An Overview on QorIQ Security Features Targeted to Secure the Embedded System

EUF-SNT-T1638

Eric Bost | FAE

O C T . 2 0 1 5

External Use

Foreword

- Security is key .. isn't it ?
- Related to and impacting: Reliability, Privacy, Safety, Economics, Intellectual Property .. and more
- In this session we will speak of **system security** (not network security)
- QorIQ SoC processors offer multiple system security features , they are refered to as ''**Trust Architecture**''
- New QorIQ Layerscape families (LS1, LS2) combine the Freescale-originated Trust Architecture that was already in P- & T-series with the ARM®-centric TrustZone® architecture
- This session will review these so-called ''Trust'' functionalities, outline the objectives and basic functionalities

Agenda

- Intro / Background
- QorIQ major security components across families
- Detail of Trust Architecture key components
- Trust Architecture (TA) and TrustZone (TZ) in Layerscape
- Advanced Features

Introduction

System Security – Use Cases

- Handheld / User Appliance
  - It is key to isolate a very secure minimal set of functionalities besides the rich set of illimited applications managed by a full-featured OS (those functions related to user/owner ID, user interface, sensitive data and actions, DRM ..)

- Networking / Infrastructure
  - Network security as Key requirement (tunnels, protocols ..)
  - Unlike Network Security, System Security not considered a ''must''
  - More concerns arising for IP protect, cloning, authentication

- Industrial & Critical Embedded (Transport, Avionics, Defense)
  - Overall goal is to ensure and monitor deterministic behaviour (for safety & reliabilty)
  - The most sensitive functions incl. those for monitoring must be guaranteed
  - When multiple partitions act on the same system, some form of robust partitioning must be implemented
  - System security required in some cases for IP protect, authentication, tamper protect ..

Trust Architecture in QorIQ Communications Platform

- Trusted System
  - A system which does what its stakeholders expect it to do, resisting attackers with both remote and physical access, else it fails safe

- Freescale Trust Architecture (TA)
  - a set of OEM-controlled silicon features which simplify the development of trustworthy systems
  - Included for long time and enriched thru several generations of i.MX and QorIQ

- ARM TrustZone (TZ)
  - Both an Architecture specification and feature of ARM CPUs processors and IPs

- FSL TA and ARM TZ merged in ARM-based QorIQ LS-series resulting in TA 2.1 and above.

QorIQ Trust Architecture in Effect (1)

- In the context of the device's implementation of the Trust Architecture, if developers properly leverage the hardware hooks in the device, they can 'trust' that the software they loaded into the system during manufacturing (or during authorized software updates) is the software that executes following system boot.

- Once trusted software is in control, the developer can leverage additional Trust Architecture features to keep the trusted code in control of the system and defend against potential threats.

Question: *Ok but what is to be considered the "trusted software " in the system ? Up to which point ?*

QorIQ Trust Architecture in Effect (2)

The security mechanisms within the Trust Architecture allow users to define and enforce security policies. Examples of security policy violations that can be prevented or heavy mitigated by the Trust Architecture are :

- Unauthorized modifications to developer software and system configuration information (code/data, device trees, certificates). Protection consists of both prevention and after the fact detection mechanisms
- Unauthorized exposure of system persistent secrets. These are secrets that are intended to persist between resets of the system. Trust architecture 2.0 persistent secrets include the chip's One Time Programmable Master Key (OTPMK) and any code, factory installed private asymmetric, and pre-shared symmetric keys encrypted by the OTPMK and stored to non-volatile memory
- Unauthorized exposure of system ephemeral secrets. These are secrets that are intended to be cleared by the system's next reset (or sooner). Trust architecture ephemeral secrets include the chip's Job Descriptor Key Encryption Keys (JDKEKs) and session keys negotiated during normal operation that are encrypted with a JDKEK (also known as, "Black Keys").
- Unauthorized physical external access to the system eg. thru debug interface or any tamper
- Accidental or deliberate use of the private resources of one software partition by any other software partition

Note: *on this aspect, some QorIQ multicore key architecture features including robust partitioning thru MMU + IO-MMU + Hypervisor model are considered a component of the overall Trust Architecture in freescale litterature.*

QorIQ Trust Architecture in Effect (3)

- While some developers consider security policy enforcement to be a critical feature of the device, other developers may not. Consequently the Trust Architecture is disabled by default. Developers not implementing trust features can ignore their existence.
- Developers who choose to leverage the Trust Architecture are not dependent on Freescale to provision devices or sign code. Freescale is not part of the system development or manufacturing chain of trust. Developer provisioning of devices is designed to be simple, with minimal impacts on manufacturing cycle times.

How Goals Relate to Basic Security Hardware "Tools"

Authentication

I can prove to another party that I am User 12345 and only User 12345

Asymmetric Ciphers 'Public Key'

Non-repudiation

I can send a message, but cannot later claim that I did not send it.

Validation

I can send a message & guarantee it was not altered in transmission

I can run firmware and guarantee that it has not been altered

Hashes, secure boot, runtime checks

Secrecy

I can send a message and know that, even if it is intercepted, it cannot be read

Symmetric Ciphers

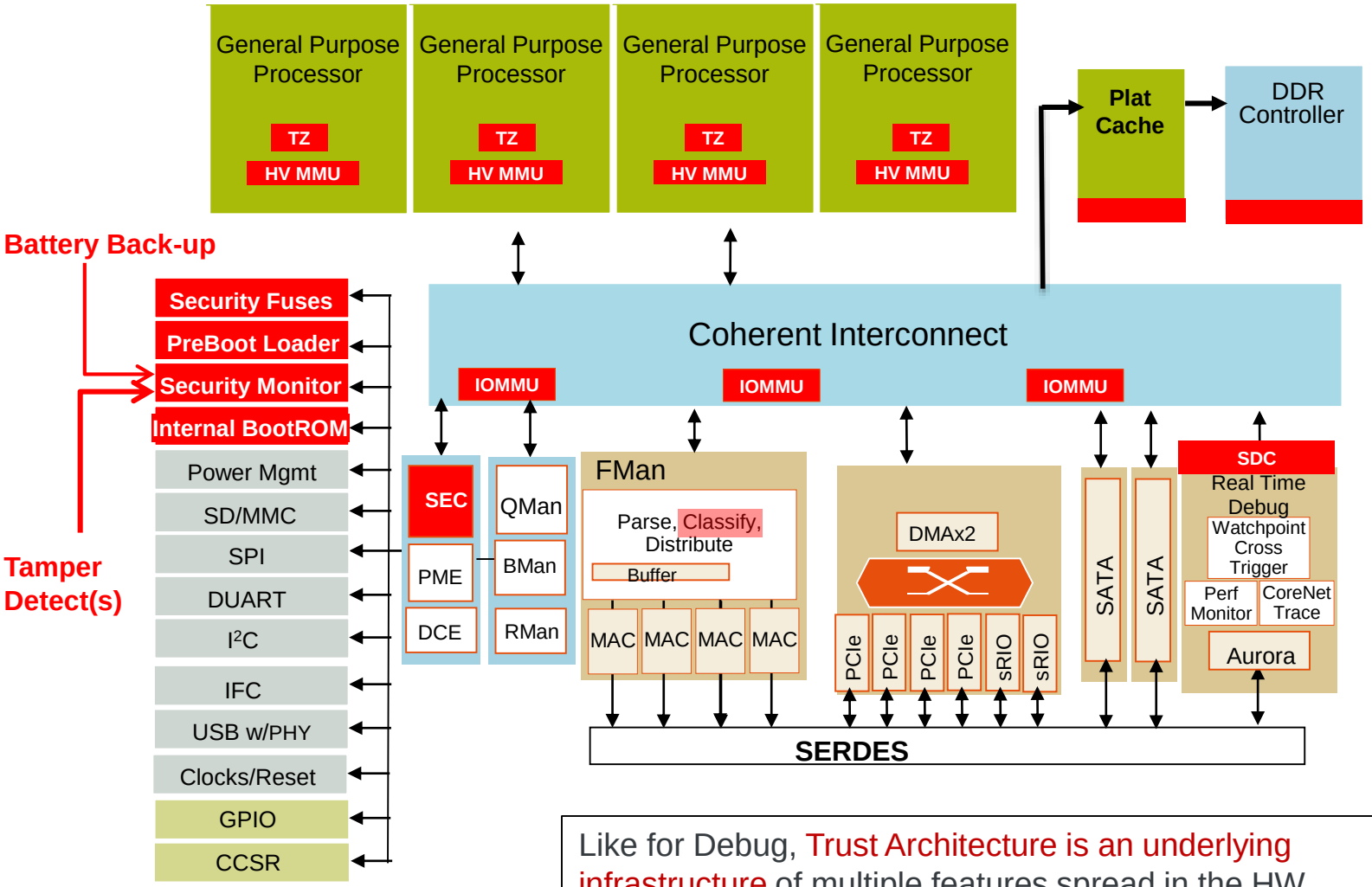
Protection

My equipment must:
- Prevent compromises in security,
- Detect if system is tampered with
- Store secret keys

Hardware (Fuses, Tamper detect, Zeroization)

QorIQ Major Security Components Across Families

Where Does Trust Architecture Reside in the SoC ?



