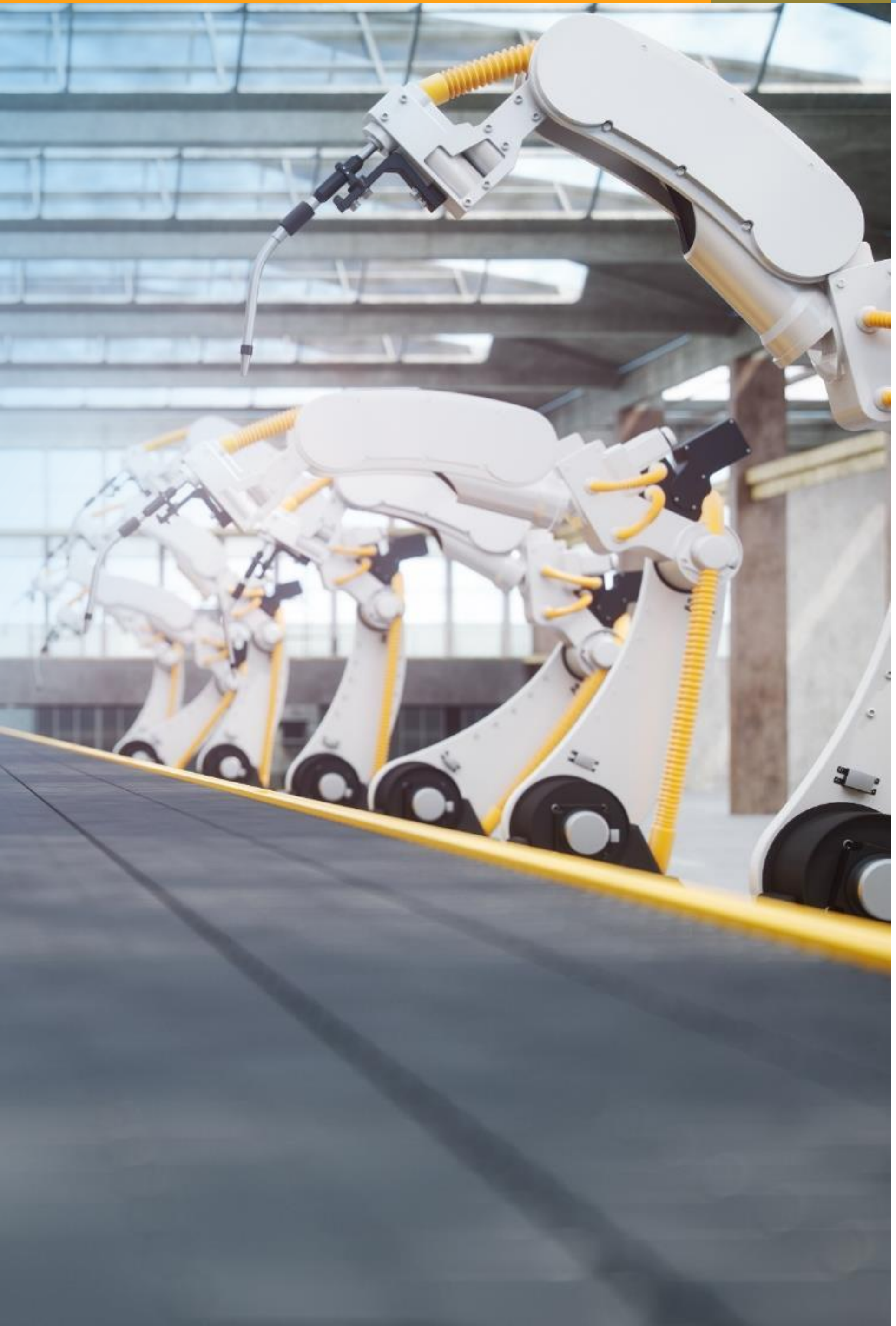
NXP MCU SECURITY SOLUTION

David Chen/ NXP MCU FAE

AUG 2021

SECURE CONNECTIONS
FOR A SMARTER WORLD

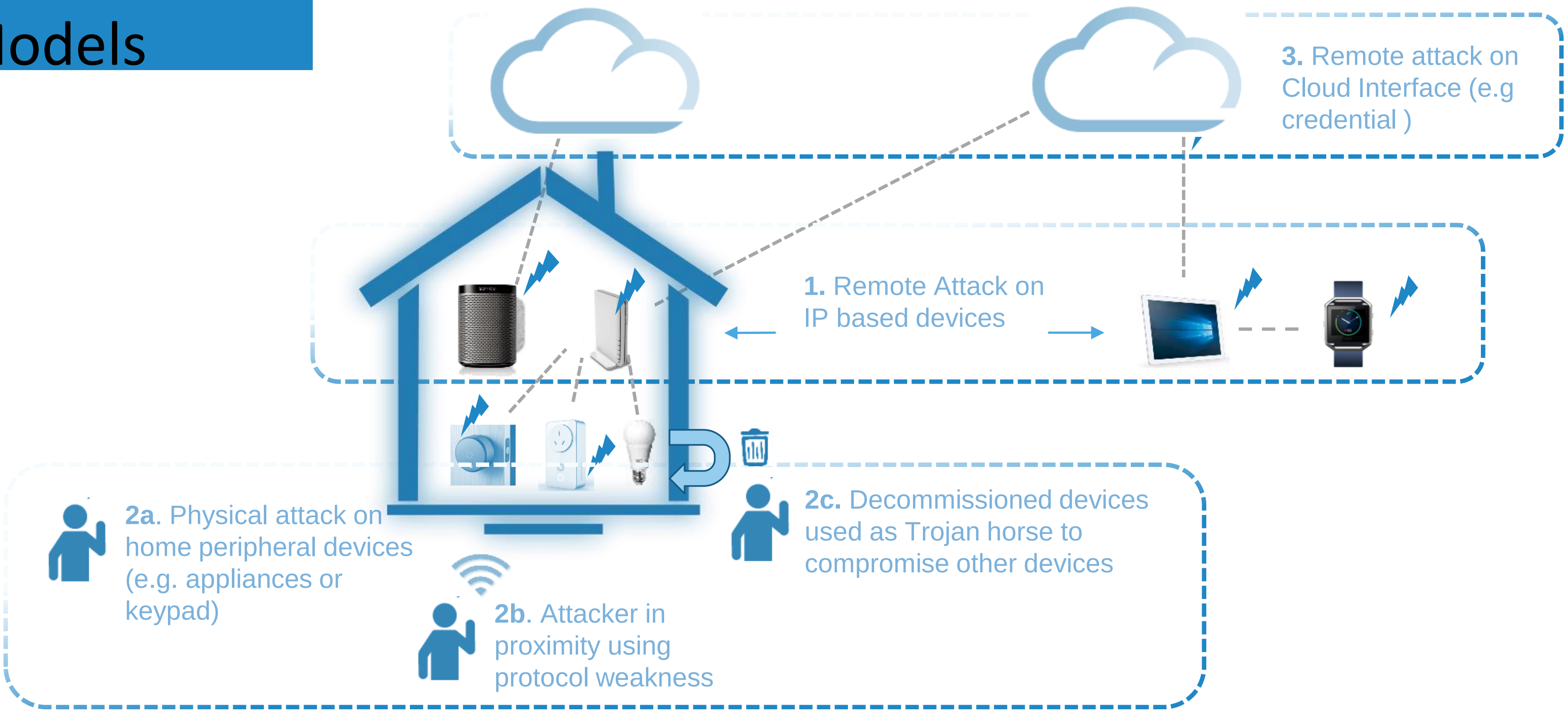
PUBLIC



Agenda

- Start With a Security Model
- Secure Boot Architecture
- LPC55Sxx Secure Technology
  - Secure AHB Bus Matrix
  - TrustZone-M
  - PRINCE
  - SRAM PUF
- MCUXpresso ENABLEMENT

Home Automation Threat Models



3. Remote attack on Cloud Interface (e.g credential)

1. Remote Attack on IP based devices

2a. Physical attack on home peripheral devices (e.g. appliances or keypad)

2b. Attacker in proximity using protocol weakness

2c. Decommissioned devices used as Trojan horse to compromise other devices

1 Remote Attacks on **Gateway/Bridge/IP-based Devices**

Reach of single attack: ● ● ◐

2 Local Attacks on **Nodes**

Reach of single attack: ● ○ ○

3 Remote Attacks on **Cloud Service Providers**

Reach of single attack: ● ● ●

START WITH A SECURITY MODEL

Policies

The rules in place that identify the data that should be protected

For example
The management of firmware, secret keys, user and application data Passwords, personal information, network credentials

