TROPIC01

A Cryptographic Coprocessor and Secure Storage IC

FEATURES

- Non-volatile memory
  - 32 slots for ECC keys
  - 238 kB for general purpose data
  - X.509 certificate signed by Tropic Square
  - ISAP encryption
  - Flash and AntiFuse technology
- Encrypted communication channel:
  - Strong forward secrecy
  - Authentication with up to 4 Hosts
  - Asymmetric key exchange
  - Encrypted and authenticated communication
  - User Access Privileges
- 2 x True Random Number Generator:
  - NIST800-90b, AIS31 compliant
  - Up to 128 kBit/s
  - Separate instance for internal engines and user
- Physically Unclonable Function (PUF):
  - 256-bit per-chip unique fingerprint
- Elliptic Curve Cryptography (ECC):
  - Key setup and signing operations
  - P-256 (ECDSA) and Ed25519 (EdDSA) curve support
- Atomic PIN verification:
  - Keccak based MAC-and-Destroy scheme
- On-chip physical security:
  - Attack sensors
  - Anti-tampering design
  - Active shield
- 16 x 32 bit Monotonic Counters
- SPI comminucation interface

APPLICATIONS

- Hardware Wallets
- Secure Embedded Systems
- FIDO/U2F Authenticators
- Hardware Root of Trust
- Digital identity
- Unique silicon-based fingerprint

DESCRIPTION

TROPIC01 is an openly auditable secure element used to store cryptographic keys and general purpose data. Each chip provides a unique silicon-based fingerprint for its internal cryptographic engines to enhance protection.

TROPIC01 uses various algorithms such as ECDSA, EdDSA, ECDH, AES-GCM, Keccak, and ISAP. TROPIC01 is also protected against side-channel attacks, voltage glitching attacks, laser attacks, and Focused Ion Beam (FIB) reverse engineering. The security protocol of this Integrated Circuit (IC) makes it well suited for HW Root of Trust (ROT) applications.

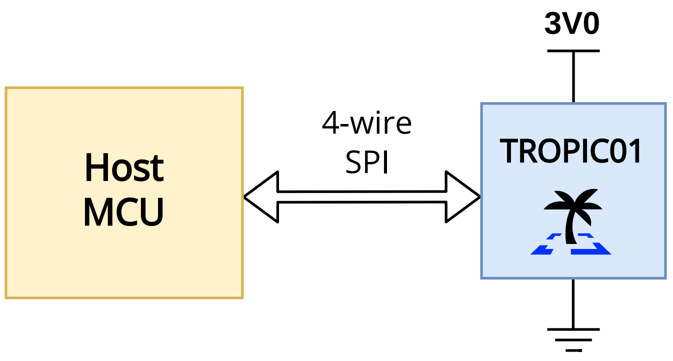


Figure 1: Typical application

For more information visit:
www.tropicsquare.com
Tropic Square s.r.o
Kundratka 2359/17A, Praha 8
180 00, Czech Republic

Contents

1 INTRODUCTION 3

2 BLOCK DIAGRAM 5

3 PIN DESCRIPTION 6

4 GLOSSARY 8

5 SECURITY POLICY 9

5.1 Identity Attestation 9
5.2 Mutual Authentication 10
5.3 Secure Communication 10
5.4 Security Lifecycle Management 10
5.5 User Access Privileges 13
5.6 Transparency and Auditability . 13
5.7 Assets Security 13
5.8 Attack Resilience 15

6 CHIP MODES 17

6.1 Device Start-up 18
6.2 Idle Mode 18
6.3 Secure Channel Mode 18
6.4 Sleep Mode 18
6.5 Alarm Mode 19
6.6 Chip Restart 19

7 FUNCTIONAL DESCRIPTION 21

7.1 Non-Volatile Memory 21
7.2 Chip Configuration 24
7.3 Communication Protocol 25
7.4 Secure Channel Session 38
7.5 User Access Privileges 42
7.6 User Data Access 46
7.7 Elliptic Curve Cryptography . . . 46
7.8 X.509 Certificate 51
7.9 PIN Verification 51
7.10 Monotonic Counters 52
7.11 Ping 52
7.12 Random Number Generation . . 53

8 ARCHITECTURE OVERVIEW 54

8.1 FW architecture 54
8.2 Secure Design Part 55
8.3 Security Provisioning 55
8.4 Manufacturing Test Modes . . . 55

9 ELECTRICAL SPECIFICATIONS 56

9.1 SPI Interface Timing 57

10 ABSOLUTE MAXIMUM RATINGS 58

11 TYPICAL APPLICATION 59

12 PACKAGE DESCRIPTION 60

13 REVISION HISTORY 62

14 REFERENCES 66

1 INTRODUCTION

This datasheet describes TROPIC01 functionality and consists of the following major sections:

- Security Policy – Describes the security features important for users.
- Chip Modes – Describes TROPIC01 modes and their behavior.
- Functional Description – Describes the chip operation.
- Architecture Overview – Briefly describes the internal architecture.

This datasheet familiarizes users with TROPIC01. To better understand the rationale of TROPIC01's security design, use section 5 as a supplement for sections 6 and 7.

For implementation details, refer to Functional Specification [2].

Audience

This datasheet is intended for system architects, embedded developers, software and hardware developers, application developers, cryptographers, and security certification authorities.

Related Documents

The following documents are available:

- *TROPIC01 – User API* [1].
- *TROPIC01 – Functional Specification* [2].
- *TROPIC01 – Memory Map* [3].
- *TROPIC01 – PIN Verification* [5].

Feedback

We intend for the datasheet content to be as clear as possible to better assist your use of TROPIC01. Your feedback is welcomed so we can continue to improve our documentation.

Please forward any questions or comments regarding the datasheet to inquiry@tropicsquare.com

Functional Overview

TROPIC01 is a cryptographic coprocessor IC with non-volatile memory and a set of cryptographic functions.

The chip is controlled by a Host MCU via a communication protocol that consists of the following layers:

- Physical (L1)
- Data Link (L2)
- Secure Session (L3)

L3 communication is encrypted and is only available once TROPIC01 and the Host MCU establish a Secure Channel Session.

A Secure Channel Session is established via an NOISE protocol [8] based procedure called a Secure Channel Handshake.

The Host MCU configures TROPIC01 via the Configuration Objects (CO) stored in TROPIC01's non-volatile memory. The Host MCU sends commands to TROPIC01 and reads the chip's results.

For detailed explanations of:

- Communication Protocol – see section 7.3
- Secure Channel Session – see section 7.4
- Secure Channel Handshake – see section 7.4.1
- Chip Configuration – see section 7.2
- Chip Modes – see section 5

2 BLOCK DIAGRAM

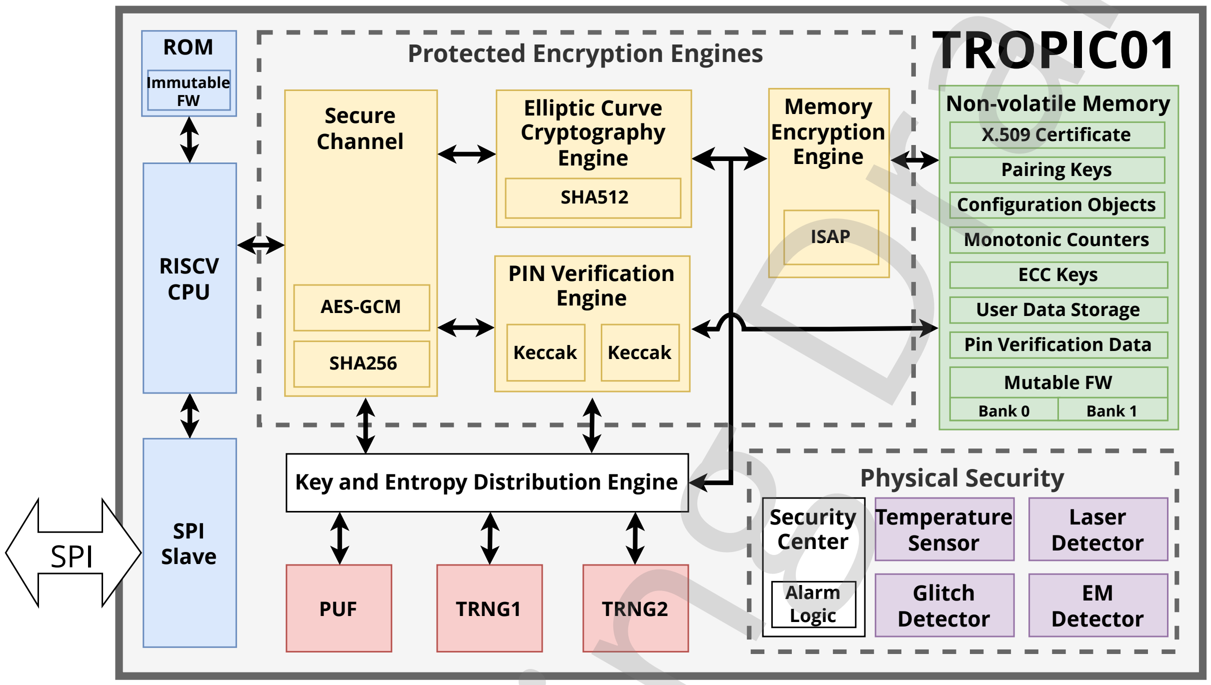


Figure 2: Functional block diagram

3 PIN DESCRIPTION

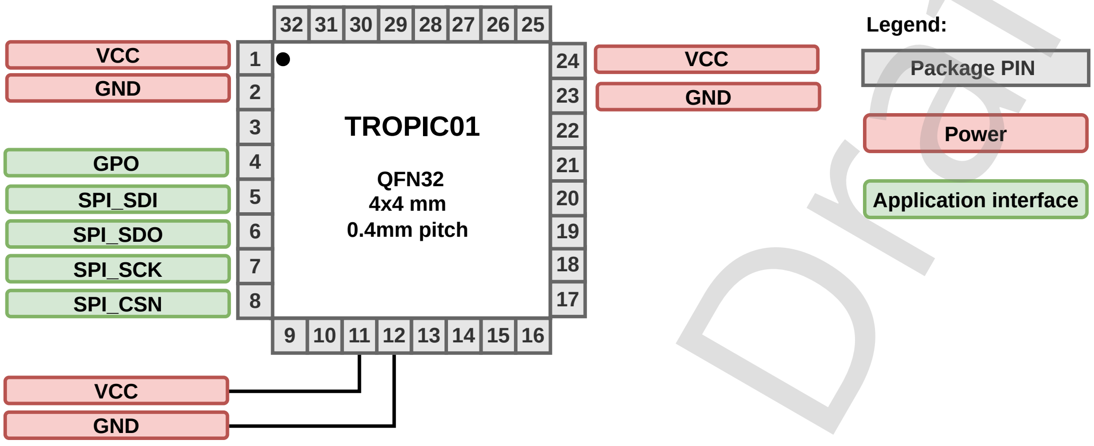


Figure 3: QFN32 package pinout

Note

NC (Non Connected) pins have an alternative functionality for testing during manufacturing and debugging during chip development. Such functionalities are irreversibly disabled during TROPIC01 production for security purposes.

Note

For the description of NC pin functionalities and timing, refer to [2].

Table 1: TROPIC01 pinout

Pin #	Name	Pin type	Description
1	VCC	Power	VCC
2	GND	Ground	Ground
3	NC	—	Do not connect/use
4	GPO	Digital Output	General Purpose Output (Reserved for future use)
5	SPI_SDI	Digital Input	Serial Data Input
6	SPI_SDO	Digital Output	Serial Data Output
7	SPI_SCK	Digital Input	Serial Clock
8	SPI_CSN	Digital Input	Chip Select
9–10	NC	—	Do not connect/use
11	VCC	Power	Power
12	GND	Ground	Ground
13	NC	—	Do not connect/use
14	NC	—	Do not connect/use
15–16	NC	—	Do not connect/use
23	GND	Ground	Ground
24	VCC	Power	Power
25–32	NC	—	Do not connect/use

4 GLOSSARY

Abbreviated Terms

IC	Integrated Circuit
HW	Hardware
FW	Firmware
CO	Configuration Object
I-Config	Irreversible Configuration
R-Config	Reversible Configuration
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
MCU	Microcontroller
MSB	Most Significant Bit
OTP	One Time Programmable
PUF	Physically Unclonable Function
ROT	Root of Trust
TRNG	True Random Number Generator

Functions

$\cdot$	Elliptic Curve Scalar Point Multiplication
$\|\|$	Byte-wise string concatenation

Variables

S_{TPUB}	TROPIC01 X25519 public key for a Secure Channel Handshake
S_{TPRIV}	TROPIC01 X25519 private key for a Secure Channel Handshake
S_{HiPRIV}	X25519 private key of the Host MCU to execute a Secure Channel Handshake on Pairing Key slot i
S_{HiPUB}	X25519 public key of the Host MCU to execute a Secure Channel Handshake on Pairing Key slot i
K_{FXA}, K_{FXB}	Keys for the PIN verification procedure
T_{HCMD}	L3 Command packet Authentication Tag computed and sent by the Host MCU
T_{TCMD}	L3 Command packet Authentication Tag received by TROPIC01
T_{TRES}	L3 Result packet Authentication Tag computed and sent by TROPIC01
T_{HRES}	L3 Result packet Authentication Result Tag received by the Host MCU
n_T	Secure Channel Session Nonce as kept by TROPIC01
n_H	Secure Channel Session Nonce as kept by the Host MCU
k_{CMD}	Encryption and decryption key for L3 Command packet
k_{RES}	Encryption and decryption key for L3 Result packet
h	Secure Channel Handshake hash
E_{TPUB}	TROPIC01 ephemeral public key
E_{TPRIV}	TROPIC01 ephemeral private key
E_{HPUB}	Host MCU ephemeral public key
E_{HPRIV}	Host MCU ephemeral private key
k_{AUTH}	Handshake Authentication key during a Secure Channel Handshake
d	ECDSA private key 1
w	ECDSA private key 2
s	EdDSA private key 1
$prefix$	EdDSA private key 2
A	ECDSA/EdDSA public key
(r, s)	ECDSA/EdDSA signature result

5 SECURITY POLICY

TROPIC01's security policy consists of the major features:

- Identity Attestation
- Mutual Authentication
- Secure Communication
- Security Lifecycle Management
- User Access Privileges
- Transparency and Auditability
- Assets Security
- Attack Resilience

TROPIC01's security implements multiple domains. Each domain requires different protection mechanisms as described in Table 2.

5.1 Identity Attestation

TROPIC01 can prove its identity and authenticity upon the Host MCU's request. The Host MCU can obtain certificate data from TROPIC01's non-volatile memory.

TROPIC01's non-volatile memory (section 7.1) stores the X.509 certificate signed by Tropic Square during manufacturing. The X.509 certificate contains TROPIC01's X25519 public key (S_{TPUB}).

Tropic Square is the author of S_{TPRIV}, S_{TPUB} key pair.

Meeting the following conditions guarantees the TROPIC01 instance was manufactured by Tropic Square:

- S_{TPUB} from the X.509 certificate is identical to the public key of the Tropic Square Certification Authority.
- Successful signature verification of TROPIC01's X.509 certificate

For details on how to read the X.509 certificate, see section 7.8.

Table 2: Security domains

Security Domain	Attack path	Protection mechanisms
Cryptographic Algorithms	Mathematical crypto-analysis	Standardized algorithms with defined key strength: ▪ Keccak ▪ AES-GCM ▪ SHA256 and SHA512 ▪ Elliptic Curve Cryptography with P-256, Curve25519, and Ed25519 curves Internal entropy source and reproducible instance specific randomness (section 5.8.5).
Algorithm implementation	Side-channel attacks	Algorithm protection mechanisms (section 5.8.1). HW implementations of encryption algorithms. Implementation protection mechanisms (section 5.8).
Physical implementation	Fault injection attacks	On-chip attack sensors (section 5.8). Redundancy, error correction codes, and separate HW datapaths for important cryptographic objects (section 5.8).
Physical implementation	Invasive attacks	Active shield and fuzzing design approaches (section 5.8.4).

5.2 Mutual Authentication

Mutual Authentication is the process where TROPIC01 and the Host MCU verify each other's identity. TROPIC01 and the Host MCU can then establish a secure communication.

TROPIC01 enforces mutual authentication using NOISE protocol during a Secure Channel Handshake.

TROPIC01 can contain up to 4 X25519 public keys (S_{HiPUB}). Each S_{HiPUB} is stored in a single Pairing Key slot (i) to allow authentication with different remote Host MCUs.

Establishing a Secure Channel Session with an X25519 key from Pairing Key slot i guarantees:

- The TROPIC01 instance is authentic – Is is the owner of S_{TPRIV} used to generate the S_{TPUB} in the X.509 certificate.
- The Host MCU is authentic – It is the owner of S_{HiPRIV} used to generate the S_{HiPUB} in the Pairing Key slot i.

For details on the pairing keys, see section 7.1.

For details on the Secure Channel Handshake, see section 7.4.1.

5.3 Secure Communication

Secure communication is established once the Host MCU and TROPIC01 successfully verify each other's identities.

TROPIC01 implements a multi-layered communication protocol (section 7.3) to create a Secure Channel Session for sensitive data communication.

When the Host MCU and TROPIC01 are in a Secure Channel Session, communication is encrypted and authenticated in both directions:

- Host MCU to TROPIC01 communication is verified by the Host MCU's Authentication Tag.
- TROPIC01 to Host MCU communication is verified by TROPIC01's Authentication Tag.

The architecture of Secure Channel Session and L3 Communication prevents a third party from performing a "man-in-the-middle" attack (i.e. An attacker cannot intercept the communication between TROPIC01 and the Host MCU to observe or alter the plaintext data).

The Host MCU is the author of the S_{HiPRIV}, S_{HiPUB} key pair needed for secure communication.

Thus, an attacker cannot force TROPIC01 to execute an action that can only be requested by the Host MCU.

For details on L3 communication, see section 7.3.4

For details on the Secure Channel Session, see section 7.4.

5.4 Security Lifecycle Management

Lifecycle Management is the process of changing TROPIC01's configuration during the chip's lifetime. As a TROPIC01 instance moves across the supply chain (i.e. moves from manufacturer to customer, then final application), each party can change the chip's configuration.

Changing TROPIC01's configuration can restrict future holders of that chip from executing certain actions.

For further explanation of Chip Configuration, see section 7.2.

At the time of manufacturing, Tropic Square configures every TROPIC01 with:

- A signed X.509 certificate (section 7.8).
- An X25519 public key (S_{H0PUB}) generated from S_{H0PRIV}.

Tropic Square provides S_{H0PRIV} to TROPIC01 customers. The customer can then establish a Secure Channel Session with Pairing Key slot 0 and configure TROPIC01.

The customer can generate their own X25519 key pair (S_{HiPRIV} and S_{HiPUB}) and store S_{HiPUB} into Pairing Key slots 1-3 (section 7.5.3).

If the customer stores their own S_{HiPUB} in TROPIC01, only that customer (i.e. the only party that knows S_{HiPRIV}) can establish a Secure Channel Session via Pairing Key slot i.

The customer can invalidate S_{HiPUB} (including S_{H0PUB}) and thereby prevent the owner of S_{HiPRIV} from establishing a Secure Channel Session.

5.4.1 Chip Ownership

Chip Ownership is a property of each TROPIC01 as the chip passes between parties during its lifetime. Each Pairing Key slot i has a separate ownership.

Consider the following roles:

- Author – The party who generates the S_{HiPUB}, S_{HiPRIV} key pair for Pairing Key slot i.
- Owner – The party who has control over the content of Pairing Key slots and the actions executed by other parties.
- User – The party who has access to S_{HiPRIV} and can establish a Secure Channel Session via Pairing Key slot i.

Note

A TROPIC01 user can be in one or more of the three roles at the same time.

A TROPIC01 owner has the following guarantees:

- They can control whether TROPIC01 can establish a Secure Channel Session with any other party.
- They can restrict other parties from executing certain actions.
- They can transfer ownership to another party by delivering the TROPIC01 instance and S_{HiPRIV}.
- They cannot execute actions that a previous owner restricted.
- They can decide whether the User is able to restrict Owner's User Access Privileges.

The principle of Security Lifecycle Management and Chip Ownership is demonstrated in Figure 4.

For details on handling Pairing Key slots, see section 7.5.3.

For details on restricting actions available for each Pairing Key slot, see section 5.5.

For a use case of ownership transfer, see section 7.5.4.