Like for Debug, Trust Architecture is an underlying infrastructure of multiple features spread in the HW

Trust Architecture – Features Summary

TRUST Version	Trust 1.0	Trust 1.1	Trust 2.0	Trust 2.1	Trust 3.0
Related devices ('E' devices)	P4080, P1010	P2040, P3041, P5020	T1040, T2080, T4240, B4860, C290	LS1020, LS1043	LS2080, LS1088
Base Features					
Secure Boot	Y				
Secure Boot -offloaded thru SEC (HW accel.)	N		Y		
Secure Debug Controller	Y			Y + TrustZone "Secure World" add'l protections	
Security Fuse Processor (SFP)	Y				
Security Monitor	Y				
Security Monitor Dual-power sections (incl. Key zeroization)	N	Y	N	Y	
External Tamper Detection	Y				
Real-Time Integrity Checker (RTIC)	Y				
SEC-supported Blobs based on Master Key	Y				
SEC-supported Ephemeral Key Encryption Keys	Y				
CPU Memory Access Control	Power ISA MMU w/HV			ARM ISA MMU w/HV and TrustZone	
I/O Memory Access Control	Platform MMU (PAMU)			Platform MMU (SMMU)	
ARM TrustZone	N			Y	
Advanced Features					
Secure Boot - Alternate (secondary) signed Image	N		Y		
Secure Boot - Key list and Key revocation	N		Y (List of 4 keys)	Y (List of 8 keys)	
Monotonic Counters	N		1 (not in T1 & B4)	1	
HW Key Pair (aka Trusted Mfg)	N			Y	

Reference Literature / Spec

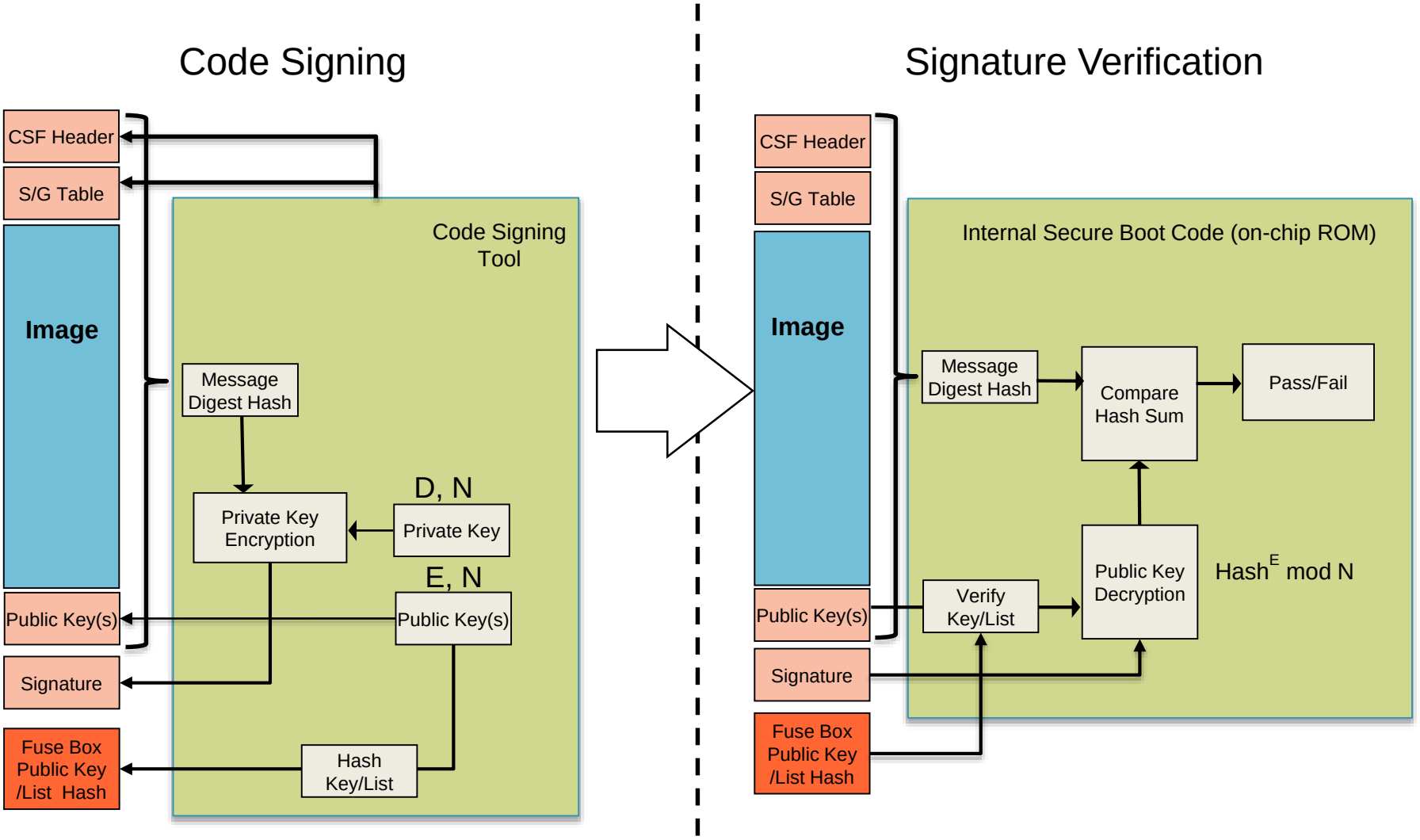
- QorIQ processors Reference Manuals:
  - Chapter “Secure Boot and Trust Architecture”
  - Chapter “Security Engine (SECx.y)”
- White papers including *QORIQTAWP: “An introduction to the QorIQ platform Trust Architecture” and QORIQSECBOOTWP : “Secure Boot for QorIQ Communications Processors”*
- *“Manufacturing guide for Trust architecture” and Code Signing Tool (CST)* – ***under NDA***
- QorIQ processors Reference Manuals (for PAMU, SoC-level partitioning)
- Core (e500mc, e5500, e6500) Reference Manuals (for Hypervisor model, MMU, Core-level partitioning …)
- ARM Datacenter (for ARM related oncl. TrustZone)
- NIST Crypto Algorithm Validation Program Certificates for Digital Networking Products
- Export Control summary sheets (ECCN, CCATS, Algorithms, Key Lengths)

Detail of Trust Architecture Key Components

Secure Boot – in Principle

- Optional Secure Boot implemented by OEM
- Boot validation performed by running Internal Secure Boot Code (ISBC) provided by Freescale in internal ROM.
- Principle:
  - at **manufacturing**, External boot code is signed by OEM : Hash value calculated thru SHA-256 on boot image, secured (encrypted) with RSA Private Key and appended to boot in external non-volatile memory.
  - in **operation**, at boot time, ISBC (in internal ROM) validates the boot image by decrypting signature with Public Key, recalculating hash value on boot image and compare to hash value from signature.
  - Public Key itself is kept in internal fuses (in fact just a hash of it)
  - Any mismatch fails booting in secure mode

Secure Boot – in Action



Secure Boot – a few more Things and Q&As

- Using secure boot requires some Pre-Boot Initialization (PBL) at least for loading an ESBC pointer to the SCRATCHRW1 before ISBC starts executing from core0.
- If ISBC 1st stage secure boot succeeds, what happens ?
  - if no further boot and validation required, then enters ‘’Trusted’’ state and the SEC is allowed to use the provisioned keys OTPMK, ZMK, and secret KEKs.
  - if ESBC is Trusted Bootscript and Bootloader, then next stage image is validated as in previous stage
- If ISBC 1st stage secure boot fails, what’s next ?
  - can optionally validate an alternate boot image
  - transition Secure Monitor to either soft or hard failed state leading to either reduced security usage mode or leading to Hard Reset
- on QorIQ ARM-based LS-series, the ISBC and ESBC run in TrustZone secure world

Questions:

• until which point do we need to secure ?

• can other TA features (eg. secure debug, RTIC, Blobs) be implemented without implementing secure boot ? In that case, what happens if there is a HW security violation ?

Chain of Trust

Secure Boot can validate several non-contigous memory ranges and can chain several validation sequences

ESBC U-Boot can be written to use the SEC to decrypt some or all of the client. Portions of the client could be decrypted back into main memory, while particularly sensitive code or data could be decrypted into internal SRAM (portions of the Corenet platform cache configured as SRAM).