Threat landscape

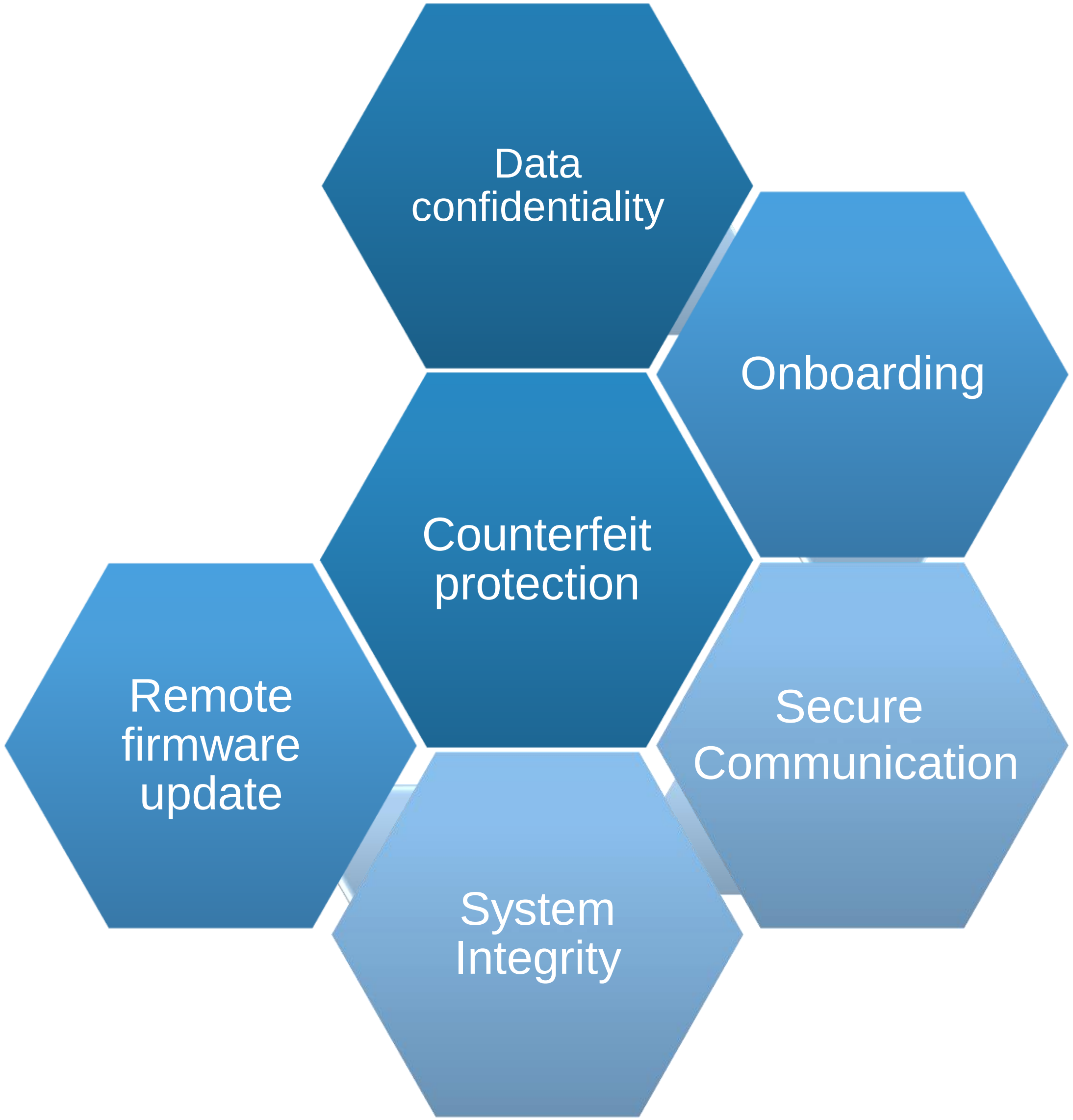
The definition of the attacks and attackers that the end device will face and protect against. Considers the access to the device, and cost of the attack

For example
Expert attackers who will use off the shelf tools to gain access and insert malware

Methods

The means by which the policies for the device are enforced. Involves the application of security technology to achieve product goals

For example
Disabling debug access to restrict the availability of secret data on a processor



ESSENTIAL SECURITY GOALS

- Counterfeit protection
  - Every device has a unique identity that can not be reproduced by an attacker
- Onboarding
  - Shared credentials between the end device and the back end system
- System integrity
  - Trust in the functionality provided by the end device
- Secure communication
  - Cryptography applied to the communications for the device
- Data confidentiality
  - Protection of data in the device
- Remote firmware update
  - Safe updating of the software on the end device

ALIGNMENT TO SECURITY GOALS

Counterfeit protections

With the authenticated and encrypted boot, security is enforced by a unique secret available only to the individual chip (OTPMK)

As the application code must be linked to the chip for it to be used, this establishes a link between known devices and the application functions to protect against clones

Onboarding

Chip specific unique and protected keys along with secure boot flow protect OEM installed cloud credentials

During manufacturing cloud credentials are encrypted with chip specific unique & protected keys

Cloud credentials become part of the secure boot image that is protected for integrity and confidentiality

System Integrity

Secure boot functions upon every reset and is the foundation for establishing trust in the device operation

Chip hardware and ROM provides an immutable secure boot flow to support recovery from system run away scenarios once the device is rebooted.