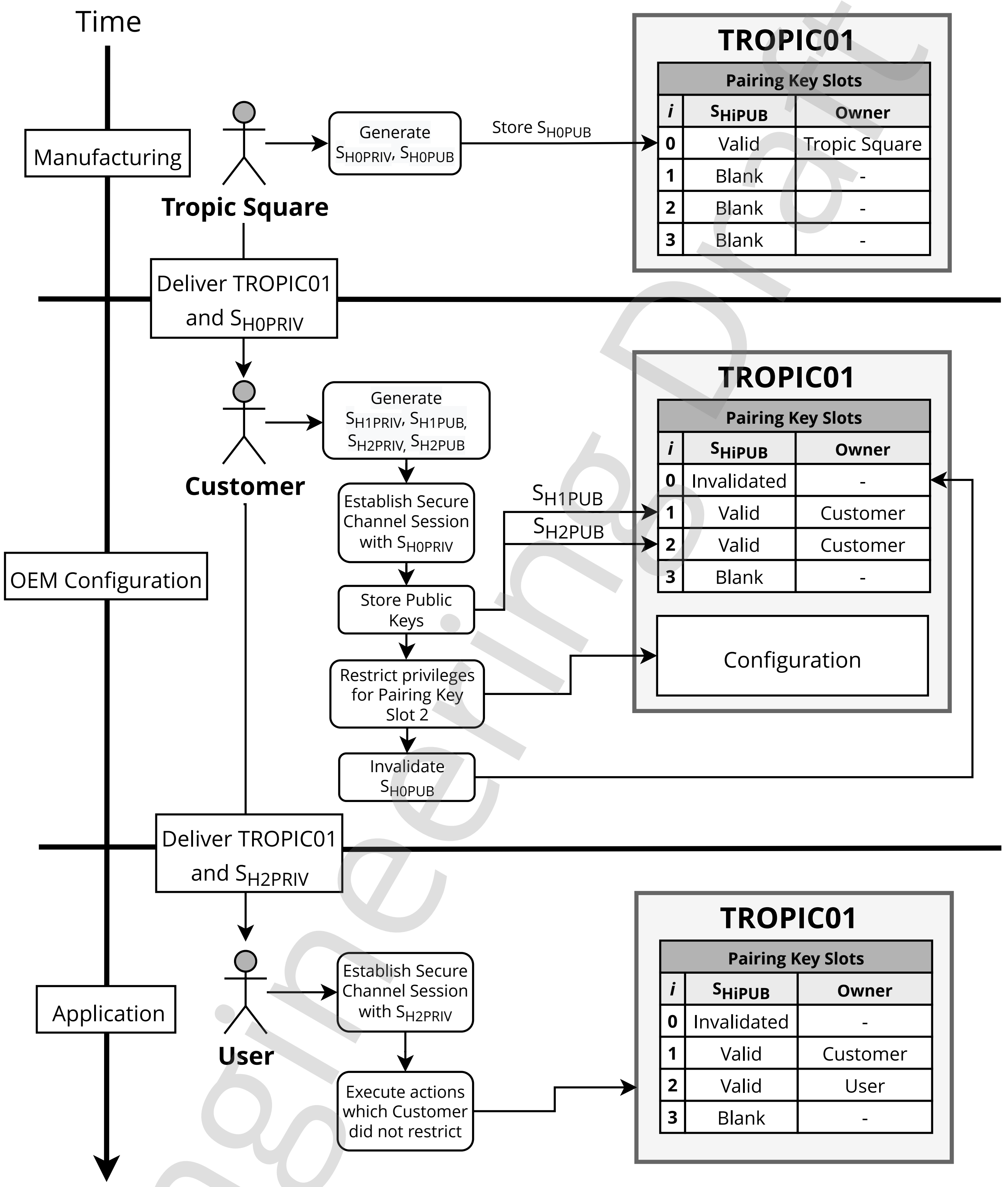


Figure 4: Security Lifecycle Management

5.5 User Access Privileges

User Access Privileges allows the current chip holder to restrict future holders from executing certain actions based on the selected Pairing Key slot.

There are two types of restrictions on User Access Privileges:

- Reversible: The restricted action can be allowed again.
- Irreversible: The restricted action cannot be allowed again.

Reversible and Irreversible restrictions on User Access Privileges are set by R-Config (section 7.2.2) and I-Config (section 7.2.1) configurations.

For details on Chip Configuration, see section 7.2.

The option to irreversibly restrict another party from executing certain actions is a building block of Chip Ownership (section 5.4.1).

For further explanation on User Access Privileges, see section 7.5.

5.6 Transparency and Auditability

TROPIC01 is a secure element designed with the security implementations transparent and accessible for review. The goal of transparency is to abide by Kerckhoffs's principle and guarantee maximum auditability of the system.

The following materials are available for the general public:

- Documentation of the internal chip architecture and digital logic.
- Documentation of test modes and test mode locking.
- Documentation of the RTL (Register Transfer Level) of TROPIC01's digital logic.
- Firmware source code running on the internal RISC-V CPU (section 8.1.2).

Note

TROPIC01 analog components (TRNG, PUF, Flash and OTP macros, and Power Supply) and TROPIC01 layout are not publicly available due to NDAs with the component suppliers and the semiconductor fabricator.

5.7 Assets Security

User assets are:

- General purpose data
- ECC keys
- Pairing keys
- Chip configuration

TROPIC01 has a non-volatile memory that securely stores user assets.

5.7.1 Memory Types

TROPIC01 contains two non-volatile memories:

- R-Memory
- I-Memory

R-Memory is a non-secure flash memory where cells containing 1 and 0 can be distinguished by a scanning electron microscope (SEM).

R-Memory provides the following protection mechanisms:

- Slots with sensitive data are encrypted by an ISAP algorithm with 256-bit key strength and a unique per chip encryption key (section 5.7.3).
- Each chip's address bus is randomly scrambled.

- Error correction code (Single Error Correction and Double Error Detection).

I-Memory is an anti-fuse One Time Programmable (OTP) memory. OTP memory cells containing 0 and 1 cannot be easilly distinguished by optical or SEM inspection, while known reverse-engineering methods are highly expensive.

The I-Memory address bus is also uniquely scrambled in each TROPIC01 instance to increase security of stored assets.

For more details on both non-volatile memories, see section 7.1.

5.7.2 Memory Security

All user assets within the R-Memory and within the I-Memory reside in different memory partitions.

TROPIC01 allows only authorized hardware blocks the access to each memory partition.

To maximize TROPIC01's security, it is physically impossible for unauthorized parties to read certain parts of non-volatile memories.

Dedicated HW datapaths are implemented for each party accessing TROPIC01's memories (this implementation protects against redirected bus transfers caused by glitching attacks).

TROPIC01's encryption engines need to obtain data from the non-volatile memories in order to perform calculations.

To maximize TROPIC01's security, the encryption engines:

- Autonomously fetch necessary data.
- Hold that data in volatile memory elements during the calculation.
- Discard the data when the calculation finishes.

5.7.3 R-Memory Encryption

TROPIC01 encrypts two R-Memory partitions:

- ECC Key
  - Encrypted when programmed during processing of an ***ECC_Key_Generate/Store*** L3 Command packet.
  - Decrypted when read by the Elliptic Curve Encryption engine during processing of an ***ECDSA_Sign*** or ***EDDSA_Sign*** L3 Command packet.
- User Data
  - Encrypted when programmed during processing of an ***R_Mem_Data_Write*** L3 Command packet.
  - Decrypted when read during processing of an ***R_Mem_Data_Read*** L3 Command packet.

For more details on R-Memory, see section 7.1.1.

TROPIC01 configures R-Memory encryption and automatically encrypts R-Memory content.

The Host MCU does not need to encrypt R-memory content. The Host MCU can additionally encrypt its own data before sending to TROPIC01.

TROPIC01 uses an ISAP encryption scheme based on Keccak permutation (400-bit state with 128-bit security level) [16].

TROPIC01 derives encryption keys to encrypt the R-Memory from a device specific fingerprint given by PUF (section 5.8.5).

For more information about the encryption scheme, refer to [2].

5.7.4 MAC-and-Destroy

When TROPIC01 executes a MAC-and-Destroy sequence (section 7.9), the chip interleaves two calculations of Keccak permutation [5].

TROPIC01 uses the K_{FXA} and K_{FXB} keys, and unique per-instance randomness from PUF to guarantee the security of the MAC-and-Destroy sequence.

Both TROPIC01's Keccak engines are separated physical implementations. Both Keccak engines are protected by a multi-threshold sharing scheme [17].

5.7.5 Signature Uniqueness

When TROPIC01 calculates ECDSA and EdDSA signatures (section 7.7), the Secure Channel Session hash (h) and Secure Channel Session Nonce (n) are used to:

- Diversify each signature
- Avoid nonce-reuse attacks ([23])

Refer to the TMAC function inputs in Tables 19 and 18.

For details on the Secure Channel Session, see section 7.4

For the description of the Secure Channel Session Nonce, see Table 15.

5.8 Attack Resilience

TROPIC01 features a wide range of attack sensors and countermeasures to prevent:

- Side-channel attacks
- Glitching attacks
- Laser and EM attacks
- Micro-probing attacks

If TROPIC01's sensors detect an attack, TROPIC01 may move to Alarm Mode if configured to do so (section 6.5).

TROPIC01 implements a Physically Unclonable Function (PUF) for further security measures against physically invasive attacks.

Thus, the protection mechanism knowledge of one TROPIC01 instance cannot be applied to other TROPIC01 instances.

5.8.1 Side-channel Attacks

TROPIC01 protects its cryptographic engines from side-channel attacks by implementing:

- Keccak engines (used in PIN verification) that are protected by threshold sharing across three shares. Each share is masked by a random number generated by TRNG1. For further description of Keccak protection mechanisms, refer to [17].
- An AES-GCM engine (used during Secure Channel Handshake and encrypted communication) that uses first order masking from [26] and [27]
- An Elliptic Curve Cryptography engine that uses multiple randomization and masking methods (e.g. group scalar randomization [18] or point randomization).
- Cryptographic engines that perform computations in constant cycle counts. For further explanation of attacks exploiting non-constant time operations, refer to [25].
- An internal clock source that generates a clock with frequency dithering and random clock cycle stealing.
- Register pre-charge to a random value to mitigate Hamming distance leakage.

5.8.2 Glitching Attacks

TROPIC01 implements glitching attack protections:

- Glitch detectors.
- Voltage monitors.
- Multiple-majority encoding on important internal storage elements.
- FSM state encodings with protection (parity, Hamming distance, etc.).
- Error Correction Code protection on important data structures within the design.

5.8.3 Laser and EM Attacks

TROPIC01 implements protection against laser and EM attacks:

- Laser detectors
- EM field detectors
- Temperature sensor

5.8.4 Micro-probing Attacks

TROPIC01 implements protection against micro-probing attacks:

- Active metal shield.
- Scrambling of memory address busses on the R-Memory and I-Memory.
- TROPIC01 test modes are disabled by a set of dedicated e-Fuses [2] and values in I-Memory dedicated addresses.
- Multi-bit information to disable and gate test features (This approach avoids a single point of failure in case of microscopic modifications by Focused Ion Beam (FIB)).

5.8.5 Physically Unclonable Function (PUF)

PUF provides a unique random 256-bit per-instance value.

The PUF value is:

- Diversified by a 32-bit challenge (controlled by TROPIC01).
- Unique and reproducible for each chip instance.
- Non-reproducible across other chip instances.

TROPIC01 uses PUF output to:

- Diversify protections in each TROPIC01 instance like memory address scrambling or PIN verification keys.

6 CHIP MODES

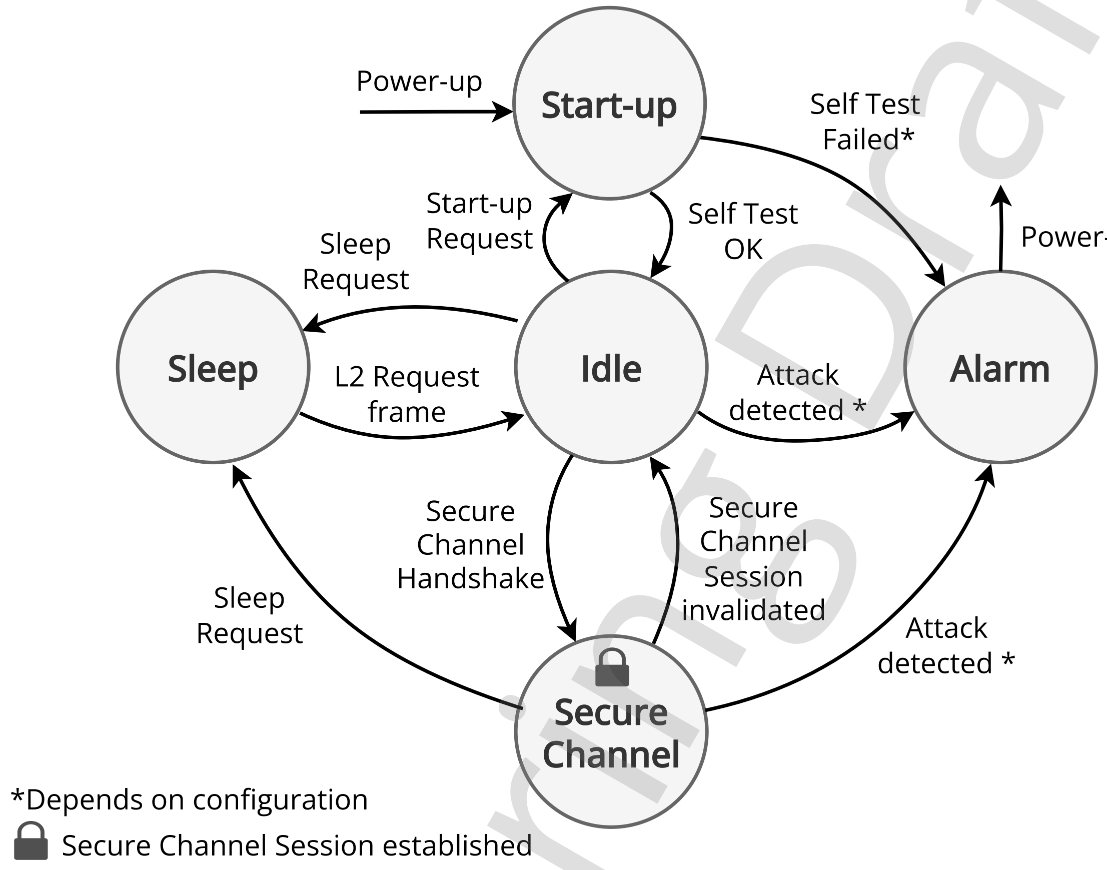


Figure 5: Chip Modes

Table 3: TROPIC01 Chip Modes

Mode	Description
Start-up	The transient state after power-up. TROPIC01 automatically performs internal self-tests.
Idle	Encrypted communication via a Secure Channel Session is not possible. Only L2 Request frames are accepted (7.3.3).
Secure Channel	A Secure Channel Session is established between the Host MCU and TROPIC01. L2 Request frames and L3 Command packets (7.3.4) are accepted.
Sleep	Reduced TROPIC01 power consumption. Sending an L2 Request frame moves TROPIC01 to Idle Mode.
Alarm	An attack attempt detected. Any L2 Request frame or L3 Command packet is rejected.

6.1 Device Start-up

The system integrating TROPIC01 shall guarantee the VCC supply voltage ramps up to the final value within the $T_{VCCRAMPUP}$ time.

After the VCC power-up, TROPIC01 performs an internal start-up sequence.

If configured, the start-up sequence executes a set of internal self tests (refer to **CFG_START_UP** CO in [1]).

If the start-up tests fail, TROPIC01 moves to Alarm Mode.

If the start-up sequence finishes without error, TROPIC01 moves to Idle Mode.

After power-up TROPIC01 responds with **CHIP_STATUS[READY]**=0 until TROPIC01 is ready to process an L2 Request frame.

TROPIC01 ignores any L2 Request frames sent before it has responded with **CHIP_STATUS[READY]**=1 for the first time.

During start-up, TROPIC01 reads its configuration from its non-volatile memories (section 7.2) and applies the configuration.

TROPIC01 stays in Start-up mode, if it is restarted by Maintenance restart. For more detail, see section 6.6.

6.2 Idle Mode

In Idle Mode, TROPIC01 accepts L2 Request frames and responds with L2 Response frames (section 7.3.3).

The Host MCU may initialize a Secure Channel Session by sending a ***Handshake_Req*** L2 Request frame.

TROPIC01 moves to Secure Channel Mode upon completion of a Secure Channel Handshake (section 7.4.1).

6.3 Secure Channel Mode

In Secure Channel Mode, a Secure Channel Session (section 7.4) is established between the Host MCU and TROPIC01.

TROPIC01 accepts L2 Request frames and L3 Command packets (section 7.3.4) while Secure Channel Mode is active.

6.4 Sleep Mode

In Sleep Mode, TROPIC01 power consumption is reduced.

TROPIC01 enters Sleep mode when it receives a ***Sleep_Req*** L2 Request frame with **SLEEP_KIND**=**SLEEP_MODE**.

The Host MCU can disable the effect of ***Sleep_Req*** in the **CFG_SLEEP_MODE** CO.

When entering Sleep Mode, TROPIC01 invalidates the current Secure Channel Session.

If TROPIC01 receives any L2 Request frames during Sleep Mode, TROPIC01 moves to Idle Mode and processes the L2 Request frame.

TROPIC01 discards all security sensitive volatile data when entering Sleep Mode.

In Sleep Mode, TROPIC01 does not detect attack attempts and does not move to Alarm Mode.

If TROPIC01 transfers to Sleep Mode from Secure Channel Mode, the current Secure Channel Session is invalidated.

Thus, the Host MCU needs to execute a new Secure Channel Handshake to execute any further L3 Commands.

For details on executing a Secure Channel Handshake, see section 7.4.1.

6.5 Alarm Mode

In Alarm Mode, TROPIC01 remains in Alarm mode until the next power-cycle.

When TROPIC01 is in Alarm Mode, TROPIC01:

- Transmits **CHIP_STATUS[ALARM]** = 1 (if configured).
- Ignores any incoming L2 Request frames.

When entering Alarm Mode, TROPIC01 invalidates the current Secure Channel Session (section 7.4).

The Host MCU configures events that trigger an Alarm Mode in the **CFG_SENSORS** CO [1].

If configured to do so, TROPIC01 moves to Alarm Mode when its internal sensors detect an attack attempt (section 5.8):

- TROPIC01 stops any L2 Request or L3 Command processing.
- Alarm Mode is activated within the T_{ATTACK} time.

Figure 6 shows an example of an attack during an L2 Request frame processing.

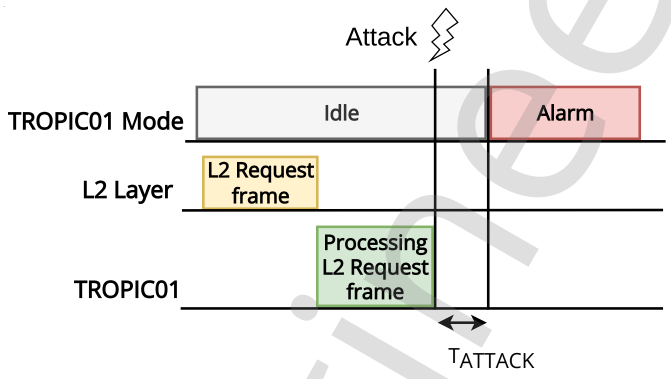


Figure 6: Alarm Mode timing

6.6 Chip Restart

The Host MCU can restart TROPIC01 by sending ***Startup_Req*** L2 Request frame. TROPIC01 supports following types of restart:

- Regular restart
- Maintenance restart

Upon regular restart, TROPIC01 executes self-tests in Start-up Mode and transfers to Idle Mode.

Upon maintenance restart, TROPIC01 stays in the Start-up Mode, until next power-cycle or next regular restart.

Maintenance restart can be disabled by setting **CFG_STARTUP[MAINTENANCE_ENA]** CO to 0.

Note

TROPIC01 responds to ***Startup_Req*** by a regular L2 Response frame. TROPIC01 restarts only after Host MCU reads this L2 Response frame.

When TROPIC01 is in Start-up mode due to maintenance restart it:

- Does not execute Secure Channel Handshake.
- Upon receiving ***Handshake_Req***, respond with **STATUS**=**HSK_ERR**
- TROPIC01 internal FW can be updated. See section 8.1 for details.

7 FUNCTIONAL DESCRIPTION

TROPIC01 is a secure element with on-chip non-volatile memory and HW acceleration of cryptographic algorithms. TROPIC01 has the following functionalities:

- Communicates with the Host MCU via encrypted commands (section 7.3.4).
- Sets up an encrypted session between TROPIC01 and the Host MCU via Secure Channel Handshake (section 7.4.1).
- Securely stores general purpose user data (section 7.6).
- Stores ECC key(s) provided by the Host MCU (section 7.7.1).
- Generates and stores ECC key(s) for the ECDSA/EdDSA algorithms (section 7.7.1).
- Signs messages from the Host MCU with an EdDSA/ECDSA signature algorithm (section 7.7.4 and 7.7.5).
- Provides the X.509 certificate signed by Tropic Square (section 7.8).
- Executes secure and atomic PIN verification from the Host MCU's PIN, (section 7.9).
- Initializes/Updates/Provides value of Monotonic Counters (section 7.10).
- Proves its authenticity via the X.509 certificate and succesful Secure Session establishment.

