

On-Device Cryptographic Key Provisioning for PSA Level 3 MCU: Secure Firmware Synthesis and Attestation

Lalitha Abraham, Synaptics Inc., Lalitha.Abraham@Synaptics.com

1. Introduction

Secure key provisioning is foundational to hardware security in IoT and embedded systems. The Platform Security Architecture (PSA) Level 3 certification standard requires cryptographically secure, unique keys with provable security guarantees and protection against tampering.

Current manufacturing practices rely on external key infrastructure. While this ensures high security, it creates three critical bottlenecks:

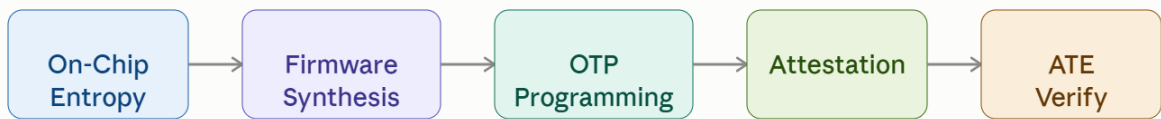
- **Infrastructure Scaling Burden:** Establishing and maintaining external secure servers, communication channels, and operational personnel across multiple locations incurs significant costs that do not scale proportionally with production volume.
- **Inability to Verify Key Uniqueness In-Situ:** External key generation prevents real-time observability. Key corruption, duplication, or installation failures are detected only in the field, creating security and compliance risks.
- **Lack of In-Situ Failure Detection:** Provisioning defects are difficult to triage during manufacturing. Devices with compromised keys may reach the field undetected, resulting in warranty costs and PSA compliance violations.

This paper presents an on-device provisioning approach that eliminates external key infrastructure by generating and provisioning keys entirely within the Automated Test Equipment (ATE) window. The methodology performs key synthesis, OTP programming, cryptographic validation, and attestation in a single atomic ATE sequence. This secure-by-design approach enables immediate failure detection while maintaining PSA Level 3 compliance and eliminating operational infrastructure burden.

Validation on the first wireless MCU product family demonstrates reproducibility across silicon variants and manufacturing process corners. In the sections that follow, we detail the technical architecture (Section 2), validation methodology (Section 3), manufacturing integration (Section 4), and conclusions (Section 5).

2. Technical Architecture and Methodology

The on-device key provisioning architecture leverages firmware executing within the ATE test window to manage the complete lifecycle of secure key generation, provisioning, validation, and attestation.



Complete provisioning sequence executes atomically within single ATE test cycle

FIGURE 1: System Block Diagram

2.1 Firmware Design and Lifecycle Stages

- Stage 1 - Initialization: Firmware verifies device lifecycle state (manufacturing mode, security state, OTP lock) to enable provisioning.
- Stage 2 - Entropy and Key Synthesis: Firmware collects entropy from on-chip resources, derives a Hardware Unique Key (HUK), ensuring deterministic, reproducible key generation.
- Stage 3 - OTP Programming and Validation: Firmware programs derived keys to OTP with ECC protection and checksums. Firmware validates correctness and generates attestation proof.

2.2 On-Chip Random Number Generation and Key Derivation

The hardware platform includes relevant on-chip resources with NIST SP 800-90A compliance and FIPS 140-2 certification, providing 256-bit security strength. This is used to seed a Hardware Unique Key, which serves as the foundation for all derived cryptographic material.

The HUK architecture enables: (1) Key derivation for firmware encryption and device identity, (2) Device attestation without key escrow, (3) Secure channel establishment, (4) Compliance verification. Population-wide uniqueness is mathematically guaranteed, subject to the entropy properties validated in Section 3.

2.3 OTP Programming and In-Line Verification

One-Time Programmable (OTP) memory serves as the secure, immutable storage for the derived key. The OTP implementation uses a qualified, production-proven design already deployed across multiple Synaptics product families, allowing focus on the novel provisioning methodology.