ALIGNMENT TO SECURITY GOALS (CONTINUED)

Secure Communication

Authenticated application code includes TLS Stacks (WolfSSL or Arm MbedTLS)

Option for AES engine to use OTP or application generated keys

Hardware acceleration for AES and SHA-2 (SHA-256) with DCP key protection

Data Confidentiality

Based on device policies, data stored in system is protected by hardware managed keys

Option for AES engine to use OTP or Applicaiton generated keys

Hardware acceleration for AES and SHA-2 (SHA-256)

Secure firmware update

New firmware applied to the system must pass the secure boot flow

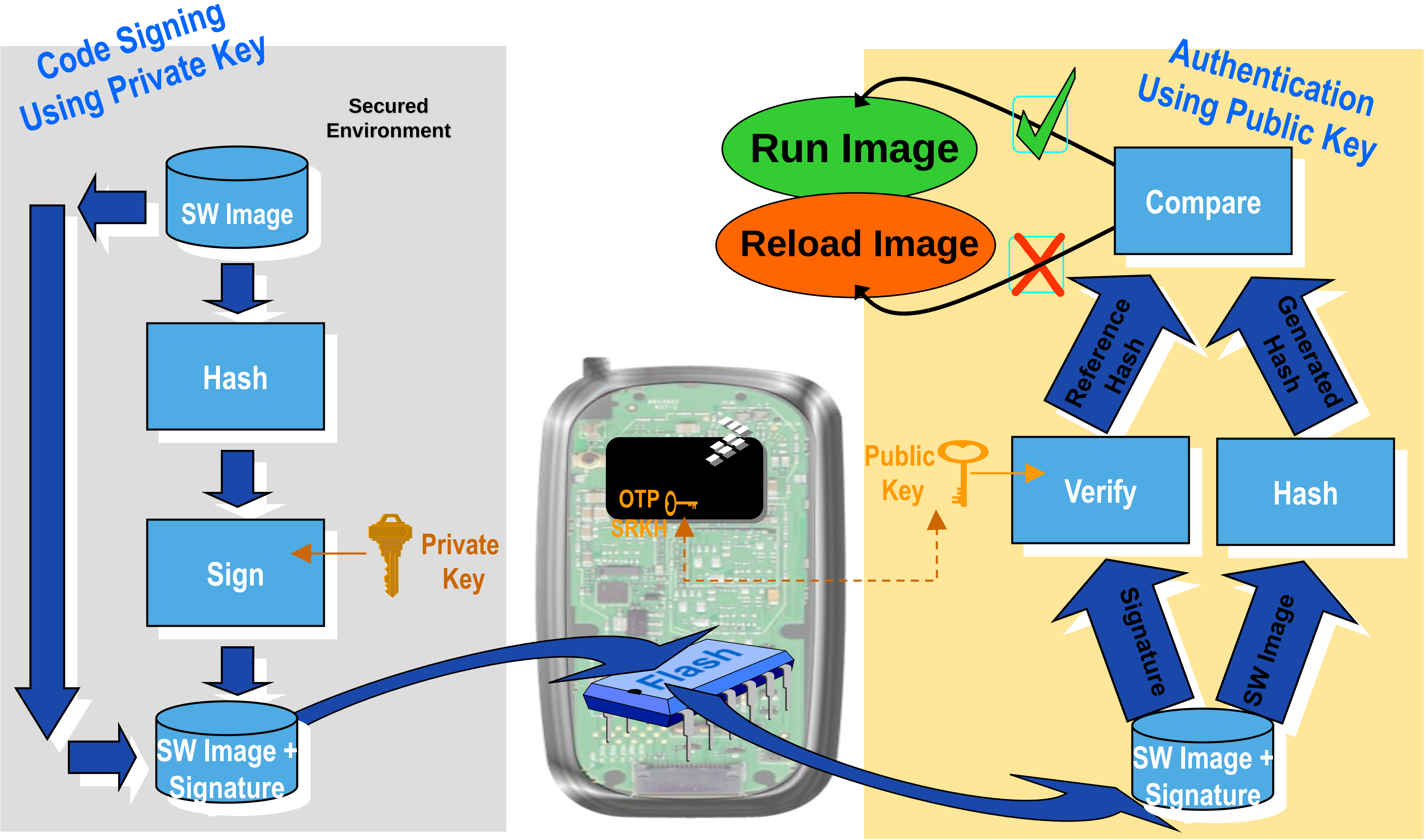
ROM support for up to 3 revocations using SRK Revoke