TROPIC01 is controlled by the Host MCU via a multi-layer serial protocol that consists of Commands and Requests sent by the Host MCU to TROPIC01. The communication protocol is further described in section 7.3.

7.1 Non-Volatile Memory

TROPIC01 contains the following non-volatile memories:

- Reversible Memory (R-Memory)
- Irreversible Memory (I-Memory)

Both non-volatile memories have a hardwired memory layout as shown in Tables 4 and 5.

The Host MCU has access to most of the memory partitions via dedicated L2 Request frames or L3 Command packets (section 7.3).

Certain TROPIC01 encryption engines have access to some partitions (e.g. MAC-and-Destroy data partition is available exclusively to PIN verification engine, see section 7.9).

7.1.1 R-Memory

R-Memory consists of slots that are allocated to partitions as shown in Figure 4. Each slot has size of 512 Bytes. R-Memory supports the following operations:

- Read (R): Returns value from a memory location.
- Write (W): Changes memory location from 1 to 0.
- Erase (E): Changes memory location from 0 to 1.

R-Memory contains 1s in all memory locations after manufacturing. The Host MCU can write 1s to 0s by sending L3 Command packets (section 7.3.4).

After the R-Memory is written, the memory shall be erased before being programmed again.

The Host MCU shall erase each slot of the R-Memory's User Data partition before writing (section 7.6).

For details on the R-Memory's security aspects, see section 5.

Table 4: R-Memory Partitions

Object	User Slots	User Access	Encrypted	Description
R-Config	1	R/W/E	No	Reversible chip configuration. Access with a ***R_Config_**** L3 Command packet (section 7.2.2).
ECC Keys	32	R/W/E	Yes	Keys for Elliptic Curve Cryptography (ECDSA/EdDSA) algorithms. Accessible with a ***ECC_Key_**** L3 Command packets. Only the public part is accessible for read.
User Data	512	R/W/E	Yes	General purpose storage for user data. Access with a ***R_Mem_Data_**** L3 Command packet (7.6).
Monotonic Counters	16	R/W/E	No	Data for Monotonic Counters (7.10). Accessible by TROPIC01 while processing ***MCounter_**** L3 Command packets.
CPU FW	2	W/E	No	CPU application firmware. (24KB). Description (512B) + signature (512B). Access with a ***Mutable_FW_**** L2 Request frame in Start-Up Mode.
ECC FW	2	W/E	No	ECC engine application firmware. (12KB). Description (512B) + signature (512B). Access with a ***Mutable_FW_**** L2 Request frame in Start-Up Mode.
MAC-and-Destroy data	128	—	No	Used by the PIN verification engine when TROPIC01 processes a ***MAC_And_Destroy*** L3 Command packet (7.9).

Table 5: I-Memory Partitions

Object	Size [Bytes]	Slots [X]	User access	Description
I-Config	256	1	R/W	Irreversible chip Configuration (7.2.1) Access with an ***I_Config_**** L3 Command packet.
X.509 certificate	1024	1	R	The X.509 certificate signed by Tropic Square (7.8). Contains the S_{TPUB} used during a Secure Channel Handshake. To access, send a ***Get_Info_Req*** L2 Request frame.
S_{TPRIV}	32	1	—	The X25519 private key used during a Secure Channel Handshake (7.4.1).
S_{TPUB}	32	1	—	The X25519 public key used during a Secure Channel Handshake. It is the same S_{TPUB} as in the X.509 certificate. It is used by cryptography engines during the Secure Channel Handshake.
S_{HiPUB}	32	4	R/W	The Host's X25519 public key used during a Secure Channel Handshake. Accessible with a ***Pairing_Key_**** L3 Command packet (7.5.3).
K_{FX}	16	2	—	Parts of keys used by the PIN verification engine when TROPIC01 processes a ***MAC_And_Destroy*** L3 Command packet (7.9). There are two key slots: K_{FXA} and K_{FXB}.

7.1.2 I-Memory

I-Memory consists of partitions shown in Table 5.

Each I-Memory partition consists of slots.

If a single partition has multiple slots, the partition contains the same kind of item (e.g. 4 S_{HiPUB} are stored in 4 I-Memory slots).

I-Memory contains 1s in all memory locations after manufacturing and supports the following operations:

- Read (R) – Return a value from a memory location.
- Write (W) – Change a memory location from 1 to 0.

If the Host MCU writes a memory location in the I-Memory, it is impossible to revert the value back to 1.

Certain I-Memory partitions can be written in via dedicated L3 Command packets.

The other partitions are written in only during TROPIC01 manufacturing and are read-only for the Host MCU.

Partitions with crucial information for TROPIC01's security are inaccessible to the Host MCU (e.g. S_{TPRIV}).

For details on the I-Memory security aspects, see section 5.7.1.

7.2 Chip Configuration

The Host MCU configures TROPIC01 by modifying the Configuration Objects (CO).

TROPIC01 provides the following COs:

- I-Config – **I**rreversible **Config**uration stored in the I-Memory.
- R-Config – **R**eversible **Config**uration stored in the R-Memory.

Both configuration objects have equal memory layouts defined in [1].

The actual configuration TROPIC01 uses is determined by the bit-wise *AND* of the configuration values defined in I-Config and in R-Config.

For an example on how final configuration is determined, see section 7.2.4.

Note

When the Host MCU changes CO content, TROPIC01 applies the new configuration in the next power-cycle.

7.2.1 I-Config

I-Config COs are stored in TROPIC01's I-Memory.

All bits of I-Config COs are 1 after manufacturing. Thus, all configurations are enabled by default.

If the Host MCU sends an ***I_Config_Write*** L3 Command packet (section 7.3.4), 1 is irreversibly written to 0 in the single bit of a single I-Config CO.

When writing bits in I-Config COs, the Host MCU selects:

- The CO address offset by the **ADDRESS** field of ***I_Config_Write***.
- The CO bit by the **BIT_INDEX** field of ***I_Config_Write***.

To read I-Config COs, the Host MCU sends an ***I_Config_Read*** L3 Command packet. The Host MCU selects the CO address to read by the **ADDRESS** field of ***I_Config_Read***.

TROPIC01 then returns an L3 Result packet with one CO (32-bit word) at a time.

For the description of L3 Commands for I-Config manipulation, refer to [1].

7.2.2 R-Config

R-Config COs are stored in a single slot of TROPIC01's R-Memory.

All bits of R-Config COs are 1 after manufacturing. Thus, all configuration options are enabled.

When the Host MCU sends a ***R_Config_Write*** L3 Command packet (section 7.3.4), the Host MCU writes a single CO at a time. The Host MCU selects the CO address to write by the **ADDRESS** field of ***R_Config_Write***.

When the Host MCU sends a ***R_Config_Read*** L3 Command packet, the Host MCU reads one R-Config CO (32-bit word) given by the **ADDRESS** field.

To erase all of R-Config content (i.e. change all bits of all COs to 1), the Host MCU sends a ***R_Config_Erase*** L3 Command packet.

For the description of L3 Commands for R-Config, refer to [1].

Note

R-Config has to be first erased before new content is writen by ***R_Config_Write*** L3 Command packet. Thus, the Host MCU should first read whole R-Config, perform desired bit changes and then write it back to TROPIC01.

7.2.3 Chip Configuration Privileges

The commands to modify R-Config and I-Config COs are subject to User Access Privileges (section 7.5).

The Host MCU can permanently disable the option to read and modify R-Config and/or I-Config COs by forbidding the processing of ***R_Config_**** and/or ***I_Config_**** L3 Command packets.

7.2.4 Sleep Mode Configuration Example

TROPIC01 moves to Sleep Mode after receiving a ***Sleep_Req*** L2 Request frame (if configured to do so by the **CFG_SLEEP_MODE[SLEEP_MODE_EN]** CO).

The actual value of **CFG_SLEEP_MODE[SLEEP_MODE_EN]** is given by merging the **CFG_SLEEP_MODE** COs from R-Config and I-Config.

An example scenario:

- I-Config: **CFG_SLEEP_MODE[SLEEP_MODE_EN]**=0x0

  Sleep_Req effect is disabled.

- R-Config: **CFG_SLEEP_MODE[SLEEP_MODE_EN]**=0x1

  Sleep_Req effect is enabled.

The final applied configuration is:

$$0x0 \wedge 0x1 = 0x0 \tag{1}$$

Thus, ***Sleep_Req*** effect is disabled. I-Config forbids TROPIC01 from transferring to Sleep Mode upon receiving the L2 Request frame.

7.3 Communication Protocol

The Host MCU communicates with TROPIC01 via a multi-layer serial protocol that consists of the following layers:

- Physical Layer (L1)
- Data Link Layer (L2)
- Secure Session Layer (L3)

The protocol layers are described in Table 6.

The Host MCU and TROPIC01 communicates by request-and-response.

The Host MCU initiates communication and

Table 6: TROPIC01 Protocol Layers

Short Name	Full Name	Description
L1	Physical Layer	4-wire SPI slave interface. The Host MCU executes L1 transfers.
L2	Data Link Layer	For non-secure information about TROPIC01 and L3 communication set up. Guarantees the integrity of transmitted and received data. Communication on L2 is not encrypted.
L3	Secure Session Layer	For application functionality of TROPIC01. Requires an established a Secure Channel Session between the Host MCU and TROPIC01 for communication (See section 7.4). Communication on L3 is encrypted.

directs TROPIC01 to execute an action. Then, TROPIC01 returns the result of the action.

TROPIC01 does not initiate communication by itself. TROPIC01 responds to actions requested by the Host MCU.

TROPIC01 executes only one action at a time.

The Host MCU shall wait until TROPIC01 finishes processing the previously requested action before requesting the next action.

Table 7: L1 **CHIP_STATUS** Byte

Bit	Name	Description
0	**READY**	TROPIC01 is ready to receive L2 Request frame or L3 Command packet (7.3.3 and 7.3.4).
1	**ALARM**	TROPIC01 is in Alarm Mode (6.5).
2	**START**	TROPIC01 is in Start-up Mode (6.1).
3 – 7	—	Undefined.

7.3.1 Byte order

TROPIC01 uses Little-Endian byte order to communicate with Host MCU. TROPIC01 uses Little-Endian order:

- For all multi-byte protocol fields.
- On all protocol layers.

Note

If exceptions to the little-endian rule exist, they are described next to the relevant protocol field in [1].

7.3.2 L1 Layer

A transfer is a unit of communication on the L1 Layer.

L1 communication uses a 4-wire SPI interface and supports **CPOL** = 0 and **CPHA** = 0. Transfers are byte oriented, with MSB sent first.

When TROPIC01 receives the first byte of L1 transfer, it simultaneously sends a **CHIP_STATUS** byte as defined in Table 7 and shown in Figure 8.

7.3.3 L2 Layer

A frame is a unit of communication on the L2 Layer. The structure of an L2 frame is organized into fields.

For L2 communication, the Host MCU sends an L2 Request frame and TROPIC01 returns an L2 Response frame, as shown in Figure 8.

Tables 8 and 9 show the structure of L2 Request frames and L2 Response frames.

The value of the first byte (the **REQ_ID** field) determines the type of L2 Request frame:

- **REQ_ID** != ***Get_Response*** (0xAA):
  The Host MCU is sending an L2 Request frame.
- **REQ_ID** == ***Get_Response***:
  The Host MCU is requesting an L2 Response frame from TROPIC01.

For more valid **REQ_ID** values, see Table 10.

When the Host MCU sends an L2 Request frame:

1. TROPIC01 processes the L2 Request frame.
2. The Host MCU shall send **REQ_ID** == ***Get_Response***.
3. TROPIC01 returns an L2 Response frame.

Figure 8(b) shows the L2 communication structure.

If L2 Request frame processing is in progress, TROPIC01 responds with L2 Response frames containing **STATUS**=**NO_RESP** until the end of the L1 transfer, as shown in Figure 8 (c).

If the Host MCU attempts to read an L2 Response frame without sending any L2 Request frame previously, TROPIC01 also responds with **STATUS**=**NO_RESP** until the end of the L1 transfer.

If during an L1 transfer TROPIC01 becomes ready to send an L2 Response frame (i.e. TROPIC01 finishes processing the previous L2 Request frame), TROPIC01 continues to respond with **STATUS**=**NO_RESP** until the end of the L1 transfer.

If the Host MCU detects **STATUS**=**NO_RESP** during the second byte of L1 transfer, the Host MCU shall finish the transfer.

L2 Request Frame Types

The types of L2 Request frames supported by TROPIC01 are summarized in Table 10.

For detailed explanation of all L2 Request frames and corresponding TROPIC01 L2 Response frames, refer to [1].

STATUS Response Field

The content and length of an L2 Response frame depends on the **STATUS** field.

The value of **STATUS** determines if TROPIC01 sends further bytes, as shown in Figure 7.

Valid **STATUS** field values of an L2 Response frame are shown in Table 11.

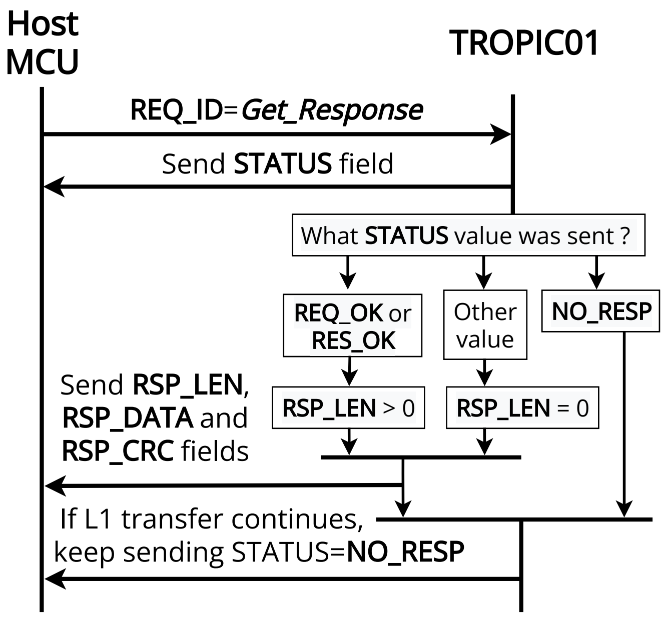

Host MCU
TROPIC01
REQ_ID=Get_Response
Send STATUS field
What STATUS value was sent ?
REQ_OK or RES_OK
Other value
NO_RESP
RSP_LEN > 0
RSP_LEN = 0
Send RSP_LEN, RSP_DATA and RSP_CRC fields
If L1 transfer continues, keep sending STATUS=NO_RESP

Figure 7: L2 Response frame logic

Note

When the Host MCU reads the **STATUS** field as other than **NO_RESP**, the Host MCU shall read the whole L2 Response frame within the current L1 transfer.

The Host MCU shall not split the reading of the L2 Response frame into multiple L1 transfers.

Cyclic Redundancy Check

The Host MCU and TROPIC01 calculate the value of **REQ_CRC** (Table 8) and **RSP_CRC** (Table 9) when:

- The Host MCU sends an L2 Request frame.
- TROPIC01 responds with an L2 Response frame.

CRC calculation on the L2 layer has the following properties:

- CRC16 Algorithm.
- 0x8005 Polynomial.
- 0x0000 Initialization vector.

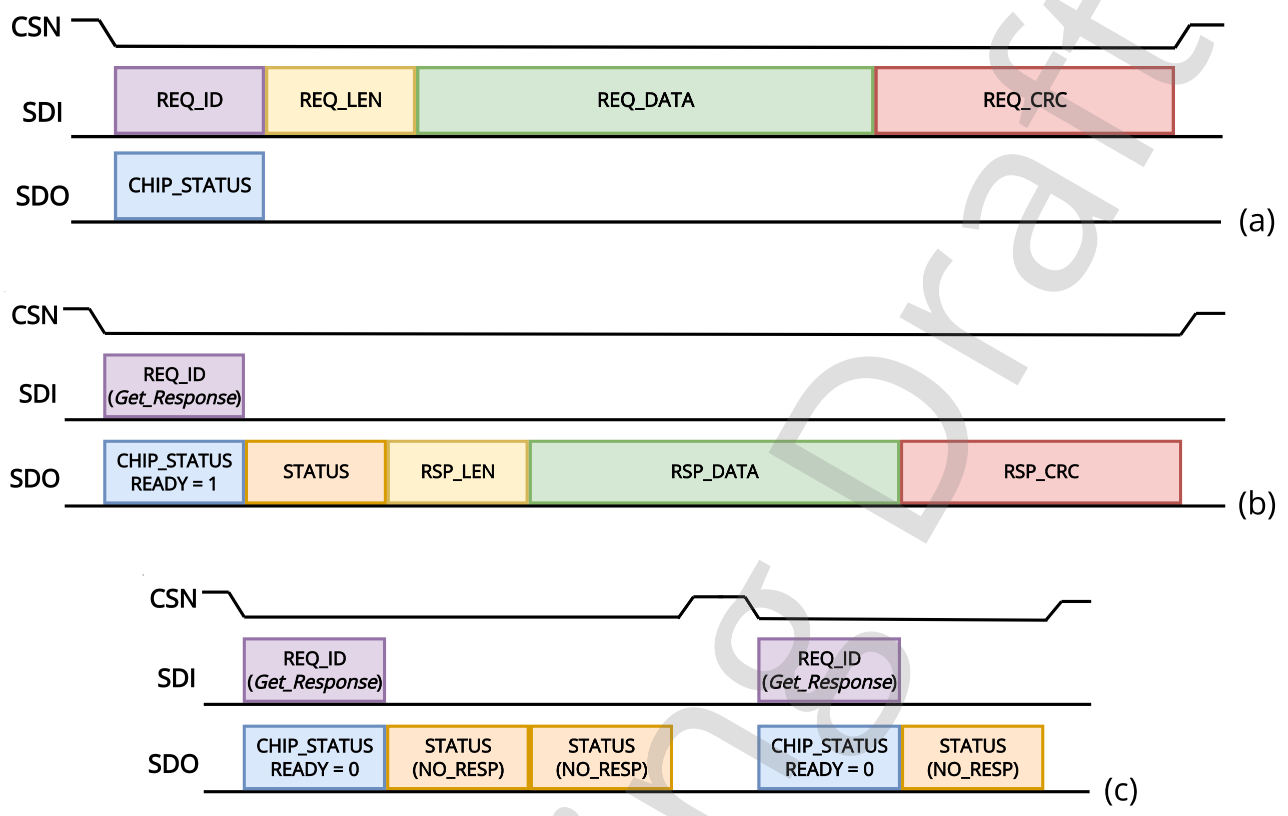


Figure 8: L2 frames structure

Table 8: L2 Request frame structure

Field	Size [bytes]	Field name	Description
REQ_ID	1	ID	Request Identifier.
REQ_LEN	1	Length	Request data length in bytes.
REQ_DATA	**REQ_LEN**	Data	Request data.
REQ_CRC	2	Checksum	Request CRC checksum calculated from the **REQ_ID**, **REQ_LEN**, and **REQ_DATA** fields.

Table 9: L2 Response frame structure

Field	Size [bytes]	Field name	Description
STATUS	1	Status	Response status code.
RSP_LEN	1	Length	Response data length in bytes.
RSP_DATA	**RSP_LEN**	Data	Response data.
RSP_CRC	2	Checksum	Response CRC checksum calculated from the **STATUS**, **RSP_LEN**, and **RSP_DATA** fields.

When TROPIC01 receives an L2 Request frame, TROPIC01 compares the **REQ_CRC** field with the internally calculated CRC value.

If CRC mismatches, TROPIC01:

- Ignores the L2 Request frame
- Responds with **STATUS**=**CRC_ERR** upon the Host MCU's next attempt to read L2 Response frame.

When the Host MCU reads the L2 Response frame, the Host MCU shall compare the **RSP_CRC** field with the internally calculated CRC value.

If CRC mismatches, the Host MCU shall ignore the L2 Response frame.

Resending L2 Response frame

The Host MCU might ask TROPIC01 to retransmit the last L2 Response frame by sending a ***Resend_Req*** L2 Request frame.

If TROPIC01 receives ***Resend_Req***, TROPIC01 resends the previous L2 Response frame.

Figure 9 shows how TROPIC01 handles ***Resend_Req***.