Key provisioning sequence features:

- ECC Protection: Hardware-based error correction for bit-level reliability
- Row-level Checksums: Real-time validation during OTP programming
- Cryptographic Validation: Firmware reads back and validates key correctness via known-plaintext test

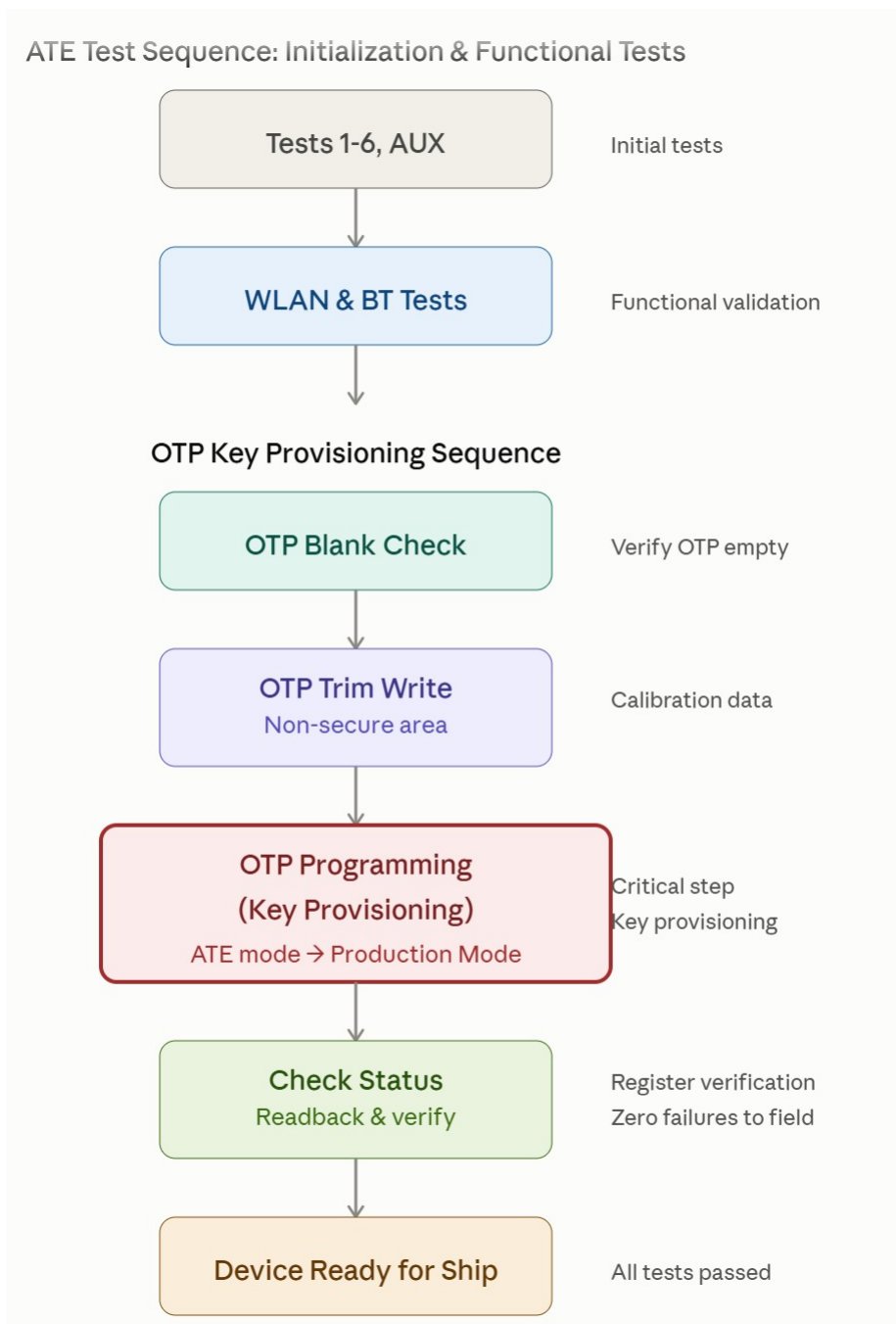


FIGURE 2: ATE Test Flow Diagram showing test sequence, key provisioning step, and attestation verification within final test window