Security Fuses

(Showing QorIQ T1040 case)

In RED not readable

FSL Section fuses

- 1b - FSL section write protect
- 32b - FSL unique ID
- 64b - FSL scratchpad
- 32b - FSL CRC

OEM Section fuses

- 1b - OEM section write protect
- **1b - Intent to secure**
- 1b - Clear_SFF
- 1b - **SEC disable**
- 4b - Key0-2 revocation
- 3b - Debug permissions
  - Open
  - Conditionally closed w/o notification
  - Conditionally closed w notification
  - Closed

(for secure boot)

- **256b - Super Root Key Hash (hash of Public key)**

(for secure debug)

- **64b - Debug challenge value**
- **64b - Debug response value**

(for long-term keys & data protection)

- **256b - One time programmable master key (OTPMK)**
- 32b - OEM unique ID
- 64b - OEM scratchpad
- 32b - OEM CRC

Generated by SEC at each boot, not in fuses

(for short-term session keys protection)

- **256b - Key Encryption Key (KEK)**

Internal Fuses

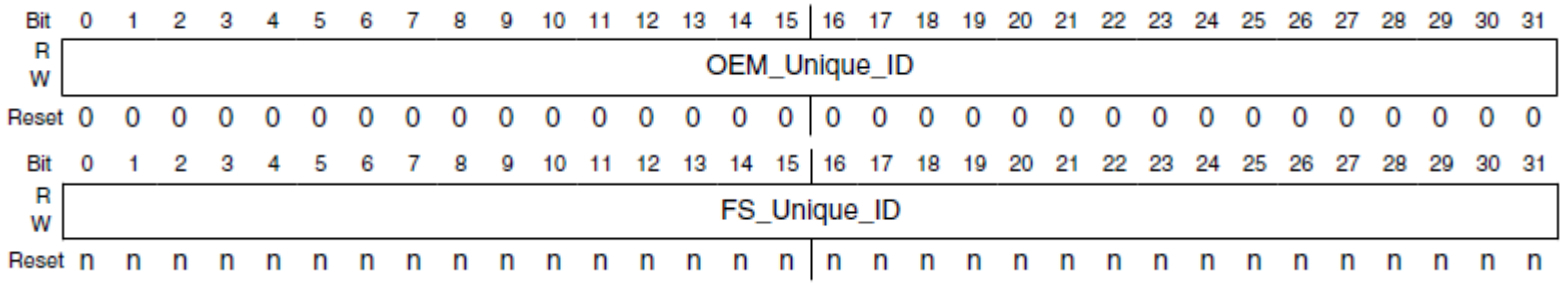
- Includes FSL and OEM sections. Sensitive content cannot be read, modified or scanned-out

- OEM section burned during device provisionning thru SW + external pin powering or thru Debug ctrl interface

- Holds secret values used for secure boot and further ‘’chain of trust’’, including:
  - **ITS** (Intent to Secure) bit which enforces secure Boot thru Internal ROM code (ISBC)
  - **OTPMK** (One Time Programmable Master Key): a 256bit key to be used by the SEC as an AES key. Primary purpose being encryption/decryption of additionnal session keys further used by the SEC to protect Data.
  - Super Root Key (**SRK**) -Hash: a 256bit SHA-256 hash of the RSA Public Key (called Super Root Key- SRK) that is used by the ISBC (Internal Secure Boot Code) to validate the ESBC (External Secure Boot code). The full SRK is included in the ESBC; ISBC first hash and validates it with regard to the fused SRK-Hash.

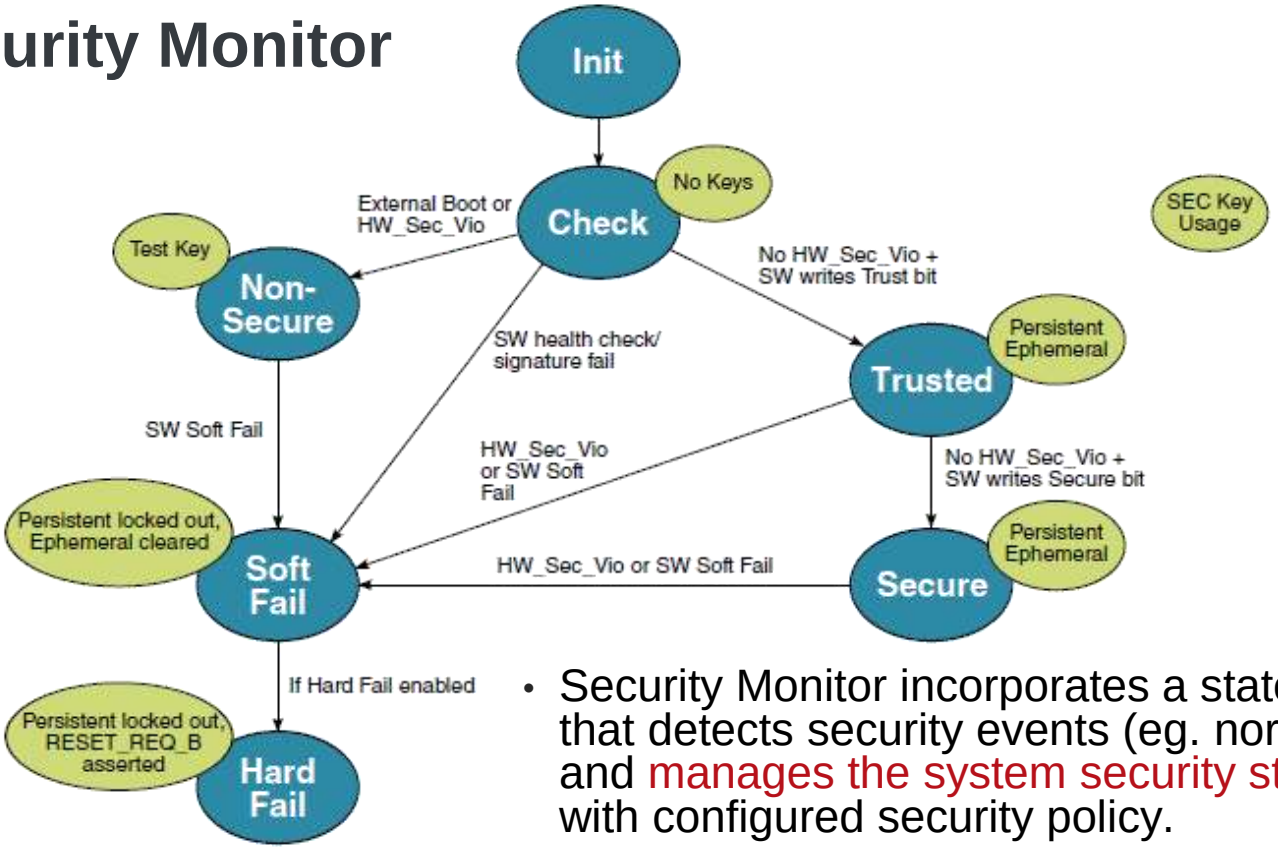
- Holds other key fixed values:
  - Debug access permission & challenge/response values, OEM unique ID, Fuse values checksums, scratchpad values, FSL unique ID, SEC disabling

On Freescale and OEM Unique ID Fuses



- The **OUIDR** is set by OEMs to configure an unalterable 32-bit software readable value which can be (optionally) included as part of the ESBC for the purposes of digital signature validation. It is up to the OEM to determine whether each device is provisioned with a truly unique ID.
- The **FUIDR** is set (and write protected) by Freescale prior to device shipment to provision a pseudo-random software readable value.
- Note that to bind an image to a specific device, the FUIDR must also be included in the ESBC for digital signature validation. Otherwise it would be possible for an attacker with access to a new chip to program the OUIDR to match the value found in a fielded chip.

Security Monitor



- Security Monitor incorporates a state machine (SSM) that detects security events (eg. normal or violations) and manages the system security state in accordance with configured security policy.
- On some QorIQs (all except P4, T1, B4) , a portion of the Security Monitor can operate on battery power while the rest of the SoC is powered off.
- Once in the trusted/secure state, the SecMon providesongoing monitoring of the QorIQ processor's security, with security violations causing configurable actions ranging from master key lock-out or zeroization to full SoC reset.

Security Monitor - Security Violation Sources

- Hardware:
  - External Tamper Detection via TMP_DETECTs
  - Secure Debug Controller (if set to Conditionally Closed with Notification)
  - Run Time Integrity Checker
  - Security Fuse Processor
    - On fuse array read fails, including hamming code check
  - Security Monitor (OTPMK hamming code check)
  - All sensitive flops upon detection of scan entry (expert mode debug)
  - Monotonic Counter Rollover
- Software:
  - Internal Secure Boot Code
  - Trusted U-Boot
  - Any SW with write access to the Security Monitor can declare a security violation.

Security Violation Response

The response is configurable.

- Soft Fail
  - Persistent Device Secrets are locked out
  - Ephemeral device secrets (if in use) are cleared
  - All SEC registers containing sensitive data are cleared
  - Sec_Mon generates IRQ.
- Hard Fail
  - Fatal security violations start a High Assurance Counter, when counter reaches zero, the device initiates Soft Fail consequences plus:
    - Battery backed Device Secret and non-secret values are cleared
    - Active zeroization of the device platform caches and system main memory, while concurrently triggering the RESET_REQ signal.
    - System designer must ensure that the RESET_REQ output signal triggers a device reset (HRESET or PORESET).

External Tamper Detection

- QorIQ Trust Architecture enables an OEM to add **external tamper detection circuitry** to the system, such as:
  - access-panel-open switch
  - light sensor inside the electronics chassis
  - voltage out of range

… and route an event signal from this external circuitry into the Sec_Mon by means of a dedicated signal **TMP_DETECT**.

- If this input signal is interrupted, the Sec_Mon **transitions to a non-secure or soft fail state** preventing the system from decrypting secrets that were previously protected by the OTPMK or KEK while in the secure or trusted states.