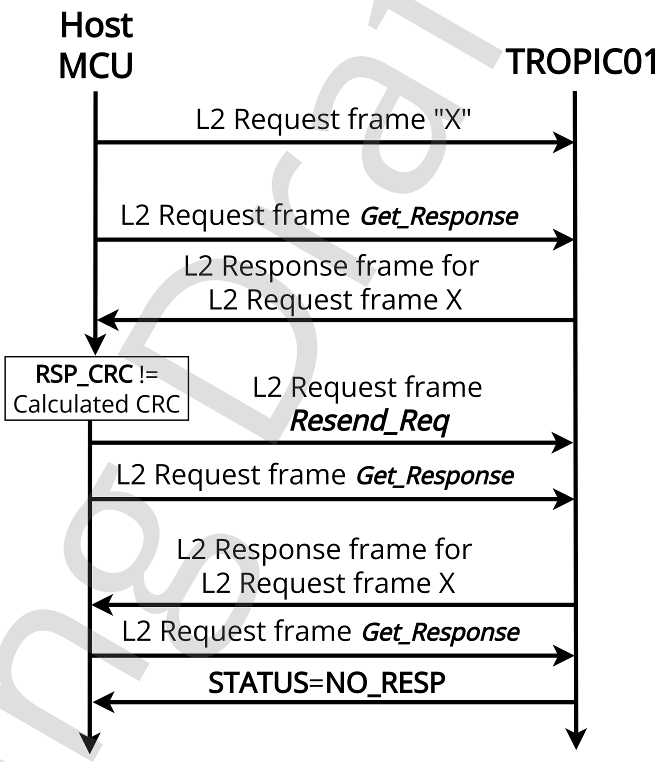

Host MCU
TROPIC01
L2 Request frame "X"
L2 Request frame Get_Response
L2 Response frame for L2 Request frame X
RSP_CRC != Calculated CRC
L2 Request frame Resend_Req
L2 Request frame Get_Response
L2 Response frame for L2 Request frame X
L2 Request frame Get_Response
STATUS=NO_RESP

Figure 9: ***Resend_Req*** handling

Table 10: Valid **REQ_ID** field values

REQ_ID	Value	Description
Get_Response	0xAA	Request to read the L2 Response frame.
Get_Info_Req	See [1]	Request to obtain public information from TROPIC01.
Handshake_Req	See [1]	Request to start a new Secure Channel Handshake. (For details on the Secure Channel Handshake. See section 7.4.1).
Encrypted_Cmd_Req	See [1]	Request to process an L3 Command packet in the **REQ_DATA** field of the L2 Request frame.
Encrypted_Session_Abt	See [1]	Request to abort current Secure Channel Session and execution of L3 Command packet command (TROPIC01 moves to Idle Mode).
Resend_Req	See [1]	Request to resend the previous L2 Response frame.
Sleep_Req	See [1]	Request to go to Sleep Mode. See sections 6.4.
Startup_Req	See [1]	Request restart. See section 6.6
Mutable_FW_Update_Req	See [1]	Request to write a mutable firmware to R-Memory.
Mutable_FW_Erase_Req	See [1]	Request to erase a mutable firmware from R-Memory.

Table 11: Valid **STATUS** field values

STATUS	Value	Description
REQ_OK	0x01	TROPIC01 has received, checked for CRC validity, and processed the L2 Request frame. The **RSP_DATA** field (of the L2 Response Frame) contains the results of the action invoked from the L2 Request frame.
RES_OK	0x02	TROPIC01 has processed the L3 Command packet. The **RSP_DATA** field contains an L3 Result packet.
REQ_CONT	0x03	Similar to ***REQ_OK***, see section 7.3.7.
RES_CONT	0x04	Similar to ***REQ_CONT*** when split result is transmitted from TROPIC01 to the Host MCU.
RESP_DISABLED	0x78	The L2 Request frame is disabled and can't be executed.
HSK_ERR	0x79	Secure Channel Handshake failed and Secure Channel Session is not established (e.g Pairing Key slot from the **PKEY_INDEX** field of a ***Encrypted_Cmd_Req*** L2 Request frame has an invalid X25519 public key).
NO_SESSION	0x7A	TROPIC01 is not in Secure Channel Mode and the Host MCU has sent ***Encrypted_Cmd_Req***. TROPIC01 ignores this L2 Request frame.
TAG_ERR	0x7B	Invalid L3 Command packet Authentication Tag. L3 Command packet is ignored. TROPIC01 invalidates the current Secure Channel Session and moves to Idle Mode. (For details on L3 Authentication Tags, see section 7.3.4).
CRC_ERR	0x7C	Incorrect CRC-16 checksum. TROPIC01 ignores the associated L2 Request frame.
UNKNOWN_REQ	0x7E	Unknown L2 Request frame (value of **REQ_ID**) received.
GEN_ERR	0x7F	Generic error (cannot be classified under other status codes).
NO_RESP	0xFF	No L2 Response frame available.

7.3.4 L3 Layer

A packet is a unit of communication on the L3 Layer.

L3 communication requires an established Secure Channel Session. Then, the Host MCU can communicate through sending L3 Command packets and TROPIC01 responds with L3 Result packets.

Communication is executed on an encrypted channel (Secure Channel) with strong forward secrecy based on a Noise Protocol Framework [8].

L3 communication has authenticated encryption where each encryption and decryption tag acts as a Message Authentication Code (MAC) [9].

For details on how to establish a Secure Channel Session, see section 7.4.1.

TROPIC01 ignores all of the Host MCU's L3 Command packets if there is no established Secure Channel Session.

If the Host MCU sends an ***Encrypted_Cmd_Req*** L2 Request frame without an active Secure Channel Session, TROPIC01 responds with **STATUS**=**NO_SESSION** in the L2 Response frame.

The Host MCU sends an L3 Command packet in the **REQ_DATA** field of the L2 Request frame and TROPIC01 returns an L3 Result packet in the **RSP_DATA** field of the L2 Response frame as shown in Figure 11.

Valid **RESULT** field values are defined in Table 14.

Figure 12 shows an example of L3 communication between the Host MCU and TROPIC01.

Command and Result Packets

L3 Command packets and L3 Result packets consist of an encrypted part and a plaintext part.

The sender encrypts part of the L3 packet and the receiver decrypts the same part of the packet before interpreting.

Table 12 shows L3 Command packet structure and descriptions.

Table 13 shows L3 Result packet structure and descriptions.

The Host MCU uses the **CMD_ID** value to distinguish the type of L3 Command packet.

The content of the decrypted **CMD_DATA** differs for each type of L3 Command packet.

For a further L3 Command packet description, refer to [1].

The structure of an L3 Command packet and an L3 Result packet in relation to L1 and L2 layers is shown in Figures 10 and 11.

Command and Result Packet Encryption

The Host MCU and TROPIC01 encrypt and decrypt L3 packets with AES256-GCM [10].

32 LSB bits of Initialization Vector (IV) for L3 packet encryption and decryption are given by the Secure Channel Session Nonce (n) of the current Secure Channel Session.

For details on the Secure Channel Session Nonce, see section 7.4 and Table 15.

The Host MCU and TROPIC01 use 256-bit keys (Table 15) for L3 packet encryption and decryption:

- k_{CMD} : The encryption key for L3 Command packets.
- k_{RES} : The encryption key for L3 Result packets.

Due to the Secure Channel Handshake, the Host MCU and TROPIC01 settle on equal values of k_{CMD} and k_{RES}.

Thus, the receiver is always able to decrypt incoming L3 packets.

Correct handling of Secure Channel Session Nonce by both parties guarantees correct L3 packet encryption and decryption.

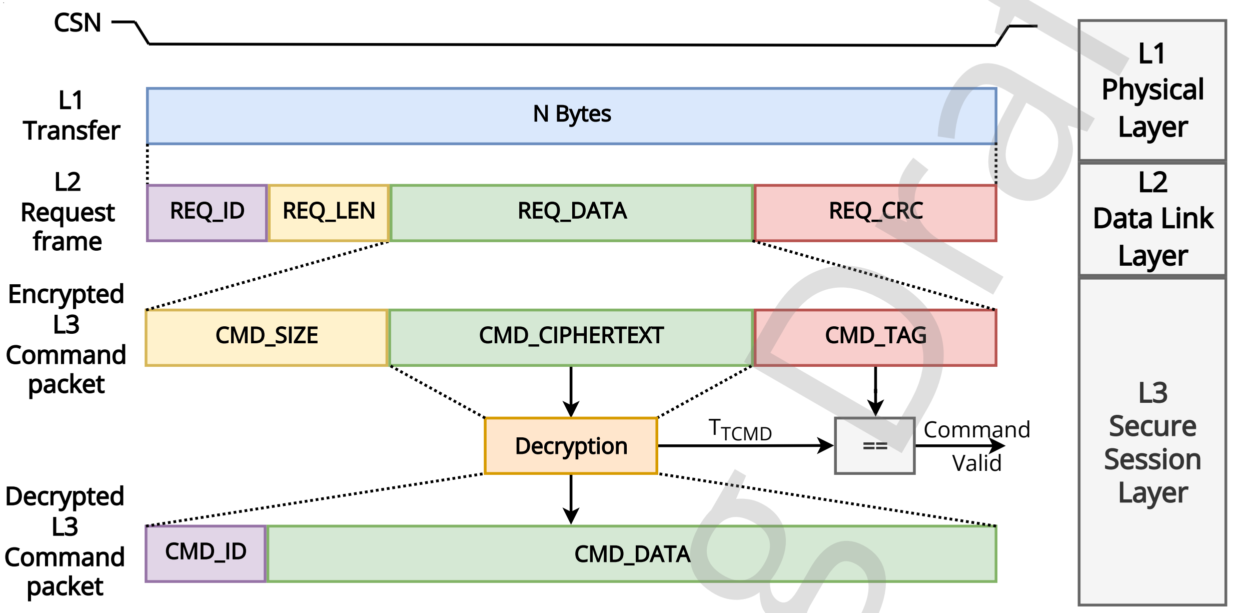


Figure 10: Nesting of communication [The Host MCU → TROPIC01]

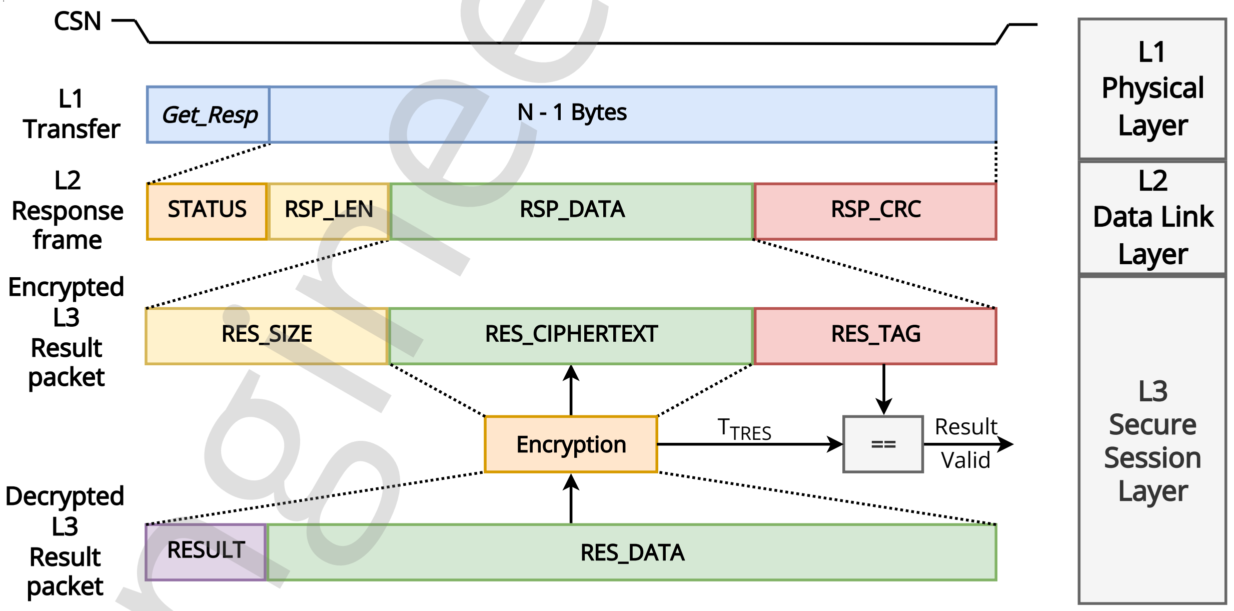


Figure 11: Nesting of communication [TROPIC01 → the Host MCU]

Table 12: L3 Command packet structure

Field	Size [Bytes]	Encr.	Description
CMD_SIZE	2	No	Command size in bytes.
CMD_CIPHERTEXT	**CMD_SIZE**	Yes	Command content.
CMD_TAG	16	No	Command Authentication tag.
Content of **CMD_CIPHERTEXT** after decryption			
CMD_ID	1	—	Command Identifier (refer to [1]).
CMD_DATA	**CMD_SIZE** - 1	—	Command Data.

Table 13: L3 Result packet structure

Field	Size [Bytes]	Encr.	Description
RES_SIZE	2	No	Result size in bytes.
RES_CIPHERTEXT	**RES_SIZE**	Yes	Result content.
RES_TAG	16	No	Result Authentication tag.
Content of **RES_CIPHERTEXT** before encryption			
RESULT	1	—	Result status indication.
RES_DATA	**RES_SIZE** - 1	—	Result data content.

Table 14: L3 **RESULT** field Values

Value	Name	Description
0xC3	**OK**	Command successfully executed.
0x3C	**FAIL**	Error during processing of the command.
0x1	**UNAUTHORIZED**	Insufficient User Access Privileges, see section 7.5.
0x2	**INVALID_CMD**	Unknown L3 Command packet (Invalid **CMD_ID**) .
—	—	Other values are Command specific defined in [1].

L3 Communication Example

Figure 12 shows an example of L3 Layer communication between the Host MCU and TROPIC01.

L3 communication flow from the Host MCU to TROPIC01:

1. The Host MCU encrypts **CMD_ID**, **CMD_DATA** of an L3 Command packet to create **CMD_CIPHERTEXT** and Authentication Tag (T_{HCMD}).

2. The Host MCU sends an ***Encrypted_Cmd_Req*** L2 Request frame to TROPIC01 and puts an L3 Command packet (**CMD_SIZE**, **CMD_CIPHERTEXT** and **CMD_TAG** = T_{HCMD}). in the **REQ_DATA** field of ***Encrypted_Cmd_Req***.

3. TROPIC01 performs a CRC on the incoming ***Encrypted_Cmd_Req***. If the CRC value is correct, TROPIC01 responds with **STATUS**=**REQ_OK** in the L2 Response frame. Otherwise, it responds with **STATUS**=**CRC_ERR**.

4. TROPIC01 interprets the **REQ_DATA** field of ***Encrypted_Cmd_Req*** as an L3 Command packet. It decrypts **CMD_CIPHERTEXT** and obtains the L3 Command packet and Authentication tag (T_{TCMD}).

5. If T_{TCMD} is identical to the **CMD_TAG** from the decrypted text, TROPIC01 executes the action invoked by the L3 Command packet. Otherwise, TROPIC01 drops the L3 Command packet, responds with **STATUS**=**TAG_ERR**, and invalidates the current Secure Channel Session.

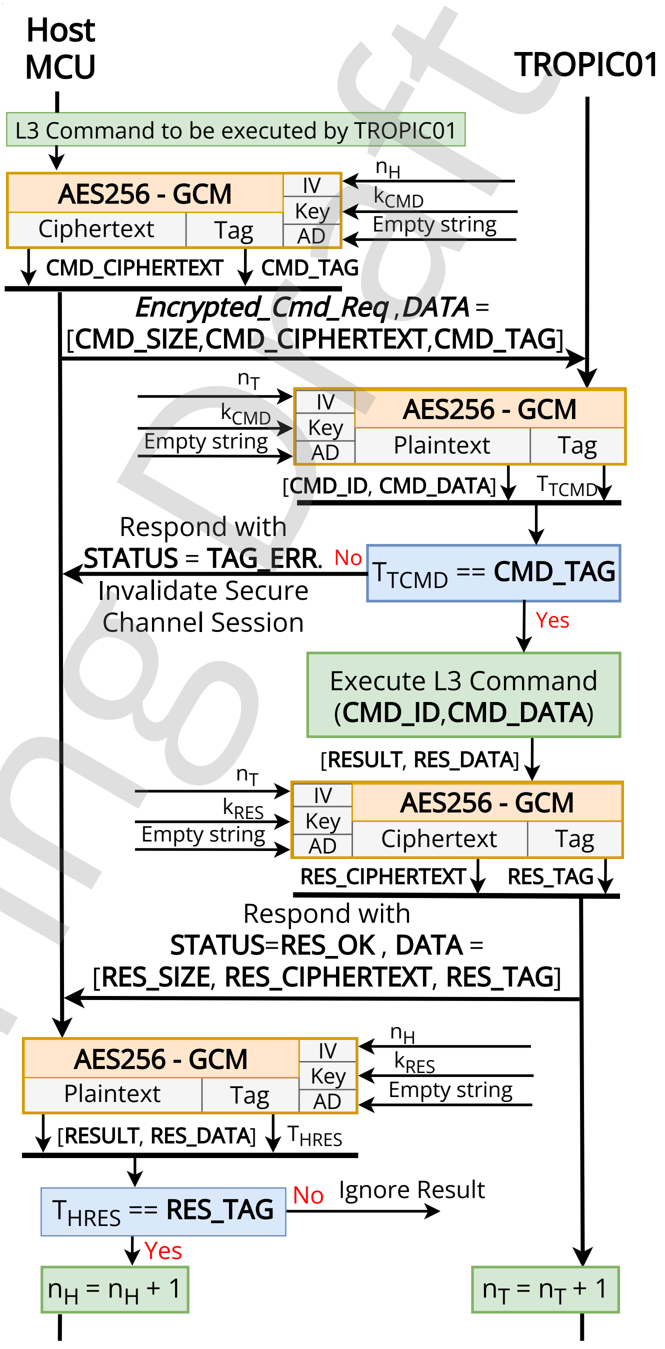


Figure 12: L3 Command packet processing

Note

All L3 communication requires an active Secure Channel Session. For details on establishing a Secure Channel Session, see section 7.4.

When TROPIC01 executes an action invoked by an L3 Command packet, TROPIC01 returns a L3 Result packet to the Host MCU.

L3 communication flow from TROPIC01 to the Host MCU:

1. TROPIC01 executes the action invoked by the L3 Command packet from the Host MCU. TROPIC01 encrypts **RESULT**, **RES_DATA** of an L3 Result packet to create **RES_CIPHERTEXT** and Authentication Tag (T_{TRES}).
2. When the Host MCU sends a ***Get_Response*** L2 Request frame, TROPIC01 responds with an L2 Response frame including **STATUS**=**RES_OK** and an L3 Result packet (**RES_SIZE**, **RES_CIPHERTEXT** and **RES_TAG** = T_{TRES}) in the **RSP_DATA** field.
3. The Host MCU receives the L2 Response frame. If **STATUS**=**RES_OK**, the Host MCU knows **RSP_DATA** contains an L3 Result packet.
4. The Host MCU decrypts **RES_CIPHERTEXT** and obtains **RES_DATA** and Authentication Tag (T_{HRES}). If T_{HRES} is identical to **RES_TAG** from the decrypted text, the Host MCU can interpret the L3 Result packet. Otherwise, the Host MCU shall discard the L3 Result packet.

7.3.5 Control Flow Management

When an L2 Request frame or an L3 Command packet is being processed, TROPIC01 responds with **CHIP_STATUS[READY]** = 0 during the first byte of L1 transfer.

The Host MCU shall use **CHIP_STATUS[READY]** to synchronize with the end of the L3 Command packet processing before sending TROPIC01 another L3 Command packet.

If the Host MCU sends ***Encrypted_Cmd_Req*** and TROPIC01 processes it without error, TROPIC01 provides two L2 Response frames in the following order:

1. **STATUS**=**REQ_OK**, **RSP_LEN** = 1 (L2 Layer)
2. **STATUS**=**RES_OK** (L3 Layer)

These responses confirm a successful frame processing and a successful packet processing on the respective protocol layer.

This feature ensures each L3 Command packet chunk (within an L2 Request frame) is acknowledged, after a long L3 Command packet splits between multiple L2 Request frames.

Note

The Host MCU shall send a maximum of one L3 Command packet at a time and shall read the L3 Result packet from TROPIC01 before sending the next L3 Command packet.

7.3.6 Stopping L3 Command Execution

When the Host MCU sends an ***Encrypted_Session_Abt*** L2 Request frame, TROPIC01:

- Immediately stops processing the L3 Command packet (if any L3 Command packet is being processed).
- Erases all of its temporary storage elements holding security sensitive information (e.g. buffers, registers and RAM memories).
- Invalidates the current Secure Channel Session.

For further details of ***Encrypted_Session_Abt***, refer to [1].