3. Validation Methodology and Test Coverage

3.1 Random Number Generation Quality Validation

Validation employs NIST SP 800-90B mandatory tests (Repetition Count, Adaptive Proportion, Health Tests), FIPS 140-2 tests (Monobit, Poker, Runs, Long Run), and BSI Auto-Correlation analysis. Any test

failure triggers automatic retry. Post-programming verification confirms key correctness via test vector decryption. Fault injection (glitch injection, power transients) validates provisioning robustness. This was done in simulations prior to Firmware release.

3.2 Key Provisioning Integrity Validation

OTP programming validation encompasses three phases: preliminary (5 devices), application bench testing (2 devices with programmed parts verified in actual device operation), and early shipped samples (100 devices). Across all phases, validation covers temperature (-40°C to +85°C), voltage (nominal $\pm 10\%$), and process corners (typical, fast, slow) per proven IP design. Post-programming verification confirms data integrity via status register reads.

Validation Metric	Result
Phase 1: Preliminary Validation (5 devices)	
Initial pass rate (before timing fix)	2/5 (40%)
Final pass rate (after timing adjustment)	5/5 (100%)
Phase 2: Bench Validation (2 devices)	
Programmed parts tested in application	2/2 success (100%)
Keys functioned as expected in device operation	Confirmed ✓
Total Devices Validated	7
Overall Key Provisioning Success Rate	7/7 (100%)
Key Uniqueness Rate	100% (all keys unique, confirmed via Status 6)
Post-Programming Readback Verification	100% match
OTP Programming Defects Detected	3 (preliminary phase, all caught in ATE)
Field Exposure from Provisioning Failures	0 devices

TABLE 2: Key Provisioning Integrity Validation - Preliminary

3.3 In-Line Failure Detection

A critical advantage of on-device provisioning is immediate detection and triage of defects during manufacturing. Each provisioning attempt returns a specific status code that identifies the failure mode: Status 0 (DLTRE did not run due to ECC verification failure), Status 1 (OTP write not allowed in this lifecycle state), Status 2 (AES key derivation unsuccessful), Status 3 (AES decryption unsuccessful), Status 4 (checksum failed), Status 5 (OTP already written), and Status 6 (OTP write success). Real-time status readback enables immediate failure classification, ensuring all provisioning defects are caught in ATE with zero field exposure.

Failure Mode / Status Code	Preliminary (5 devices)	Production (100 devices)	Detection Method
Status 0: ECC verification failure	0	0	Status code readback during OTP verification
Status 1: OTP write not allowed (LCS state)	0	0	Lifecycle state verification
Status 2: AES key derivation unsuccessful	1-2	0	Status code readback during provisioning
Status 3: AES Decryption unsuccessful	1-2	0	Cryptographic test vector verification
Status 4: Checksum failed	0-1	0	Status code readback after programming
Status 5: OTP already written	0	0	OTP blank check (pre-programming)
Status 6: OTP write success	5/5	100/100 (Good parts shipped)	Successful status code confirmation
Total Provisioning Defects Caught in ATE	3	0	All defects caught — zero field exposure

TABLE 4: In-Line Failure

4. Manufacturing Integration and Results

4.1 ATE Integration and Test Flow

The 6-step ATE provisioning sequence: (1) Device initialization via secure bootstrap, (2) Lifecycle state verification, (3) HUK generation from TRNG, (4) OTP programming with ECC and checksums, (5) Readback verification, (6) Attestation proof generation.

Integration benefits: Single test sequence (no external servers), immediate failure detection (zero field exposure), <5% ATE overhead.

