electronics

MDPI

Systematic Review

Advanced Hardware Security on Embedded Processors: A 2026 Systematic Review

Ali Kia [1], Aaron W. Storey [2] and Masudul Imtiaz [1,*]

[1] Department of Electrical and Computer Engineering, Clarkson University, Potsdam, NY 13699, USA
[2] Department of Computer Science, Clarkson University, Potsdam, NY 13699, USA; storeyaw@clarkson.edu
* Correspondence: mimtiaz@clarkson.edu

Abstract

The proliferation of Internet of Things (IoT) devices and embedded processors has recently spurred rapid advances in hardware-level security. This paper systematically reviews developments in securing microcontroller units (MCUs) and constrained embedded platforms from 2020 to 2026, a period marked by the finalization of NIST's post-quantum cryptography standards and accelerated commercial deployment of hardware security primitives. Through analysis of the peer-reviewed literature, industry implementations, and standardization efforts, we survey five critical areas: post-quantum cryptography (PQC) implementations on resource-constrained hardware, physically unclonable functions (PUFs) for device authentication, hardware Roots of Trust and secure boot mechanisms, side-channel attack mitigations, and Trusted Execution Environments (TEEs) for microcontroller-class devices. For each domain, we analyze technical mechanisms, deployment constraints (power, memory, cost), security guarantees, and commercial maturity. Our review distinguishes itself through its integration perspective, examining how these primitives must be composed to secure real-world embedded systems, and its emphasis on post-standardization PQC developments. We highlight critical gaps including PQC memory overhead challenges, ML-resistant PUF designs, and TEE developer friction, while documenting commercial progress such as PSA Level 3 certified components and 500+ million PUF-enabled devices deployed. This synthesis provides practitioners with practical guidance for securing the next generation of IoT and embedded systems.

Keywords: embedded security; IoT; hardware Root of Trust; post-quantum cryptography; PUF; secure boot; side-channel attacks; Trusted Execution Environments

Academic Editors: John Vourvoulakis and John Kalomiros

Received: 21 January 2026
Revised: 8 February 2026
Accepted: 23 February 2026
Published: 9 March 2026

1. Introduction

Billions of embedded devices are now deployed in safety-critical and consumer environments [1], making hardware security essential. Unlike general-purpose computing, embedded processors such as microcontrollers (MCUs) operate under severe resource constraints (limited memory, power, and cost) yet must resist a wide range of attacks. Recent years have seen sustained research and industry effort to build security features into the hardware of these systems, moving beyond software-only protections. This review examines key advances from 2020 to 2026 in hardware-based security techniques for embedded processors, with an emphasis on microcontroller-class devices and other constrained platforms, including popular single-board systems like Raspberry Pi and NVIDIA Jetson.

We begin by summarizing the threat landscape for embedded hardware. Attackers can target the underlying silicon via physical attacks (probing, fault injection, side-channel

power analysis [2–4]), exploit the lack of secure boot to install malicious firmware, or extract secret keys stored in non-volatile memory. Meanwhile, emerging cryptographic requirements, particularly the transition to post-quantum cryptography (PQC [5]), pose new challenges for small devices. To address these challenges, researchers and practitioners have developed specialized hardware security primitives and architectures. In this paper, we survey five major areas of progress:

1. **Post-Quantum Cryptography (PQC) on Constrained Hardware:** Implementations and accelerators for quantum-resistant algorithms (e.g., lattice-based encryption and signatures) optimized for MCUs.
2. **Physical Unclonable Functions (PUFs):** Use of intrinsic manufacturing variations to generate device-unique secrets and keys without storing them in memory.
3. **Hardware Root-of-Trust and Secure Boot:** Establishing a Chain of Trust from reset, anchored in immutable hardware, to guarantee code integrity and authenticity.
4. **Side-Channel and Fault Attack Mitigations:** Circuit and architectural techniques to thwart power analysis, electromagnetic leakage, timing side-channels, and injection of faults.
5. **Trusted Execution Environments (TEEs):** Hardware-enforced isolation (such as Arm TrustZone and others) to protect sensitive code and data on embedded processors.

For each topic, we review representative developments from the 2020–2026 timeframe, highlighting both the achieved solutions and the limitations or open problems. We also discuss how these techniques are being tailored to real-world constraints in IoT devices (limited CPU performance, memory footprint, battery power, and cost). Finally, we provide an outlook on future research and standardization efforts required to further harden embedded systems against evolving threats. Table 1 summarizes the main security primitives and mechanisms discussed in this paper, highlighting their underlying benefits, the associated challenges they face, and the recent solutions proposed in the literature to address these challenges.

Before examining these technical domains in detail, we first position our work within the broader context of the hardware security literature. Section 2 reviews related surveys and highlights how our integrated, post-standardization perspective differs from prior work, establishing the unique contribution of this review.

Table 1. Summary of recent developments in securing hardware systems.

Security Primitives	Introduced (and Evolved)	Benefits	Challenges	Solutions and Recent Works
PQC	2001 (Implementations emerging)	Resistance against quantum computer attacks, long-term security, diverse algorithms	Physical attacks, memory usage and performance, implementation complexity, interoperability and standardization, energy consumption	Instruction-level constant-time implementations, noise injection, threshold cryptography, software optimization, hardware accelerators, hybrid HW/SW
PUF	2002 (Standardization and mainstream adoption: 2020s and beyond)	Device-unique identities without storing secret keys	Modeling attacks, reliability, quality, optimization of methods	Feed-forward or non-linear PUFs, response obfuscation, on-die stabilizing circuits and sensors, error correction and fuzzy extraction

Table 1. *Cont.*

Security Primitives	Introduced (and Evolved)	Benefits	Challenges	Solutions and Recent Works
RoT and Secure Boot	Mid-2000s (Integration of secure hardware into the CPU/SoC: 2020s and beyond)	Trusted firmware execution and authentication of system state	Run-time trust measures, vulnerabilities in ROM code or signing algorithms, performance and power	TEE, periodic attestation, tamper detection, fail-safe recovery mechanisms, crypto-agility
TEE	Mid-2000s (Evolving challenges and applications: 2020s and beyond)	Isolated execution of sensitive code and data	Physical attacks, technique of implementation, secure partitioning, complexity for the developer	Combination with other security techniques, separated security core, simple and efficient APIs
Side-Channel and Fault Attack Mitigation	1996 (Widespread implementations: ∼2010s; modern attacks and countermeasures: 2020s and Beyond)	Protects cryptographic keys and operations from leakage, intentional errors and manipulation	Attack discovery, countermeasure development, security evaluation, power, area and performance overheads vs. cost	Noise generation and masking for cryptographic accelerators, dual-rail logic or current balancing, random clocking, fault detection units, tamper detection sensors, secure instruction set and co-processors

Review Methodology

This systematic review was conducted following the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) 2020 guidelines [6] to ensure methodological rigor and transparency. The review protocol was registered with OSF Registries (DOI: `10.17605/OSF.IO/E3967`) after the systematic search was conducted; this retrospective registration documents the methodology as implemented. The complete PRISMA 2020 checklist, database-specific search queries, eligibility criteria, and data extraction forms are provided in the supplementary materials.

Information Sources and Search Strategy. We systematically searched four major databases covering peer-reviewed publications and preprints in hardware security: IEEE Xplore (conference proceedings and journals), IACR ePrint Archive and Transactions on Cryptographic Hardware and Embedded Systems (TCHES), ACM Digital Library (including CCS, ASIA CCS, and TECS), and arXiv (cs.CR category). The search was conducted in February 2026, covering publications from January 2020 through January 2026, a period encompassing NIST's post-quantum cryptography standardization (FIPS 203/204/205, August 2024) and the maturation of embedded security primitives.

The primary search query combined hardware security terminology with embedded platform identifiers and the five survey domains:

```
(''hardware security'' OR ''embedded security'') AND (''microcontroller'' OR
''MCU'' OR ''IoT'' OR ''Cortex-M'' OR ''RISC-V'') AND (''PQC'' OR ''post-quantum''
OR ''PUF'' OR ''secure boot'' OR ''side-channel'' OR ''TEE'' OR ''TrustZone'')
```

This query was adapted for each database's syntax while maintaining semantic equivalence. Domain-specific keywords (e.g., CRYSTALS-Kyber, SRAM PUF, TrustZone-M) supplemented the primary query to ensure broad coverage.

Eligibility Criteria. Studies were included if they: (1) were published between January 2020 and January 2026; (2) addressed implementation on resource-constrained platforms (ARM Cortex-M series, RISC-V embedded cores, or low-end FPGAs with $\leq$512 KB Flash and $\leq$256 KB RAM); (3) focused on hardware-level security mechanisms within at least one

of the five survey domains (PQC, PUF, RoT, SCA, or TEE); (4) provided implementation details or empirical evaluation; and (5) appeared in peer-reviewed venues or recognized archives with institutional affiliations. Studies were excluded if they addressed only server-class or unconstrained platforms, presented purely theoretical analyses without implementation, duplicated or were superseded by later versions, or lacked sufficient methodological detail for reproducibility.

Selection Process. The search yielded 2100 records across all databases (IEEE Xplore: 847; IACR: 312; ACM DL: 523; arXiv: 418). After removing 853 duplicates, 1247 unique records underwent title and abstract screening by two independent reviewers using the inclusion criteria, achieving inter-rater reliability of Cohen's $\kappa = 0.83$. This initial screening excluded 1035 records (412 non-embedded platforms, 287 purely theoretical, 156 scope mismatch, 89 duplicates, 18 non-English, 73 other). The remaining 212 articles underwent full-text assessment against all eligibility criteria, resulting in the exclusion of 118 articles (47 scope, 31 insufficient detail, 23 quality concerns, 17 superseded versions). The final corpus comprised 94 studies included in the synthesis. Figure 1 presents the complete PRISMA flow diagram detailing this selection process.

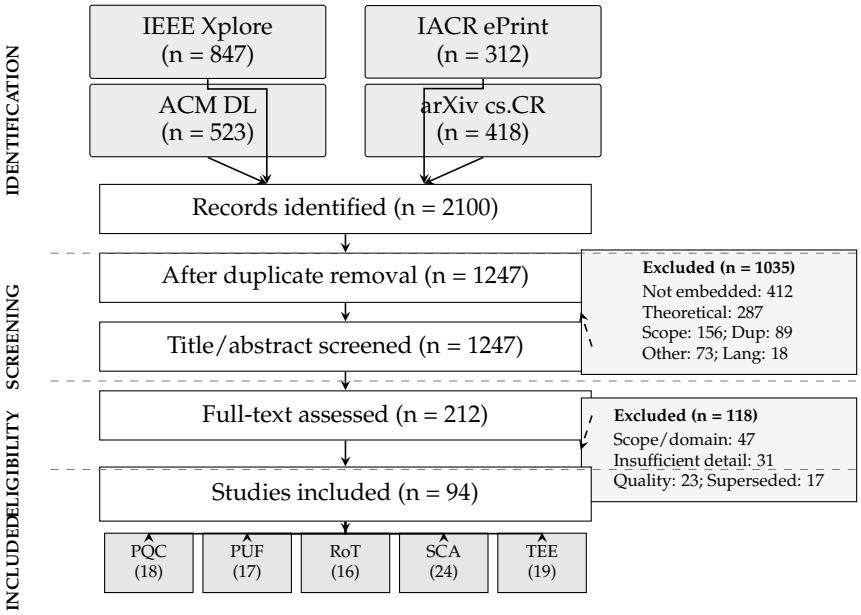


Figure 1. PRISMA 2020 flow diagram showing the systematic review selection process. From 2100 initial records, 94 studies were included after duplicate removal, title/abstract screening, and full-text eligibility assessment.

Data Extraction and Quality Assessment. Two reviewers independently extracted data using a standardized form capturing: target platform, memory footprint (ROM/RAM), performance metrics (cycles, latency), security claims and threat models, and key contributions. Discrepancies were resolved through discussion. Quality was assessed across four dimensions, each scored 0–25 points: (1) *Methodology clarity*: Is the experimental protocol described with sufficient detail for reproduction? (2) *Reproducibility*: Is the source code or implementation data publicly available? (3) *Evaluation rigor*: Was testing performed on real hardware rather than the simulation alone? (4) *Security analysis depth*: Does the work provide formal proofs, empirical attack evaluation, or both? Of the 94 included studies, 53 (56%) achieved high aggregate quality scores ($\geq$75/100), 34 (36%) medium quality (50–74), and 7 (8%) were included with noted limitations (<50). The risk of bias was mitigated through broad database coverage, independent dual screening, and the inclusion of grey literature (arXiv preprints).

Distribution of Included Studies. The 94 included studies are distributed across domains as: side-channel analysis (24), Trusted Execution Environments (19), post-quantum cryptography (18), physically unclonable functions (17), and Roots of Trust (16). And by platform as: ARM Cortex-M (40), RISC-V (22), FPGA (18), and multi-platform studies (14). This distribution reflects both research activity and commercial deployment trends in embedded hardware security during 2020–2026.

Citation Classification. This review cites 133 works total, comprising the 94 systematically reviewed studies plus 39 supporting materials. Supporting materials include: standard documents and specifications (NIST FIPS, PSA, Common Criteria), vendor technical reports and datasheets, foundational works predating our 2020–2026 search window, and industry deployment statistics. These supporting materials provide essential context but are distinguished from the peer-reviewed corpus; claims about security effectiveness and performance are grounded in the 94 systematically reviewed studies.

2. Related Work

Hardware security for embedded systems has been the subject of numerous surveys and reviews in recent years, reflecting the growing importance of securing IoT and embedded devices. However, existing surveys typically focus on specific aspects of hardware security or pre-date critical developments such as the finalization of NIST's post-quantum cryptography standards in 2024. This section positions our work in the context of recent related surveys.

General Hardware Security Surveys: Several broad surveys have examined hardware security broadly. A 2023 IEEE survey on "Hardware Security: Current Trends and Challenges" [7] investigated various hardware security approaches in the literature, addressing concerns arising from IC supply chain globalization and IoT device connectivity. Another 2023 survey examined the "Security Analysis of Embedded Devices" [8], documenting that over 32% of firmware images contain more than 10 critical vulnerabilities. While these surveys provide valuable overviews of hardware security threats and countermeasures, they do not deeply examine the integration challenges specific to resource-constrained microcontrollers nor the recent shift toward post-quantum cryptography following NIST standardization.

IoT-Specific Security Reviews: The ACM Computing Surveys published "A Systematic Review of IoT Security" in 2023 [9], examining research potential, challenges, and future directions for IoT security. This broad review categorizes vulnerabilities across hardware, software, cloud, and network layers, but maintains a broader IoT system perspective rather than focusing specifically on hardware-level security primitives for embedded processors. Industry reports continue to document widespread IoT security incidents, but these reports emphasize operational security and network-level threats rather than the hardware security mechanisms we examine.

Post-Quantum Cryptography for Embedded Systems: The emergence of quantum-resistant cryptography has spawned specialized surveys. A January 2024 arXiv survey examined "Post-Quantum Cryptography for Internet of Things: Performance and Optimization" [10], reviewing 35 publications on PQC performance, software/hardware optimization, and GPU acceleration for IoT. The IACR published "SoK: The Engineer's Guide to Post-Quantum Cryptography for Embedded Devices" in 2024 [11], providing practical guidance for embedded developers. A February 2024 survey addressed "Post-Quantum Based Approaches for Edge Computing Security" [12]. While these surveys excel in PQC analysis, they do not integrate PQC with other hardware security primitives (PUFs, TEEs, secure boot, side-channel protections) that must work in concert for complete embedded system security.

Specific Security Primitive Surveys: Individual security mechanisms have received dedicated attention. Research on Physical Unclonable Functions has been surveyed in the context of IoT authentication and key generation. Trusted Execution Environments have been reviewed in Shepherd and Markantonakis's 2024 book "Trusted Execution Environments" [13], though with emphasis on mobile- and server-class processors rather than microcontroller-class devices. Side-channel attack countermeasures have been extensively documented in the cryptographic engineering literature, but rarely in the context of emerging PQC algorithms.

Our Contribution: Our review complements prior surveys by synthesizing five security domains through the lens of practical embedded deployment. We acknowledge substantial overlap with recent PQC surveys [10,11] and TEE coverage in Shepherd's 2024 book [13]; our contribution lies not in exhaustively re-reviewing these domains but in examining their *integration* for resource-constrained platforms. Specifically, we emphasize: (1) *post-standardization context*: developments after NIST's PQC finalization (August 2024), which changes deployment calculus for embedded systems; (2) *composition challenges*: how PQC, PUFs, RoT, SCA countermeasures, and TEEs interact, including trade-offs and conflicts not visible in single-domain surveys; (3) *MCU-class constraints*: Cortex-M, RISC-V embedded cores, and platforms like Raspberry Pi face different challenges than server-class systems; and (4) *commercial deployment*: PSA certification levels, vendor implementations, and real-world adoption data ground our analysis in industrial practice.

Existing surveys excel in depth within individual domains; our review provides breadth across domains with emphasis on integration and deployment realities for time-constrained practitioners.

3. Post-Quantum Cryptography on Embedded Systems

Having established the gaps in the existing literature, we now examine each of the five critical security domains outlined in Section 1. We begin with post-quantum cryptography, as the 2024 NIST standardization is the most consequential recent development affecting all other security primitives discussed in this review. The impending transition to post-quantum cryptography has far-reaching implications for embedded devices. In 2022, the U.S. National Institute of Standards and Technology (NIST) announced [14] the first algorithms selected for standardization (e.g., CRYSTALS-Kyber key encapsulation [15] and CRYSTALS-Dilithium signatures [16]), with final standards published in 2024. Unlike classical RSA [17] or elliptic-curve cryptography (ECC [18]), these lattice-based [19] and hash-based [20] schemes involve much larger key and signature sizes as well as heavier computation. This poses a major challenge for resource-constrained processors: memory and speed can become bottlenecks.