7.3.7 Splitting L3 Command Packets

REQ_DATA and **RSP_DATA** fields are both limited to a maximum length of 252 bytes each. Upon sending **REQ_LEN** of 253, 254 and 255 TROPIC01 returns **STATUS**=**GEN_ERR**.

An L3 packet size might be longer (up to 4KB for an ***EDDSA_Sign*** L3 Command packet).

To support long L3 packets, the Host MCU can split the L3 Command packet between multiple ***Encrypted_Cmd_Req*** L2 Request frames.

When TROPIC01 receives multiple ***Encrypted_Cmd_Req*** L2 Request frames, TROPIC01 sends back L2 Response frame(s) with:

- **STATUS**=**REQ_CONT** to all but the last ***Encrypted_Cmd_Req***.
- **STATUS**=**REQ_OK** to the last ***Encrypted_Cmd_Req***.

When an L3 Command packet is split, the Host MCU shall send each chunk in an ***Encrypted_Cmd_Req*** L2 Request frame as shown in Figure 13.

TROPIC01 internally processes the L3 Command packet only when the last chunk is received (TROPIC01 expects incoming **CMD_SIZE** bytes).

If the Host MCU sends an ***Encrypted_Session_Abt*** L2 Request frame before all the chunks of the L3 Command packet are sent to TROPIC01, TROPIC01 flushes the part of the L3 Command that is already received.

Thus, TROPIC01 interprets the **REQ_DATA** field of the next ***Encrypted_Cmd_Req*** as containing a new L3 Command packet.

If an L3 Result packet does not fit into a single L2 Response frame, TROPIC01 responds with the L3 Result packet split into multiple up-to 128 byte long L2 Response frames.

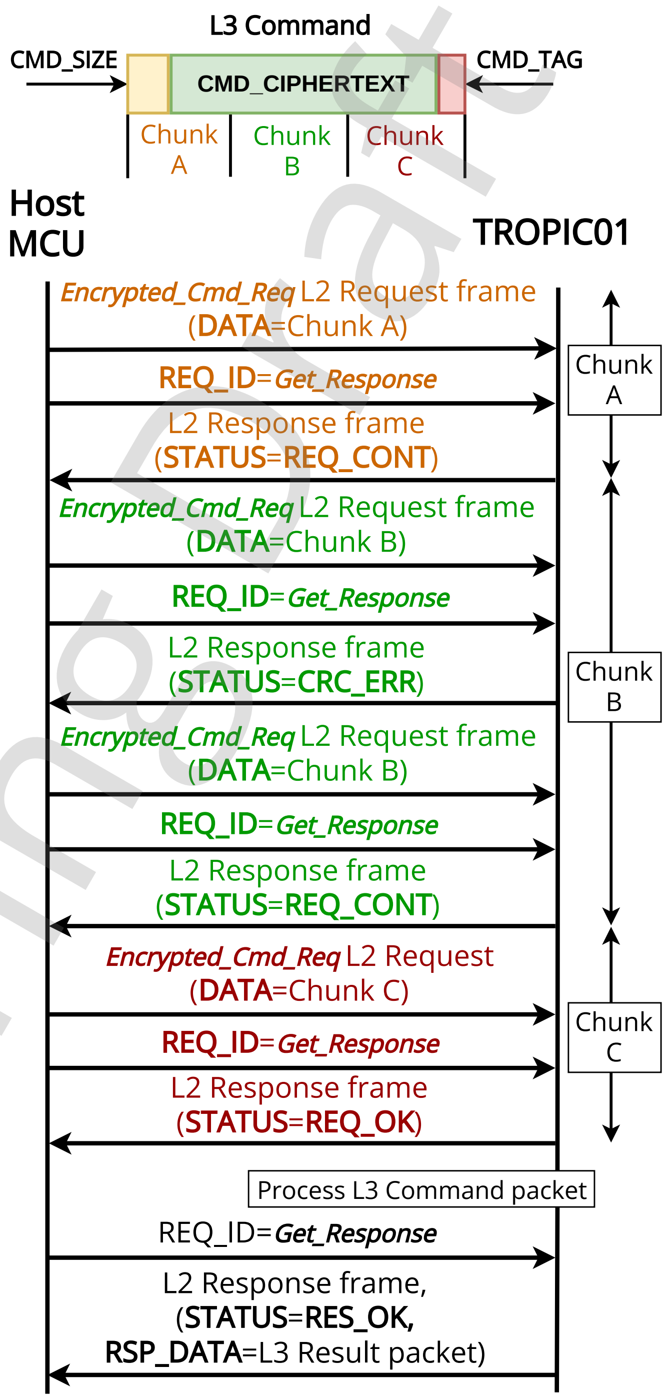

L3 Command
CMD_SIZE
CMD_CIPHERTEXT
CMD_TAG
Chunk A
Chunk B
Chunk C
Host MCU
TROPIC01
Encrypted_Cmd_Req L2 Request frame (DATA=Chunk A)
REQ_ID=*Get_Response*
L2 Response frame (STATUS=REQ_CONT)
Encrypted_Cmd_Req L2 Request frame (DATA=Chunk B)
REQ_ID=*Get_Response*
L2 Response frame (STATUS=CRC_ERR)
Encrypted_Cmd_Req L2 Request frame (DATA=Chunk B)
REQ_ID=*Get_Response*
L2 Response frame (STATUS=REQ_CONT)
Encrypted_Cmd_Req L2 Request (DATA=Chunk C)
REQ_ID=*Get_Response*
L2 Response frame (STATUS=REQ_OK)
Process L3 Command packet
REQ_ID=*Get_Response*
L2 Response frame, (STATUS=RES_OK, RSP_DATA=L3 Result packet)

Figure 13: Splitting an L3 Command packet to multiple L2 Request frames.

Note

STATUS=**CRC_ERR** in response to **Encrypted_Cmd_Req** with a part of an L3 Command packet means the Host MCU shall resend only the last chunk and not the whole L3 Command packet.

7.4 Secure Channel Session

Secure Channel Session is the protocol link used by the Host MCU and TROPIC01 to:

- Encrypt all L3 Layer communication
- Authenticate both parties

A Secure Channel Handshake establishes a Secure Channel Session as described in section 7.4.1.

Secure Channel Mode is activated after a successful Secure Channel Handshake.

In any other chip mode, TROPIC01 responds with **STATUS**=**NO_SESSION** in L2 Response frames to ***Encrypted_Cmd_Req*** L2 Request frames.

For more details on TROPIC01 chip modes, see section 6.

TROPIC01 exits Secure Channel Mode (and invalidates the current Secure Channel Session) when TROPIC01:

- Receives an L3 Command packet with a **CMD_TAG** not equal to T_{TCMD}.
- Receives an ***Encrypted_Session_Abt*** L2 Request frame (refer to [1]).
- Has a Secure Channel Session Nonce $n = 2^{32} - 1$.
- Detects a security breach.
- Detects alarm sensor activation – TROPIC01 might also move to Alarm Mode (section 6.5).
- Always after responding with **STATUS**=**GEN_ERR**.

Each Secure Channel Session is defined by a set of objects that are equal in the Host MCU and in TROPIC01. Table 15 describes these objects.

The ephemeral keys (E_{TPUB}, E_{TPRIV}, E_{HPUB}, E_{HPRIV}) make sure this set of objects is unique for each established Secure Channel Session.

Nonce Policy Compliance

To comply with The Noise Protocol Framework [8], the Host MCU and TROPIC01 post-increment their Secure Channel Session Nonce (n) value by 1 after encrypting or decrypting an L3 packet.

The Host MCU and TROPIC01 holds their own value of the Secure Channel Session Nonce (n).

As long as the Host MCU correctly sends L3 Command packets and handles L3 Result

Table 15: Secure Channel Session Objects

Object	Size [bytes]	Description
k_{CMD}	32	Key used by the Host MCU and TROPIC01. Encrypts and decrypts L3 Commands.
k_{RES}	32	Key used by the Host MCU and TROPIC01. Encrypts and decrypts L3 Results.
n	4	Secure Channel Session Nonce used during encryption and decryption of L3 Packets. Unique to each L3 Command-Result pair. Initialized to 0 upon completion of Secure Channel Handshake.
h	32	Secure Channel Session hash – Unique to each established Secure Channel Session.

packets, both the Host MCU and TROPIC01 maintain the same value of the Secure Channel Session Nonce.

The Host MCU shall handle its Secure Channel Session Nonce (n_H) by incrementing n_H by 1 after the L3 Result packet is received and decrypted.

TROPIC01 handles its Secure Channel Session Nonce (n_T) by incrementing n_T by 1 after processing an L3 Command packet and encrypting an L3 Result packet.

7.4.1 Secure Channel Handshake

Secure Channel Handshake is a series of actions the Host MCU and TROPIC01 executes to establish a new Secure Channel Session (section 7.4).

The Secure Channel Handshake consists of an Elliptic Curve Diffie-Hellman (ECDH) key exchange between the Host MCU and TROPIC01.

ECDH uses an X25519 function with a Curve25519 Elliptic Curve [11].

The X25519 Curve parameters are listed in Table 24.

To execute a Secure Channel Handshake, the Host MCU shall meet these conditions:

- Knows the X25519 private key (S_{HiPRIV}) for Pairing Key slot i, and keeps S_{HiPRIV} a secret.
- Knows the X25519 public key (S_{HiPUB}) corresponding to S_{HiPRIV}.
- Ensures TROPIC01's Pairing Key slot i (in the I-Memory) contains S_{HiPUB} (section 7.5.3).
- Knows TROPIC01's X25519 public key (S_{TPUB}).

For details on how to read TROPIC01's public key, see section 7.8

Note

The Host MCU never knows TROPIC01's X25519 private key (S_{TPRIV}). S_{TPRIV} is securely stored in TROPIC01.

Figure 14 shows the Secure Channel Handshake data flow between the Host MCU and TROPIC01.

To initiate a Secure Channel Handshake, the Host MCU sends a ***Handshake_Req*** L2 Request frame and selects a Pairing Key slot by the **PKEY_INDEX** field.

When TROPIC01 receives ***Handshake_Req***, TROPIC01 uses the **PKEY_INDEX** field to select the Host MCU's public key (S_{HiPUB}) stored in the corresponding Pairing Key slot of TROPIC01's I-Memory.

Establishing a Secure Channel Session with the Pairing Key from **PKEY_INDEX** authenticates both the Host MCU and TROPIC01.

TROPIC01 can be certain the Host MCU initiating the Secure Channel Session is the genuine author of the public key S_{HiPUB} and therefore the owner of the corresponding private key S_{HiPRIV}.

During Secure Channel Handshake, the Host MCU and TROPIC01:

- Generate ephemeral X25519 keys, see section 7.7.7.
- Calculate the SHA256 function [12].
- Calculate the X25519 function [11].
- Calculate the $HKDF$ function, as explained below in this subsection.

During a Secure Channel Handshake, TROPIC01 responds with **CHIP_STATUS[READY]**=0 and completes the Secure Channel Handshake within $T_{HANDSHAKE}$ time.

Protocol Name

The Secure Channel Handshake between the Host MCU and TROPIC01 uses a specific protocol string that is also an input for the Secure Channel Handshake:

```
protocol_name =
Noise_KK1_25519_AESGCM_SHA256\x00\x00\x00
```

Note

To align the protocol_name length to 32 bytes, 3 padding bytes of 0x00 are added at the end.

HMAC Key Derivation Function

The HMAC key derivation function [15] ($HKDF(ck, input, nouts)$) executes:

```
tmp = HMAC-SHA256(ck, input)
output_1 = HMAC-SHA256(tmp, 0x01)
if (nouts == 1)
  return output_1
else
  output_2 = HMAC-SHA256(tmp,
                output_1 || 0x02)
return (output_1, output_2)
```

Security Measures

To maximize security, when the Host MCU completes its part of a Secure Channel Handshake, the Host MCU shall erase the ck and k_{AUTH} values computed during the Secure Channel Handshake.

When TROPIC01 completes a Secure Channel Handshake, TROPIC01 erases the ck and k_{AUTH} values computed during the Secure Channel Handshake.

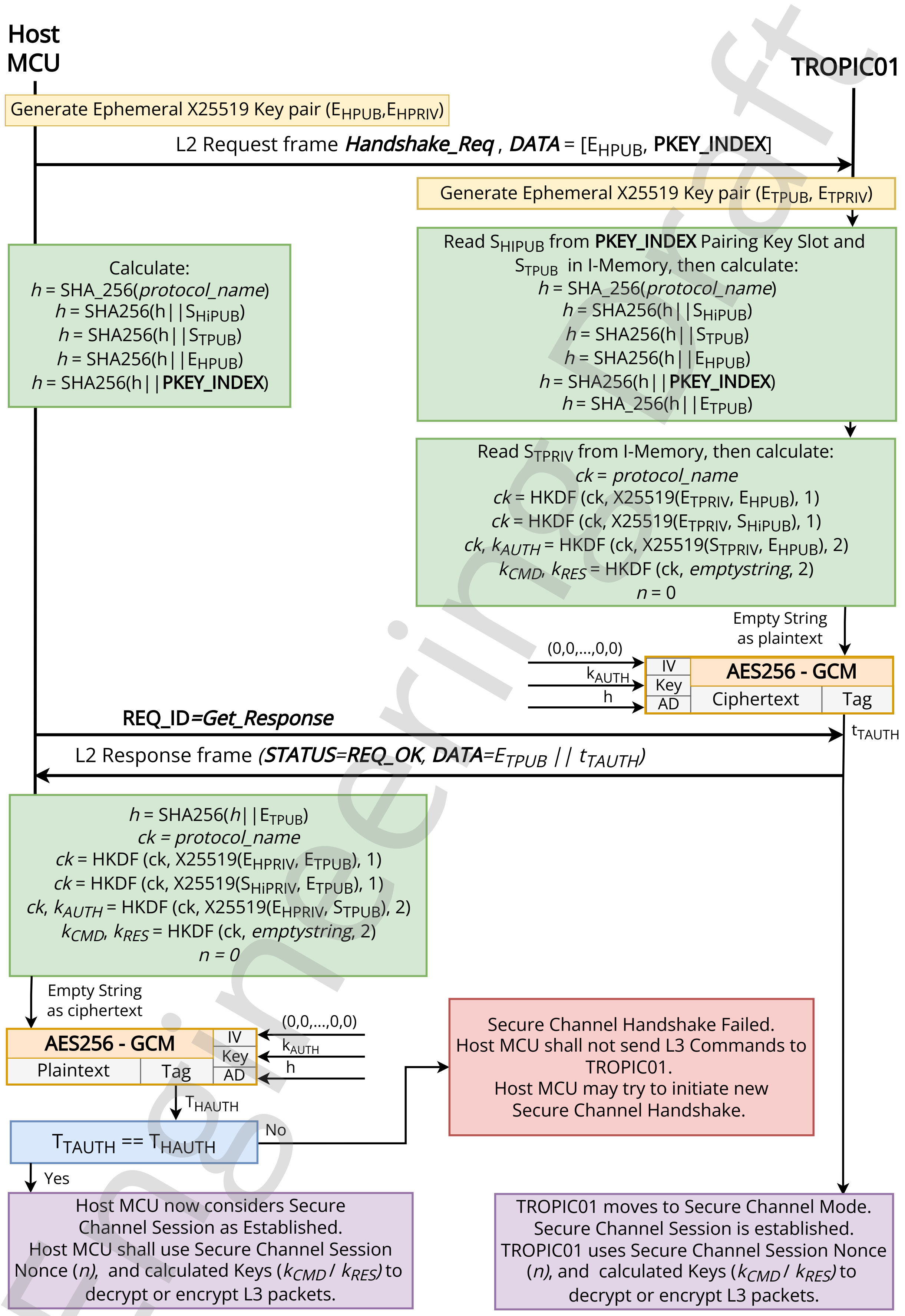

Host MCU
TROPIC01
Generate Ephemeral X25519 Key pair (E_{HPUB},E_{HPRIV})
L2 Request frame *Handshake_Req* , *DATA* = [E_{HPUB}, **PKEY_INDEX**]
Generate Ephemeral X25519 Key pair (E_{TPUB}, E_{TPRIV})
Read S_{HiPUB} from **PKEY_INDEX** Pairing Key Slot and S_{TPUB} in I-Memory, then calculate:
h = SHA_256(*protocol_name*)
h = SHA256(h||S_{HiPUB})
h = SHA256(h||S_{TPUB})
h = SHA256(h||E_{HPUB})
h = SHA256(h||**PKEY_INDEX**)
h = SHA_256(h||E_{TPUB})
Calculate:
h = SHA_256(*protocol_name*)
h = SHA256(h||S_{HiPUB})
h = SHA256(h||S_{TPUB})
h = SHA256(h||E_{HPUB})
h = SHA256(h||**PKEY_INDEX**)
Read S_{TPRIV} from I-Memory, then calculate:
ck = *protocol_name*
ck = HKDF (ck, X25519(E_{TPRIV}, E_{HPUB}), 1)
ck = HKDF (ck, X25519(E_{TPRIV}, S_{HiPUB}), 1)
ck, k_{AUTH} = HKDF (ck, X25519(S_{TPRIV}, E_{HPUB}), 2)
k_{CMD}, k_{RES} = HKDF (ck, *emptystring*, 2)
n = 0
Empty String as plaintext
(0,0,...,0,0)
k_{AUTH}
h
IV
Key
AD
AES256 - GCM
Ciphertext
Tag
t_{TAUTH}
REQ_ID=*Get_Response*
L2 Response frame (*STATUS*=*REQ_OK*, *DATA*=E_{TPUB} || t_{TAUTH})
h = SHA256(h||E_{TPUB})
ck = *protocol_name*
ck = HKDF (ck, X25519(E_{HPRIV}, E_{TPUB}), 1)
ck = HKDF (ck, X25519(S_{HiPRIV}, E_{TPUB}), 1)
ck, k_{AUTH} = HKDF (ck, X25519(E_{HPRIV}, S_{TPUB}), 2)
k_{CMD}, k_{RES} = HKDF (ck, *emptystring*, 2)
n = 0
Empty String as ciphertext
AES256 - GCM
Plaintext
Tag
IV
Key
AD
(0,0,...,0,0)
k_{AUTH}
h
T_{HAUTH}
T_{TAUTH} == T_{HAUTH}
No
Yes
Secure Channel Handshake Failed. Host MCU shall not send L3 Commands to TROPIC01. Host MCU may try to initiate new Secure Channel Handshake.
Host MCU now considers Secure Channel Session as Established. Host MCU shall use Secure Channel Session Nonce (*n*), and calculated Keys (k_{CMD} / k_{RES}) to decrypt or encrypt L3 packets.
TROPIC01 moves to Secure Channel Mode. Secure Channel Session is established. TROPIC01 uses Secure Channel Session Nonce (*n*), and calculated Keys (k_{CMD} / k_{RES}) to decrypt or encrypt L3 packets.

Figure 14: Secure Channel Handshake

7.5 User Access Privileges

TROPIC01 executes actions invoked by any of the Host MCU's L3 Command packets (defined in [1]), only if the Host MCU has sufficient User Access Privileges.

Otherwise, TROPIC01 responds with **RESULT** = **UNAUTHORIZED** in the L3 Result packet.

Note

When TROPIC01 responds with **RESULT**=**UNAUTHORIZED**, it is a valid L3 Result packet that is encrypted as specified in section 7.3.4. TROPIC01 increments its Secure Channel Session Nonce n accordingly and the current Secure Channel Session remains established.

User Access Privileges restrict L3 Command packet processing from other Host MCUs based on:

- The Pairing Key slot (selected by the **PKEY_INDEX** field of the ***Handshake_Req*** L2 Request frame) used to establish the current Secure Channel Session.
- L3 Command packet specific information (e.g. ECC Key slot, refer to [1]).

The Host MCU configures User Access Privileges by the **CFG_UAP_*** R-Config and I-Config COs (section 7.2).

Table 17 shows mapping between User Access Privileges and COs.

When the Host MCU sets a bit of **CFG_UAP_*** CO in I-Config to 0, L3 Command packet processing is permanently forbidden.

Each **CFG_UAP_*** CO forbids and allows L3 Command packet processing based on the Pairing Key slot used to establish the current Secure Channel Session.

User Access Privileges restrict L3 Command packet processing from different Host MCUs.

! Warning !

Certain **CFG_UAP_*** CO modifications can permanently impair TROPIC01 functions (e.g. writing all bits of all **CFG_UAP_*** COs to 0 forbids all L3 Command packet processing).

Each **CFG_UAP_*** CO field contains User Access Privileges for each Pairing Key slot according to Table 16.

Table 16: **CFG_UAP_*** CO field legend

Field Bit	Description
0	Pairing Key slot 0.
1	Pairing Key slot 1.
2	Pairing Key slot 2.
3	Pairing Key slot 3.
4–7	Reserved.

7.5.1 Privileges Read and Modification

The Host MCU can read and modify User Access Privileges by accessing the corresponding R-Config and I-Config COs as specified in section 7.2.