Secure Debug Controller

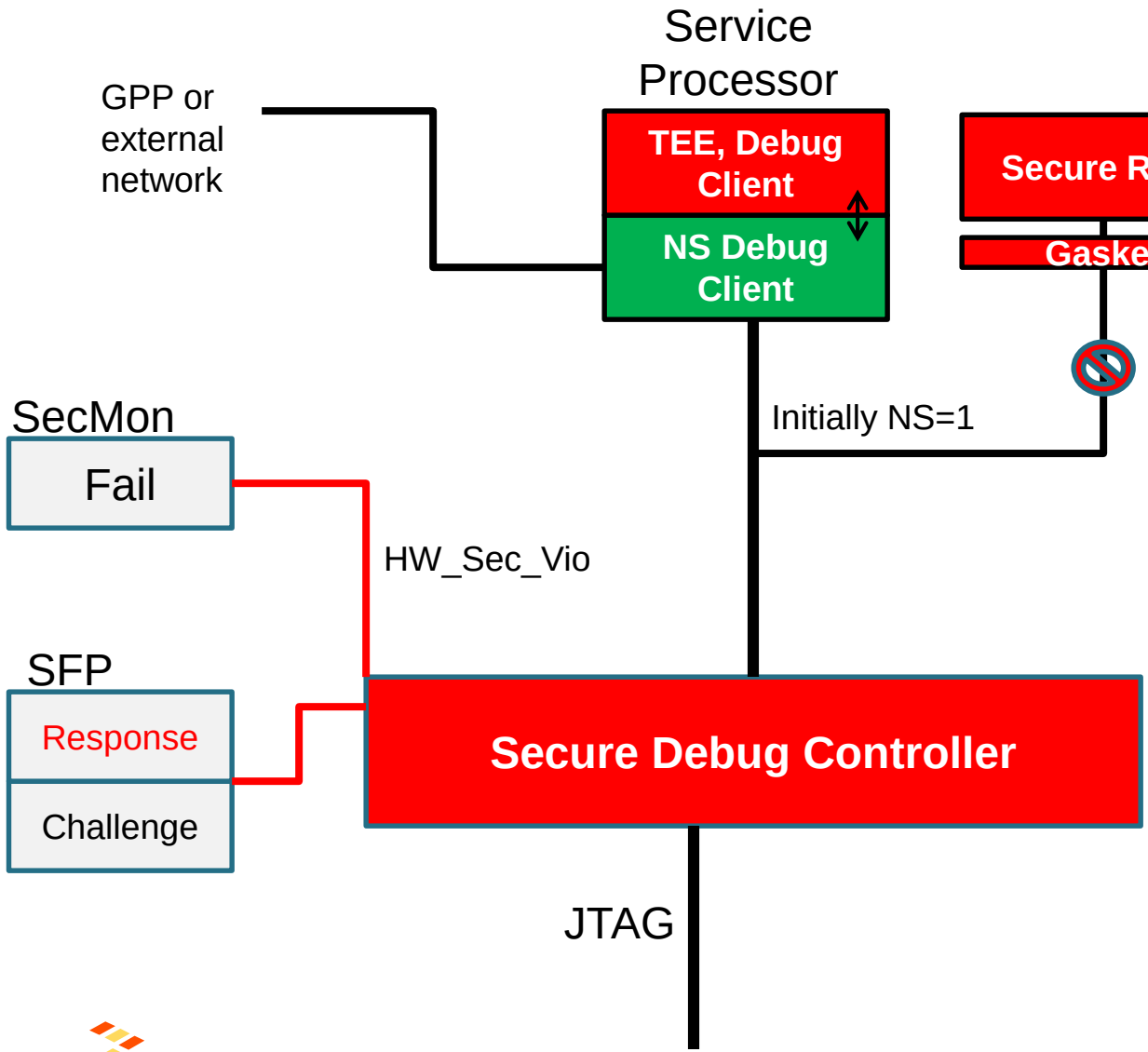
OEMs can control the level of debug access available to external debuggers. The Secure Debug Controller supports 4 levels of access.

- **Open:** external debug agents connected to the Aurora or SAP/JTAG interfaces have full access to P4080 memory space. Activation of debug is not considered a security violation, and if the Sec_Mon is in Trusted/Secure state, it will remain there. This setting is appropriate to secure system in a lab environment or for non-secure system.

- **Conditionally Closed:** (with or without Notification):. External debug agents are blocked until successful challenge/response sequence. 64b response value compared to secret value in the security fuse processor.
  With/without notification refers to how the system continues or not to operate if challenge/response fails, with OTMK and KEK usage enabled or disabled.

- **Closed:** attempts by external debug agents to access P4080 memory space are always blocked, and are not reported as security violations. The JTAG interface can still be used for boundary scan physical interconnect testing.

Secure Debug
(case of LS devices with TA+TrustZone)



Once beyond HW secure debug (and depending on configuration), external debugger can perform TrustZone signature based authentication to TEE/Secure World Debug Client, which can give SDC secure world access, or access secure world registers itself.

.

Hardware challenge/response and violation control methods for HW debug interfaces.

Low-Power Domain

Available in all QorIQ from P3/P5 except T1 & B4

- Implemented thru **two separate voltage domains** (SM_LP and SM_HP)
- SM_LP is typically supplied thru battery, this is a separate always-powered-up domain with its own power supply. Its purpose is to store and protect system data and enforce security regardless of the main system power state .
- There is an option for a battery backed Zeroizable Secret Key (ZSK) which can be used instead of the OTPMK in the security fuses. On a security violation, rather than being locked out until the next successful secure boot cycle , the ZSK is zeroized. Anything encrypted with the ZSK is unrecoverable, potentially including encrypted part of system software This significantly raises the consequence of a security violation.
- a secondary external tamper detection input is featured for when the QorIQ device's main power is off.
- On QorIQ LS, Low Power domain adds more advanced features (eg. monotonic counter)

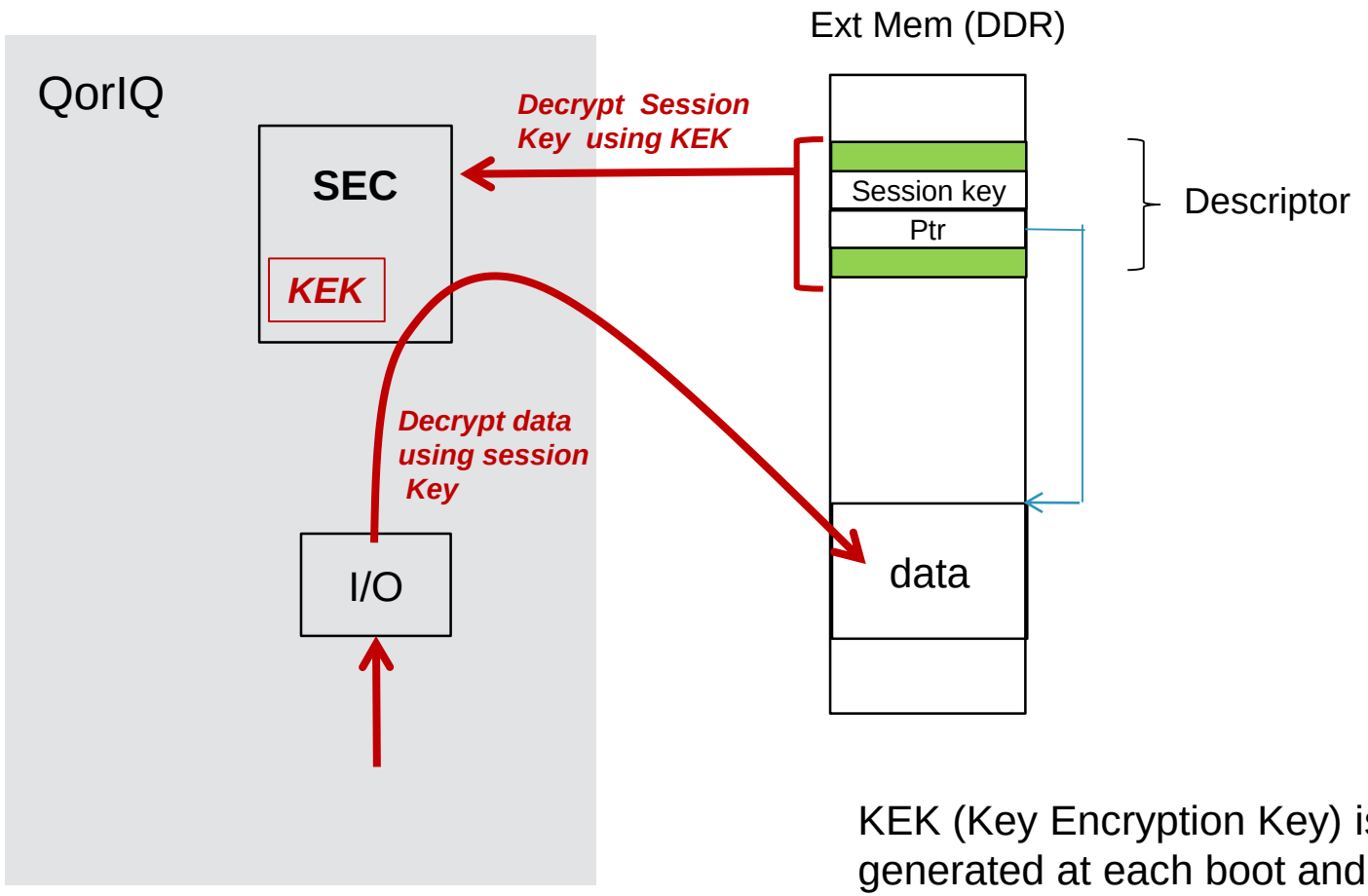
Sensitive Data Protection

- Trust Architecture provides support for protection of internal and external storage of developer-provisioned sensitive instructions and data. Implemented thru SEC assistance + some externally protected keys

- **Long-term secrets** = code/data/key in Non-volatile memory, decrypted for use in on-chip memory, do not change accross boot
  - ex: system private keys, pre-shared session keys, certificates , sensitive data …
  - Principle: Data + assoc. Key stored externally (notion of **Blob**) , key itself is encrypted with a AES Key derived from fused OTPMK, decrypted directly in the SEC without being ever exposed unencrypted externally