Experts consider memory footprint the primary hurdle for PQC adoption in embedded contexts. As Joppe Bos, a cryptographic researcher at NXP Semiconductors, emphasized [21], the keys and signatures in PQC (often on the order of kilobytes) vastly exceed those in ECC (tens of bytes), straining microcontroller RAM/flash capacities. For example, a typical 256-bit ECC public key is 64 bytes, whereas a lattice-based Dilithium signature can be ∼2–3 KB. Figure 2 illustrates this difference, comparing key and output sizes of classical vs. post-quantum schemes. These larger data sizes mean that embedded developers must account for increased memory usage and transmission overhead in IoT applications. Post-quantum schemes require dramatically larger key and output sizes, which poses challenges for memory-constrained devices.

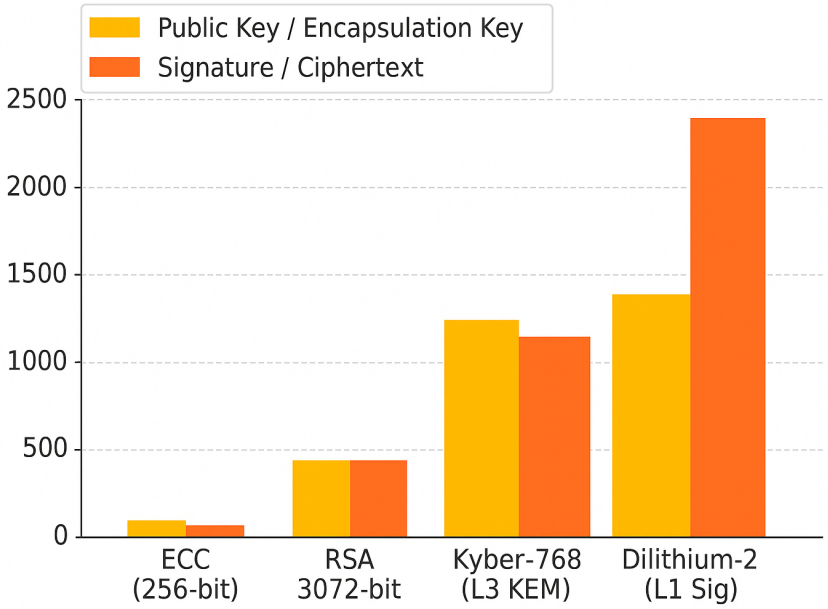


Figure 2. Public key and signature/ciphertext sizes for classical cryptography versus post-quantum schemes (Kyber, Dilithium, SPHINCS+).

Despite these overheads, the importance of PQC in ensuring resilience against quantum adversaries has driven broad adoption efforts. Developers and standardization bodies recognize that long-term security guarantees justify the additional memory and bandwidth costs. Table 2 highlights this trade-off by presenting the differences between classical cryptographic schemes and emerging PQC candidates, outlining their key specifications and design considerations.

Table 2. Comparison of classic and post-quantum cryptography features.

Feature	Classic Crypto (RSA, ECC)	Post-Quantum Crypto (Kyber, Dilithium, etc.)
Security basis	Factorization, discrete log	Lattices, hashes, codes, multivariate polynomials
Quantum resistance	Broken by Shor's algorithm	Designed to resist quantum attacks
Key sizes	Small (e.g., 256-bit ECC)	Large (e.g., 1–2 KB for lattice PQC)
Performance	Fast, efficient	Often slower, higher memory
Standardization	Mature, globally adopted	Ongoing (NIST PQC standards 2022–2024)
Hardware support	Widely available (AES-NI, RSA accelerators)	Emerging, requires optimizations
Physical attack protection	Established and practiced in hardware implementations	Under active development; countermeasures are still being studied

In response, researchers from 2020 onward have focused on optimizing PQC algorithms for the embedded domain. This includes: (a) software optimizations such as using hand-tuned assembly for big integer arithmetic and number-theoretic transforms [22–24], with the pqm4 benchmark suite [25] establishing standardized performance metrics for Cortex-M4; (b) hardware accelerators and instruction set extensions to offload expensive operations (e.g., polynomial multiplication in lattice cryptography) [26]; and (c) hybrid HW/SW co-designs with side-channel protections including polynomial masking techniques [27]. For instance, one 2021 study proposed a masked hardware/software co-design for accelerating the NIST PQC finalists Kyber and Saber on a RISC-V core, integrating countermeasures to mitigate power analysis leakage [28]. By carefully partitioning the algorithm

between the CPU and dedicated co-processor, they achieved acceptable performance while maintaining resistance to side-channel attacks. Many microcontroller vendors are likewise preparing cryptographic libraries and updates to support PQC. As of 2026, industry leaders including NXP and Infineon are developing PQC-capable secure elements and MCU crypto libraries targeting the NIST-standardized algorithms Kyber and Dilithium, with initial implementations focusing on secure boot and firmware authentication use cases.

Another important consideration is the physical security of PQC implementations. The first generation of PQC schemes have not yet been vetted by decades of real-world attack testing, and initial research indicates they may have new side-channel vulnerabilities. For example, certain lattice operations might be susceptible to timing or cache attacks, and some PQC algorithms have shown sensitivity to fault injection [29]. Thus, embedded security engineers face the dual challenge of fitting PQC into limited devices and simultaneously hardening it against physical attacks. NXP researchers have noted the difficulty of hardening some PQC KEMs against side-channel attacks, calling for unconventional approaches to protect these schemes [30]. Techniques like instruction-level constant-time implementations [31], noise injection [32], and threshold cryptography (splitting secrets across multiple shares [33]) are being explored to secure PQC on IoT endpoints.

Despite these challenges, the consensus is that planning for PQC in embedded systems should begin now. Security agencies are recommending a phased migration (e.g., hybrid schemes combining classical and PQC algorithms) for high-value devices starting in 2025 [34]. A concrete near-term use case is secure firmware updates and secure boot. Industry experts advise that embedded devices should prioritize making their bootloader signatures quantum-resistant, since a device's longevity may exceed the timeframe of quantum adversaries becoming practical. For instance, by using Dilithium or SPHINCS+ signatures [35] to validate firmware and boot code, an IoT device can ensure that even a future quantum attacker cannot forge software updates or boot images. By deploying PQC-based secure boot today (in parallel with existing RSA/ECC verification), manufacturers create a "hedge" that can be fully switched to PQC-only by the 2030s.

FALCON Considerations: While this review focuses on Kyber and Dilithium as the primary NIST-standardized lattice schemes, FALCON (FIPS 206 [36]) merits consideration for specific embedded use cases [29,37]. FALCON offers significantly smaller signatures than Dilithium (666 bytes vs. 2420 bytes at comparable security levels), making it attractive for bandwidth-constrained applications such as over-the-air firmware updates or certificate chains. However, FALCON's reliance on floating-point arithmetic presents implementation challenges on MCUs lacking FPU hardware, requiring costly software emulation on Cortex-M0/M3 class devices. Additionally, FALCON's Gaussian sampling is notoriously difficult to protect against timing attacks, with side-channel secure implementations showing 3–5× overhead compared to unprotected versions. Light-weight authenticated encryption schemes like ASCON [38,39] complement PQC by providing efficient symmetric primitives. Consequently, most embedded PQC deployments favor Dilithium despite its larger signatures, reserving FALCON for scenarios where signature size critically impacts system performance or cost.

Critical Analysis: PQC Deployment Trade-offs. The PQC migration for embedded systems presents a complex cost-benefit calculation that varies dramatically across device tiers and deployment contexts. For high-value, long-lifetime devices (industrial controllers, medical implants, critical infrastructure sensors), the "harvest now, decrypt later" threat model justifies immediate PQC adoption despite 2–10× memory overhead and potential performance penalties. However, for cost-optimized IoT endpoints with sub-dollar BOM targets and 2–3 year replacement cycles, the economic case remains ambiguous: quantum computers capable of breaking 2048-bit RSA may not materialize within device

lifetimes [40], while PQC memory requirements could necessitate moving from $0.50 MCUs to $2.00 parts, a 4× cost increase that fundamentally alters product economics. The hybrid approach (dual RSA+Dilithium signatures) superficially appears optimal but carries hidden costs: doubled signature verification time at boot (critical for battery-powered devices), increased attack surface (vulnerabilities in either algorithm compromise security), and increased code complexity. Practitioners must navigate three interconnected decisions: (1) timing (migrate now vs. wait for hardware acceleration and optimized libraries), (2) scope (full PQC vs. selective application to high-risk operations like firmware updates), and (3) algorithm selection (Dilithium's performance vs. SPHINCS+'s stateless simplicity vs. Kyber's smaller ciphertext). The performance data in Table 3 reveals that Kyber512 key encapsulation (536K cycles) is feasible on even modest MCUs, while Dilithium5 signing (9.04M cycles) approaches one-second latency on 24 MHz Cortex-M4, acceptable for infrequent operations like firmware authentication but prohibitive for per-packet cryptography. Looking forward, the critical enabler will be dedicated PQC accelerators that reduce cycle counts by 10–100×, transforming PQC from a premium feature to a baseline expectation by 2028–2030.

Taken together, post-quantum cryptography is rapidly moving from theory to practice on embedded processors. The last five years have delivered initial solutions for making PQC feasible on constrained devices, through optimized libraries, hardware support, and hybrid designs, but more work is needed. Memory usage and performance optimizations remain critical, as does developing standardized side-channel countermeasures. The coming years will likely see the first PQC-enabled microcontrollers, specialized PQC crypto accelerators, and widespread integration of PQC in IoT security protocols.

Table 3. Performance comparison of cryptographic algorithms on Arm Cortex-M4 (24 MHz).

Algorithm	Key Size (Bytes)	Sig/CT Size (Bytes)	Keygen (Cycles)	Sign/Encaps (Cycles)	Verify/Decaps (Cycles)	Reference
Post-Quantum Algorithms (NIST-standardized)						
Kyber512	1632	736	443K	536K	513K	[41]
Kyber768	2400	1088	745K	899K	839K	[41]
Kyber1024	3168	1568	1.19M	1.37M	1.29M	[41]
Dilithium2	2528	2420	1.60M	4.09M	1.57M	[41]
Dilithium3	4000	3293	2.83M	6.72M	2.70M	[41]
Dilithium5	4864	4595	4.72M	9.04M	4.72M	[41]

Note: K = thousand, M = million cycles. Kyber provides key encapsulation (KEM); Dilithium provides digital signatures. All values measured on ARM Cortex-M4 @ 24 MHz from optimized implementations with ARM assembly and NTT optimizations [41]. Kyber and Dilithium represent the NIST-standardized lattice-based PQC algorithms (FIPS 203 and FIPS 204). For comparison, classical RSA-2048 keygen requires ∼2 billion cycles (1000× slower than Dilithium2), while ECDSA P-256 requires ∼1.6–3.8 million cycles for keygen/sign/verify operations.

While post-quantum cryptography addresses algorithmic vulnerabilities to quantum attackers, cryptographic operations themselves require secure key storage and device authentication. Simply having quantum-resistant algorithms is insufficient if secret keys can be extracted from the device memory or if attackers can impersonate legitimate devices. This motivates our next topic: hardware-based device identity and key generation through Physical Unclonable Functions.

4. Physical Unclonable Functions (PUFs) for Device Identity

A Physical Unclonable Function (PUF) is a hardware primitive that leverages inherent manufacturing variability to produce a unique, device-specific response (e.g., a binary string) that serves as a "silicon fingerprint" [42,43]. The idea is that each chip harbors slight random variations (threshold voltages, doping concentrations, etc.) introduced during

fabrication; these variations are unpredictable and practically impossible to replicate even by the manufacturer [44]. PUF circuits harness these differences. Standard implementations include SRAM startup patterns, ring oscillator frequency differences, or arbiters' delay races to derive secret keys or identifiers on-chip [45]. Over the past two decades, PUF research has matured steadily, moving from early physical one-way functions to silicon-based designs and, more recently, to light-weight and AI-resilient architectures. Figure 3 provides a timeline of key milestones in PUF development, illustrating this progression.

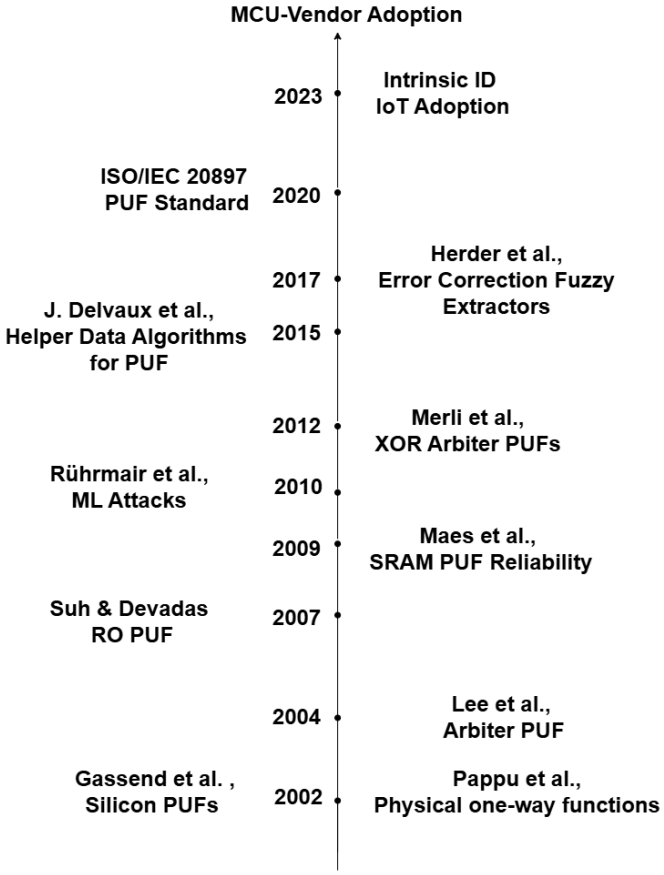


Figure 3. Timeline of key developments in Physical Unclonable Functions (PUFs) [43,44,46–54].

PUFs gained widespread traction in the 2020s as a means to establish a hardware Root of Trust in IoT devices without storing any permanent secret in memory. Instead of fuses or non-volatile memory for key storage (which can be read out or tampered with), a PUF allows the device to *derive* a secret key internally whenever needed, using its unique physical fingerprint. An external entity (e.g., the device manufacturer or a server) can be provided with the expected PUF response (or a processed form of it) during provisioning, enabling remote authentication of the device in the field. Crucially, if an attacker power-cycles or probes the device, the secret is erased (since it is only reconstructed on demand and not stored persistently).

In the past five years, there have been several notable developments in PUF technology for embedded systems.

SRAM PUFs in mainstream MCUs: Static RAM (SRAM) PUFs, which use the random power-up values of SRAM cells, have become one of the most deployed types. Companies like Intrinsic ID report that [54] their SRAM PUF IP has been integrated into over 500 million devices by 2023, including microcontrollers, FPGAs, and ASICs. This demonstrates a maturation from research concept to real-world adoption at scale. Many secure IoT MCUs

(from NXP, Microchip, etc.) now tout "PUF-based key storage" [55] where the master root key is derived from silicon each time, rather than stored in flash.

Software-only PUF solutions for existing chips: Beyond new silicon, there have been efforts to retrofit PUF capabilities into devices via software. For example, Intrinsic ID introduced a software-based solution (the Zign X00 series) that can generate keys from SRAM patterns on chips already deployed, without any custom hardware [56]. This approach leverages the fact that even general-purpose SRAM in a microcontroller can act as a PUF source. Such software PUFs enable adding a root-of-trust to COTS (commercial off-the-shelf) devices post-fabrication.

Integration with secure element chips and modules: PUFs have also been integrated into dedicated secure elements and TPM-like chips for IoT [57]. For instance, some secure element devices use a PUF to protect an internally stored key: the key is combined with a PUF-derived value so that even if memory is read out, the true key cannot be reconstructed without the device's unique PUF response. This adds a layer of protection against physical extraction.

PUF quality improvements: Recent research has focused on improving the reliability and randomness of PUF responses. Techniques like better error-correcting codes (to handle noise and environmental drift in PUF readings), fuzzy extractors [58], and the lockdown of responses via one-time programming of helper data have been refined [52]. There is also an emphasis on evaluating PUFs across wider temperature and voltage ranges to ensure they remain reliable for IoT applications that may experience harsh conditions.

A practical application of PUFs is enabling each device to "create its own secrets" on first boot, avoiding any secret distribution over communications. In practice, during manufacturing, a device can run a key derivation from its PUF and use that to generate a public/private key pair or symmetric key. The public portion (or a hash of the PUF key) can be registered to a server or printed as a QR code on the device, while the private key never leaves the chip. This zero-touch provisioning enhances supply chain security: even the manufacturer might not need to know the device's private key, reducing insider attack risk. As Pim Tuyls (CEO of Intrinsic ID) described, "no secrets are sent over the air; a device creates its own secrets" using PUF-based key generation [56].

Despite their promise, PUFs are not without challenges. A major concern is susceptibility to modeling attacks, where an attacker gathers many challenge–response pairs (CRPs) from a PUF and uses machine learning to predict its behavior [50]. This is especially relevant for so-called "strong PUFs" that have a large CRP space (like Arbiter PUFs). Research in the early 2020s demonstrated that deep neural networks can successfully learn models of many PUF architectures, effectively breaking their unpredictability [59]. For example, XOR Arbiter PUFs and other complex PUFs, once proposed as secure, have been modeled with high accuracy by ML, given enough CRP data. This has diminished the viability of strong PUFs for authentication protocols. In response, new PUF designs (e.g., feed-forward or non-linear PUFs [60]) and obfuscation techniques including lockdown mechanisms [61,62] are being studied to thwart ML attacks, but this remains an ongoing challenge.

Another issue is environmental reliability: PUF responses can drift with temperature, aging, or supply voltage. Thus, error correction and fuzzy extraction are essential parts of any practical PUF system. The 2020–2026 period saw improved error-correcting schemes and reliability testing. For instance, incorporating temperature sensors or stabilizing circuits on-die can help ensure consistent PUF output across conditions [63].