The actual User Access Privileges configuration TROPIC01 uses is given by merging I-Config and R-Config COs as explained in section 7.2.

Table 17: User Access Privileges

L3 Command Packet	Access Control Register
Functionality	
R_Mem_Data_Write	**CFG_UAP_R_MEM_DATA_WRITE**
R_Mem_Data_Read	**CFG_UAP_R_MEM_DATA_READ**
R_Mem_Data_Erase	**CFG_UAP_R_MEM_DATA_ERASE**
Random_Value_Get	**CFG_UAP_RANDOM_VALUE_GET**
ECC_Key_Generate	**CFG_UAP_ECC_KEY_GENERATE**
ECC_Key_Store	**CFG_UAP_ECC_KEY_STORE**
ECC_Key_Read	**CFG_UAP_ECC_KEY_READ**
ECC_Key_Erase	**CFG_UAP_ECC_KEY_ERASE**
ECDSA_Sign	**CFG_UAP_ECDSA_SIGN**
EDDSA_Sign	**CFG_UAP_EDDSA_SIGN**
MCounter_Init	**CFG_UAP_MCOUNTER_INIT**
MCounter_Get	**CFG_UAP_MCOUNTER_GET**
MCounter_Update	**CFG_UAP_MCOUNTER_UPDATE**
MAC_And_Destroy	**CFG_UAP_MAC_AND_DESTROY**
Ping	**CFG_UAP_PING**
Configuration	
I_Config_Write	**CFG_UAP_I_CONFIG_WRITE**
I_Config_Read	**CFG_UAP_I_CONFIG_READ**
R_Config_Write	**CFG_UAP_R_CONFIG_WRITE_ERASE**
R_Config_Erase	**CFG_UAP_R_CONFIG_WRITE_ERASE**
R_Config_Read	**CFG_UAP_R_CONFIG_READ**
Pairing_Key_Write	**CFG_UAP_PAIRING_KEY_WRITE**
Pairing_Key_Read	**CFG_UAP_PAIRING_KEY_READ**
Pairing_Key_Invalidate	**CFG_UAP_PAIRING_KEY_INVALIDATE**

7.5.2 User Access Privileges Example

Consider the following TROPIC01 CO content:

- I-Config : **CFG_UAP_PING**=0x00000001
  - A ***Ping*** L3 Command packet is authorized only when a Secure Channel Session is established with Pairing Key slot 0.
- R-Config : **CFG_UAP_PING**=0x0000000F
  - A ***Ping*** L3 Command packet is authorized only when a Secure Channel Session is established with Pairing Key from slots 0–3.

Then the actual applied configuration is:

$$0x00000001 \land 0x0000000F = 0x00000001 \quad (2)$$

Thus, ***Ping*** is authorized only via a Secure Channel Session from Pairing Key slot 0.

If a Secure Channel Session is established with keys from Pairing Key slots 1–3 and the Host MCU sends ***Ping***, TROPIC01 then responds with **RESULT**=**UNAUTHORIZED** in the L3 Result packet.

7.5.3 Pairing Key Slot Access

All public keys of X25519 pairing keys are stored in the I-Memory.

When TROPIC01 establishes a Secure Channel Session (section 7.4), TROPIC01 uses an X25519 public key from Pairing Key slot i (S_{HiPUB}).

Note

Tropic Square stores its public key in Pairing Key slot 0 (S_{H0PUB}) during manufacturing. By default it is possible to establish a Secure Channel Session via Pairing Key slot 0.

To read S_{HiPUB}, the Host MCU can send a ***Pairing_Key_Read*** L3 Command packet and TROPIC01 returns S_{HiPUB} in the L3 Result packet.

To write an X25519 public key S_{HiPUB} in a TROPIC01 Pairing Key slot, the Host MCU shall send a ***Pairing_Key_Write*** L3 Command packet.

Note

The Host MCU needs to know the private key used to generate the public key in Pairing Key slot 0 (S_{H0PRIV}). For details on how to access this key, see section 5.4.

Each Pairing Key slot can be in one of these states:

- **Blank**: The Pairing Key slot contains 1 in all bits (default state after manufacturing for Pairing Key slots 1–3).
- **Valid**: The Pairing Key slot contains a valid X25519 public key.
- **Invalidated**: The Pairing Key slot contains 0 in all bits.

If Host MCU sends ***Pairing_Key_Write*** on a Pairing Key Slot that is **Valid** or **Invalidated**, TROPIC01responds with TsApiCmdFldRESULT = **FAIL**.

If the Pairing Key slot is **Blank** or **Invalidated**, it is impossible to set up a Secure Channel Session with S_{HiPUB} from that Pairing Key slot.

TROPIC01 responds with **STATUS**=**HSK_ERR** in the L2 Response frame to a ***Handshake_Req*** L2 Request frame in such case.

The Host MCU can set a Pairing Key slot to **Invalidated** state by sending a ***Pairing_Key_Invalidate*** L3 Command packet.

Once the Pairing Key slot is **Invalidated**, the state is impossible to change.

Making the **Invalidated** state irreversible is fundamental to Chip Ownership (section 5.4.1).

! Warning !

At least one Pairing Key slot in the I-Memory must be **Valid**. When all Pairing Key slots are **Invalidated**, Secure Channel Sessions cannot be established and TROPIC01 is unusable.

7.5.4 Ownership Transfer Use-case

Assume the following roles:

- **Tropic Square** – The chip manufacturer.
- **OEM** – The customer of Tropic Square who integrates TROPIC01 into their product.
- **End Customer** – The customer of the OEM who uses the OEM's product to execute cryptographic calculations.

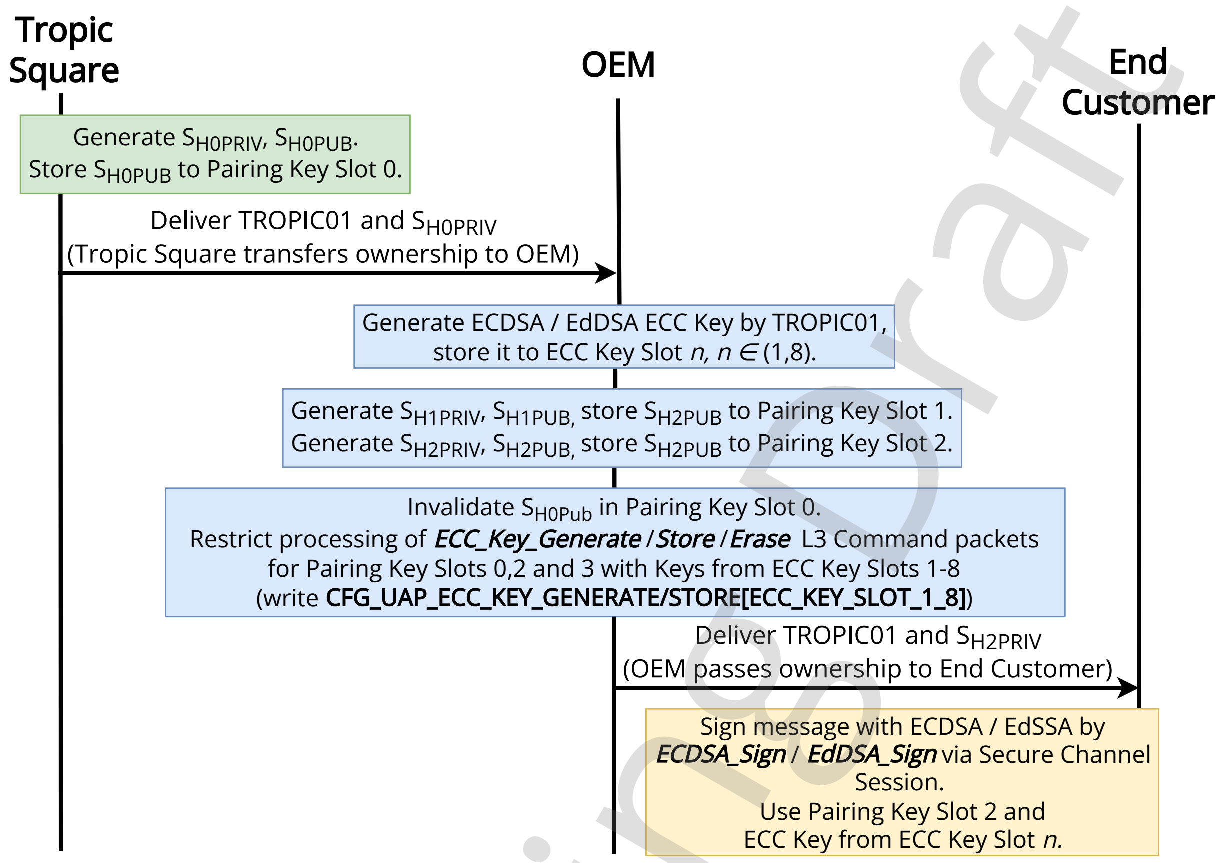

Tropic Square
OEM
End Customer
Generate S_{H0PRIV}, S_{H0PUB}. Store S_{H0PUB} to Pairing Key Slot 0.
Deliver TROPIC01 and S_{H0PRIV} (Tropic Square transfers ownership to OEM)
Generate ECDSA / EdDSA ECC Key by TROPIC01, store it to ECC Key Slot n, $n \in (1,8)$.
Generate S_{H1PRIV}, S_{H1PUB}, store S_{H2PUB} to Pairing Key Slot 1. Generate S_{H2PRIV}, S_{H2PUB}, store S_{H2PUB} to Pairing Key Slot 2.
Invalidate S_{H0Pub} in Pairing Key Slot 0. Restrict processing of ECC_Key_Generate / Store / Erase L3 Command packets for Pairing Key Slots 0,2 and 3 with Keys from ECC Key Slots 1-8 (write CFG_UAP_ECC_KEY_GENERATE/STORE[ECC_KEY_SLOT_1_8])
Deliver TROPIC01 and S_{H2PRIV} (OEM passes ownership to End Customer)
Sign message with ECDSA / EdSSA by ECDSA_Sign / EdDSA_Sign via Secure Channel Session. Use Pairing Key Slot 2 and ECC Key from ECC Key Slot n.

Figure 15: Ownership use-case

Assume the OEM wants to provide the End Customer a system where the End Customer can:

- Verify the system is genuinely from the OEM (Identity Attestation).
- Sign a message with ECDSA or EdDSA by an ECC key that is guaranteed to be from the OEM.
- Store and generate their own ECC keys and sign messages with ECDSA/EdDSA.

To deploy this system, the OEM has the following guarantees:

- No third party can modify the ECC keys loaded by the OEM.
- No third party can sign messages with ECDSA/EdDSA.
- The TROPIC01 instance is genuinely manufactured by Tropic Square.
- The End Customer cannot modify ECC keys loaded by the OEM, but can sign messages with the keys.

An ownership transfer use-case is shown in Figure 15.

OEM can modify TROPIC01's I-Config to prevent the End Customer from changing the ECC keys in slots 1 – 8 of ECC Key Partition in TROPIC01's R-Memory.

7.6 User Data Access

The Host MCU can store general purpose data in any of the 512 slots of the User Data partition of TROPIC01's R-Memory.

Each User Data partition slot is encrypted to protect User Data (section 5.7.3).

Note

Each R-Memory slot has a 512 Byte capacity. 476 Bytes are available for Host MCUs data. The remaining 68 bytes are overhead imposed by the encryption scheme (section 5.7.3).

To write data in the User Data partition of the R-Memory, the Host MCU sends a ***R_Mem_Data_Write*** L3 Command packet.

The Host MCU selects a number of bytes to write into the User Data slot by the following calculation using the data from ***R_Mem_Data_Write***:

$$byte\_length(\textbf{CMD\_DATA}) - 2 \quad (3)$$

(3) corresponds to **CMD_SIZE** – 3

To read from User Data partition slot, the Host MCU sends a ***R_Mem_Data_Read*** L3 Command packet.

The **UDATA_SLOT** field of ***R_Mem_Data_Write***/***R_Mem_Data_Read*** determines the index of the User Data Partition slot to write/read (0–511).

Each ***R_Mem_Data_**** L3 Command packet can only access a single slot.

If the Host MCU writes data into a slot of the User Data partition, the Host MCU shall write in the same slot again only after erasing the slot first.

If the Host MCU attempts to write in the same slot multiple times without erasing first, TROPIC01 responds with **RESULT**=**WRITE_FAIL** in the L3 Result packet.

Note

The Host MCU shall erase data from the same slot even if the Host MCU stores data only in part of the slot. (e.g. if the Host MCU writes 64 bytes to the User Data partition slot, the Host MCU needs to erase the entire slot to write data in the remaining 412 bytes)

When writing or reading the User Data partition, the Host MCU shall write or read data from address 0 of the slot. The Host MCU cannot start from the middle of a slot.

The Host MCU can erase a slot from the R-Memory User Data partition by sending a ***R_Mem_Data_Erase*** L3 Command packet.

The **UDATA_SLOT** field of ***R_Mem_Data_Erase*** determines the index of the slot to erase (0–511).

7.7 Elliptic Curve Cryptography

TROPIC01 implements Elliptic Curve Cryptography (ECC) via a dedicated HW engine.

The ECC engine performs these ECC operations:

- Generates an ECC key pair inside TROPIC01 and writes it to the R-Memory.
- Gets an private part of the ECC key pair from the Host MCU, computes the public part and writes the key pair to the R-Memory.
- Signs a message with an ECC key from the R-Memory (refer to [19] and [20]).
- Reads the public part of the ECC key pair from the R-Memory.

All ECC operations are supported on the following Elliptic Curves:

- ECDSA – NIST Curve P-256 (secp256r1)
- EdDSA – Curve Ed25519

TROPIC01 supports key setup and ECDH on Curve25519 (using X25519 function) during a Secure Channel Handshake (section 7.4.1).

TROPIC01 uses a TMAC function during ECC key setup, ECDSA sign, or EdDSA sign. TMAC function is a custom KECCAK based MAC function similar to KMAC256 ([14]). For further definition of the TMAC function, see [7].

7.7.1 ECC Key Setup

TROPIC01 stores the keys for ECC in the ECC Key partition of the R-Memory. The partition has 32 ECC Key slots. Each ECC Key slot stores a single key pair.

TROPIC01 supports these types of ECC key setup:

- TROPIC01 writes the private ECC key provided by the Host MCU to the R-Memory (the Host MCU sends an ***ECC_Key_Store*** L3 Command packet).
- TROPIC01 generates an ECC key pair and writes the key pair to the R-Memory (the Host MCU sends an ***ECC_Key_Generate*** L3 Command packet).

Note

When the Host MCU sends ***ECC_Key_Store***, the TROPIC01 automatically computes public key part and stores ECC key as a key pair.

The Host MCU selects the slot where the key will be stored by the **SLOT** field of ***ECC_Key_Generate*** or ***ECC_Key_Store***.

If the slot already contains an ECC key, TROPIC01 returns **RESULT**=**FAIL** in the L3 Result packet.

To erase a slot that contains an ECC key, the Host MCU:

- Selects the slot to be erased in the ***SLOT*** field of an ***ECC_Key_Erase*** L3 Command packet.
- Sends the ***ECC_Key_Erase***.

Note

TROPIC01 automatically masks the ECC private keys with 256-bit mask from internal TRNG before they are written to R-Memory. The private part never occurs in the chip in a raw form again.

Note

TROPIC01 automatically encrypts the ECC key when the keys are written to the R-Memory to maximize the security of the stored keys (section 5.7.3).

The Host MCU selects the type of ECC key TROPIC01 generates and stores by the **CURVE** field of ***ECC_Key_Generate*** or ***ECC_Key_Store***.

When TROPIC01 generates an ECC key (invoked by ***ECC_Key_Generate***), TROPIC01 uses TRNG1 to generate a random value k.

When the Host MCU inserts an ECC key into TROPIC01 (via ***ECC_Key_Store***), the Host MCU sends k in the **K** field of ***ECC_Key_Store***.

The Host MCU shall ensure k is a valid private key for the respective curve:

- P-256: $k \in \langle 1, q-1 \rangle$, where q is the order of the P-256 curve.
- Ed25519: $k \in \langle 0, 2^{256} - 1 \rangle$

TROPIC01 uses k to derive the public part of the ECC key pair.

When TROPIC01 receives ***ECC_Key_**** with **CURVE**=**P256**, TROPIC01 executes an ECC key setup for ECDSA as shown in Figure 16.

During the ***ECC_Key_Generate*** processing, TROPIC01 generates a random $k \in \langle 2^{512} - 1, 0 \rangle$

and computes:

$$d = k \pmod q, \qquad (4)$$

where q is the order of the P-256 curve as defined in Table 22.

If $d = 0$, then the ECC key setup failed and TROPIC01 responds with **RESULT**=**FAIL** in the L3 Result packet.

Otherwise, TROPIC01 proceeds with the key setup and responds with **RESULT**=**OK** when the key setup is done.

When TROPIC01 receives ***ECC_Key_Generate*** or ***ECC_Key_Store*** with **CURVE**=**Ed25519**, TROPIC01 performs an EdDSA key setup [21].

TROPIC01 then responds with **RESULT**=**OK**.

Figure 17 shows the EdDSA key setup.

After the Host MCU sets up an ECC key, the Host MCU can use the key to sign messages.

For details on ECDSA signing, see section 7.7.4.

For details on EdDSA signing, see section 7.7.5.

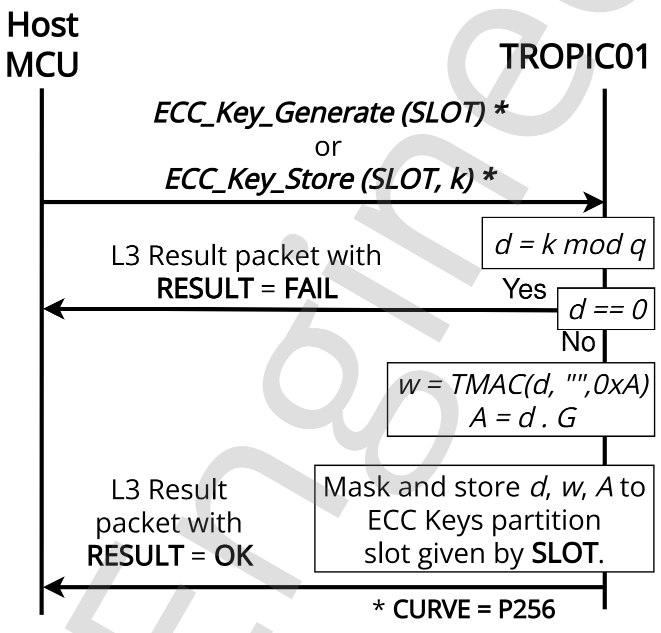

Host MCU
TROPIC01
ECC_Key_Generate (SLOT) *
or
ECC_Key_Store (SLOT, k) *
d = k mod q
L3 Result packet with RESULT = FAIL
Yes
d == 0
No
w = TMAC(d, "",0xA)
A = d . G
L3 Result packet with RESULT = OK
Mask and store d, w, A to ECC Keys partition slot given by SLOT.
* CURVE = P256

Figure 16: ECDSA Key Setup

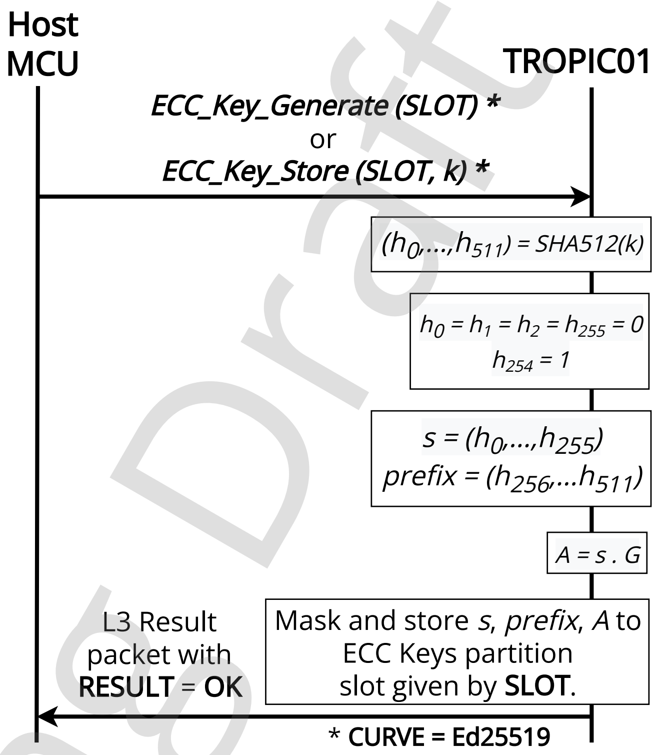

Host MCU
TROPIC01
ECC_Key_Generate (SLOT) *
or
ECC_Key_Store (SLOT, k) *
(h0,...,h511) = SHA512(k)
h0 = h1 = h2 = h255 = 0
h254 = 1
s = (h0,...,h255)
prefix = (h256,...h511)
A = s . G
L3 Result packet with RESULT = OK
Mask and store s, prefix, A to ECC Keys partition slot given by SLOT.
* CURVE = Ed25519

Figure 17: EdDSA Key Setup

7.7.2 ECC Key Memory Layout

The memory layout of ECC keys in the R-Memory slot is shown in Table 18.