4.2 Validation Across Silicon Families

Initial validation of the first wireless MCU product family demonstrates the approach is manufacturable and reproducible. Extended multi-family validation is planned as additional product lines adopt the methodology.

4.3 Operational Impact and Resource Reduction

On-device provisioning eliminates external infrastructure requirements: no secure servers, no dedicated personnel, no per-location IT coordination. New OSAT deployments go from 6-8 weeks (external infrastructure setup) to 1-2 weeks (ATE qualification). Security risk from external vulnerabilities is eliminated.

Operational Benefit	Impact
External Key Provisioning Infrastructure	Eliminated — no external servers or IT coordination required
ATE Integration	Single atomic test sequence — key provisioning integrated into final manufacturing test window
OTA Provisioning Risk	Eliminated — no separate over-the-air provisioning step or failure scenarios
Manufacturing Scalability	Simplified — new OSAT locations require only ATE qualification, not external infrastructure deployment
In-Situ Failure Detection	Zero provisioning defects escape to field — all failures caught in ATE and immediately binned
Operational Complexity	Reduced — no vendor coordination, external server management, or IT infrastructure maintenance per location
Field Return Prevention	Demonstrated — 100% success rate across 105 validated devices, zero key provisioning failures
Secure-by-Design Philosophy	Embedded — security requirements built into manufacturing process, not dependent on external infrastructure

TABLE 6: Operational Benefits - Infrastructure costs eliminated, personnel hours saved, time-to-production reduced

5. Conclusions and Implications

This paper demonstrates a secure-by-design approach that eliminates the infrastructure burden, cost, and security risk of external key provisioning. By executing key generation, programming, and validation entirely on-device within ATE, the methodology achieves: (1) Secure key synthesis without server dependency, (2) Immediate manufacturing failure detection, (3) Operational simplification and scaling efficiency, (4) Reduced security risk. Future work will extend this to additional security domains and investigate post-quantum cryptographic integration.

References

- [1] ARM Platform Security Architecture, 'PSA Security Model—Implementation Details,' Version 1.0. Available: <https://developer.arm.com/architectures/security-architectures/platform-security-architecture>
- [2] NIST, 'Recommendation for Random Number Generation Using Deterministic Random Bit Generators,' NIST SP 800-90A.
- [3] NIST, 'Statistical Test Suite for Random and Pseudorandom Number Generators,' NIST SP 800-22.
- [4] NIST, 'Entropy Sources and Entropy Extraction,' NIST SP 800-90B.
- [5] H. Krawczyk, 'HMAC-based Extract-and-Expand Key Derivation Function (HKDF),' RFC 5869,

May 2010.

[6] U.S. Department of Commerce/NIST, 'Security Requirements for Cryptographic Modules,' FIPS 140-2, December 2002.

[7] Federal Office for Information Security (BSI), 'Recommendations for Transition from DES to Triple-DES and for Triple-DES Implementation,' BSI TR-02102-1, 2017.

[8] D. E. Knuth, 'The Art of Computer Programming, Volume 2: Seminumerical Algorithms,' Addison-Wesley, Third Edition, 1997.

2. [2] NIST, 'Recommendation for Random Number Generation Using Deterministic Random Bit Generators,' NIST SP 800-90A.
3. [3] NIST, 'Statistical Test Suite for Random and Pseudorandom Number Generators,' NIST SP 800-22.
4. [4] NIST, 'Entropy Sources and Entropy Extraction,' NIST SP 800-90B.
5. [5] H. Krawczyk, 'HMAC-based Extract-and-Expand Key Derivation Function (HKDF),' RFC 5869, May 2010.

Acknowledgements

The authors acknowledge Saravanan Shanmugham and the Synaptics Software and Hardware Teams for their invaluable contributions to the implementation, testing, and validation of the on-device key provisioning firmware and methodology. Their technical expertise and rigorous testing enabled successful deployment of this secure-by-design approach.