Critical analysis: PUF selection and integration strategies. The PUF technology landscape presents a stark dichotomy between commercial maturity (SRAM PUF) and research-stage innovations (memristor, quantum PUFs), requiring practitioners to carefully match PUF selection to threat models and deployment constraints. Table 4 reveals that

SRAM PUF dominates with 500M+ deployed devices because it offers zero marginal cost (exploiting existing SRAM) and moderate security that is suitable for device authentication and key derivation in non-adversarial physical environments. However, the 1–5% BER and vulnerability to ML modeling attacks (when CRPs are exposed) make SRAM PUF unsuitable for high-security applications where attackers have physical access and computational resources. The critical decision framework involves three factors: (1) attack surface (is CRP exposure possible?), (2) environmental range (industrial −40 to +85 °C vs. consumer 0–50 °C affects reliability), and (3) security budget (PUF alone vs. PUF + secure element). For applications requiring ML resistance, memristor PUF shows promise with ∼50% ML attack accuracy (equivalent to random guessing) but remains research-stage with unknown long-term reliability. The integration question, PUF versus discrete secure element (ATECC608, OPTIGA), hinges on cost–security trade-offs: secure elements provide CC EAL5+ assurance and side-channel resistance but add $0.50–2.00 BOM cost and board space, while SRAM PUF is effectively free but offers PSA Level 1–2 equivalent security. The optimal architecture for many mid-tier devices combines both: PUF-derived keys protect secure element access, creating defense-in-depth where extracting secrets requires both physical PUF readout (difficult due to unique device characteristics) and secure element attack (difficult due to tamper protections). The 500M deployment figure demonstrates commercial viability, but practitioners must recognize that PUF is a component, not a complete solution; it must integrate with secure boot (Section 5), side-channel protections (Section 6) and TEEs (Section 7) to form a cohesive security architecture. The emerging ISO/IEC 20897 standardization will help, but gaps remain in cross-vendor interoperability and in standardized metrics for ML resistance evaluation.

Table 4. Comparison of Physical Unclonable Function (PUF) technologies for embedded devices.

PUF Type	Area Overhead	BER (%)	Uniqueness (%)	ML Resistant	Deployment Status	Reference
SRAM PUF	Minimal	1–5	46–50	Moderate	Commercial, 500M+ devices	[54]
Ring Oscillator	Low	2–8	48–52	Moderate	Commercial, widely deployed	[59]
Arbiter PUF	Low	5–15	45–55	Low	Research, ML vulnerable	[59]
XOR Arbiter	Medium	10–20	48–52	Low	Research, broken by ML	[59]
Feed-Forward	Medium	8–15	47–53	High	Research, ML-resistant	[60]
Memristor PUF	Low–Med	1–4	48–52	High	Research, ∼50% ML accuracy	[64]
Quantum PUF	High	Unknown	Unknown	Very High	Theoretical, proof-of-concept	[65]

Note: BER = Bit Error Rate (environmental noise sensitivity); uniqueness = inter-device variation (ideal: 50%); ML resistant = robustness against machine learning modeling attacks. BER and uniqueness ranges represent typical values from the PUF literature; actual values vary by implementation, process technology, and operating conditions. SRAM PUF dominates commercial deployment due to zero area overhead (uses existing SRAM), with 500M+ devices deployed globally [54]. Memristor and quantum PUFs show promise for ML resistance (achieving ∼50% attack accuracy, equivalent to random guessing) but remain in early research stages. Feed-forward architectures provide the best balance of ML resistance and practicality for future deployment [60].

5. Hardware Roots of Trust and Secure Boot

Establishing a hardware Root of Trust (RoT) is a foundational step in securing any computing system, and embedded devices are no exception [66,67]. A Root of Trust is a minimally complex, highly reliable hardware block (or combination of hardware and immutable software in ROM) that is implicitly trusted to perform security-critical functions like device identity, secure boot, and cryptographic operations [68]. In the embedded context, this often means a region of the MCU (or a separate secure element) that stores a secret key or certificate, and boot ROM code that uses that key to verify the initial firmware. Light-weight RoT architectures such as TrustLite [69], Sancus [70], and SANCTUARY [71] have been specifically designed for resource-constrained embedded devices, with LiteHAX [72] focusing on minimal attestation overhead.

The secure boot process is typically as follows: on reset, the CPU executes a bootloader from on-chip ROM (which cannot be modified). This bootloader code contains a public key

or hash (the RoT key) that is considered a trust anchor. It uses that key to verify a digital signature on the next stage of the boot code (e.g., the user's firmware in flash). Only if the signature is valid (i.e., the code is authenticated and untampered) will it allow the execution to proceed. In this manner, a *chain-of-trust* is established, where each stage of software is validated by a previously trusted stage, all the way from hardware to application. The hardware Root of Trust usually incorporates cryptographic accelerators (for signature verification), secure storage for keys, and sometimes monotonic counters or anti-rollback features to prevent older vulnerable firmware from being loaded [73].

From 2020 to 2026, we have witnessed an increasing adoption of secure boot and hardware RoTs in even the smallest devices:

Built-in Secure Boot in MCUs: Many modern MCU families (Arm Cortex-M based and others) now come with secure boot functionality out-of-the-box. For example, Microchip's SAMA7G54 (a 1 GHz Cortex-A7 microprocessor for IoT) advertises secure boot with an onboard key store and cryptographic engine, preventing unauthorized code execution at startup. Similar capabilities exist in STMicroelectronics STM32L5 series, NXP's LPC/IMX RT series, etc. Typically, these devices allow the user to burn a public key or hash into a one-time programmable (OTP) memory or fuse. On reset, internal hardware will compute a hash of the firmware and compare it to the trusted hash (or verify a signature using the stored public key), thereby ensuring only signed firmware can run.

External Secure Elements and TPMs: In cases where the main CPU is too small to include strong security, an external secure element chip can act as the Root of Trust. These chips (e.g., Microchip ATECC608 [74], Infineon OPTIGA Trust, or TPM 2.0 modules) store keys and perform cryptographic verification. The main MCU at boot can establish a secure channel to the secure element to ask it to sign or verify something, ensuring that secrets never leave the secure hardware. While these add cost, they have become popular in industries like automotive and industrial IoT where strong assurance is required.

Platform Security Architecture (PSA) Certified RoT Components: Arm's Platform Security Architecture (PSA) has defined levels of security for IoT devices. By 2023, we even saw the first *PSA Level 3 certified* Root of Trust IP, which includes protection against hardware attacks (side-channel, fault injection). Intrinsic ID's QuiddiKey 300 (a PUF-based key storage IP) achieved PSA Level 3 certification as a Root of Trust component [75], indicating it has strong defenses even against sophisticated physical attacks. This reflects a broader trend of certifying hardware RoT implementations to meet security standards (e.g., EMVCo [76] for payments, FIPS 140-3 Level 3 [77], etc.) for IoT sectors that demand high trust. Table 5 lists the PSA certified levels, their focus, typical threats covered, and common use cases, providing a clear view of how security assurance increases from Level 1 to Level 3.

Table 5. PSA certified levels with focus, threats, and typical use cases.

PSA Certified Level	Description
Level 1	Documentation and design review; covers basic software attacks (no physical access); typical use: IoT devices, consumer products.
Level 2	Independent lab evaluation with penetration testing (white-box); covers software and limited physical/logical attacks; typical use: connected home, smart energy, medical devices.
Level 3	Highest assurance with advanced testing, side-channel and fault injection resistance; covers sophisticated physical and logical attacks; typical use: critical infrastructure, payment systems, automotive, defense.

Roots of Trust in open architectures: Even the open-source RISC-V architecture is being used to build custom Roots of Trust. Projects like OpenTitan [78] (an open silicon Root of Trust design) emerged to create transparent, auditable RoT hardware for applications like data center servers and potentially IoT devices. While not an MCU per se, OpenTitan's concepts (secure boot, immutable ROM, crypto engines, key manager, etc.) inform embedded design principles.

A hardware RoT is closely tied to secure boot. Together, they ensure that a device only runs trusted software from the very first instruction. This prevents persistent malware from injecting itself at boot or adversaries from installing custom firmware (a common attack on IoT devices). The importance of secure boot is underscored by regulatory and industry guidelines: for instance, recent IoT cyber-security frameworks consider secure boot a mandatory requirement for any "reasonable security" device.

However, implementing secure boot in small devices is not without difficulties. One challenge is how to update the Root of Trust itself (e.g., if a vulnerability is found in the ROM code or the signing algorithm). Generally, the ROM is immutable, so upgradability is limited. Some systems include a secondary updatable bootloader that can be patched, but the initial verification key remains fixed unless physically reprogrammed. Another issue is balancing performance and power: signature verification (e.g., an RSA-2048 or ECDSA signature) at boot can be expensive on a tiny MCU. This has driven interest in lighter-weight signature schemes or accelerators to perform verification quickly. PQC adds a new wrinkle here, as discussed: switching to a Dilithium signature at boot means verifying a 2KB signature with a potentially expensive algorithm on a microcontroller. Research is ongoing into optimizing PQC verification for secure boot [79], or using a hybrid approach (one recent idea is a two-step boot where a fast classical signature boots a firmware that then verifies a PQC signature out-of-band).

Secure boot must also be complemented by run-time trust measures. Simply authenticating firmware at boot is insufficient if an attacker can later exploit a vulnerability and take control. Thus, many systems combine secure boot with a TEE or with periodic attestation. Remote attestation protocols [80–82], for instance, can allow a device to prove to a server that it's still running untampered code (by signing a measurement of memory using its RoT key). Control-flow attestation [83,84] extends this by verifying execution paths, not just static code. These are being actively explored in the IoT space, often using the same RoT that secure boot establishes. The overview of the explained security primitives is illustrated in Figure 4. At the foundation lies the hardware Root of Trust (hRoT), which satisfies the requirements of PSA Level 3. Built upon the hRoT is the Chain of Trust, where secure boot ensures the integrity and authenticity of each software stage before transferring control to the Trusted Execution Environment (TEE). Finally, the application layer relies on the TEE to access secure services. This layered structure demonstrates how the previously discussed concepts, along with recent security achievements, are integrated into a coherent architecture.

Critical Analysis: Root-of-Trust Certification and Crypto-Agility Tensions. The democratization of hardware RoT from premium smartphones to sub-dollar MCUs masks a fundamental tension between security assurance (via certification) and adaptability (crypto-agility for algorithm migration). Table 5 illustrates the PSA certification ladder where Level 3 (requiring side-channel and fault injection resistance with independent lab evaluation) represents the gold standard but imposes 12–18 month certification timelines and $100K–500K costs comparable to Common Criteria EAL3+ evaluations [85], economically viable only for high-volume products (automotive ECUs, payment terminals, medical devices). This creates a bifurcated market: premium devices achieve PSA L3 or Common Criteria EAL5+ with multi-year development cycles and extensive documentation, while

mass-market IoT endpoints settle for PSA L1 (self-certification via security checklist) that provides minimal assurance against sophisticated attackers. The crypto-agility challenge compounds this: RoT designs hardcode trust anchors in OTP fuses or immutable ROM, creating long-term lock-in to specific algorithms, problematic given PQC migration timelines and the possibility of cryptographic breaks (as occurred with SHA-1 and MD5). A Dilithium public key burned into OTP in 2025 might need replacement by 2035 if vulnerabilities emerge, but OTP is by definition one-time programmable. Industry responses include multi-tier bootloaders (immutable ROM validates updateable bootloader, which validates firmware), but each indirection adds latency and complexity. The OpenTitan project's open-source silicon RoT represents an intriguing transparency-through-auditability approach, but reaching production (Feb 2025) required 5+ years and significant investment, timescales incompatible with fast-moving IoT markets. The practical deployment strategy for most embedded systems involves risk-tiered architectures: critical path operations (initial boot, secure element communication) use certified RoT components (even if external secure elements), while less critical functions rely on MCU-integrated features. The emerging pattern combines PUF-derived keys (Section 4) for device-unique secrets, PSA L2–L3 certified crypto accelerators for operations, and TEE isolation (Section 7) for runtime protection; no single component provides complete security, but defense-in-depth creates acceptable risk profiles. Looking forward, the industry needs standardized crypto-agile RoT architectures that support algorithm migration without device replacement, and simplified certification processes (perhaps via compositional security arguments) that make PSA L2–L3 economically viable for mid-tier IoT products.

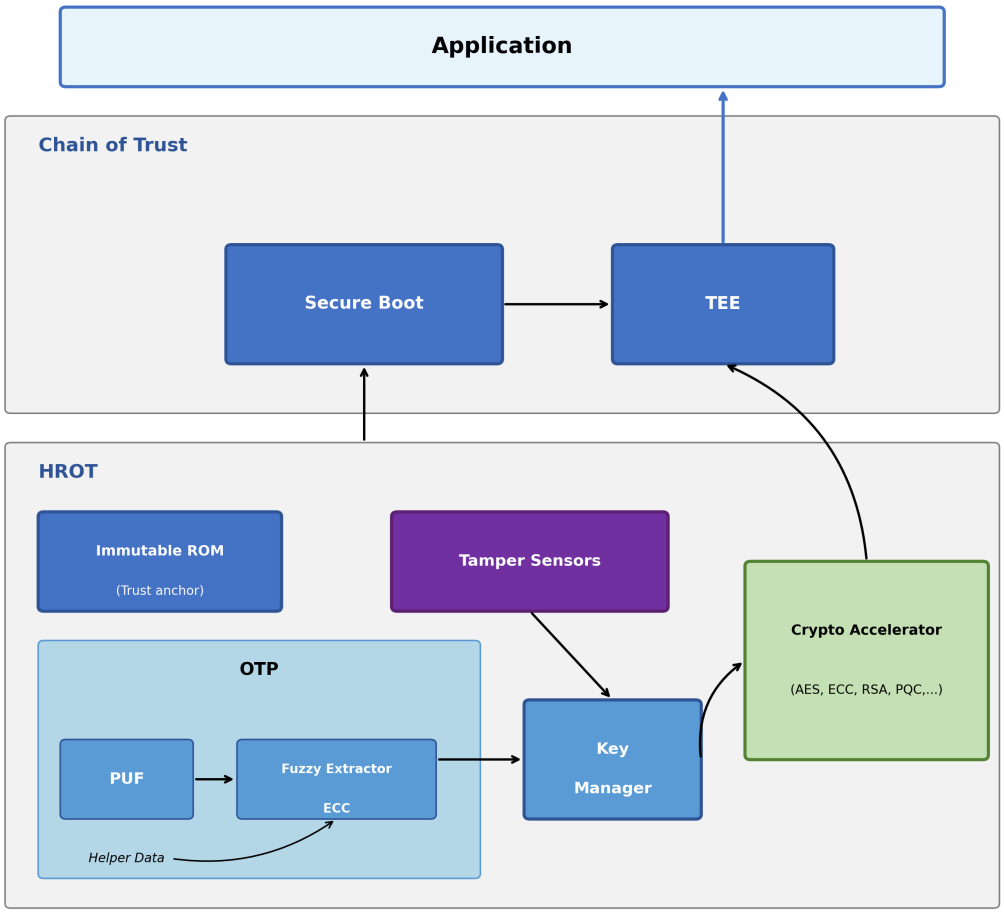


Figure 4. Layered Root of Trust and Chain of Trust architecture.

Over the 2020–2026 period, secure boot and hardware Roots of Trust have moved from high-end devices down to microcontrollers and IoT gadgets. Techniques once reserved for smartphones or PCs, such as verified boot, unique device keys, and isolated key storage, are now common even in relatively simple embedded processors. This trend is driven by both technological need and by recognition that insecure IoT devices can have far-reaching consequences (botnets [86], infrastructure attacks, etc.). The challenge ahead lies in making these Roots of Trust as light-weight and foolproof as possible: incorporating features like tamper detection, fail-safe recovery (if verification fails), and crypto-agility (to migrate to new algorithms), all under tight resource constraints.

Hardware Roots of Trust and secure boot establish architectural security, but their cryptographic operations (signature verification, key derivation, PUF readout) are themselves vulnerable to physical attack. An attacker with physical access to a device can exploit side-channel leakage during these security-critical operations to extract the very secrets that the Root of Trust is designed to protect. This threat motivates dedicated countermeasures against physical attacks.

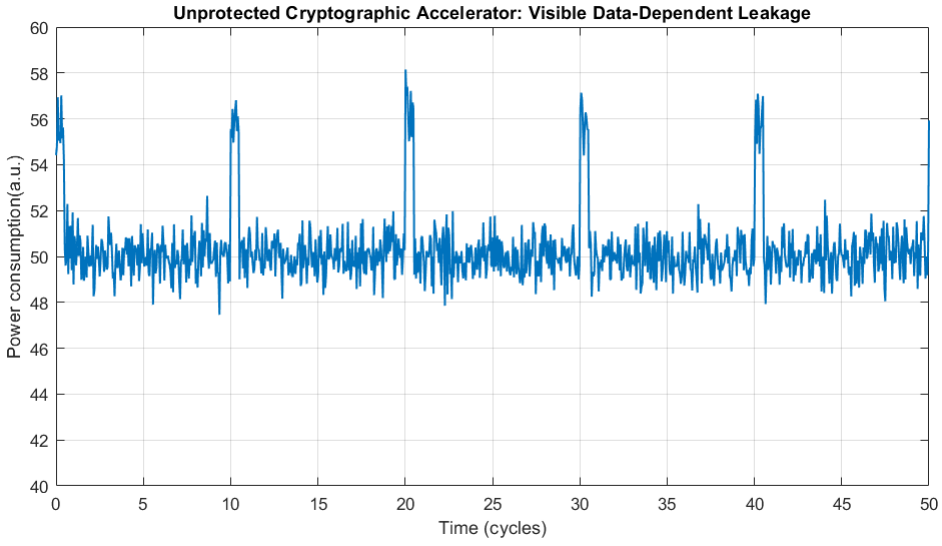
6. Side-Channel Attack Mitigations in Hardware

Side-channel attacks (SCAs) exploit unintentional information leakage from a system's physical implementation to compromise security. The foundational work by Kocher on timing attacks [87] and differential power analysis [88] established the theoretical basis for these threats, which have grown steadily more sophisticated since the late 1990s. Common examples include measuring power consumption, electromagnetic emissions, or execution timing to infer secret keys [89,90]. Embedded devices are especially vulnerable to SCAs because an attacker can often obtain physical proximity or possession of the device (e.g., a consumer IoT gadget or a remote sensor) and because these devices lack the layers of abstraction that might obscure such signals [91]. Fault injection attacks (inducing glitches in clock or power, or using lasers) can force a device into abnormal states that reveal secrets or bypass security checks [92].