Table 18: ECC Key Memory Layout

Key Part	Size [Bytes]	Description
ECDSA		
d	32	Private key 1
w	32	Private key 2
A	64	Public key
EdDSA		
s	32	Scalar
$prefix$	32	Prefix
A	32	Public key

7.7.3 ECC Key Read

The Host MCU can read the public key of ECC keys.

When the Host MCU sends an ***ECC_Key_Read*** L3 Command packet, the Host MCU selects the ECC Key slot that contains the key using the **SLOT** field.

When TROPIC01 receives ***ECC_Key_Read***, TROPIC01 responds with an L3 Result packet containing:

- The type of key using the **CURVE** field.
- Information weather the key was generated on the device by ***ECC_Key_Generate*** or imported by ***ECC_Key_Store command*** using **GEN_STORED** field.
- The public key of the selected ECC key.

7.7.4 ECDSA Signature

To sign a message M (of arbitrary meaning) by an ECDSA algorithm (defined in [20]), the Host MCU:

1. Calculates an arbitrary cryptographic hash of the message M by a hash function of its choice (defined in [20]).
2. Sends an ***ECDSA_Sign*** L3 Command packet with the 256 left-most bits of the calculated hash result in the **MSG_HASH** field (corresponds to z as denoted in [20]).

When TROPIC01 receives ***ECDSA_Sign***, TROPIC01:

1. Selects the R-Memory slot (in the ECC Key partition) specified by the **SLOT** field of ***ECDSA_Sign***.
2. Reads the public and private ECC keys from that slot.
3. Computes an ECDSA signature.
4. Returns the signature result (r, s) in an L3 Result packet with **RESULT**=**OK**.

Figure 18 shows TROPIC01's calculation for an ECDSA signing. (r, s) is defined in [20].

If the ECC Key slot from the **SLOT** field does not contain a valid ECDSA signature key (e.g. the slot does not contain any key, or the slot contains an EdDSA key), TROPIC01 responds with **RESULT**=**INVALID_KEY** in the L3 Result packet.

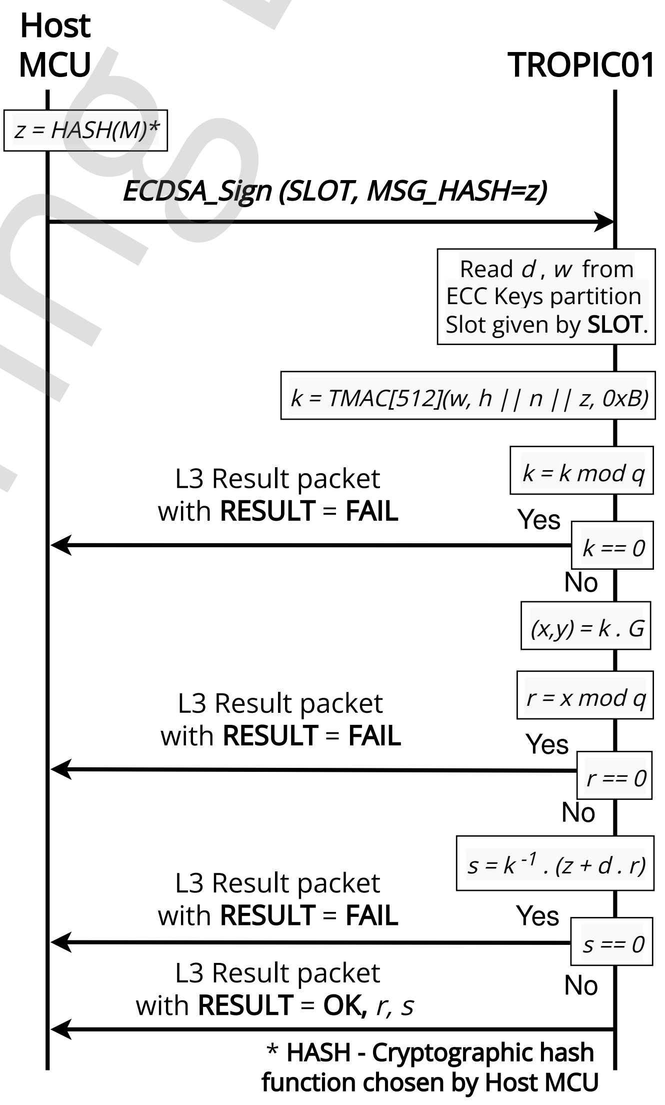

Host MCU
TROPIC01
z = HASH(M)*
ECDSA_Sign (SLOT, MSG_HASH=z)
Read d, w from ECC Keys partition Slot given by SLOT.
k = TMAC[512](w, h || n || z, 0xB)
k = k mod q
L3 Result packet with RESULT = FAIL
Yes
k == 0
No
(x,y) = k . G
r = x mod q
L3 Result packet with RESULT = FAIL
Yes
r == 0
No
s = k^-1 . (z + d . r)
L3 Result packet with RESULT = FAIL
Yes
s == 0
L3 Result packet with RESULT = OK, r, s
No
* HASH - Cryptographic hash function chosen by Host MCU

Figure 18: ECDSA Sign

If an ECDSA sign operation fails, TROPIC01 responds with **RESULT**=**FAIL**.

When TROPIC01 responds with **RESULT=FAIL** or **RESULT=INVALID_KEY**, TROPIC01 does not return any signature result (r, s) in the L3 Result packet.

7.7.5 EdDSA Signature

To sign a message M (of arbitrary meaning) with an EdDSA digital signature algorithm (defined in [21]), the Host MCU:

1. Selects the ECC Key slot for the EdDSA signature by the **SLOT** field of an ***EDDSA_Sign*** L3 Command packet.
2. Puts message M in the **MSG** field of the same L3 Command packet.
3. Sends ***EDDSA_Sign*** to TROPIC01.

When TROPIC01 receives ***EDDSA_Sign***, TROPIC01:

1. Reads the selected ECC keys from the ECC Key partition in the R-Memory.
2. Signs the message M with the ECC key.
3. Returns a signature result (R, S) in the L3 Result packet.

Figure 19 shows TROPIC01's steps when signing a message with an EdDSA signature.

Note

The maximum size of message M TROPIC01 signs with an EdDSA algorithm is 4 kB (4096 Bytes).

7.7.6 ECC User Access Privileges

All Elliptic Curve Cryptography (ECC) L3 Command packets are subject to User Access Privileges (section 7.5).

If an ECC L3 Command packet is not authorized for the current Secure Channel Session, TROPIC01 responds with **RESULT=UNAUTHORIZED** in the L3 Result packet.

The privilege levels for ECC L3 Command packets can be used to restrict unauthorized users from:

- Setting up ECC keys
- ECDSA/EdDSA message signing in particular slots
- Accessing public ECC keys

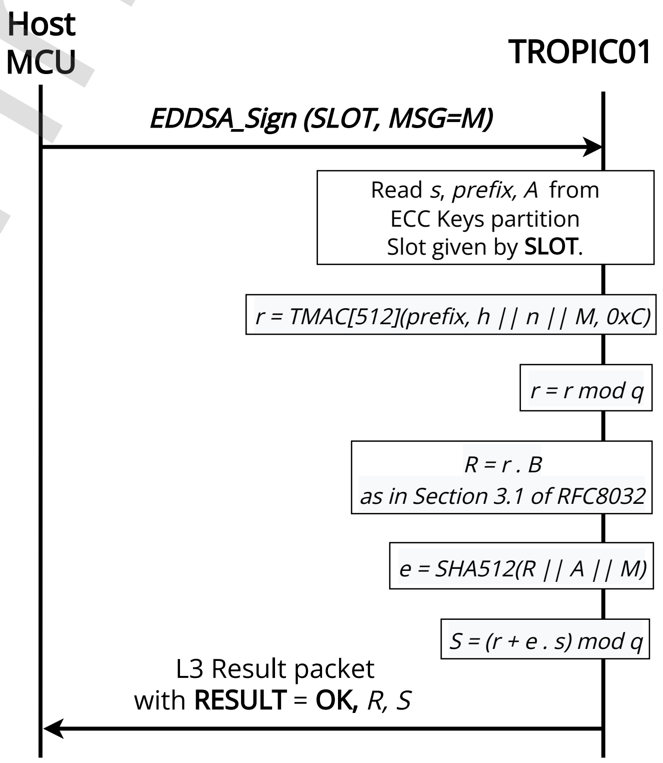

Host MCU
TROPIC01
EDDSA_Sign (SLOT, MSG=M)
Read s, prefix, A from ECC Keys partition Slot given by SLOT.
r = TMAC[512](prefix, h || n || M, 0xC)
r = r mod q
R = r . B as in Section 3.1 of RFC8032
e = SHA512(R || A || M)
S = (r + e . s) mod q
L3 Result packet with RESULT = OK, R, S

Figure 19: EdDSA Sign

7.7.7 X25519 Ephemeral Key Setup

When TROPIC01 executes a Secure Channel Handshake (section 7.4.1), TROPIC01 generates an X25519 ephemeral key pair (E_{TPUB}, E_{TPRIV}).

For more X25519 specifications, refer to [22].

7.8 X.509 Certificate

When the Host MCU sends a ***Get_Info_Req*** L2 Request frame with **OBJECT**=0x00, the Host MCU reads the data from the X.509 Certificate Store from TROPIC01's I-Memory.

The X.509 Certificate Store contains Tropic01 PKI certificate chain with following certificates:

- Tropic01 eSE (Device) Certificate
- Tropic01-XXXX Certificate
- Tropic01 CA Certificate
- Tropic Square Root CA Certificate

The certificate format is defined in [24].

The eSE (Device) Certificate contains a 32-byte S_{TPUB} X25519 public key TROPIC01 uses during a Secure Channel Handshake.

Tropic Square signs the Tropic01 eSE (Device) Certificate. The signature algorithm used might vary and can be read from the certificate.

The Host MCU shall verify the certificate chain before using the S_{TPUB} for Secure Channel Handshake.

For details on X.509 Certificate, see [6].

7.9 PIN Verification

TROPIC01 implements a custom procedure for Atomic PIN verification through a dedicated PIN verification engine.

TROPIC01 executes a part of a PIN verification procedure called a MAC-And-Destroy sequence.

To initiate the MAC-And-Destroy sequence, the Host MCU:

1. Selects the R-Memory slot from the MAC-And-Destroy partition (by the **SLOT** field of a ***MAC_And_Destroy*** L3 Command packet).
2. Sends the ***MAC_And_Destroy***.

When TROPIC01 receives ***MAC_And_Destroy***, TROPIC01 executes the MAC-And-Destroy sequence:

- Calculates a hash from the **DATA_IN** field of ***MAC_And_Destroy***.
- Reads, writes, and erases the R-Memory slots from the MAC-and-Destroy data partition.

For more details of the PIN verification procedure, refer to [5]. The external document explains:

- Actions the Host MCU shall perform to execute the PIN verification.
- Actions TROPIC01 executes while processing ***MAC_And_Destroy***.
- The cryptographic rationale behind the PIN verification design.

For details on TROPIC01's internal operation when executing a MAC-And-Destroy sequence, refer to [2]

7.10 Monotonic Counters

TROPIC01 implements 16 x 32 bit Monotonic Counters with:

- A count that starts from a user defined value and ends at 0
- Non-volatility – The counter value persists across power cycles
- The option to reset, decrement, or read from TROPIC01
- Protection against glitching attacks

Note

Each Monotonic Counter spans 2 slots in R-Memory.

7.10.1 Counter Initialization

To initialize a Monotonic Counter, the Host MCU sends a ***MCounter_Init*** L3 Command packet.

The Host MCU selects:

- A Monotonic Counter instance with the **MCOUNTER_INDEX** field.
- An initialization value of the counter with the **MCOUNTER_VAL** field.

When TROPIC01 completes a Monotonic Counter initialization, TROPIC01 responds with **RESULT**=**OK** in the L3 Result packet.

Note

The Host MCU must initialize a Monotonic Counter before reading and decrementing the counter.

7.10.2 Get Counter Value

When the Host MCU sends a ***MCounter_Get*** L3 Command packet, TROPIC01 responds with a Monotonic Counter value.

The Host MCU selects the counter instance by the **MCOUNTER_INDEX** field of ***MCounter_Get***.

TROPIC01 returns the value of the Monotonic Counter in the **MCOUNTER_VAL** field of the L3 Result packet.

7.10.3 Counter Update

When the Host MCU sends a ***MCounter_Update*** L3 Command packet, TROPIC01 attempts to decrement the value of the Monotonic Counter instance given by the **MCOUNTER_INDEX** field of ***MCounter_Update***.

If the value of the Monotonic Counter is 0, then the value stays at 0 and TROPIC01 responds with **RESULT**=**UPDATE_ERR** in the L3 Result packet.

Otherwise, TROPIC01 decrements the chosen Monotonic Counter value by 1 and responds with **RESULT**=**OK**.

Figure 20 shows an example of Monotonic Counter in use.

7.11 Ping

To check for a correct encryption operation of the current Secure Channel Session, the Host MCU can send a ***Ping*** L3 Command packet.

TROPIC01 responds with a fixed return value in the L3 Result packet.

Ping is a non-actionable command (i.e. TROPIC01 does not execute any action, apart from responding).

Since ***Ping*** is on the L3 Layer, processing ***Ping*** increments the Secure Channel Session Nonce n.

The Host MCU can use ***Ping*** to prevent TROPIC01 from moving to Sleep Mode.

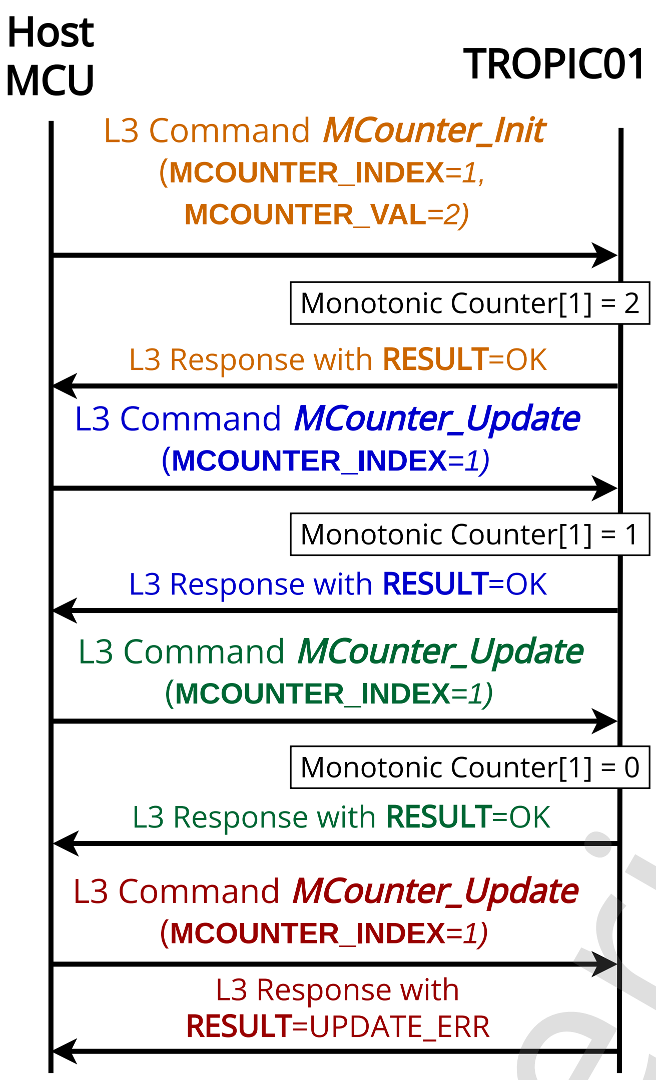

Host MCU
TROPIC01
L3 Command MCounter_Init (MCOUNTER_INDEX=1, MCOUNTER_VAL=2)
Monotonic Counter[1] = 2
L3 Response with RESULT=OK
L3 Command MCounter_Update (MCOUNTER_INDEX=1)
Monotonic Counter[1] = 1
L3 Response with RESULT=OK
L3 Command MCounter_Update (MCOUNTER_INDEX=1)
Monotonic Counter[1] = 0
L3 Response with RESULT=OK
L3 Command MCounter_Update (MCOUNTER_INDEX=1)
L3 Response with RESULT=UPDATE_ERR

Figure 20: Monotonic Counter use-case

For further details on ***Ping***, refer to[1].

7.12 Random Number Generation

TROPIC01 features two True Random Number Generators (TRNG): TRNG1 and TRNG2.

TRNG1 is for TROPIC01's internal encryption engines. (For details, see section 5.8.1).

TRNG2 is for the Host MCU to obtain a random number.

For the Host MCU to obtain random number:

1. The Host MCU selects a number of random bytes TROPIC01 returns in the **N_BYTES** field of a ***Random_Value_Get*** L3 Command packet.
2. The Host MCU sends the ***Random_Value_Get*** .
3. TROPIC01 returns a random number.

TROPIC01 provides a random number with at most BR_{RNG} bit-rate.

A separate TRNG for the Host MCU and TROPIC01 guarantees:

- Independent entropy sources.
- Random data from ***Random_Value_Get*** cannot be used to weaken security of other TROPIC01 functionalities.

8 ARCHITECTURE OVERVIEW

This subsection provides a brief overview of TROPIC01's internal architecture.

For more details on the internal architecture, refer to [2] and [3].

The external documents describe TROPIC01's internal operation such as:

- Clock Domains
- Supply Voltages
- Functionality
- Security Countermeasures
- Manufacturing Testability

8.1 FW architecture

TROPIC01 contains the following FW execution engines:

- RISCV CPU
- ECC engine

The FW running in TROPIC01 is split into the following parts:

- **Immutable FW** – Loads after power-up and boots mutable FWs
- **RISCV mutable FW** – Synchronizes all the encryption engines and processes L3 Command packets.
- **ECC engine mutable FW** – Handles the big number arithmetics of Elliptic curves.

TROPIC01 executes the immutable FW during Start-up mode. When TROPIC01 transfers to Idle Mode, it executes both mutable FWs.

Immutable FW is stored in ROM memory of RISCV CPU. Mutable FWs are stored in TROPIC01 R-Memory. For each FW there are two banks.

If no bank contains a valid mutable FW, TROPIC01 stays in Start-up mode (as if a maintenance restart was executed). If only one bank contains a valid mutable FW, TROPIC01 executes the FW from this bank. If both banks contain a valid FW, TROPIC01 executes the newer FW.

To change mutable FWs, TROPIC01 must be in Start-up mode after a maintenance restart.

The Host MCU updates TROPIC01 mutable FW by sending a ***Mutable_FW_Update_Req*** L2 Request frame.

The Host MCU erases TROPIC01 mutable FW by sending a ***Mutable_FW_Erase_Req*** L2 Request frame.

8.1.1 FW protection

TROPIC01 FWs are protected by the following mechanisms:

- Error Correction Code protected memories.
- All FWs are CRC protected.
- Mutable FWs are signed with EdDSA by Tropic Square.

If a protection mechanism reveals an error in the immutable FW, TROPIC01 ends in Alarm mode.

If a protection mechanism reveals an error in the mutable FW, the FW is invalid and TROPIC01 does not execute it.

Note

To run custom firmware on the CPU, (e.g custom encryption scheme, custom L2 Request frame/L3 Command packet), contact Tropic Square for further details.

Note

The TROPIC01 allows only FW signed by TropicSquare to be executed. If no signed FW is found, the TROPIC01 enters Maintenance loop. While in Maintenance loop new FW can be uploaded by command ***Mutable_FW_****.

8.1.2 RISC-V CPU

TROPIC01 contains an on-chip 32-bit RISC-V CPU (IBEX core), with the following parameters:

- 24 kB Instruction RAM
- 16 kB Instruction ROM
- 16 kB Data RAM
- 2x32 bit Timers
- 32 channel Interrrupt Controller
- Dual core lock-step

The on-chip CPU handles:

- L2 Request frame and L3 Command packet processing.
- L2 Request frame and L3 Command packet distribution to encryption engines.

8.1.3 Elliptic Curve Cryptography engine

Elliptic Curve Cryptography engine is a custom 256-bit CPU with instructions for modular arithmetics [4] and the following features:

- 32 256-bit registers
- Arithmetics for P-256, Ed25519 and Curve25519
- SHA512 unit
- TMAC unit

8.2 Secure Design Part

TROPIC01 encryption engines are designed to be secure.