- **Short-term secrets** = code/data/key in external volatile memory, change accross boot
  - ex: volatile session keys, data/packets flowing through the system
  - Implemented thru SEC and a randomly initialized key **KEK** (Key Encryption Key)
  - Principle: KEK is initialized to a cryptographically secure random value after each boot cycle and is never exposed outside the SEC

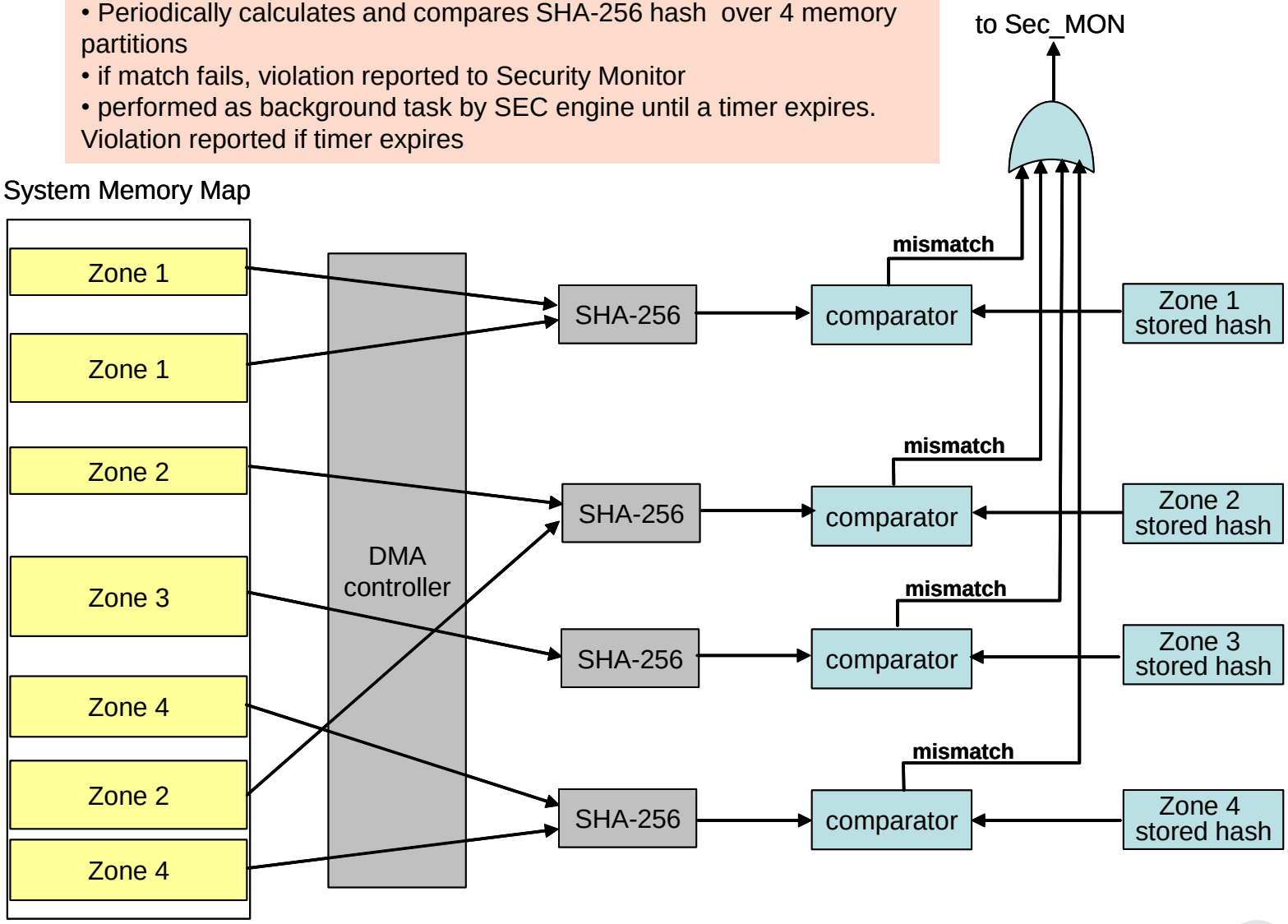
Protecting Short Term Secrets With SEC-4.0



KEK (Key Encryption Key) is generated at each boot and never exposed outside SEC

Run Time Integrity Checker

- Periodically calculates and compares SHA-256 hash over 4 memory partitions
- if match fails, violation reported to Security Monitor
- performed as background task by SEC engine until a timer expires. Violation reported if timer expires

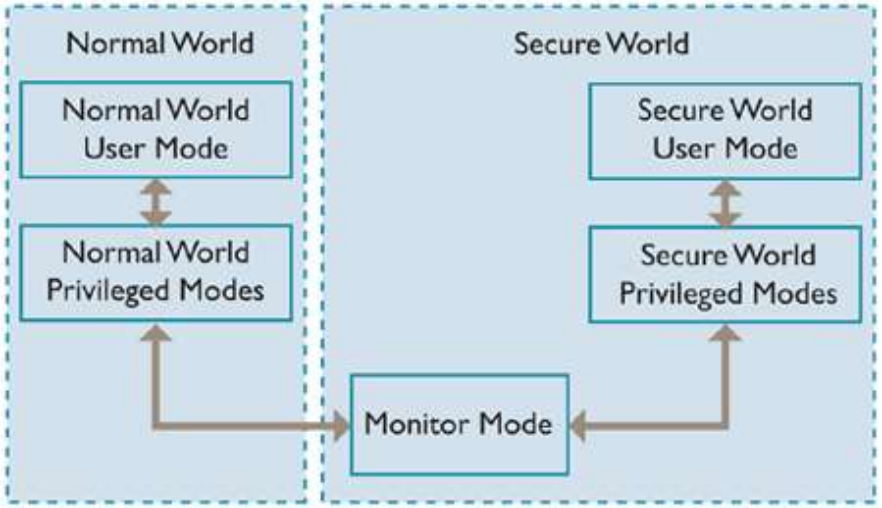


Trust Architecture (TA) and TrustZone (TZ)

TrustZone Concept/Foundation

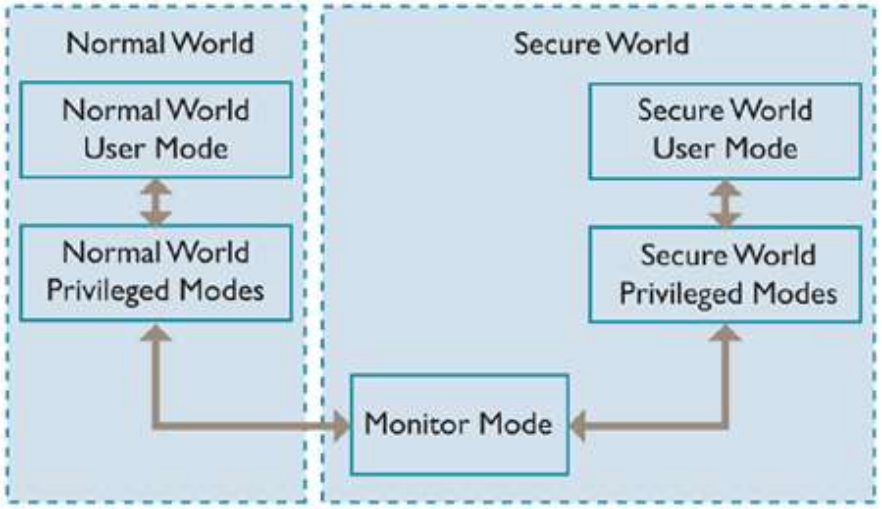
- Developed 10 years ago as a versatile & flexible Trust platform module Vs previous more specific TPM solutions
- Initially targeted to ARM Cortex-A main market eg. Mobile devices
- TrustZone uses a hardware-enforced security domain in order to systemize the implementation of secure systems.
- Typically, a device will run its rich conventional OS, like Linux or Android, in the **normal world**, while running a small vendor specific secure OS and its applications in the **secure world**
- Typical Secure Applications:
  - Secure PIN Entry
  - SIMLock Security
  - Terminal Identity
  - Over-the-Air Reprogramming
  - Managing Content
  - Virtual Private Networks
  - Digital Rights Management (DRM)

TrustZone – Principles (1)



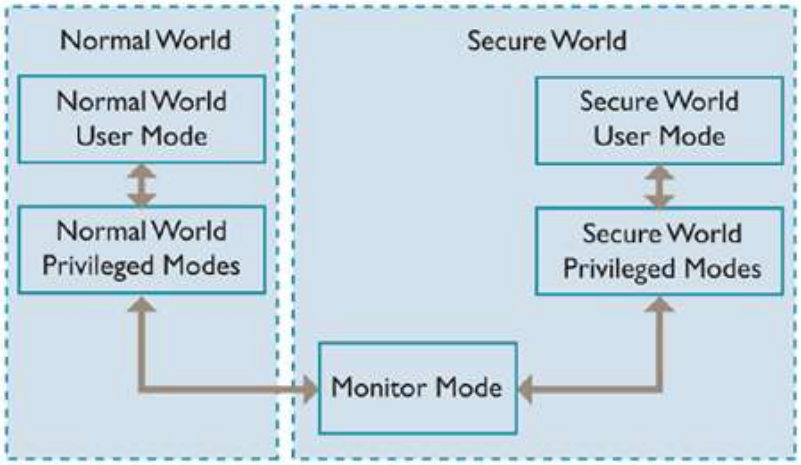
- The ARM GPP execution modes are divided into two “worlds”, secure and non-secure.
- The secure world supports a trusted execution environment, with:
  - A monitor mode dedicated to switching between worlds
  - Separate user and supervisor modes for secure applications and OS
  - Increasing levels of trust from user to supervisor
  - Secure debug modes, with isolation from non-secure world debug modes