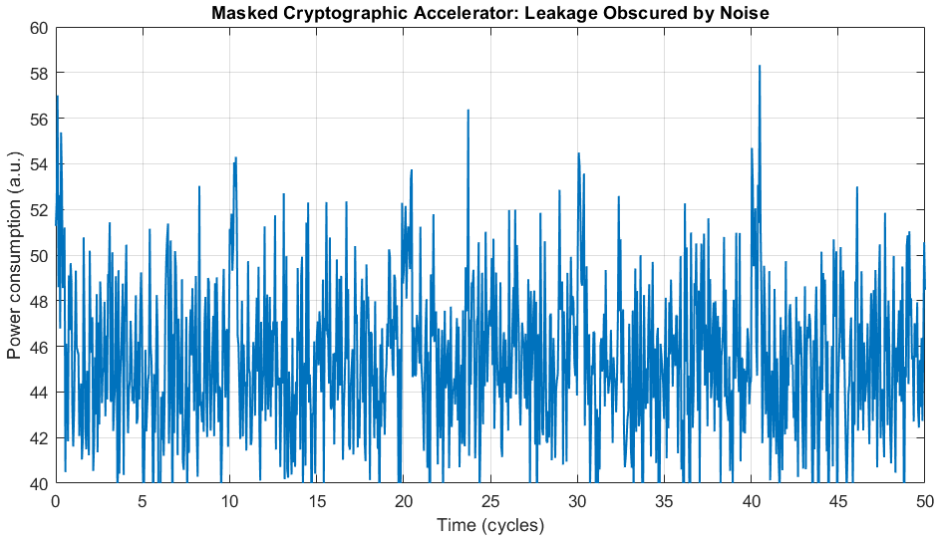
Countering side-channel and fault attacks has become a major focus in hardware security research during the past five years. Some notable advancements and strategies include:

Integrated Hardware Countermeasures: Secure cryptographic accelerators within microcontrollers now commonly implement countermeasures like noise generation [93,94] and masking [95,96]. Noise generation might involve adding random delays or dummy operations to confuse timing analysis, or shuffling the order of operations [97–100]. Masking involves splitting sensitive computations (e.g., an AES S-box operation) into parts using random masks so that the power consumption depends on randomized intermediate values rather than the true secret [101,102]. Figure 5 conceptually illustrates the effect of masking and noise generation on power consumption traces. While unprotected operations exhibit clear data-dependent patterns, the application of noise and masking countermeasures significantly obfuscates the leakage, making side-channel analysis more difficult.

In 2026, many secure MCUs (especially those used in payment, automotive, or military contexts) feature hardware AES and RSA engines that are advertised as "DPA-resistant" or "SPA-resistant" (resistant to differential or simple power analysis). This is often achieved by dual-rail logic [103] or current balancing in the circuit, making the power trace less correlated to the data.



(**a**)



(**b**)

Figure 5. Effect of masking and noise on power consumption traces: (**a**) before, showing data-dependent leakage; (**b**) after, with countermeasures obscuring patterns and reducing side-channel leakage.

6.1. Electromagnetic Side-Channel Attacks

While power analysis attacks have received extensive attention, electromagnetic (EM) side-channel attacks have emerged as an equally potent threat to embedded systems, particularly for modern applications involving machine learning inference and biometric authentication. EM emanations offer attackers several advantages over power-based attacks: they can be captured non-invasively from greater distances, they provide spatial resolution allowing targeting of specific circuit regions, and they bypass power-filtering countermeasures that attenuate supply-line leakage [3].

EM Attacks on Machine Learning Accelerators: Batina et al. demonstrated that electromagnetic emanations from embedded processors can be exploited to reverse engineer neural network architectures [104]. Using simple and differential EM analysis on an ARM Cortex-M3 microcontroller, the authors successfully recovered hyperparameters including activation functions, the number of hidden layers and neurons, and pre-trained weights from multi-layer perceptrons and convolutional neural networks. This attack requires no access to training data and poses a significant threat to proprietary machine learning models

deployed on edge devices such as wearables, smart cameras, and industrial vision systems. The implications extend beyond intellectual property theft: recovered model architectures enable targeted adversarial attacks and model inversion, potentially compromising user privacy or safety-critical decision systems.

EM Attacks on Biometric Systems: More directly relevant to physical security, Ni et al. introduced FPLogger, the first attack exploiting electromagnetic side-channel leakage from in-display fingerprint sensors to recover user fingerprints [105]. By capturing EM emanations when users authenticate on smartphones equipped with optical or ultrasonic in-display sensors, the attack reconstructs fingerprint images that can be used to fabricate 3D spoofing artifacts. Evaluation across five commercial smartphones achieved 24% top-1 and 54% top-3 success rates in spoofing attacks, prompting vulnerability acknowledgments from Samsung, Xiaomi, OPPO, and OnePlus. This attack is particularly concerning because in-display fingerprint sensors are increasingly deployed in banking applications and access control systems where fingerprint compromise has long-term consequences (unlike passwords, biometric credentials cannot be changed).

EM Countermeasures: Defending against EM side-channel attacks requires a multi-layered approach with quantifiable attenuation targets. Physical countermeasures include EM shielding (metal enclosures providing 20–40 dB attenuation, conductive coatings achieving 10–25 dB), signal filtering on power and I/O lines, and careful PCB layout with ground planes (10–20 dB attenuation) to minimize EM radiation from sensitive traces [3]. At the circuit level, techniques developed for power analysis resistance (dual-rail logic, current balancing, noise injection) provide partial protection against EM attacks. However, unlike global power consumption, EM emanations can be spatially localized: attackers with near-field probes can target specific circuit regions, potentially bypassing countermeasures applied elsewhere. First-order masking that defeats power analysis may remain vulnerable to localized EM probing that captures intermediate values before they combine with masks. Consequently, EM-resistant designs require either higher masking orders or dedicated shielding of sensitive circuit blocks. For biometric systems vulnerable to FPLogger-style attacks, sensor-level countermeasures (randomized readout patterns, encrypted sensor-to-processor communication, timing jitter) are essential complements to algorithmic protections. The emergence of EM attacks on ML models motivates research into constant-time neural network inference and architecture-hiding accelerator designs.

Error Detection for Fault Attacks: To mitigate fault injections, hardware modules can include error detection and redundancy. For example, a crypto core might compute a result twice (in different manners or with bits reversed) and check consistency, so that an induced fault causing a mismatch can be caught and trigger a reset. Some microcontrollers also lock up or erase secrets if a fault is detected in certain critical registers (to prevent something like a successful glitch from allowing privilege escalation). As an illustration, Maxim Integrated's MAX32510 secure microcontroller includes a Fault Management Unit (FMU) that monitors for abnormal conditions and can actively clear sensitive data if a tamper event is sensed.

Active Tamper Sensors: Building on the above, many secure elements and some MCUs have tamper detection sensors; these can include light sensors (to detect depackaging attempts), voltage monitors (for brown-out/glitch detection), frequency monitors (for clock glitch detection), temperature sensors (extreme temperatures could indicate an attack to alter timing), and even accelerometers (to detect invasive physical movement). If these sensors trip, the device can either wipe critical secrets (one approach known as "rapid zeroization" or fast wipe memory) or at least send an alert. For instance, certain smartcard MCUs will erase their key storage SRAM in microseconds if a tamper pin reports a breach in the device enclosure. Designing these features in a power-efficient way is challenging,

but in 2026, they have become standard in high-security chips and will gradually appear in more commodity IoT security chips (for example, secure battery authentication ICs that zeroize if probed).

Circuit-Level Balancing Techniques: On the analog side, designers use dual-rail logic and current flattening to mitigate power analysis. Dual-rail logic uses pairs of wires that carry complementary signals, so ideally, the power consumption is constant regardless of bit values. While perfect balance is hard to achieve in practice, these techniques increase difficulty for attackers. Some crypto modules also add random pre-charging of buses and memory or automatic insertion of random wait states in computations.

Formal Verification of SCA Countermeasures: A research trend has been to formally verify that an implementation is leakage-free (to certain models). Tools and methodologies for side-channel leakage evaluation (like TVLA: Test Vector Leakage Assessment [106]) are now widely used in development: before deploying, engineers will collect power traces from the device performing cryptography and apply statistical tests to ensure no leakage above the noise level can be detected with reasonable samples. If it can, they iterate on the design. The academic community has been pushing for rigorous proofs that masked implementations are secure under given assumptions (e.g., glitches or certain leakage models), although the challenge is high.

Secure Instruction Set and Co-processors: There are proposals for entire processors that are designed to be side-channel resilient. For example, some research has explored processors that randomize instruction scheduling or insert dummy instructions at the hardware level to foil simple power analysis on secret-dependent branches [107]. Another idea is co-processors that handle all secret operations in a way that is opaque to the main CPU, so the CPU's activity (which might be more easily measured) does not directly involve the secrets.

One specific example of progress is in the intersection of PQC and side-channel security: since lattice-based operations are quite different from AES/RSA, they introduce new leakage patterns. Researchers have implemented masked versions of lattice polynomial multiplications and studied their leakage. By 2024, at least one team demonstrated a hardware accelerator for Kyber (a lattice KEM) with first-order masking, maintaining performance while preventing key extraction via power analysis [108]. Such developments are critical to making PQC algorithms deployable in hostile environments.

It should be noted that side-channel mitigation often comes at the cost of additional power, area, or performance overhead, which is difficult to justify in cost-sensitive IoT markets. Thus, a key challenge is tuning the level of protection to the device's threat model. A $1 disposable sensor might not warrant heavy-duty side-channel protection, whereas a $10 crypto-secure microcontroller for a payment token absolutely does. The concept of *security levels* has emerged, with vendors offering different product tiers: for instance, Silicon Labs classifies parts into "Secure Vault High" and "Secure Vault Mid" tiers [109] where the high tier has more advanced side-channel protections and meets regulatory security certifications, whereas the mid tier might omit some expensive countermeasures.

6.2. Quantitative Countermeasure Evaluation

Recent systematic studies have established quantitative benchmarks for side-channel countermeasure effectiveness on embedded platforms, providing practitioners with concrete guidance for protection level selection. Table 6 summarizes trace requirements for successful key extraction across protection levels, demonstrating the multiplicative security benefit of layered countermeasures.

Table 6. Trace requirements for key extraction under various protection levels.

Protection Level	Algorithm	Traces Required	Reference
Unprotected Implementations			
None	AES-128	20–375	[110]
None	TinyAES (RISC-V)	375	[110]
Protected Implementations			
Affine masking + shuffling	AES-128	>100,000 (no leakage)	[111]
1st-order masking	Kyber-768	>100,000 (hardened)	[112]
Software countermeasures	TinyAES (RISC-V)	>2,000,000 (unsuccessful)	[110]
RDFS protection	TinyAES (RISC-V)	>5,000,000 (withstands)	[110]
Full protection suite	TinyAES (RISC-V)	>20,000,000 (secure)	[110]
Higher-Order Attacks on Masked PQC			
2nd-order masked	Dilithium	700 (HOCPA)	[113]
3rd-order masked	Dilithium	2400 (HOCPA)	[113]
2nd-order masked	Kyber	2200 (HOCPA)	[113]
3rd-order masked	Kyber	14,500 (HOCPA)	[113]

Note: HOCPA = Higher-Order Correlation Power Analysis. Trace requirements increase super-linearly with masking order. Combined countermeasures (masking + shuffling + RDFS) provide the strongest protection but with significant overhead. PQC algorithms remain more vulnerable than AES due to arithmetic operation complexity and Boolean-to-arithmetic conversion bottlenecks [114].

The performance overhead of countermeasures has been systematically characterized. First-order masking overhead varies by algorithm: for AES-128 on Cortex-M4, cycles increase from ∼1500 (unprotected) to 5290–7422 (masked), representing 3.5–5× overhead [115]. For simpler operations or hardware-assisted implementations, overhead can be as low as 30–80%. Higher-order masking scales super-linearly: for Kyber and Saber, second-order masking adds 5.58× overhead while third-order reaches 8.68× [114]. Memory overhead scales more linearly, with each additional masking order adding approximately 6–7 KB of dynamic RAM for PQC schemes. These quantitative trade-offs inform the countermeasure selection framework discussed in the critical analysis below.

Critical Analysis: Threat Model-Matched Countermeasure Selection. The proliferation of side-channel countermeasures from Table 7 creates a combinatorial design space where practitioners must navigate cost–security–performance trade-offs against realistic threat models rather than worst-case scenarios. The overhead ranges (30–80% for first-order masking, 200–400% for dual-rail logic, 5–20% for noise injection) reveal that full protection can triple the device cost and halve performance, which is economically infeasible for mass-market IoT. The critical insight from two decades of smartcard security is that defense-in-depth through layered countermeasures (masking + noise + shielding) provides better cost-effectiveness than single heavyweight solutions, as each layer forces attackers to employ progressively more sophisticated and expensive techniques. However, the PQC transition disrupts established SCA mitigation playbooks: lattice-based operations involve large polynomial multiplications with complex data dependencies that are difficult to mask efficiently, and TVLA testing (the validation gold standard) lacks standardized methodologies for PQC algorithms. The Bos 2021 work demonstrating second-order masked Kyber with 120–250% overhead represents early progress, but practical deployment requires reducing this to sub-50% overhead through dedicated accelerators that implement masking in hardware rather than software. The threat model matching question is particularly acute: a smart door lock faces different adversaries (opportunistic home burglars with $100 equipment) than a payment terminal (organized crime with $50K oscilloscopes and expertise), yet both use similar MCUs. The vendor response of tiered product lines (Secure

Vault High vs. Mid) provides optionality but increases ecosystem complexity and prevents code portability. A more refined approach involves runtime-configurable countermeasures where devices enable expensive protections (higher-order masking, dual-rail logic) only during high-value operations (key generation, signing financial transactions) while using light-weight protections (noise injection, constant-time) for routine cryptography. This selective protection approach can achieve 80% of security benefits at 20% of performance cost, but requires careful analysis to avoid vulnerabilities at mode-switching boundaries. The ML-powered SCA attack trend (deep learning to extract keys from fewer traces) further complicates defense strategies, as traditional masking security arguments assume attackers cannot exploit higher-order statistical dependencies, an assumption increasingly violated by neural networks. Looking forward, the critical research gaps include: (1) automated tools for synthesizing masked PQC implementations with provable security, (2) standardized TVLA-equivalent methodologies for PQC validation, (3) on-chip attack detection mechanisms that trigger defensive responses when abnormal power patterns suggest active SCA attempts, and (4) cross-vendor certification schemes that enable comparable evaluation of countermeasure effectiveness across different hardware platforms.

Table 7. Side-channel attack countermeasures: effectiveness and implementation cost.

Technique	Prot. Level	OH (%)	Impl. Diff.	Best Suited For	Ref.
Algorithmic Countermeasures					
First-Order Masking	Moderate	30–80	Medium	AES, RSA (mature)	[95]
Higher-Order Masking	High	100–300	High	PQC (Kyber, Dilithium)	[108]
Constant-Time Impl.	Moderate	10–30	Medium	All crypto (standard)	[31]
Threshold Crypto	Very High	150–400	Very High	High-security, multi-party	[33]
Hardware Countermeasures					
Noise Injection	Low–Moderate	5–20	Low	IoT devices (optimized)	[93]
Dual-Rail Logic	High	200–400	Very High	ASICs, smartcards	[103]
Random Instr. Injection	Moderate	15–40	Medium	Software-based processors	[107]
Shielding/Filtering	Low	5–15	Low	EM attacks (physical layer)	[3]
Validation Methods					
TVLA Testing	N/A	Offline	Medium	All impl. (Gold standard)	[106]

Note: Overhead (OH) percentages represent typical ranges; actual values vary significantly by algorithm, implementation, and hardware. First-order masking overhead ranges from 30–80% (hardware-assisted or simple operations) to 250–500% (software AES on Cortex-M). PQC algorithms require higher-order masking (2nd–3rd order) due to larger secret space and complex operations. TVLA (Test Vector Leakage Assessment) is the industry standard for validating countermeasure effectiveness. Practical implementations often combine multiple countermeasures for defense-in-depth.

7. Trusted Execution Environments (TEEs) in Embedded Processors

A Trusted Execution Environment (TEE) provides an isolated area of execution in a processor, such that code and data within the TEE are protected in terms of confidentiality and integrity, even if the rest of the system (the "rich OS" or normal world) is compromised. TEEs have been commonplace in mobile and desktop processors (e.g., ARM TrustZone in smartphone SoCs, Intel SGX in CPUs), and in the 2020s, they increasingly appeared in microcontroller-class and IoT processors as well [13].

The flagship TEE for embedding is Arm TrustZone, which comes in two flavors: TrustZone for Cortex-A (used in application processors) and TrustZone for Cortex-M (introduced with the Armv8-M architecture). TrustZone on Cortex-M (found in cores like Cortex-M23, M33, M35P) offers a hardware-enforced separation between Secure and Non-Secure worlds within the MCU. In essence, the memory space is partitioned: certain flash, RAM, and

peripherals are designated as Secure (accessible only by code running in the Secure world), while the rest are Non-Secure. The CPU can switch states via controlled gateways (Secure Function Calls), but Non-Secure code cannot access Secure resources except through predefined veneers. This allows a small trusted firmware (cryptography services, key storage, secure boot code, etc.) to run isolated from the main application logic and potentially from an RTOS or rich OS running in the Non-Secure domain. In 2026, many MCU vendors (ST, NXP, Renesas, Silicon Labs, etc.) have TrustZone-enabled microcontrollers on the market. Silicon Labs, for example, integrated TrustZone-based Secure Vault technology to hide wireless stack keys and sensitive data, meeting PSA Level 2 certification [116]. Real-time operating systems like FreeRTOS and mbedOS have added support for TrustZone-M, making it easier for developers to use the separation for their IoT applications.

In parallel, other TEEs exist or are emerging. On higher-end embedded Linux platforms (like Raspberry Pi or NVIDIA Jetson), TrustZone for Cortex-A is available. While early Raspberry Pi models did not expose TrustZone to users, newer 64-bit Pi systems theoretically could use it. In practice, adoption on hobbyist platforms is limited, but industrial use of TrustZone-A (with something like OP-TEE, an open-source TEE OS) is seen in some gateways and controllers. NVIDIA Jetson (Tegra SOCs) include a Secure OS and TrustZone-based boot chain since they are derived from mobile phone SOC designs. These allow, for example, running DRM or AI models in a TEE on the Jetson.