RISC-V CPU cannot access plaintext content in L3 Command packets and L3 Result packets that are targeted towards HW encryption engines.

RISC-V CPU cannot access encryption keys, these are sent to HW encryption engines by dedicated datapaths.

8.3 Security Provisioning

Security Provisioning is the process during manufacturing of TROPIC01, where each TROPIC01 instance is configured with S_{TPRIV}, S_{TPUB} X25519 key pair, X.509 certificate signed by Tropic Square CA, and initial S_{H0PUB}. Security Provisioning is executed in a secure manner ensuring that all data configured in TROPIC01 are genuine.

8.4 Manufacturing Test Modes

All test-features are locked by default after TROPIC01 is manufactured and provisioned. These features are locked by multiple irreversible fuses and silicon live-cycle data in I-Memory.

For details on Test Mode locking, refer to [2].

Note

To get TROPIC01 without Manufacturing Test Modes locked, (e.g. for better control or observability during consumer specific security evaluation of TROPIC01 's encryption engines), contact Tropic Square for further details.

9 ELECTRICAL SPECIFICATIONS

Table 19: Electrical specifications

Parameter	Test Conditions	Min	Typ	Max	Unit
Supply Voltage					
VCC	—	2.7	3.0	3.6	V
$T_{VCCRAMPUP}$	—			TBD	us
Power Consumption – Typical Case					
Idle Mode	T=25°C		TBD		mA
Sleep Mode	T=25°C		TBD		mA
During Secure Channel Handshake	T=25°C		TBD		mA
During Elliptic Curve calculation	T=25°C		TBD		mA
During PIN verification	T=25°C		TBD		mA
Power Consumption – Worst Case					
Idle Mode	T=85°C			TBD	mA
Sleep Mode	T=85°C			TBD	mA
During Secure Channel Handshake	T=85°C			TBD	mA
During Elliptic Curve calculation	T=85°C			TBD	mA
During PIN verification	T=85°C			TBD	mA
Random Number Generator					
BR_{RNG}	—			TBD	kB/s
On-Chip Sensor Parameters					
T_{HIGH}	—			TBD	kB/s
T_{LOW}	—			TBD	kB/s
Timing Parameters					
$T_{HANDSHAKE}$				TBD	ms
$T_{STARTUP}$				TBD	ms
T_{ATTACK}				TBD	ms
Memory Parameters					
n_{RPROG}				TBD	

9.1 SPI Interface Timing

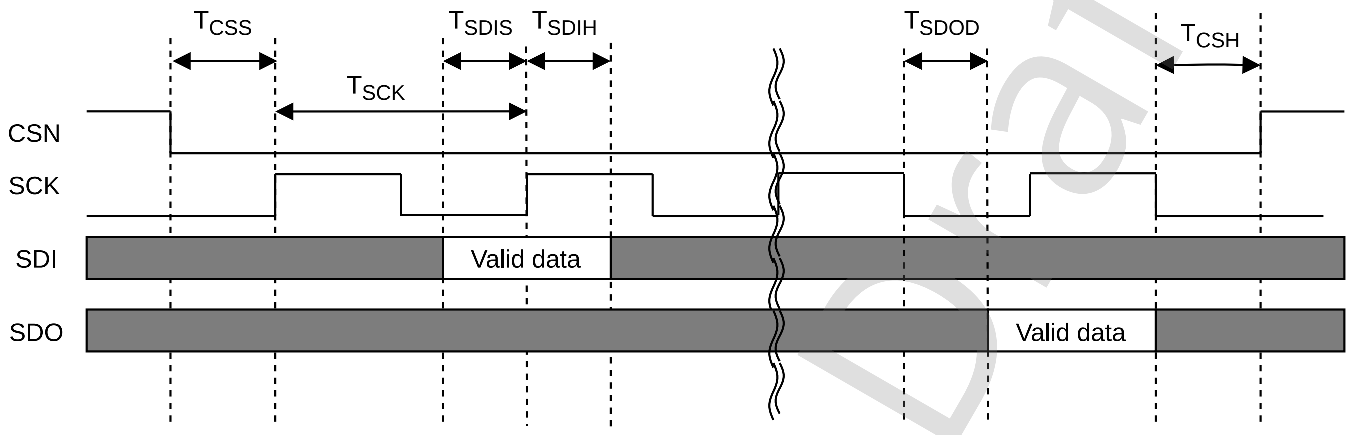


Figure 21: SPI Timing specification

Parameter	Min value [ns]	Max Value [ns]
T_{CSS}	20	-
T_{CSH}	20	-
T_{SDIS}	5	-
T_{SDIH}	5	-
T_{SDOD}	-	12
T_{SCK}	30	-

10 ABSOLUTE MAXIMUM RATINGS

Table 20: Absolute Maximum ratings

Parameter	Rating
Supply voltage	3.6 V
Voltage on IO pins	3.6 V
Temperature range	-20°C to 105°C
	-20°C to 85°C, I-Memory programming
Pin ESD rating	2 kV

11 TYPICAL APPLICATION

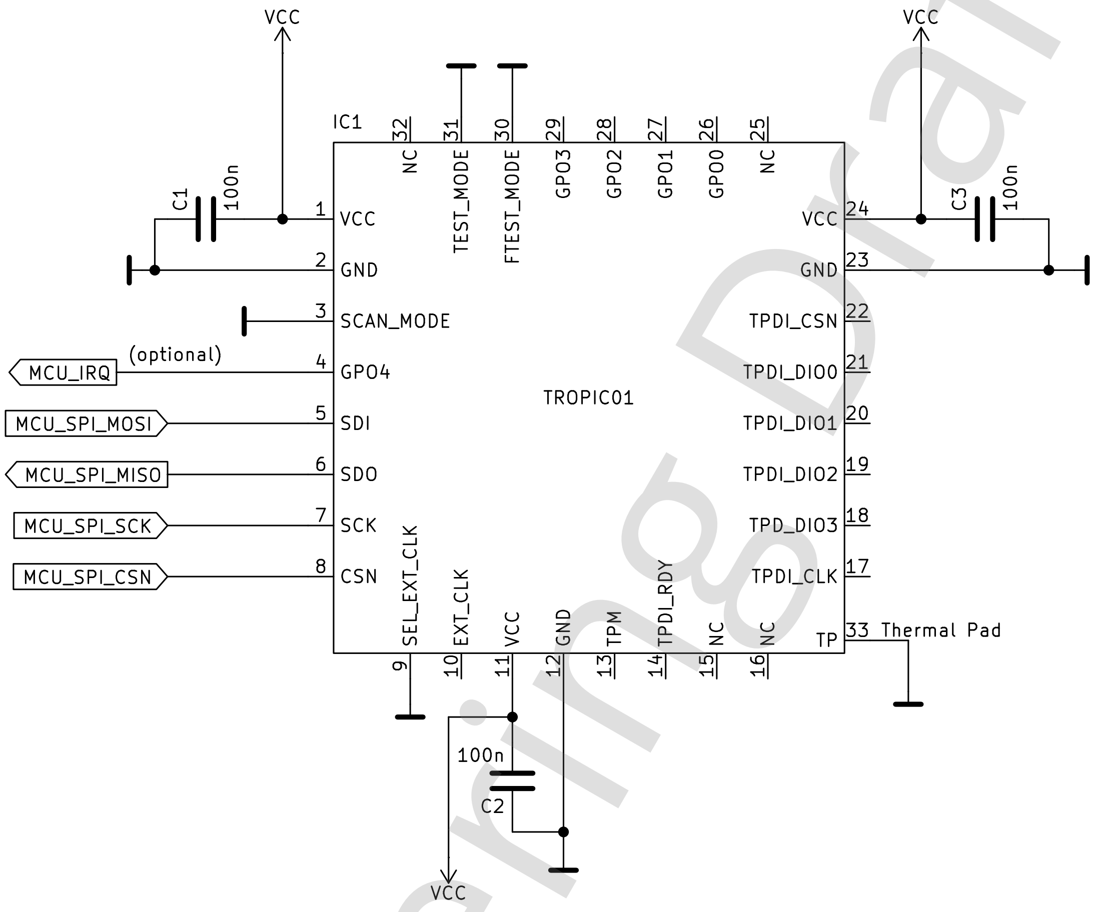
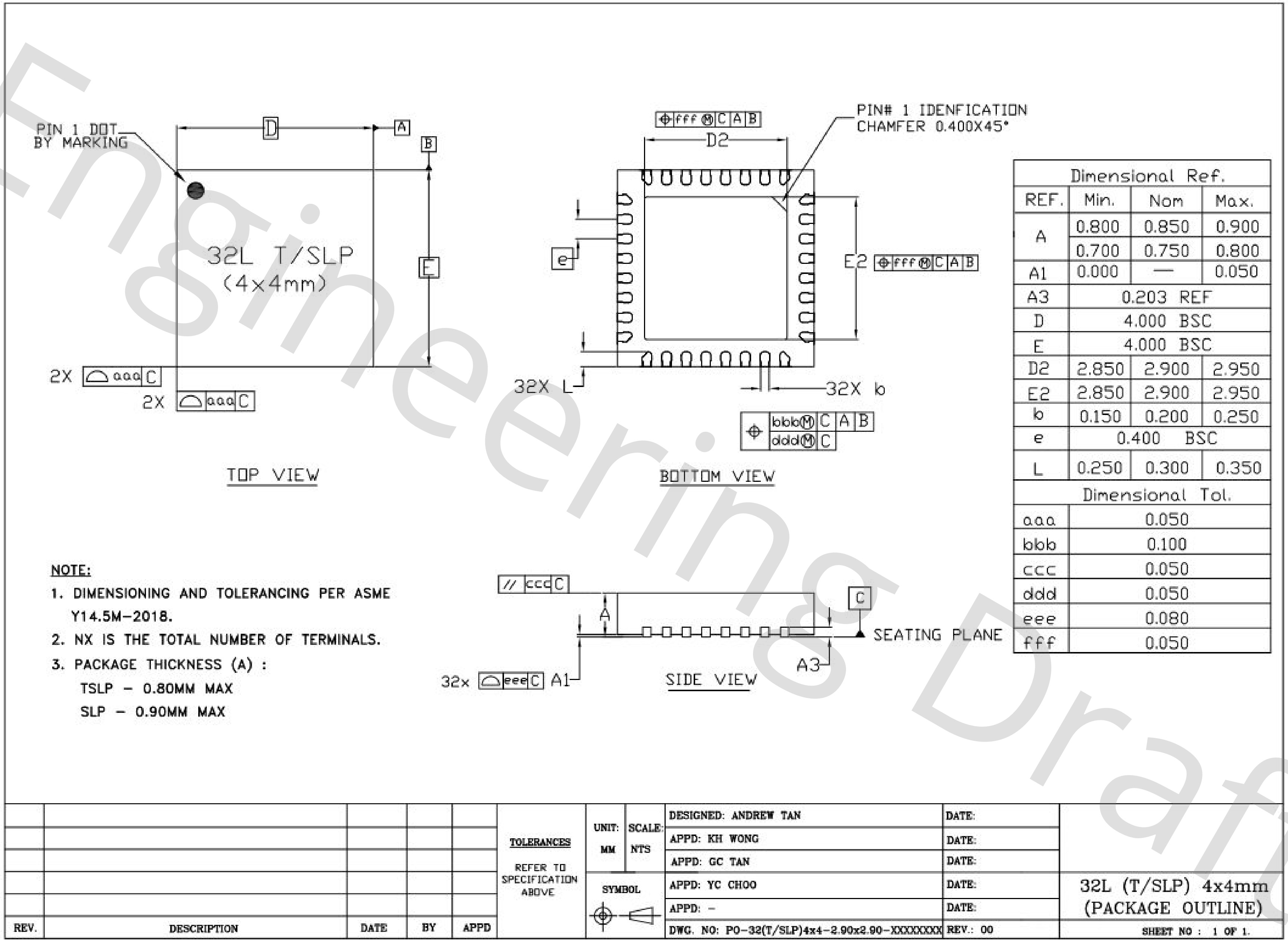


Figure 22: Typical application schematic

12 PACKAGE DESCRIPTION



Dimensional Ref.			
REF.	Min.	Nom	Max.
A	0.800	0.850	0.900
	0.700	0.750	0.800
A1	0.000	—	0.050
A3	0.203 REF		
D	4.000 BSC		
E	4.000 BSC		
D2	2.850	2.900	2.950
E2	2.850	2.900	2.950
b	0.150	0.200	0.250
e	0.400 BSC		
L	0.250	0.300	0.350
Dimensional Tol.			
aaa	0.050		
bbb	0.100		
ccc	0.050		
ddd	0.050		
eee	0.080		
fff	0.050		

NOTE:

1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M–2018.
2. NX IS THE TOTAL NUMBER OF TERMINALS.
3. PACKAGE THICKNESS (A) :
   TSLP – 0.80MM MAX
   SLP – 0.90MM MAX

REV.	DESCRIPTION	DATE	BY	APPD

TOLERANCES
REFER TO SPECIFICATION ABOVE

UNIT: MM SCALE: NTS
SYMBOL

DESIGNED: ANDREW TAN	DATE:
APPD: KH WONG	DATE:
APPD: GC TAN	DATE:
APPD: YC CHOO	DATE:
APPD: –	DATE:
DWG. NO: PO-32(T/SLP)4x4-2.90x2.90-XXXXXXXX	REV.: 00

32L (T/SLP) 4x4mm
(PACKAGE OUTLINE)
SHEET NO : 1 OF 1

Figure 23: 4x4 QFN32 Dimensions

13 REVISION HISTORY

REV A February 2023
First revision

REV A.1 April 2023
Rename Attestation Keys to ECC Keys
Add ***Sleep_Req***, ***ECC_Key_Read***, ***ECC_Key_Erase*** and **REQ_CONT**.
Fix typos and inconsistencies.
Fix X25519 arguments during Secure Channel Handshake in TROPIC01.

REV A.2 May 2023
Move **SLOT** of ***ECC_Key_Read*** to L3 Result packet.
Change **CMD_DONE** to **RES_OK**.
Add **RES_CONT**.

REV A.3 June 2023
Removed **UDATA_LEN** from ***R_Mem_Data_Read***.
Add **GEN_STORED** to ***ECC_Key_Read***.

REV A.4 September 2023
Add ***Startup_Req*** and ***Mutable_FW_****
Add FW architecture chapter.
Add **CHIP_STATUS[START]** bit.
Disable Maintenance restart by **CFG_STARTUP[MAINTENANCE_DIS]**.
Add note about chip restart.
Convert **BUSY** to **READY** bit.
Add Deep-sleep mode and wake-up token.
Add **CFG_DEBUG** CO description.
Remove resend limit of 2.
Change indexing of Pairing Keys and slots from 0.
Add ***Pairing_Key_Invalidate***.
Reduce max length of **REQ_LEN** to 252.
Add **RESP_DISABLED** response type.
Add section on endianity.
Rename ***Encrypted_Cmd_Abt***.
Remove **LEN_ERR**.

REV A.5 July 2024
Add QFN32 package.
Update for Section 10. Absolute Maximum ratings.

REV A.6 December 2024
Remove ***Serial_Code_Get*** L3 Command.
Remove (again) GPO pin since it is unimplemented.

REV A.7 January 2025
Adapt X509 Certificate details

APPENDIX A: CRYPTOGRAPHY PARAMETERS

Parameter Definitions

Table 21: Parameter Definitions

Obj.	Definition
p	Domain field size
(a, b)	Curve parameters
G	Base point generating the underlying subgroup (X,Y coordinate)
q	Order of the subgroup generated by G
h	Co-factor of the subgroup generated by G

Table 22: P-256 parameters

Obj.	Value
p	0xFFFFFFFF00000001000000000000000000000000FFFFFFFFFFFFFFFFFFFFFFFF
a	0xFFFFFFFF00000001000000000000000000000000FFFFFFFFFFFFFFFFFFFFFFFC
b	0x5AC635D8AA3A93E7B3EBBD55769886BC651D06B0CC53B0F63BCE3C3E27D2604B
G	0x6B17D1F2E12C4247F8BCE6E563A440F277037D812DEB33A0F4A13945D898C296, 0x4FE342E2FE1A7f9B8EE7EB4A7C0F9E162BCE33576B315ECECBB6406837BF51F5
q	0xFFFFFFFF00000000FFFFFFFFFFFFFFFFBCE6FAADA7179E84F3B9CAC2FC632551
h	0x1

Table 23: Ed25519 parameters

Obj.	Value
p	0x7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFED
a	0x7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEC
d	0x52036CEE2B6FFE738CC740797779E89800700A4D4141D8AB75EB4DCA135978A3
G	0x216936D3CD6E53FEC0A4E231FDD6DC5C692CC7609525A7B2C9562D608F25D51A, 0x6666666666666666666666666666666666666666666666666666666666666658
q	0x1000000000000000000000000000000014DEF9DEA2F79CD65812631A5CF5D3ED
h	0x08

Table 24: Curve25519 parameters

Obj.	Value
p	0x7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFED
a	0x76D06
b	0x01
G	0x09, 0x20AE19A1B8A086B4E01EDD2C7748D14C923D4D7E6D7C61B229E9C5A27ECED3D9
q	0x1000000000000000000000000000000014DEF9DEA2F79CD65812631A5CF5D3ED
h	0x08

14 REFERENCES

[1] TROPIC01 – User API, Tropic Square

[2] TROPIC01 – Functional Specification, Tropic Square

[3] TROPIC01 – Memory Map, Tropic Square

[4] TROPIC01 – SPECT Programmers guide, Tropic Square

[5] TROPIC01 – PIN Verification, Application Note, Tropic Square

[6] TROPIC01 – Public Key Infrastructure, Application Note, Tropic Square

[7] TROPIC01 – TMAC Specification, Tropic Square

[8] The Noise Protocol Framework, Trevor Perrin
http://www.noiseprotocol.org/noise.pdf

[9] ISO/IEC 9797-1 Information technology – Security techniques – Message Authentication Codes (MACs)

[10] NIST Special Publication 800-38D, National Institute of Standards and Technology, November 2007

[11] RFC 7748, Elliptic Curves for Security, Internet Research Task Force (IRTF), January 2016
https://datatracker.ietf.org/doc/rfc7748/

[12] FIPS 180-2, Secure Hash Standard, National Institute of Standards and Technology, February 2004

[13] FIPS FIPS 202, SHA-3 STANDARD: PERMUTATION-BASED HASH AND EXTENDABLE-OUTPUT FUNCTIONS, August 2015

[14] NIST SP 800-185, SHA-3 DERIVED FUNCTIONS: CSHAKE, KMAC, TUPLEHASH, AND PARALLELHASH, December 2016

[15] FIPS 198-1, The Keyed-Hash Message Authentication Code (HMAC), National Institute of Standards and Technology, July 2008
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.198-1.pdf

[16] ISAP, Submission to NIST, Christoph Dobrauning, May 2021
https://isap.iaik.tugraz.at/

[17] Changing of the Guards: A Simple and Efficient Method for Achieving Uniformity in Threshold Sharing, Joan Daemen, Radboud University, Nijmegen, The Netherlands, 2017
https://eprint.iacr.org/2016/1061

[18] A synthesis of side-channel attacks on elliptic curve cryptography in smart-cards, J. Danger, S. Guilley, October 2013
https://www.researchgate.net/publication/258168869_A_synthesis_of_side-channel_attacks_on_elliptic_curve_cryptography_in_smart-cards

[19] FIPS 186-4, Digital Signature Standard(DSS), National Institute of Standards and Technology, July 2013

[20] American National Standard X9.62-2005, public key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)

[21] RFC032, Edwards-Curve Digital Signature Algorithm (EdDSA)
https://datatracker.ietf.org/doc/html/rfc8032

[22] Curve25519: New Diffie-Hellman speed records, Daniel J. Bernstein
https://www.iacr.org/cryptodb/archive/2006/PKC/3351/3351.pdf

[23] Extracting the Private Key from Schnorr Signatures that reuse a Nonce
https://b10c.me/blog/009-schnorr-nonce-reuse-challenge/?=h

[24] ISO/IEC 9594-8, Information technology — Open Systems Interconnection — The Directory, Public-key and attribute certificate frameworks
https://datatracker.ietf.org/doc/html/rfc5280#section-1

[25] Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems, P. C. Kocher
https://link.springer.com/chapter/10.1007/3-540-68697-5_9

[26] Provably Secure Higher-Order Masking of AES, M. Rivain, E. Prouff
https://eprint.iacr.org/2010/441.pdf

[27] On Masked Galois-Field Multiplication for Authenticated Encryption Resistant to Side Channel Analysis, H. Oshida, R. Ueno, N. Homma, T. Aoki
https://link.springer.com/chapter/10.1007/978-3-319-89641-0_3