TrustZone – Principles (2)



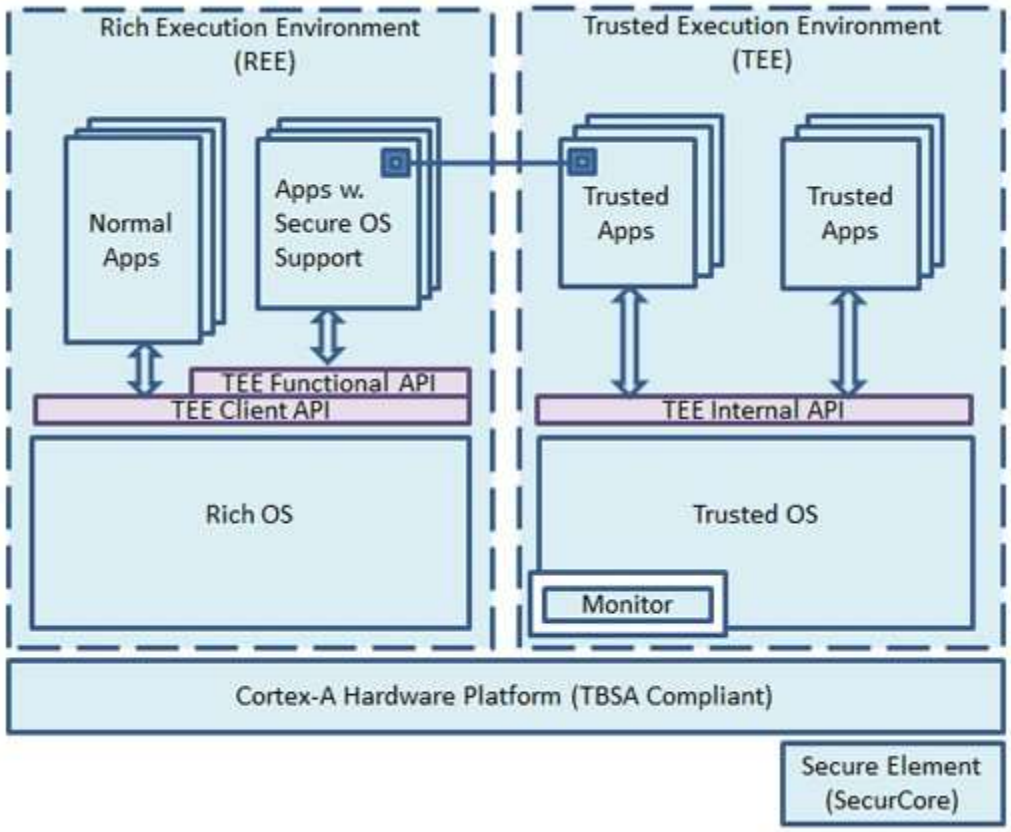
- Each of the physical processor core provides two virtual cores, one considered Non-secure and the other Secure, and a mechanism to robustly context switch between them, known as monitor mode.
- The value of the NS bit sent on the main system bus is indirectly derived from the identity of the virtual core that performed the instruction or data access.
- The Non-secure virtual processor can only access Non-secure system resources, but the Secure virtual processor can see all resources.

TrustZone – Principles (3)



- Monitor mode is used solely to mediate switching between secure and non-secure worlds. It should not be confused with any continuous or periodic platform monitoring function.
- The secure world does not include a hypervisor mode. Instead, it is useful to think of the secure world as an extra virtual machine beyond the reach of the hypervisor.
- The secure world is intended to provide strictly limited services requiring access to sensitive data or capabilities to the non-secure world via a well-controlled single point of entry.

TrustZone - TEE Model



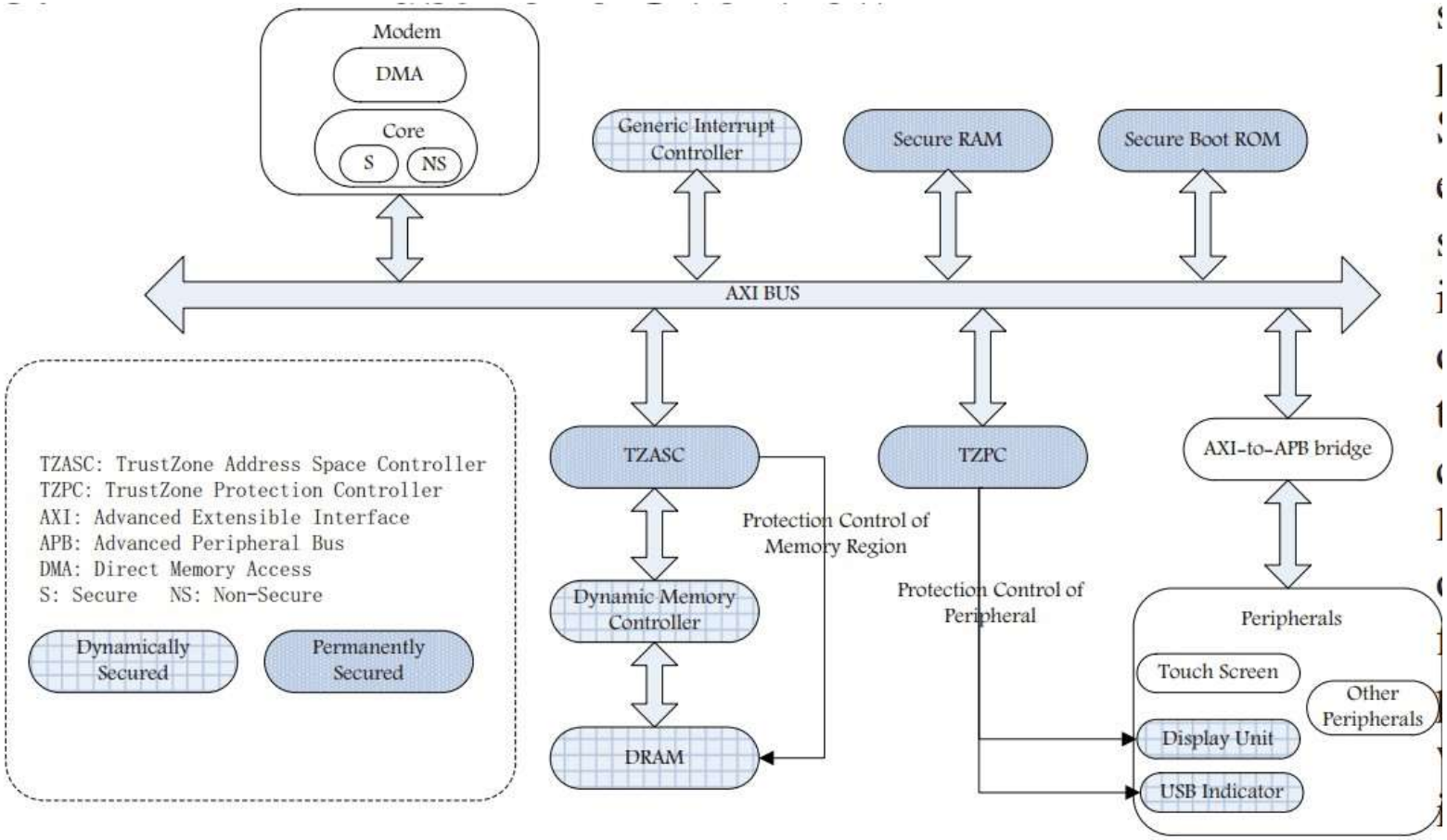
Note: showing a single core / single REE platform

TrustZone Hardware - Principles

At HW level, TrustZone is enforced at several physical blocks/IPs

- At ARM core: execution context, registers banking and a few extra instructions for switching between S/NS world
- The AXI bus adds an extra ‘’address’’ signal (NS) to differentiate between trusted and non-trusted requests. Every SoC I/O block connected to the AXI has to respect this signaling
- TrustZone Address Space Controller (TZASC) and TrustZone Protection Controller (TZPC): for setting security access permission, acting as firewalls for the various SoC resources.

TrustZone Hardware



Considerations on TA,TZ, HV in Multicore SoC (1)

- Use Trust Architecture Secure Boot to secure/authenticate those SW entities that need to be trusted. If TZ is implemented for secure/non-secure world, this should include the TZ secure world SW.

- TZ architecture has been designed for and supports a one-level of asymetric partitioning with one very trusted partition and .. the rest.