For RISC-V processors, the open nature has spurred several TEE research projects [117,118]. Since RISC-V did not initially have a standard equivalent to TrustZone, solutions like MultiZone by Hex Five [119] (a light-weight TEE using RISC-V PMP), Sanctum [120] (MIT's research TEE based on RISC-V enclave concepts), and Keystone [121] (Berkeley's open-source enclave framework) were proposed. Keystone, for instance, uses RISC-V's Physical Memory Protection and a minimal security monitor to create enclave regions somewhat akin to Intel SGX enclaves. In 2022–2023, RISC-V's Privileged Spec added the concept of "World" switching (Hypervisor extension) which can be used to implement TEEs in hardware going forward. While not mainstream yet, it's plausible that by the late 2020s, we will see RISC-V MCUs with a form of TEEs, especially as RISC-V is adopted in security-critical embedded roles. Figure 6 illustrates the deployment of TEEs across diverse platforms, highlighting their integration within different SoCs and CPUs.

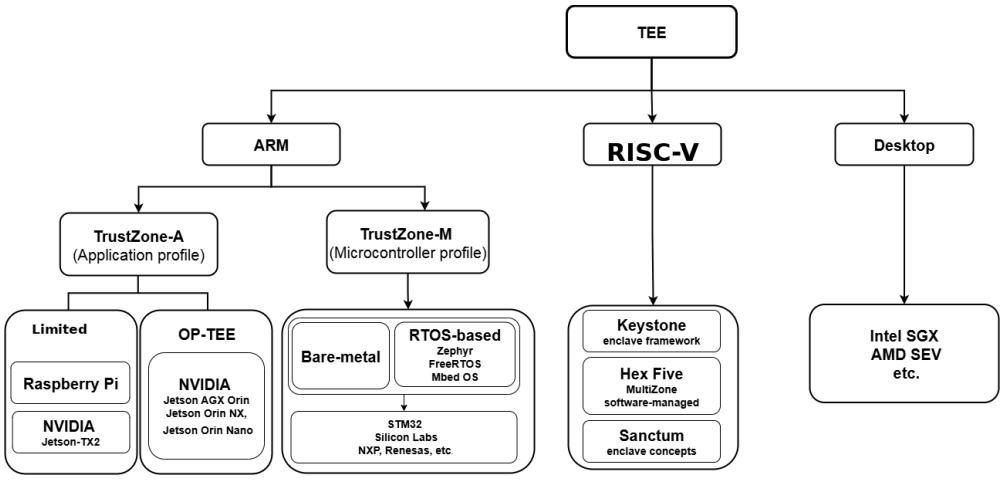


Figure 6. Overview of Trusted Execution Environment (TEE) implementations across various platforms (ARM TrustZone and RISC-V).

An alternative approach in some embedded systems is a separate security core. For example, certain automotive or IoT SoCs have a small dedicated core (maybe a Cortex-M0 or a custom state machine) that runs security functions, isolated from the main application

processor. This core can act like a TEE by virtue of being physically separate, e.g., handling crypto, secure boot verification, and secure monitoring. Microcontrollers with integrated security subsystems (like an "Arm CryptoCell" or similar secure enclave IP) fall into this category.

7.1. Quantitative Performance Overhead Analysis

Recent benchmarking studies provide concrete overhead measurements across TEE implementations, enabling informed architectural decisions. Table 8 summarizes performance overhead data from peer-reviewed evaluations of ARM TrustZone and RISC-V enclave implementations.

ARM TrustZone-M Overhead: TrustZone-M achieves minimal switching overhead due to its hardware-based design without a secure monitor mode. The secure gateway (SG) instruction executes in approximately three clock cycles, though full secure function calls include additional context save/restore overhead depending on the number of registers preserved [122–124]. Recent work has also identified vulnerabilities such as bus fault attacks that can compromise TrustZone isolation [125]. This enables real-time embedded applications to invoke secure services without significant latency penalties. The Trusted Firmware-M (TF-M) reference implementation [126] provides three scalable profiles: the Medium profile, targeting cloud-connected IoT devices requiring asymmetric cryptography for TLS/DTLS, consumes approximately 60 KB RAM and 99 KB Flash, a practical footprint for modern Cortex-M33/M55 devices.

RISC-V Enclave Overhead Variance: RISC-V TEE implementations exhibit significant performance variance depending on architectural choices. Keystone [121] demonstrates ±0.7% CPU overhead for compute-bound workloads (excluding enclave creation/destruction), though I/O-intensive operations suffer 36–41% throughput loss due to encryption and integrity verification. Penglai [127] achieves only 5–6% overhead for memory-intensive Redis workloads through its Guarded Page Table (GPT) design, while CPU-intensive benchmarks show negligible degradation. SPEAR-V [128] achieves sub-1% overall overhead (<0.95% read bandwidth, <0.09% write bandwidth) on the CVA6 RISC-V core, demonstrating that light-weight enclave designs can approach near-zero performance cost.

Table 8. Quantitative performance overhead of TEE implementations for embedded systems.

Platform	CPU Overhead	I/O Overhead	Memory Footprint	TCB Size	Ref.
ARM TrustZone Implementations					
TrustZone-M (Cortex-M33)	Negligible	N/A	60 KB RAM, 99 KB Flash (TF-M Medium)	Varies by secure OS	[126]
TrustZone-A (Cortex-A53)	3–10%	Minimal	N/A	OP-TEE: ∼100 KLoC	[129]
RISC-V Enclave Implementations					
Keystone	±0.7%	36–41%	N/A	15 KLoC (SM: 1.6K)	[121]
Penglai	Negligible	5–6% (Redis)	512 GB secure mem.	N/A	[127]
SPEAR-V	<1%	<1%	N/A	N/A	[128]
CURE	15.33% (geomean)	N/A	N/A	Few KLoC	[130]
TIMBER-V	N/A	N/A	+6.25% (tags)	N/A	[131]

Note: CPU overhead measured on compute-bound benchmarks (CoreMark, RV8); I/O overhead measured on storage/network workloads (IOZone, Redis). TCB = Trusted Computing Base. Keystone's 15 KLoC TCB includes security monitor (1.6K), crypto library (4K), and runtime (4.7K). TF-M Medium profile targets IoT devices requiring asymmetric crypto for TLS/DTLS. SPEAR-V achieves lowest overhead through light-weight enclave design on CVA6 RISC-V.

7.2. Trusted Computing Base (TCB) Size Analysis

The size of the Trusted Computing Base, the totality of code and hardware that must be trusted for security guarantees, is a critical metric for TEE security assurance, as larger TCBs present greater attack surface and complicate formal verification. Keystone's modular design achieves a 15 KLoC TCB (security monitor: 1600 LoC, crypto library: 4 KLoC, trap handling/boot: 4.7 KLoC) [121], enabling practical formal verification efforts. CURE [130] explicitly targets "few KLoC" to maintain verifiability. By contrast, full-featured secure operating systems like OP-TEE approach ∼100 KLoC, comparable to Intel SGX's reference TCB—sizes that preclude complete formal verification with current techniques. This TCB size variance reflects a fundamental design tension: minimal TCBs (Keystone, SPEAR-V) sacrifice features for verifiability, while comprehensive implementations (OP-TEE, TF-M Large profile) provide richer security services at the cost of increased trust assumptions.

The widespread adoption of TEEs in embedded systems has been driven by use cases such as:

1. Protecting cryptographic keys and operations (so the main firmware never directly handles plaintext keys; it requests operations from the secure world).
2. Isolating firmware update mechanisms (the secure world can implement a check on any new firmware binary using a key, without trusting the main application to do it correctly).
3. Running third-party or certified code (like a certified IoT communication stack or a digital payment app) in a protected environment to prevent a compromised main application from tampering with it.

One real-world example is in smart meters or protection relays (critical infrastructure devices): these often use an MCU with TrustZone-M to separate the metrology and billing code (which must be secure) from the general management code. The secure part might enforce that meter readings can be signed with a secure key that the main application can never access directly.

Despite their benefits, TEEs on microcontrollers introduce some complexity. Developers need to partition their code and manage secure gateways. Tooling and debugging become more complicated when part of the code is inaccessible. TEEs also typically do not protect against physical attacks (an attacker with hardware access could still dump memory unless combined with encryption or other hardware security). Thus, TEEs are one piece of the puzzle, often working in concert with the other techniques discussed (PUFs for key storage, secure boot to establish initial trust, side-channel protections for the secure code, etc.).

Critical Analysis: TEE Developer Friction and Architectural Integration Complexity. The widespread availability of TEE-enabled MCUs (TrustZone-M in Cortex-M33/M35P, RISC-V enclave frameworks) masks a significant adoption gap driven by developer friction and architectural complexity. Table 9 shows that performance overhead ranges from <0.01% (MultiZone hardware-only protection) to 10–18% (software-managed RISC-V enclaves), but these metrics capture only runtime cost, not the considerable development overhead of secure/non-secure code partitioning, gateway interface design, and cross-world debugging. Industrial experience reveals that TEE-enabling an existing embedded application requires 30–50% code restructuring and 2–3× longer development cycles due to complexity in managing secure function calls, preventing data leakage across boundaries, and ensuring correct privilege separation. The tooling gap compounds this: while TrustZone-M has mature vendor support (ARM MDK, IAR EWARM with TrustZone plugins), RISC-V TEE ecosystems remain fragmented with incomplete debugger support for enclave introspection. The fundamental architectural question concerns granularity trade-offs: coarse-grained separation (entire crypto library in secure world) provides simpler interfaces but larger

TCB (Trusted Computing Base), while fine-grained isolation (only key material secured, operations in normal world) minimizes TCB but requires dozens of secure gateways with associated performance and attack surface costs. The smart meter use case illustrates optimal granularity: metrology code (secure world) generates signed measurements, while communication stack (normal world) transmits them, creating a simple two-component architecture with a single gateway interface. However, more complex applications (e.g., industrial controllers with multiple secure domains) require hierarchical TEE architectures not well-supported by current hardware. The integration complexity extends to compositional security: TEEs provide memory isolation but not physical attack protection, requiring combination with PUF key storage (Section 4), side-channel-resistant crypto (Section 6), and secure boot (Section 5), creating complex dependencies where TEE secure world code must be SCA-hardened and the boot chain must establish TEE isolation before normal world executes. The RISC-V TEE landscape presents both opportunity and challenge: Keystone's 8–20% overhead and open-source flexibility enable custom security domains, but lack of standardization results in vendor-specific implementations with zero portability (code written for Keystone won't run on MultiZone or Penglai). The upcoming RISC-V WorldGuard specification may address this, but standardization timelines suggest 2026–2027 before widespread adoption. Looking forward, three developments would substantially reduce TEE adoption friction: (1) automated code partitioning tools that analyze applications and suggest optimal secure/normal world split with minimal developer input, (2) standardized TEE APIs (analogous to GlobalPlatform TEE for mobile) enabling portable secure services across ARM and RISC-V platforms, and (3) compositional security frameworks that provide pre-verified integration patterns combining TEE + PUF + secure boot + SCA countermeasures, reducing per-project security analysis burden. The maturation path requires transforming TEEs from expert-level features requiring deep security knowledge to commodity capabilities accessible via high-level APIs and automated tooling.

Table 9. Comparison of Trusted Execution Environment (TEE) implementations for embedded systems.

Platform	Isolation	Perf.	Certification	Target	Availability	Ref.
			Arm TrustZone Family			
TrustZone-M (Cortex-M33)	Memory regions	5–15%	PSA L2–L3	MCU	Commercial	[123]
TrustZone-A (Cortex-A7/A53)	Exception levels	3–10%	GP TEE, PSA L1–L2	App. Proc.	Commercial	[129]
			RISC-V TEE Implementations			
Keystone	HW enclaves	8–20%	Research	General	Open-source	[121]
MultiZone	Mem. protection	<0.01%	Commercial	MCU	Safety-certified	[119]
Penglai	HW enclaves	5–6%	Research	General	RISE project	[127]
			Hardware Security Roots of Trust			
OpenTitan (RISC-V)	Dedicated chip	Minimal	Open-source	RoT	Prod. 2025	[132]
Secure Elements	External chip	Minimal	CC EAL5+	All	Commercial	[74]

Note: Performance overhead measured as impact on secure operations vs. non-secure baseline. Overhead values represent typical ranges from the literature; exact values vary by workload and implementation. PSA = Platform Security Architecture (ARM); GP TEE = GlobalPlatform TEE; CC EAL = Common Criteria Evaluation Assurance Level. TrustZone-M dominates MCU market with 5–15% overhead and PSA L2–L3 certification. MultiZone achieves low overhead through efficient use of RISC-V Physical Memory Protection (PMP) hardware. RISC-V TEEs are emerging with competitive overhead and open-source flexibility. OpenTitan represents the first open-source silicon RoT reaching production (Feb 2025), with Chromebook deployment planned for late 2025. Secure elements provide the highest assurance (CC EAL5+, FIPS 140-3 L3) with minimal performance impact as dedicated co-processors.

8. Challenges and Future Directions Under Constraints

While significant progress has been made in embedding strong security features into processors, major challenges remain in deploying these technologies widely across the IoT ecosystem. We highlight some cross-cutting issues and potential solutions as of 2026:

Resource Constraints and Trade-offs: The foremost challenge is that security often competes with other resource demands on small devices. Every extra gate for a crypto accelerator, every additional check in software, and every milliwatt drawn by a tamper sensor is scrutinized by product designers. Ultra-low-cost devices may still skimp on security due to cost or power budgets. A key trend to address this is *scalability*: offering security building blocks that can scale down (with graceful degradation of security). For instance, a manufacturer might provide a microcontroller family with varying security levels (as Silicon Labs does with its Secure Vault tiers), allowing developers to choose a part that meets both their security needs and budget. In the future, more granular configurability could allow turning off or on certain countermeasures based on deployment scenarios (for example, enabling side-channel protections only on devices that will be physically exposed).

Interoperability and Standards: As hardware security features proliferate, there is a need for standard interfaces and protocols so that software can make use of them in a uniform way. Standards like *Trusted Firmware-M* [126] (for ARM TrustZone-M) and PSA APIs are steps in this direction, providing a reference implementation for secure services (crypto, attestation, secure storage) that can run in a TEE on various platforms. On the PQC front, NIST's finalization of algorithms will spur standards for integrating those algorithms into protocols like TLS, MQTT for IoT, etc. Ensuring that small devices can comply with these emerging standards (e.g., supporting the larger certificates and messages that PQC entails) will require collective efforts in optimization and possibly protocol redesign (to avoid excessive fragmentation or handshake steps that increase latency for constrained nodes).

Secure Lifecycle Management: Deploying a secure device is not a one-and-done effort; it must be maintained over its life. This means secure provisioning, updates, and eventually decommissioning. Many IoT products historically lacked a mechanism for firmware updates, which is unacceptable now from a security perspective. Incorporating a reliable, authenticated update mechanism (often using secure boot and RoT keys) is now seen as essential. This is challenging when devices have limited connectivity or power, but techniques like delta updates and energy-efficient protocols are being used. Industry guidelines are also increasingly mandating a "cryptographic agility": the ability to swap out algorithms if needed (e.g., in case of a vulnerability or future quantum-breaking of algorithms). Designing agility into constrained devices is hard (since including two algorithms might exceed space), but one strategy is to rely on an updatable secure element or co-processor that can be replaced or updated independently of the main MCU.

Usability vs. Security: A more practical challenge is making all these advanced security features usable for embedded developers. There is a knowledge gap: many embedded engineers are not cryptography or hardware security experts. If TEE configuration or secure boot setup proves overly complex, developers may disable or misconfigure these features (as observed in TrustZone deployments where excessive peripherals were configured as non-secure for convenience). Thus, the future likely holds improved tooling such as configuration interfaces that auto-generate secure/non-secure partitions and SDKs that abstract PUF key management, enabling developers to invoke high-level APIs without understanding the implementation details. Academic work on formal verification and automated code generation for secure enclaves may also yield tools that ensure secure code is written correctly and efficiently.

Emerging Threats: Looking forward, new types of hardware attacks are on the horizon. For example, machine learning could be used not just defensively (to detect anomalies) but offensively (to automatically find patterns in side-channel data, or to craft subtle fault injection parameters). Attacks like Spectre [133] and Meltdown [134] (transient execution attacks) taught us that even microarchitectural optimizations can be security pitfalls; embedded, simpler pipelines mostly avoid these, but as IoT processors become more powerful, they might grow similar issues. Supply chain attacks (inserting hardware Trojans during fabrication, or swapping components) also remain a concern. The development of open-source silicon (like OpenTitan) is partially to counter that by enabling the inspection of designs. Quantum computing itself could eventually threaten more than just cryptography; quantum sensors might help side-channel attacks in ways we can't yet predict.

However, these challenges come with opportunities: the convergence of safety and security in domains like automotive and healthcare is leading to unified design approaches. An example is AUTOSAR AP and adaptive platforms in automotive systems that integrate hardware security modules tightly with safety monitors. The IoT industry is also coalescing around security certifications (PSA Certified, IEC 62443, FIPS, etc.) which helps create market incentives for better security.

In summary, to meet real-world constraints, the approach is to build *efficient, composable, and user-friendly* security solutions: Efficient, so that the overhead is minimized (e.g., using hardware acceleration, optimizing memory of crypto, etc.). Composable, so that multiple protections can work together without causing conflicts or excessive cost. User-friendly, so that adoption becomes widespread, and even small companies can correctly implement security.

The advances from 2020 to 2026 form a solid foundation: we now have the raw capabilities (PQC, PUF, TEE, etc.) proven viable in embedded contexts. The next phase is refining and integrating them naturally into the design flow of every embedded system. Achieving that will lead to an ecosystem of "secure by default" devices resilient against the evolving threat landscape.