- Depending on the type of SoC (single/multicore) and usecase, the rest can range from very simple to more sophisticated models with their own security concerns

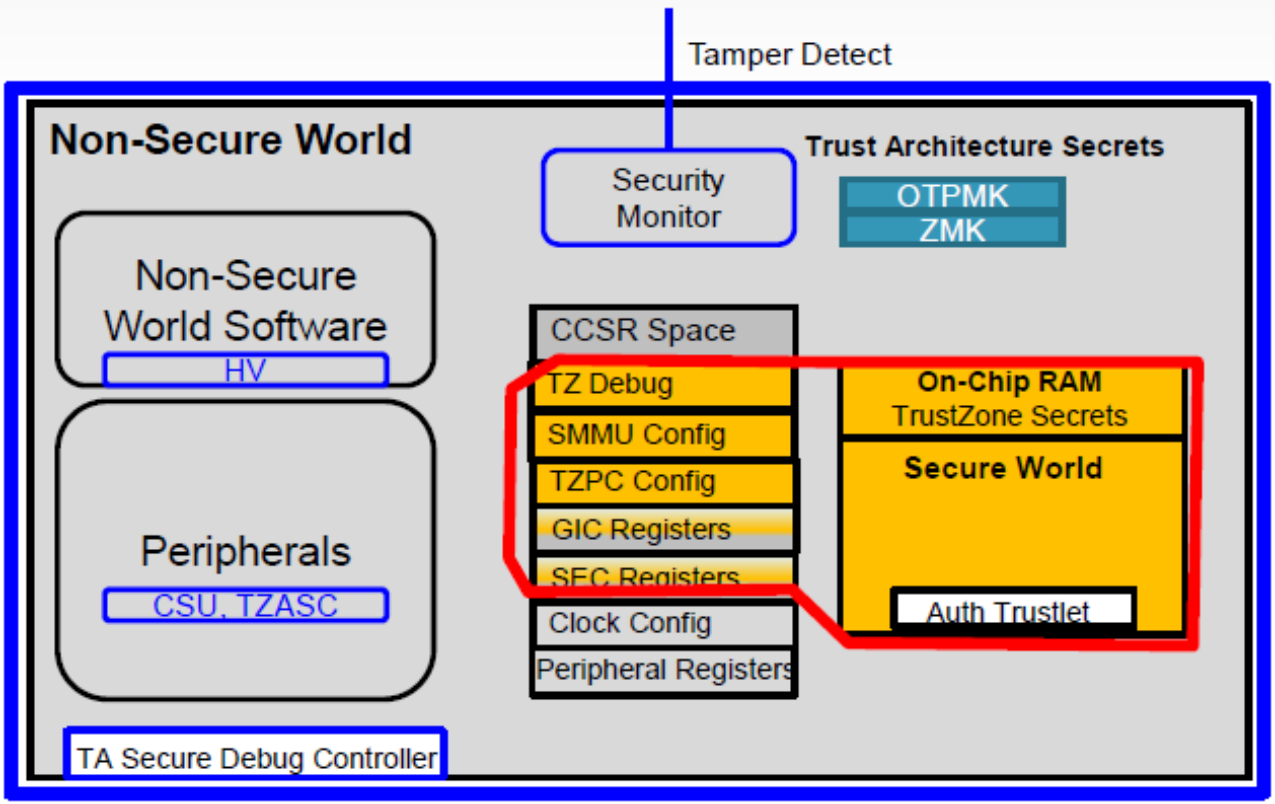
Considerations on TA,TZ, HV in Multicore SoC (2)

- Multipartition implementation with rich-set partitions allowed by multicore SoCs is better supported thru HV architecture and related built-in protections and virtualization means (eg: 3-level hierarchy, MMU + IOMMU)

- Use Trust Architecture toolbox to ensure the whole system keeps trusted during operation.

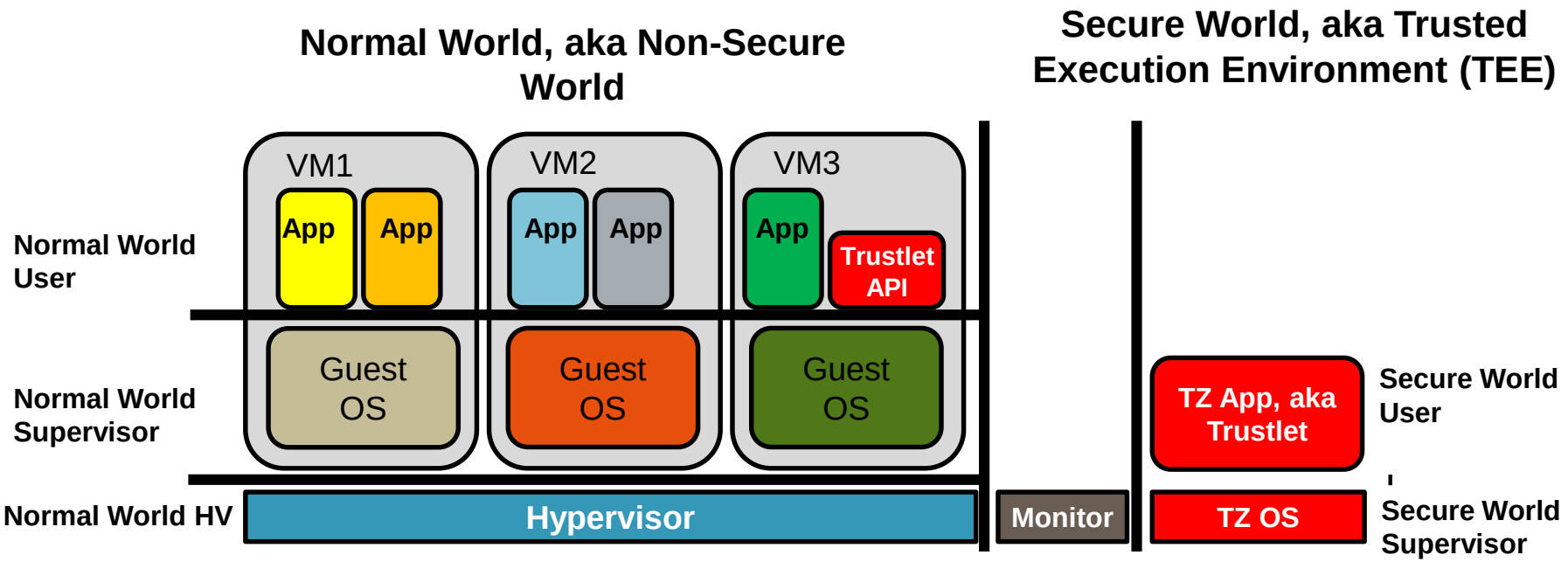
- Secure Debug restriction can be implemented thru TA and/or TZ

Trust Architecture + ARM TrustZone
Complementary Technologies



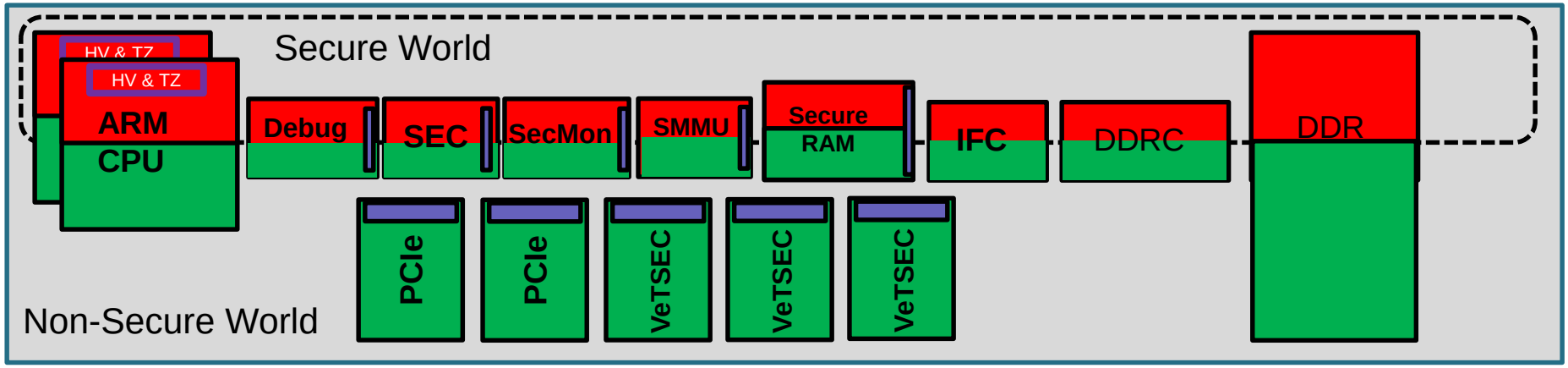
- **TrustZone** provides an inner keep for especially trusted software, it doesn't protect the whole SoC, it defends the Secure World /TEE
- **Trust Arch** provides a secure perimeter for trusted software across the whole SoC

TA+TZ in Multicore / Multi-Partition Context



Hypervisor not required, non-secure world can be single OS.

Trust Architecture with TrustZone



ARM Terminology

Secure, Secure World – A parallel execution environment, isolated from normal, non-secure world software. ARM CPUs come out of reset running in Secure World.

Non-Secure, Non-Secure World – SW running in any mode other than TZ (HV, Guest, User)

Trusted Execution Environment – (TEE), the TrustZone RTOS & Monitor. The TEE + trustlets make up Secure World.

Trustlet –
An application running in the TEE. Trustlets have access to the crown jewels, and have been developed for digital wallet, DRM, and device authentication.