9. Discussion

In 2026, hardware-based security for embedded processors has transformed from a niche topic to a baseline requirement for IoT devices. This paper has reviewed the major advancements in securing microcontrollers and small embedded systems at the hardware level: from incorporating post-quantum cryptographic agility and Physical Unclonable Function key storage to establishing hardware Roots of Trust with secure boot, mitigating side-channel leakages, and isolating execution environments for critical code. These techniques directly address the unique challenges of embedded use cases, where constraints are tight but security cannot be an afterthought.

The developments of the last five years show a clear trajectory: security features are becoming more integrated, more standardized, and more cognizant of real-world deployment constraints. Importantly, there is a growing recognition in the industry that investing in hardware security up front prevents costly vulnerabilities and fixes later. Initiatives like PSA certification and NIST's IoT cyber-security guidelines [135] are pushing the industry toward common security baselines even for low-cost devices.

Yet, our review also makes it evident that there is no single solution; each mechanism has limitations and must be carefully implemented and configured. The combination of multiple hardware security building blocks, layered defensively, is the way forward. For example, a secure boot based on a PUF-derived key, running on a TEE, with the

cryptography protected by side-channel countermeasures, brings together the strands we discussed into a secure whole.

Figure 7 illustrates this four-layer security architecture, showing how PQC, PUF, Root of Trust, side-channel countermeasures, and TEE primitives integrate to provide defense-in-depth. Key material flows from the hardware layer (PUF) through the Root of Trust to secure storage, while cryptographic operations remain protected by SCA countermeasures and isolated within the TEE. This composition ensures that compromising any single layer does not defeat the overall security posture.

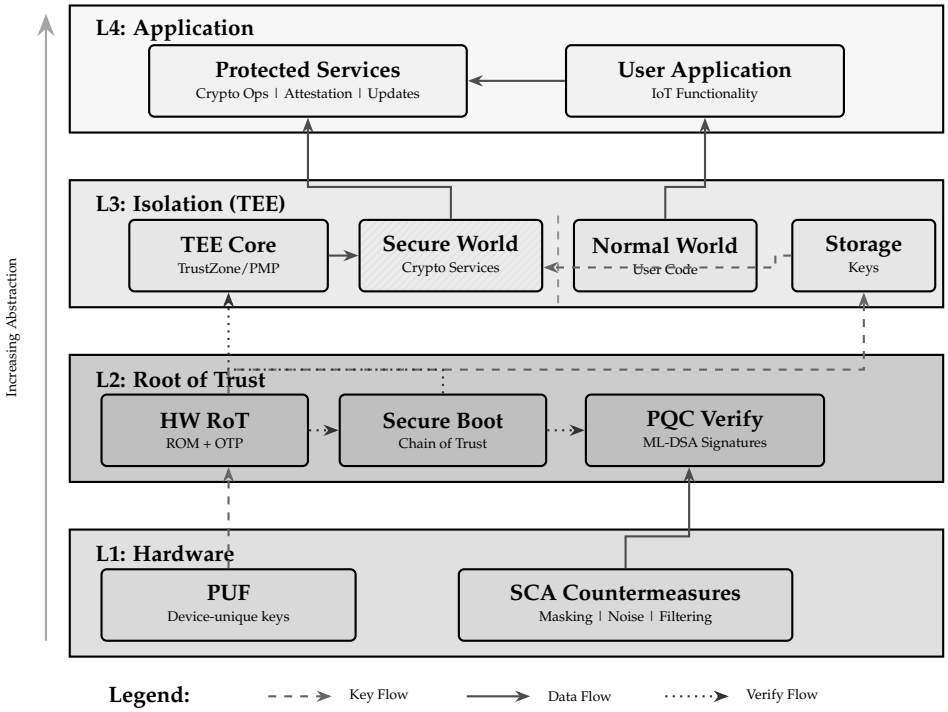


Figure 7. Unified four-layer security architecture integrating PQC, PUF, Root of Trust, side-channel protections, and TEE primitives. Arrows indicate key material flow (dashed), data/control flow (solid), and verification flow (dotted).

9.1. Practitioner Recommendations

Based on our analysis of the 2020–2026 developments across all five security domains, we provide concrete guidance for practitioners implementing hardware security in embedded systems. These recommendations synthesize the technical findings from Sections 3–7 into actionable decision frameworks.

9.1.1. Reference Integration Architecture

Figure 8 presents a reference secure boot flow demonstrating how the security primitives discussed in this review compose into a cohesive architecture. This flow represents current best practice for mid-to-high security embedded devices (Tier 2–3 in Table 10). The key integration points are: (1) PUF-derived keys eliminate static key storage vulnerabilities; (2) ROM-based verification ensures an immutable trust anchor; (3) PQC signatures provide quantum-resistant firmware authentication; and (4) TEE isolation protects runtime secrets from normal world compromise. Implementations following this pattern include ARM TF-M with SRAM PUF integration, and mcuboot with hardware-backed key storage.

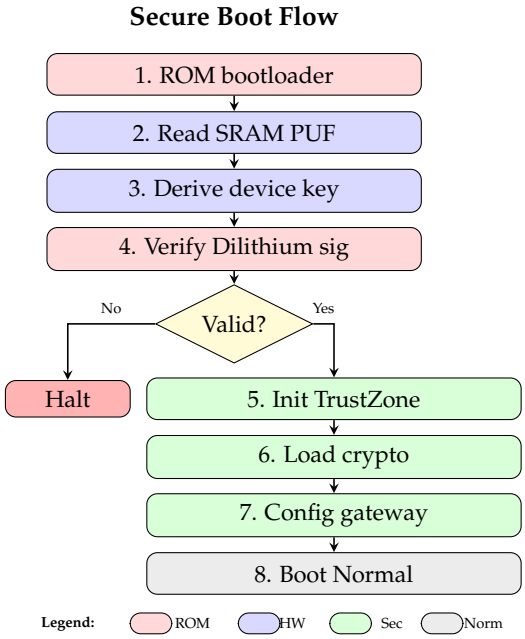


Figure 8. Secure boot integration: PUF key derivation, PQC verification, TEE initialization.

Table 10. Minimum viable security recommendations by device class.

Device Class	BOM Range	Minimum Security	Recommended Security	Certification	Example Use Cases
Ultra-Low-Cost Sensors	\$0.50–\$2	SW AES, unique ID, ROP	+ Secure boot, SRAM-PUF	PSA L1	Environmental sensors, asset tags
Consumer IoT	\$2–\$10	Secure boot, HW crypto, debug lock	+ TrustZone-M, PUF keys	PSA L1–L2	Smart home, wearables
Industrial IoT	\$10–\$50	TEE, secure boot, HW RNG, anti-rollback	+ SCA countermeasures, Ext. SE	PSA L2, IEC 62443	PLCs, gateways, medical
Critical Infrastructure	\$50–\$200+	Full TEE, certified crypto, PUF+SE, SCA/FIA protection	+ Formal verification, red team	PSA L3, FIPS 140-3	Grid, automotive, payment

Note: BOM = Bill of Materials cost increment for security features. ROP = Read-Out Protection. SE = Secure element. SCA = Side-channel attack countermeasures. FIA = Fault injection attack countermeasures. Certification targets indicate appropriate assurance levels for each tier.

9.1.2. Minimum Viable Security by Device Class

Table 10 provides tier-specific guidance for selecting security features based on cost constraints and threat profiles. The key insight is that security investment should scale with asset value and threat exposure rather than following a one-size-fits-all approach.

9.1.3. MCU Platform Comparison

Table 11 compares security features across popular MCU platforms, providing practitioners with a quick reference for platform selection. Feature availability changes rapidly; consult vendor documentation for current specifications.

Table 11. Security feature comparison of popular MCU platforms (as of February 2026).

Platform	TrustZone-M	Hardware Crypto	PUF	Secure Boot	PSA Level	BOM ($)
STM32L5 (ST)	Yes	AES, SHA, PKA, TRNG	No (ext.)	Yes	L2	3–8
LPC55S69 (NXP)	Yes	AES, SHA, RSA, ECC	SRAM	Yes	L2	4–10
EFM32PG22 (SiLabs)	Yes	AES, SHA, TRNG	No	Yes	L2 (Vault)	5–12
nRF5340 (Nordic)	Yes	AES, SHA, ECC	No	Yes	L1	4–9
RP2350 (RPi)	Yes	SHA, TRNG	OTP	Yes	L1	1–2

Note: PKA = Public Key Accelerator. TRNG = True Random Number Generator. Ext. = requires external component. BOM reflects typical unit pricing at moderate volumes (1K–10K). The LPC55S69 is notable for integrated SRAM PUF; the RP2350 offers exceptional value with TrustZone-M at sub-$2 pricing.

9.1.4. Algorithm Selection Guidance

The NIST post-quantum cryptography standardization (FIPS 203, 204, 205) provides practitioners with clear algorithm choices, but embedded constraints require careful selection:

Key Encapsulation (FIPS 203 ML-KEM/Kyber): Kyber512 (NIST Security Level 1) is recommended for most embedded applications. At 1632-byte public keys and 536K cycles for encapsulation on Cortex-M4, it offers the best balance of security and resource efficiency. Kyber768 (Level 3) is appropriate when regulatory requirements mandate higher security levels or device lifetime exceeds 15 years.

Digital Signatures (FIPS 204 ML-DSA/Dilithium): Dilithium2 (Level 2) is recommended for firmware authentication and secure boot. With 2528-byte public keys and 1.57M verification cycles, it provides practical performance for boot-time operations. Dilithium3 (Level 3) suits industrial IoT where signature verification frequency is moderate.

Hybrid Deployment Strategy: For devices with 5+ year lifetimes, we recommend hybrid signatures combining ECDSA-P256 with Dilithium2. This approach provides immediate classical security while establishing quantum resistance, with graceful degradation if either algorithm is compromised.

9.1.5. Certification Pathway Recommendations

Security certification provides external validation and is increasingly required by regulations (EU Cyber Resilience Act, US IoT Cybersecurity Improvement Act). We recommend tiered certification strategies:

Consumer IoT (PSA Certified Level 1): Self-assessment questionnaire with security architecture documentation. Minimal cost ($1K–$5K) and 2–4 week timeline. Demonstrates baseline security hygiene; sufficient for smart home devices, wearables, and non-critical sensors.

Industrial IoT (PSA Certified Level 2): Independent lab evaluation with penetration testing. Moderate cost ($15K–$50K) and 3–6 month timeline. Appropriate for connected industrial equipment, medical devices (non-implantable), building automation.

Critical Infrastructure (PSA Certified Level 3 / FIPS 140-3): Comprehensive evaluation including side-channel and fault injection resistance testing. Significant cost ($100K–$500K) and 12–18 month timeline. Required for energy grid components, automotive safety systems, payment terminals, defense applications.

9.1.6. Cost–Security Trade-Off Framework

Practitioners face constrained optimization between security investment and product economics. We propose a decision framework based on four factors: (1) *Asset Value at Risk*: low (<$100 aggregate data value) justifies basic security; high (>$10K or safety-

critical) warrants maximum practical security. (2) *Physical Access Threat*: unattended deployment with adversary physical access prioritizes PUF key storage, tamper detection, and SCA countermeasures. (3) *Device Lifetime vs. Threat Evolution*: short lifetime (<3 years) allows classical cryptography; long lifetime (>10 years) requires PQC-only or crypto-agile architecture. (4) *Regulatory and Certification Requirements*: domains such as automotive (ISO/SAE 21434), medical devices (FDA premarket guidance), and payment terminals (PCI PTS) impose mandatory security baselines that constrain the design space independently of threat modeling.

Practical BOM Impact: Moving from baseline MCU security to full hardware security typically adds: software-only enhancements (+$0 BOM, +10–20% code size); MCU with integrated security (+$0.50–$2 over baseline for TrustZone-M, crypto accelerators, PUF); external secure element (+$0.50–$2 for ATECC608-class; +$3–$8 for TPM 2.0); and certified security MCU (+$5–$15 for PSA L3 capable parts). For most consumer and industrial IoT applications, the $1–$3 incremental BOM for integrated security features provides optimal cost–security balance.

9.1.7. Selection Decision Flowcharts

Figure 9 provides entry-point decision guidance for practitioners selecting security primitives. These flowcharts encode the key trade-offs discussed throughout this review: (a) PUF selection balances cost against ML attack resistance and physical access threats; (b) TEE selection is primarily determined by platform architecture; (c) PQC migration timing depends on device lifetime and data sensitivity. These flowcharts represent simplified heuristics; practitioners should consult the detailed analysis in each domain section for context-specific decisions.

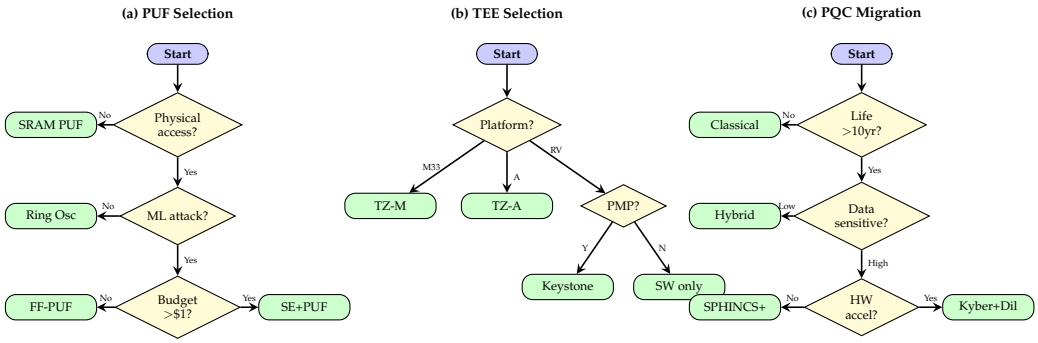


Figure 9. Decision flowcharts: (**a**) PUF selection by threat model; (**b**) TEE selection by platform; (**c**) PQC migration timing.

9.2. Synthesis and Future Directions

Future work will likely explore deeper integration of these features (e.g., PUFs feeding keys directly into cryptographic engines without ever exposing them to software, or PQC accelerators that are automatically side-channel protected by design). Research is also needed in areas like formally verifying the security of hardware designs and improving the ability to patch hardware vulnerabilities in the field.

The layered architecture depicted in Figure 4 represents the emerging best practice: hardware security is not a collection of independent features but an integrated system where each layer reinforces the others. PUF-derived keys protect the Root of Trust, which establishes the TEE, which isolates cryptographic operations that are protected by SCA countermeasures. Breaking this chain requires defeating multiple independent security mechanisms, a considerably harder proposition than attacking any single feature.

9.3. Limitations

This review has several limitations that readers should consider. First, our focus on Western standards (NIST, PSA Certified, GlobalPlatform) may underrepresent developments in Chinese IoT security standards (GM/T series) and emerging market ecosystems. Second, the temporal scope (2020–2026) excludes foundational pre-2020 works except where necessary for context; readers interested in historical evolution should consult earlier surveys [4,123]. Third, proprietary vendor implementations with limited public documentation may be underrepresented compared to open-source alternatives (OpenTitan, Keystone, TF-M). Fourth, our practitioner recommendations reflect current best practices that will evolve as quantum computing threats materialize and new attack techniques emerge. Finally, while we cover RISC-V and ARM architectures comprehensively, other embedded architectures (e.g., Xtensa, ARC) receive less attention due to their smaller market share in security-focused applications.

9.4. Conclusions

The period 2020–2026 has been transformative for embedded hardware security: many concepts matured from academic prototypes to commercial products and standards. The road ahead will involve refining these advances and ensuring they can be applied across the wide range of embedded devices. With increasingly capable attackers, continued innovation and vigilance are required. However, the progress to date provides optimism that even constrained devices can achieve strong security assurances, enabling a safer IoT-enabled world for the years to come.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/electronics15051135/s1, Table S1: PRISMA 2020 Checklist for Systematic Reviews.

Author Contributions: Conceptualization, A.K. and A.W.S.; methodology, A.K.; formal analysis, A.K. and A.W.S.; investigation, A.K., A.W.S. and M.I.; resources, M.I.; writing—original draft preparation, A.K.; writing—review and editing, A.W.S. and M.I.; visualization, A.K.; supervision, M.I.; project administration, M.I. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Science Foundation, under the Grant 2501916.

Data Availability Statement: Not applicable. This is a review article and does not report original experimental data.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AES	Advanced Encryption Standard
BER	Bit Error Rate
CC	Common Criteria
CRP	Challenge–Response Pair
DPA	Differential Power Analysis
EAL	Evaluation Assurance Level
ECC	Elliptic-Curve Cryptography
ECDSA	Elliptic-Curve Digital Signature Algorithm
FMU	Fault Management Unit
FIPS	Federal Information Processing Standards
hRoT	Hardware Root of Trust
IoT	Internet of Things

KEM	Key Encapsulation Mechanism
MCU	Microcontroller Unit
ML	Machine Learning
NTT	Number Theoretic Transform
OTP	One-Time Programmable
PMP	Physical Memory Protection
PQC	Post-Quantum Cryptography
PSA	Platform Security Architecture
PUF	Physical Unclonable Function
RoT	Root of Trust
RSA	Rivest–Shamir–Adleman
SCA	Side-Channel Attack
SoC	System on Chip
SPA	Simple Power Analysis
SRAM	Static Random-Access Memory
TCB	Trusted Computing Base
TEE	Trusted Execution Environment
TPM	Trusted Platform Module
TVLA	Test Vector Leakage Assessment

References

1. IoT Analytics. Number of Connected IoT Devices Growing 14% to 21.1 Billion, 2024. Available online: https://iot-analytics.com/number-of-connected-iot-devices/ (accessed on 15 January 2026).
2. Wang, H.; Forte, D.; Tehranipoor, M.M.; Shi, Q. Probing Attacks on Integrated Circuits: Challenges and Research Opportunities. *IEEE Des. Test* **2017**, *34*, 63–71. [CrossRef]
3. Li, Y.; Chen, M.; Wang, J. Introduction to side-channel attacks and fault attacks. In Proceedings of the 2016 Asia-Pacific International Symposium on Electromagnetic Compatibility (APEMC), Shenzhen, China, 17–21 May 2016; Volume 1, pp. 573–575. [CrossRef]
4. Randolph, M.; Diehl, W. Power Side-Channel Attack Analysis: A Review of 20 Years of Study for the Layman. *Cryptography* **2020**, *4*, 15. [CrossRef]
5. Bernstein, D.J.; Buchmann, J.; Dahmen, E. *Post-Quantum Cryptography*; Springer: Berlin/Heidelberg, Germany, 2009. [CrossRef]
6. Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ* **2021**, *372*, n71. [CrossRef]
7. Akter, S.; Khalil, K.; Bayoumi, M. A Survey on Hardware Security: Current Trends and Challenges. *IEEE Access* **2023**, *11*, 77543–77565. [CrossRef]
8. Zhou, X.; Wang, P.; Zhou, L.; Xun, P.; Lu, K. A Survey of the Security Analysis of Embedded Devices. *Sensors* **2023**, *23*, 9221. [CrossRef]
9. Fei, W.; Ohno, H.; Sampalli, S. A Systematic Review of IoT Security: Research Potential, Challenges, and Future Directions. *ACM Comput. Surv.* **2023**, *56*, 111. [CrossRef]
10. Liu, T.; Ramachandran, G.; Jurdak, R. Post-Quantum Cryptography for Internet of Things: A Survey on Performance and Optimization. *arXiv* **2024**, arXiv:2401.17538v1. [CrossRef]
11. Pursche, M.; Puch, N.; Peters, S.N.; Heinl, M.P. *SoK: The Engineer's Guide to Post-Quantum Cryptography for Embedded Devices*; Cryptology ePrint Archive, Paper 2024/1345; International Association for Cryptologic Research (IACR): Bellevue, WA, USA, 2024.
12. Karakaya, A.; Ulu, A. A Survey on Post-Quantum Based Approaches for Edge Computing Security. *WIREs Comput. Stat.* **2024**, *16*, e1644. [CrossRef]
13. Shepherd, C.; Markantonakis, K. *Trusted Execution Environments*; Springer: Berlin/Heidelberg, Germany, 2024. [CrossRef]
14. National Institute of Standards and Technology. NIST Announces First Four Quantum-Resistant Cryptographic Algorithms, 2022. Available online: https://www.nist.gov/news-events/news/2022/07/nist-announces-first-four-quantum-resistant-cryptographic-algorithms (accessed on 15 January 2026).
15. National Institute of Standards and Technology. *Module-Lattice-Based Key-Encapsulation Mechanism Standard (FIPS 203)*; Technical Report FIPS 203; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2024. Available online: https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.203.pdf (accessed on 22 August 2025).

16. National Institute of Standards and Technology. *Module-Lattice-Based Digital Signature Standard (FIPS 204)*; Technical Report FIPS 204; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2024. Available online: https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.204.pdf (accessed on 22 August 2025).
17. Rivest, R.L.; Shamir, A.; Adleman, L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
18. Koblitz, N. Elliptic Curve Cryptosystems. *Math. Comput.* **1987**, *48*, 203–209. [CrossRef]
19. Ajtai, M. Generating Hard Instances of Lattice Problems. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing (STOC '96), Philadelphia, PA, USA, 22–24 May 1996; pp. 99–108. [CrossRef]
20. Merkle, R.C. A Certified Digital Signature. In *Proceedings of the Advances in Cryptology — CRYPTO '89 Proceedings*; Springer: Berlin/Heidelberg, Germany, 1989; pp. 218–238. [CrossRef]
21. Bos, J.W.; Renes, J.; Sprenkels, A. Dilithium for Memory Constrained Devices. In *Proceedings of the Progress in Cryptology—AFRICACRYPT 2022*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2022; Volume 13503, pp. 217–237. [CrossRef]
22. Bos, J.W.; Renes, J.; van Vredendaal, C. *Post-Quantum Cryptography with Contemporary Co-Processors: Beyond Kronecker, Schönhage-Strassen & Nussbaumer*; Cryptology ePrint Archive, Paper 2020/1303; International Association for Cryptologic Research (IACR): Bellevue, WA, USA, 2020. Available online: https://eprint.iacr.org/2020/1303.pdf (accessed on 1 February 2026).
23. Satriawan, A.; Syafalni, I.; Mareta, R.; Anshori, I.; Shalannanda, W.; Barra, A. Conceptual Review on Number Theoretic Transform and Comprehensive Review on Its Implementations. *IEEE Access* **2023**, *11*, 70288–70316. [CrossRef]
24. Zhang, J.; Yan, Y.; Huang, J.; Koç, Ç.K. Optimized Software Implementation of Keccak, Kyber, and Dilithium on RV{32,64}IM{B}{V}. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2025**, *2025*, 632–655. [CrossRef]
25. Kannwischer, M.J.; Rijneveld, J.; Schwabe, P.; Stoffelen, K. pqm4: Testing and Benchmarking NIST PQC on ARM Cortex-M4. In Proceedings of the Second NIST PQC Standardization Conference, Santa Barbara, CA, USA, 22–24 August 2019.
26. Tan, W.; Wang, A.; Zhang, X.; Lao, Y.; Parhi, K.K. High-Speed VLSI Architectures for Modular Polynomial Multiplication via Fast Filtering and Applications to Lattice-Based Cryptography. *IEEE Trans. Comput.* **2023**, *72*, 2454–2466. [CrossRef]
27. Andresen, J.; Arnold, P.; Berndt, S.; Eisenbarth, T.; Faust, S.; Gourjon, M.; Landthaler, E.; Micheli, E.; Orlt, M.; Pauls, P.; et al. UP TO 50% OFF: Efficient Implementation of Polynomial Masking. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2026**, *2026*, 688–731. [CrossRef]
28. Fritzmann, T.; Beirendonck, M.V.; Roy, D.B.; Karl, P.; Schamberger, T.; Verbauwhede, I.; Sigl, G. Masked Accelerators and Instruction Set Extensions for Post-Quantum Cryptography. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2022**, *2022*, 414–460. [CrossRef]
29. Ravi, P.; Chattopadhyay, A.; D'Anvers, J.P.; Baksi, A. Side-channel and Fault-injection attacks over Lattice-based Post-quantum Schemes (Kyber, Dilithium): Survey and New Results. *ACM Trans. Embed. Comput. Syst.* **2024**, *23*, 35. [CrossRef]
30. Azouaoui, M.; Bos, J.W.; Cloostermans, C.; Davies, G.T.; Esmann, S. A Brief Outlook on the Migration to Post-Quantum Cryptography. 2024. Available online: https://www.nxp.com/company/about-nxp/smarter-world-blog/BL-A-BRIEF-OUTLOOK (accessed on 30 August 2025).
31. Erata, F.; Piskac, R.; Mateu, V.; Szefer, J. Towards Automated Detection of Single-Trace Side-Channel Vulnerabilities in Constant-Time Cryptographic Code. In Proceedings of the 2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P), Delft, The Netherlands, 3–7 July 2023; pp. 687–706. [CrossRef]
32. Huang, Z.; Wang, H.; Cao, B.; He, D.; Wang, J. A comprehensive side-channel leakage assessment of CRYSTALS-Kyber in IIoT. *Internet Things* **2024**, *27*, 101331. [CrossRef]
33. Brandão, L.T.A.N.; Mouha, N.; Vassilev, A. *Threshold Schemes for Cryptographic Primitives: Challenges and Opportunities in Standardization and Validation*; NIST Internal Report 8214; National Institute of Standards and Technology (NIST): Gaithersburg, MD, USA, 2019. [CrossRef]
34. National Security Agency. The Commercial National Security Algorithm Suite 2.0 and Quantum Computing FAQ (U/OO/194427-22, Ver. 2.1, December 2024). Available online: https://media.defense.gov/2022/Sep/07/2003071836/-1/-1/0/CSI_CNSA_2.0_FAQ_.PDF (accessed on 15 January 2026).
35. Kiktenko, E.O.; Bulychev, A.A.; Karagodin, P.A.; Pozhar, N.O.; Anufriev, M.N.; Fedorov, A.K. SPHINCS+ post-quantum digital signature scheme with Streebog hash function. *AIP Conf. Proc.* **2020**, *2241*, 020014. [CrossRef]
36. National Institute of Standards and Technology. *Stateless Hash-Based Digital Signature Standard*; Technical Report FIPS PUB 205; U.S. Department of Commerce: Washington, DC, USA, 2024.
37. Chen, K.-Y.; Chen, J.-P. Masking Floating-Point Number Multiplication and Addition of Falcon: First- and Higher-Order Implementations and Evaluations. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2024**, *2024*, 276–303. [CrossRef]
38. Gigerl, B.; Mendel, F.; Schläffer, M.; Primas, R. *Efficient First-Order Masked Ascon on 32-Bit Architectures*; Cryptology ePrint Archive, Paper 2024/755; International Association for Cryptologic Research (IACR): Bellevue, WA, USA, 2024. Available online: https://eprint.iacr.org/2024/755 (accessed on 1 February 2026).

39. Kandi, A.; Baksi, A.; Gan, P.; Guilley, S.; Gerlich, T.; Breier, J.; Chattopadhyay, A.; Shrivastwa, R.R.; Martinásek, Z.; Bhasin, S. Side-Channel and Fault Resistant ASCON Implementation: A Detailed Hardware Evaluation. In Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Knoxville, TN, USA, 1–3 July 2024; pp. 307–312. [CrossRef]
40. Mosca, M. Cybersecurity in an Era with Quantum Computers: Will We Be Ready? *IEEE Secur. Priv.* **2018**, *16*, 38–41. [CrossRef]
41. Abdulrahman, A.; Hwang, V.; Kannwischer, M.J.; Sprenkels, D. *Faster Kyber and Dilithium on the Cortex-M4*; Cryptology ePrint Archive, Paper 2022/112; International Association for Cryptologic Research (IACR): Bellevue, WA, USA, 2022. Available online: https://eprint.iacr.org/2022/112 (accessed on 1 February 2026).
42. Gao, Y.; Al-Sarawi, S.F.; Abbott, D. Physical Unclonable Functions. *Nat. Electron.* **2020**, *3*, 81–91. [CrossRef]
43. Herder, C.; Yu, M.D.; Koushanfar, F.; Devadas, S. Physical Unclonable Functions and Applications: A Tutorial. *Proc. IEEE* **2014**, *102*, 1126–1141. [CrossRef]
44. Maes, R.; Verbauwhede, I. Physically Unclonable Functions: A Study on the State of the Art and Future Research Directions. In *Towards Hardware-Intrinsic Security*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 3–37. [CrossRef]
45. Tehranipoor, F.; Karimian, N.; Yan, W.; Chandy, J.A. DRAM-Based Intrinsic Physically Unclonable Functions for System-Level Security and Authentication. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2017**, *25*, 1085–1097. [CrossRef]
46. Pappu, R.; Recht, B.; Taylor, J.; Gershenfeld, N. Physical One-Way Functions. *Science* **2002**, *297*, 2026–2030. [CrossRef] [PubMed]
47. Gassend, B.; Clarke, D.; van Dijk, M.; Devadas, S. Silicon Physical Random Functions. In Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS '02), Washington, DC, USA, 18–22 November 2002; pp. 148–160. [CrossRef]
48. Lee, J.W.; Lim, D.; Gassend, B.; Suh, G.E.; van Dijk, M.; Devadas, S. A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Applications. In Proceedings of the 2004 Symposium on VLSI Circuits, Honolulu, HI, USA, 17–19 June 2004; pp. 176–179. [CrossRef]
49. Suh, G.E.; Devadas, S. Physical Unclonable Functions for Device Authentication and Secret Key Generation. In Proceedings of the 44th Annual Design Automation Conference (DAC '07), San Diego, CA, USA, 4–8 June 2007; pp. 9–14. [CrossRef]
50. Rührmair, U.; Sehnke, F.; Sölter, J.; Dror, G.; Devadas, S.; Schmidhuber, J. Modeling Attacks on Physical Unclonable Functions. In Proceedings of the ACM Conference on Computer and Communications Security (CCS), Chicago IL, USA, 4–8 October 2010; pp. 237–249. [CrossRef]
51. Merli, D.; Schuster, D.; Stumpf, F.; Sigl, G. Semi-Invasive EM Attack on FPGA RO PUFs and Countermeasures. In Proceedings of the Workshop on Embedded Systems Security (WESS '11), Taipei, Taiwan, 9 October 2011; ACM: New York, NY, USA, 2012; pp. 1–9. [CrossRef]
52. Delvaux, J.; Gu, D.; Schellekens, D.; Verbauwhede, I. Helper Data Algorithms for PUF-Based Key Generation: Overview and Analysis. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2015**, *34*, 889–902. [CrossRef]
53. *ISO/IEC 20897:2020*; Information Security—Security Techniques—Physically Unclonable Functions. International Organization for Standardization: Geneva, Switzerland, 2020.
54. Intrinsic ID. Intrinsic ID Protects 500 000 000 Devices Globally: Leading the Way in Secure and Authenticated Connected Devices. *EEJournal Industry News*, 2023. Available online: https://www.eejournal.com/industry_news/intrinsic-id-protects-500000000-devices-globally-leading-the-way-in-secure-and-authenticated-connected-devices/ (accessed on 15 February 2023).
55. Korenda, A.R.; Afghah, F.; Cambou, B.; Philabaum, C. A Proof of Concept SRAM-based Physically Unclonable Function (PUF) Key Generation Mechanism for IoT Devices. In Proceedings of the 2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), Boston, MA, USA, 10–13 June 2019; pp. 1–8. [CrossRef]
56. Intrinsic ID. Intrinsic ID Debuts Software-Only, Chip-Based Cybersecurity (Zign X00 Series). Futuriom, 2023. Available online: https://www.futuriom.com/articles/news/intrinsic-id-debuts-software-only-chip-based-cybersecurity/2023/03 (accessed on 2 January 2026).
57. Devices, A. ChipDNA Embedded Security PUF Technology. Available online: https://www.analog.com/en/lp/001/chipdna-embedded-security-puf-technology.html (accessed on 30 August 2025).
58. Wen, Y.; Lao, Y. Efficient Fuzzy Extractor Implementations for PUF Based Authentication. In Proceedings of the 12th International Conference on Malicious and Unwanted Software (MALWARE), Fajardo, PR, USA, 11–14 October 2017; pp. 119–125. [CrossRef]
59. Khalafalla, M.; Elmohr, M.A.; Gebotys, C. Going Deep: Using deep learning techniques with simplified mathematical models against XOR BR and TBR PUFs (Attacks and Countermeasures). In Proceedings of the 2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), San Jose, CA, USA, 7–11 December 2020; pp. 80–90. [CrossRef]
60. Gupta, A.; Manhas, S.; Das, B.P. Highly Non-linear Feed-Forward Arbiter PUF Against Machine Learning Attacks. In *Proceedings of the VLSI Design and Test*; Shah, A.P., Dasgupta, S., Darji, A., Tudu, J., Eds.; Springer: Cham, Switzerland, 2022; pp. 234–248.
61. Mispan, M.S.; Halak, B.; Zwolinski, M. Lightweight obfuscation techniques for modeling attacks resistant PUFs. In Proceedings of the 2017 IEEE 2nd International Verification and Security Workshop (IVSW), Thessaloniki, Greece, 3–5 July 2017; pp. 19–24. [CrossRef]
62. Yu, M.D.; Hiller, M.; Delvaux, J.; Sowell, R.; Devadas, S.; Verbauwhede, I. A Lockdown Technique to Prevent Machine Learning on PUFs for Lightweight Authentication. *IEEE Trans. Multi-Scale Comput. Syst.* **2016**, *2*, 146–159. [CrossRef]