Trust Architecture Terminology

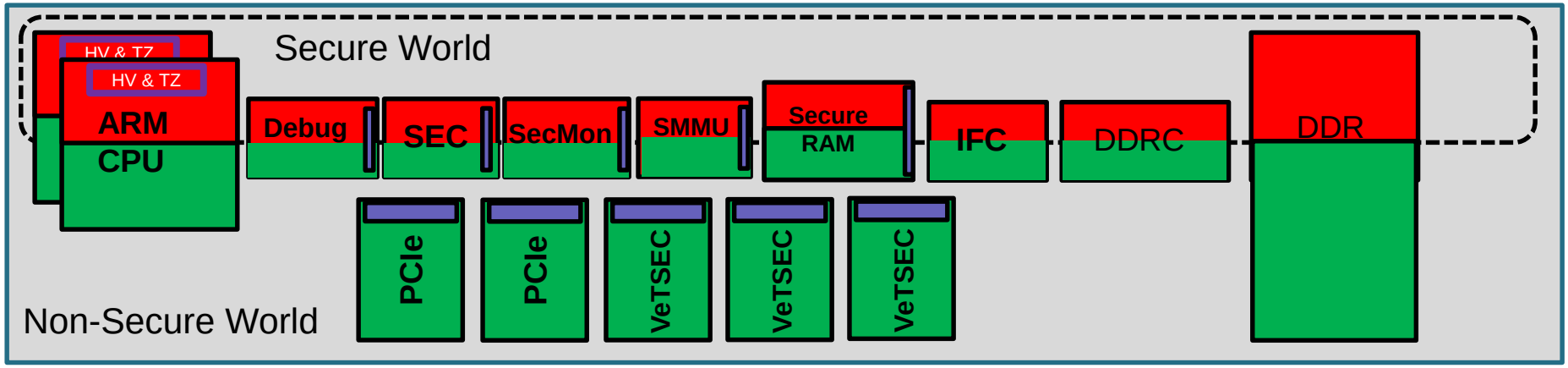
Secure – State of the HW SecMon in which trusted software can tell the SEC to use the crown jewels.

Non-Secure – State of the HW SecMon in which software can't tell the SEC to use the crown jewels (SW is untrusted).

Trusted – SW which has passed secure boot validation is Trusted to tell the SEC open the treasure chest.

Trusted/Privileged – SW which, in addition to being Trusted to do its normal applications, is allowed to access Trust Arch HW registers.

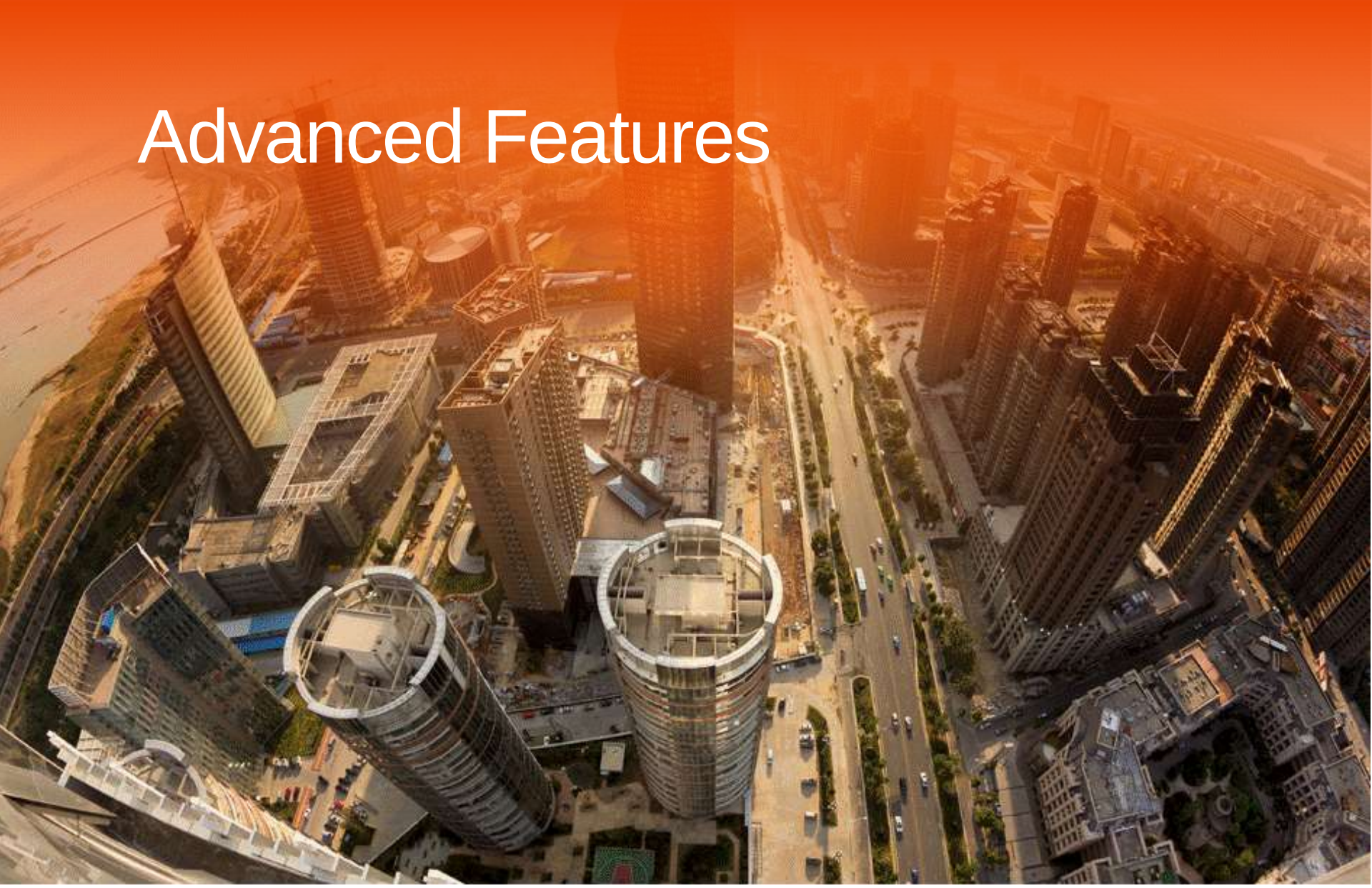
Who Trusts Who?



- Can ARM Non-Secure World be Trusted?
  - QorIQ's secure boot validates the ARM TEE (Red), plus Hypervisor & Guest Oses (Green) which run in Non-Secure world. From the QorIQ HW Security Monitor's perspective, the validated SW running in Non-Secure World is as trusted as the ARM TEE to command the SEC to use the secret key. But TrustZone access protections will stop Green from accessing Red resources, despite Green being trusted.

- Can ARM Secure World be untrusted?
  - ARM CPUs come out of reset in Secure World. The SW running on Secure World doesn't have to be validated, and consequently, the QorIQ HW Security Monitor doesn't trust it. Even if the Secure World software is successfully validated, if a hardware security violation is detected, the whole SoC and all software is considered untrusted (Fail state).

Advanced Features



Trust Architecture – Features Summary

TRUST Version	Trust 1.0	Trust 1.1	Trust 2.0	Trust 2.1	Trust 3.0
Related devices ('E' devices)	P4080, P1010	P2040, P3041, P5020	T1040, T2080, T4240, B4860, C290	LS1020, LS1043	LS2080, LS1088
Base Features					
Secure Boot	Y				
Secure Boot -offloaded thru SEC (HW accel.)	N		Y		
Secure Debug Controller	Y			Y + TrustZone "Secure World" add'l protections	
Security Fuse Processor (SFP)	Y				
Security Monitor	Y				
Security Monitor Dual-power sections (incl. Key zeroization)	N	Y	N	Y	
External Tamper Detection	Y				
Real-Time Integrity Checker (RTIC)	Y				
SEC-supported Blobs based on Master Key	Y				
SEC-supported Ephemeral Key Encryption Keys	Y				
CPU Memory Access Control	Power ISA MMU w/HV			ARM ISA MMU w/HV and TrustZone	
I/O Memory Access Control	Platform MMU (PAMU)			Platform MMU (SMMU)	
ARM TrustZone	N			Y	
Advanced Features					
Secure Boot - Alternate (secondary) signed Image	N		Y		
Secure Boot - Key list and Key revocation	N		Y (List of 4 keys)	Y (List of 8 keys)	
Monotonic Counters	N		1 (not in T1 & B4)	1	
HW Key Pair (aka Trusted Mfg)	N			Y	

Trust Architecture Enhanced Features

Differences between Trust 1.x and 2.0 (T-series):

- Reduced secure boot time by offloading cryptography to the SEC
  → full ISBC SW implementation with Trust 1.0
- Support for a primary and alternate (secondary) signed image
  → ISBC attempts to validate a secondary image if the primary fails
- Support for revocation of super root keys
- Monotonic counter
  → battery-backed zeroisable persistent secret value
- Battery-backed, general purpose storage registers

Added in Trust 2.1 & 3.0 (LS-series) :

- Trusted Manufacturing

Trust Architecture Rev2.1/3.0 (LS-Series)

Manufacturing Protection

The Trust Architecture 3.0 supports a manufacturing protection feature OEMs may use to gain the following assurances:

- The OEM is building systems using legitimate Freescale chips of the correct type.
- The contract manufacturer is burning the chip's fuses correctly (especially the SRKH)
- The system has booted securely and authorized OEM software is running on the chip prior to provisioning of credentials
- OEM additional secrets can safely be downloaded to the chip
- The manufacturing protection process includes steps taken at the OEM once per production lot and steps taken at the contract manufacturer (or upon field installation).

www.Freescale.com