63. Taneja, S.; Alvarez, A.B.; Alioto, M. Fully Synthesizable PUF Featuring Hysteresis and Temperature Compensation for 3.2% Native BER and 1.02 fJ/b in 40 nm. *IEEE J. Solid-State Circuits* **2018**, *53*, 2828–2839. [CrossRef]
64. Ibrahim, H.M.; Skovorodnikov, H.; Alkhzaimi, H. Resilience Evaluation of Memristor Based PUF Against Machine Learning Attacks. *Sci. Rep.* **2024**, *14*, 23962. [CrossRef]
65. Arapinis, M.; Delavar, M.; Doosti, M.; Kashefi, E. Quantum Physical Unclonable Functions: Possibilities and Impossibilities. *Quantum* **2021**, *5*, 475. [CrossRef]
66. GlobalPlatform. Root of Trust: Definitions and Requirements, Version 1.1. 2018. Available online: https://globalplatform.org/wp-content/uploads/2018/07/GP_RoT_Definitions_and_Requirements_v1.1_PublicRelease-2018-06-28.pdf (accessed on 29 August 2025).
67. Hu, W.; Chang, C.H.; Sengupta, A.; Bhunia, S.; Kastner, R.; Li, H. An Overview of Hardware Security and Trust: Threats, Countermeasures, and Design Tools. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2021**, *40*, 1010–1038. [CrossRef]
68. Eldefrawy, K.; Tsudik, G.; Francillon, A.; Perito, D. SMART: Secure and Minimal Architecture for (Establishing Dynamic) Root of Trust. In Proceedings of the NDSS, San Diego, CA, USA, 5–8 February 2012.
69. Koeberl, P.; Schulz, S.; Sadeghi, A.R.; Varadharajan, V. TrustLite: A Security Architecture for Tiny Embedded Devices. In Proceedings of the EuroSys Conference, Amsterdam, The Netherlands, 13–16 April 2014; pp. 1–14. [CrossRef]
70. Noorman, J.; Bulck, J.V.; Mühlberg, J.T.; Piessens, F.; Maene, P.; Preneel, B.; Verbauwhede, I.; Götzfried, J.; Müller, T.; Freiling, F. Sancus 2.0: A Low-Cost Security Architecture for IoT Devices. *ACM Trans. Priv. Secur.* **2017**, *20*, 1–33. [CrossRef]
71. Brasser, F.; El Mahjoub, U.; Sadeghi, A.R.; Wachsmann, C.; Koeberl, P. SANCTUARY: ARMing TrustZone with User-space Enclaves. In Proceedings of the NDSS Symposium, San Diego, CA, USA, 24–27 February 2019. [CrossRef]
72. Dessouky, G.; Abera, T.; Ibrahim, A.; Sadeghi, A.R. LiteHAX: Lightweight Hardware-Assisted Attestation of Program Execution. In Proceedings of the International Conference on Computer-Aided Design (ICCAD '18), San Diego, CA, USA, 5–8 November 2018; ACM: New York, NY, USA, 2018; pp. 1–8. [CrossRef]
73. Regenscheid, A.; Regenscheid, A.R. *Platform Firmware Resiliency Guidelines*; NIST Special Publication 800-193; National Institute of Standards and Technology (NIST): Gaithersburg, MD, USA, 2018. Available online: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-193.pdf (accessed on 30 August 2025). [CrossRef]
74. Microchip Technology Inc. ATECC608B CryptoAuthentication Device Summary Data Sheet, 2023. Available online: https://ww1.microchip.com/downloads/aemDocuments/documents/SCBU/ProductDocuments/DataSheets/ATECC608B-CryptoAuthentication-Device-Summary-Data-Sheet-DS40002239B.pdf (accessed on 2 January 2026).
75. Intrinsic ID. Intrinsic ID Becomes World's First IP Vendor with PSA Certified Level 3 Root of Trust Component, 2023. Available online: https://www.psacertified.org/partner-showcase/intrinsic-id/ (accessed on 15 January 2026).
76. EMVCo. EMV®Specifications, 2025. Available online: https://www.emvco.com/specifications/ (accessed on 30 August 2025).
77. National Institute of Standards and Technology. *Security Requirements for Cryptographic Modules*; Technical Report FIPS PUB 140-3; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2019. [CrossRef]
78. Musa, A.; Volante, F.; Parisi, E.; Barbierato, L.; Patti, E.; Bartolini, A.; Acquaviva, A.; Barchi, F. TitanSSL: Towards Accelerating OpenSSL in a Full RISC-V Architecture Using OpenTitan Root-of-Trust. In *Proceedings of the Computer Safety, Reliability, and Security*; Ceccarelli, A., Trapp, M., Bondavalli, A., Bitsch, F., Eds.; Springer: Cham, Switzerland, 2024; pp. 169–183.
79. Wagner, A.; Oberhansl, F.; Schink, M. To Be, or Not to Be Stateful: Post-Quantum Secure Boot using Hash-Based Signatures. In Proceedings of the 2022 Workshop on Attacks and Solutions in Hardware Security, ASHES'22, New York, NY, USA, 11 November 2022; pp. 85–94. [CrossRef]
80. Dushku, E.; Dragoni, N. Remote Attestation in IoT Devices. In *Encyclopedia of Cryptography, Security and Privacy*; Jajodia, S., Samarati, P., Yung, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; pp. 1–4. [CrossRef]
81. Abera, T.; Asokan, N.; Davi, L.; Ekberg, J.E.; Nyman, T.; Paverd, A.; Sadeghi, A.R.; Tsudik, G. C-FLAT: Control-Flow Attestation for Embedded Systems Software. In Proceedings of the ACM CCS, Vienna, Austria, 24–28 October 2016; pp. 743–754. [CrossRef]
82. Carpent, X.; Rattanavipanon, N.; Tsudik, G. ERASMUS: Efficient Remote Attestation via Self-Measurement for Unattended Settings. In Proceedings of the DATE, Dresden, Germany, 19–23 March 2018; pp. 1191–1194. [CrossRef]
83. Zeitouni, S.; Dessouky, G.; Arias, O.; Sullivan, D.; Ibrahim, A.; Jin, Y.; Sadeghi, A.R. ATRIUM: Runtime Attestation Resilient Under Memory Attacks. In Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, Irvine, CA, USA, 13–16 November 2017; pp. 384–391. [CrossRef]
84. Clements, A.A.; Almakhdhub, N.S.; Saab, K.S.; Srivastava, P.; Koo, J.; Bagchi, S.; Payer, M. APEX: A Verified Architecture for Proofs of Execution on Remote Devices under Full Software Compromise. In Proceedings of the USENIX Security Symposium, Boston, MA, USA, 12–14 August 2020; pp. 771–788.
85. Sun, N.; Li, C.T.; Chan, H.; Le, B.D.; Islam, M.Z.; Zhang, L.Y.; Islam, M.R.; Armstrong, W. Defining Security Requirements With the Common Criteria: Applications, Adoptions, and Challenges. *IEEE Access* **2022**, *10*, 44756–44777. [CrossRef]
86. Thanh Vu, S.N.; Stege, M.; El-Habr, P.I.; Bang, J.; Dragoni, N. A Survey on Botnets: Incentives, Evolution, Detection and Current Trends. *Future Internet* **2021**, *13*, 198. [CrossRef]

87. Kocher, P.C. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *Proceedings of the Advances in Cryptology—CRYPTO '96*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 104–113. [CrossRef]
88. Kocher, P.; Jaffe, J.; Jun, B. Differential Power Analysis. In *Proceedings of the Advances in Cryptology—CRYPTO '99*; Springer: Berlin/Heidelberg, Germany, 1999; pp. 388–397. [CrossRef]
89. Masure, L.; Strullu, R. Side-channel analysis against ANSSI's protected AES implementation on ARM: End-to-end attacks with multi-task learning. *J. Cryptogr. Eng.* **2023**, *13*, 163–179. [CrossRef]
90. Balon, B.; Grassi, L.; Méaux, P.; Moos, T.; Standaert, F.X.; Steiner, M.J. mid-pSquare: Leveraging the Strong Side-Channel Security of Prime-Field Masking in Software. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2025**, *2025*, 486–519. [CrossRef]
91. ANSSI. *SecAESSTM32: Secure AES Implementation for STM32*; ANSSI: Paris, France, 2019. Available online: https://github.com/ANSSI-FR/SecAESSTM32 (accessed on 15 January 2026).
92. Toprakhisar, D.; Nikova, S.; Nikov, V. *Combined Stability: Protecting against Combined Attacks*; Cryptology ePrint Archive, Paper 2025/1692; International Association for Cryptologic Research (IACR): Bellevue, WA, USA, 2025. Available online: https://eprint.iacr.org/2025/1692 (accessed on 1 February 2026).
93. Woo, J.; Seo, D.; Kim, Y.S.; Lee, N.; Cassuto, Y.; Kim, Y. Mutual Information Minimization for Side-Channel Attack Resistance via Optimal Noise Injection. *arXiv* **2025**, arXiv:2504.20556. Available online: http://arxiv.org/abs/2504.20556 (accessed on 1 February 2026). [CrossRef]
94. Das, D.; Maity, S.; Nasir, S.B.; Ghosh, S.; Raychowdhury, A.; Sen, S. High Efficiency Power Side-Channel Attack Immunity using Noise Injection in Attenuated Signature Domain. In Proceedings of the IEEE International Symposium on Hardware Oriented Security and Trust (HOST), McLean, VA, USA, 1–5 May 2017; pp. 62–67. [CrossRef]
95. Coron, J.S.; Greuet, A.; Zeitoun, R. *Side-Channel Masking with Pseudo-Random Generator*; Cryptology ePrint Archive, Paper 2019/1106; International Association for Cryptologic Research (IACR): Bellevue, WA, USA, 2019. Available online: https://eprint.iacr.org/2019/1106 (accessed on 1 February 2026).
96. Coron, J.; Rondepierre, F.; Zeitoun, R. High Order Masking of Look-up Tables with Common Shares. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2018**, *2018*, 40–72. [CrossRef]
97. Veyrat-Charvillon, N.; Medwed, M.; Kerckhof, S.; Standaert, F. Shuffling against Side-Channel Attacks: A Comprehensive Study with Cautionary Note. In *International Conference on the Theory and Application of Cryptology and Information Security*; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7658, pp. 740–757. [CrossRef]
98. Zhou, J.; Qin, G.; Li, L.; Guo, C. ISA Extensions of Shuffling Against Side-Channel Attacks. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2023**, *43*, 761–773. [CrossRef]
99. Park, J.Y.; Ju, J.W.; Lee, W.; Kang, B.G.; Kachi, Y.; Sakurai, K. A Statistical Verification Method of Random Permutations for Hiding Countermeasure Against Side-Channel Attacks. *J. Inf. Secur. Appl.* **2024**, *84*, 103797. [CrossRef]
100. Mainka, L.; Papagiannopoulos, K. Combined Masking and Shuffling for Side-Channel Secure Ascon on RISC-V. In *Constructive Approaches for Security Analysis and Design of Embedded Systems*; Springer Nature Switzerland: Cham, Switzerland, 2026; pp. 451–477. [CrossRef]
101. Gigerl, B.; Klug, F.; Mangard, S.; Mendel, F.; Primas, R. Smooth Passage with the Guards: Second-Order Hardware Masking of the AES with Low Randomness and Low Latency. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2024**, *2024*, 309–335. [CrossRef]
102. Valiveti, B.K.; Vivek, S. Higher-Order Lookup Table Masking in Essentially Constant Memory. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2021**, *2021*, 546–586. [CrossRef]
103. Sokolov, D.; Murphy, J.; Bystrov, A.; Yakovlev, A. Improving the Security of Dual-Rail Circuits. In *Proceedings of the Cryptographic Hardware and Embedded Systems—CHES 2004*; Joye, M., Quisquater, J.J., Eds.; *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3156, pp. 282–297.
104. Batina, L.; Bhasin, S.; Jap, D.; Picek, S. CSI NN: Reverse Engineering of Neural Network Architectures Through Electromagnetic Side Channel. In Proceedings of the 28th USENIX Security Symposium (USENIX Security 19), Santa Clara, CA, USA, 14–16 August 2019; pp. 515–532.
105. Ni, T.; Zhang, X.; Zhao, Q. Recovering Fingerprints from In-Display Fingerprint Sensors via Electromagnetic Side Channel. In Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23), Copenhagen, Denmark, 26–30 November 2023; pp. 253–267. [CrossRef]
106. Becker, G.T.; Cooper, J.; DeMulder, E.K.; Goodwill, G.; Jaffe, J.; Kenworthy, G.; Kouzminov, T.; Leiserson, A.J.; Marson, M.E.; Rohatgi, P.; et al. Test Vector Leakage Assessment (TVLA) Methodology in Practice. In Proceedings of the International Cryptographic Module Conference, Gaithersburg, MD, USA, 24–26 September 2013.
107. Ambrose, J.A.; Ragel, R.G.; Parameswaran, S. Randomized Instruction Injection to Counter Power Analysis Attacks. *ACM Trans. Embed. Comput. Syst.* **2012**, *11*, 1–28. [CrossRef]
108. Bos, J.W.; Gourjon, M.; Renes, J.; Schneider, T.; van Vredendaal, C. Masking Kyber: First- and Higher-Order Implementations. *IACR Trans. Cryptogr. Hardw. Embed. Syst. (TCHES)* **2021**, *2021*, 173–214. [CrossRef]

109. Silicon Labs. *AN1329: Using Silicon Labs Secure Vault Features with OpenThread*; Technical Report; Silicon Laboratories Inc.: Austin, TX, USA, 2021. Available online: https://www.silabs.com/documents/public/application-notes/an1329-using-secure-vault-openthread.pdf (accessed on 2 January 2026).
110. Sajadi, A.; Zidaric, N.; Stefanov, T.; Mentens, N. A Systematic Comparison of Side-channel Countermeasures for RISC-V-based SoCs. In Proceedings of the Nordic Circuits and Systems Conference (NorCAS), Lund, Sweden, 29–30 October 2024. [CrossRef]
111. Bronchain, O.; Standaert, F. Side-Channel Countermeasures' Dissection and the Limits of Closed Source Security Evaluations. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2020**, *2020*, 1–25. [CrossRef]
112. Heinz, D.; Kannwischer, M.J.; Land, G.; Pöppelmann, T.; Schwabe, P.; Sprenkels, A. *First-Order Masked Kyber on ARM Cortex-M4*; Cryptology ePrint Archive, Paper 2022/058; International Association for Cryptologic Research (IACR): Bellevue, WA, USA, 2022. Available online: https://eprint.iacr.org/2022/058 (accessed on 1 February 2026).
113. Tosun, M.; Oswald, E.; Savaş, E. Non-Profiled Higher-Order Side-Channel Attacks against Lattice-Based Post-Quantum Cryptography. *IACR Commun. Cryptol.* **2025**, *2*, 31. [CrossRef]
114. Bos, J.W.; Gourjon, M.; Renes, J.; Schneider, T.; van Vredendaal, C. *Masking Kyber: First- and Higher-Order Implementations*; Cryptology ePrint Archive, Paper 2021/483; International Association for Cryptologic Research (IACR): Bellevue, WA, USA, 2021. Available online: https://eprint.iacr.org/2021/483 (accessed on 1 February 2026).
115. Schwabe, P.; Stoffelen, K. All the AES You Need on Cortex-M3 and M4. In *Proceedings of the Selected Areas in Cryptography—SAC 2016*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2017; Volume 10532, pp. 180–194. [CrossRef]
116. Silicon Labs. *Third Party Accreditation—PSA Certified*; Silicon Labs Security: Austin, TX, USA, 2023. Available online: https://www.silabs.com/security/third-party-accreditation (accessed on 2 January 2026).
117. Boubakri, M.; Zouari, B. A Survey of RISC-V Secure Enclaves and Trusted Execution Environments. *Electronics* **2025**, *14*, 4171. [CrossRef]
118. RISC-V International. Towards Generic RISC-V TEE Ecosystem with Penglai and OP-TEE. RISC-V Blog, 2024. Available online: https://riscv.org/blog/2024/10/towards-generic-risc-v-tee-ecosystem-with-penglai-and-op-tee/ (accessed on 2 January 2026).
119. Hex Five Security, Inc. MultiZone Security: Trusted Execution Environment for RISC-V. GitHub, 2024. Available online: https://github.com/hex-five/multizone-sdk (accessed on 2 January 2026).
120. Costan, V.; Lebedev, I.; Devadas, S. Sanctum: Minimal Hardware Extensions for Strong Software Isolation. In *Proceedings of the 25th USENIX Security Symposium (USENIX Security 16)*; USENIX Association: Austin, TX, USA, 2016; pp. 857–874.
121. Lee, D.; Kohlbrenner, D.; Shinde, S.; Asanović, K.; Song, D. Keystone: An open framework for architecting trusted execution environments. In *Proceedings of the Fifteenth European Conference on Computer Systems, EuroSys '20*; Association for Computing Machinery: New York, NY, USA, 2020. [CrossRef]
122. Ma, Z.; Zhang, N.; Wei, S.; Chen, Y.; Guan, N. Return-to-Non-Secure Vulnerabilities on ARM Cortex-M TrustZone: Attack and Defense. In Proceedings of the 2023 60th ACM/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, 9–13 July 2023; pp. 1–6. [CrossRef]
123. Pinto, S.; Santos, N. Demystifying ARM TrustZone: A Comprehensive Survey. *ACM Comput. Surv.* **2019**, *51*, 1–36. [CrossRef]
124. Ma, Z.; Zhang, N.; Wei, S.; Chen, Y.; Guan, N. Return-to-Non-Secure Vulnerabilities on ARM Cortex-M TrustZone. Technical Report, NSF-PAR ID: 10438091, 2023. Available online: https://par.nsf.gov/biblio/10438091 (accessed on 1 February 2026).
125. Mishra, N.; Chakraborty, A.; Mukhopadhyay, D. Faults in Our Bus: Novel Bus Fault Attack to Break ARM TrustZone. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*; Internet Society: San Diego, CA, USA, 2024. [CrossRef]
126. Trusted Firmware Project. Trusted Firmware-M: Secure Software for Arm Cortex-M and Armv8-M. TrustedFirmware.org, 2024. Available online: https://www.trustedfirmware.org/projects/tf-m/ (accessed on 2 January 2026).
127. Feng, E.; Lu, X.; Du, D.; Yang, B.; Jiang, X.; Xia, Y.; Zang, B.; Chen, H. Scalable Memory Protection in the PENGLAI Enclave. In *Proceedings of the 15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*; USENIX Association: Berkeley, CA, USA, 2021; pp. 275–294.
128. Schrammel, D.; Waser, M.; Lamster, L.; Unterguggenberger, M.; Mangard, S. SPEAR-V: Secure and Practical Enclave Architecture for RISC-V. In Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security (ASIA CCS '23), Melbourne, VIC, Australia, 10–14 July 2023; pp. 457–468. [CrossRef]
129. Amacher, J.; Schiavoni, V. On the Performance of ARM TrustZone. In *Proceedings of the Distributed Applications and Interoperable Systems (DAIS 2019)*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11534, pp. 133–151. [CrossRef]
130. Bahmani, R.; Brasser, F.; Dessouky, G.; Jauernig, P.; Klimmek, M.; Sadeghi, A.R.; Stapf, E. CURE: A Security Architecture with CUstomizable and Resilient Enclaves. In *Proceedings of the 30th USENIX Security Symposium (USENIX Security 21)*; USENIX Association: Berkeley, CA, USA, 2021; pp. 1073–1090.

131. Weiser, S.; Werner, M.; Brasser, F.; Malenko, M.; Mangard, S.; Sadeghi, A.R. TIMBER-V: Tag-Isolated Memory Bringing Fine-grained Enclaves to RISC-V. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS 2019)*; Internet Society: San Diego, CA, USA, 2019. [CrossRef]
132. lowRISC C.I.C.; OpenTitan Coalition. OpenTitan Partnership Makes History as First Open-Source Silicon Project to Reach Commercial Availability, 2024. Available online: https://lowrisc.org/news/opentitan-commercial-availability/ (accessed on 15 January 2026).
133. Kocher, P.; Horn, J.; Fogh, A.; Genkin, D.; Gruss, D.; Haas, W.; Hamburg, M.; Lipp, M.; Mangard, S.; Prescher, T.; et al. Spectre Attacks: Exploiting Speculative Execution. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 19–23 May 2019; pp. 1–19. [CrossRef]
134. Lipp, M.; Schwarz, M.; Gruss, D.; Prescher, T.; Haas, W.; Fogh, A.; Horn, J.; Mangard, S.; Kocher, P.; Genkin, D.; et al. Meltdown: Reading Kernel Memory from User Space. In Proceedings of the 27th USENIX Security Symposium (USENIX Security 18), Baltimore, MD, USA, 15–17 August 2018; pp. 973–990.
135. Fagan, M.; Megas, K.N.; Scarfone, K.; Smith, M. *Foundational Cybersecurity Activities for IoT Device Manufacturers*; NIST Interagency Report NISTIR 8259; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2020. [CrossRef]