Secure Boot Architecture

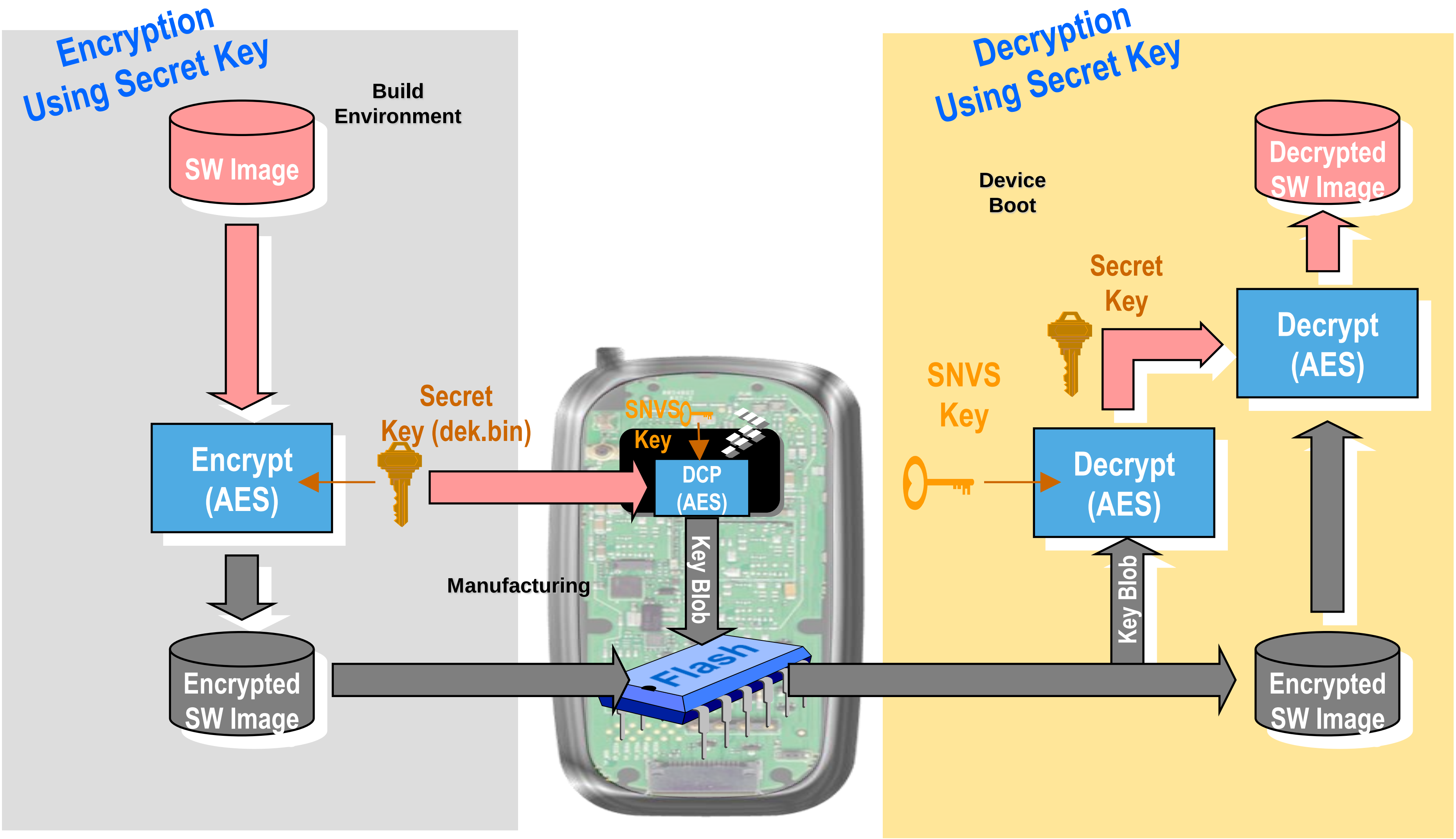
- Components of a Secure Boot

SECURE CONNECTIONS
FOR A SMARTER WORLD

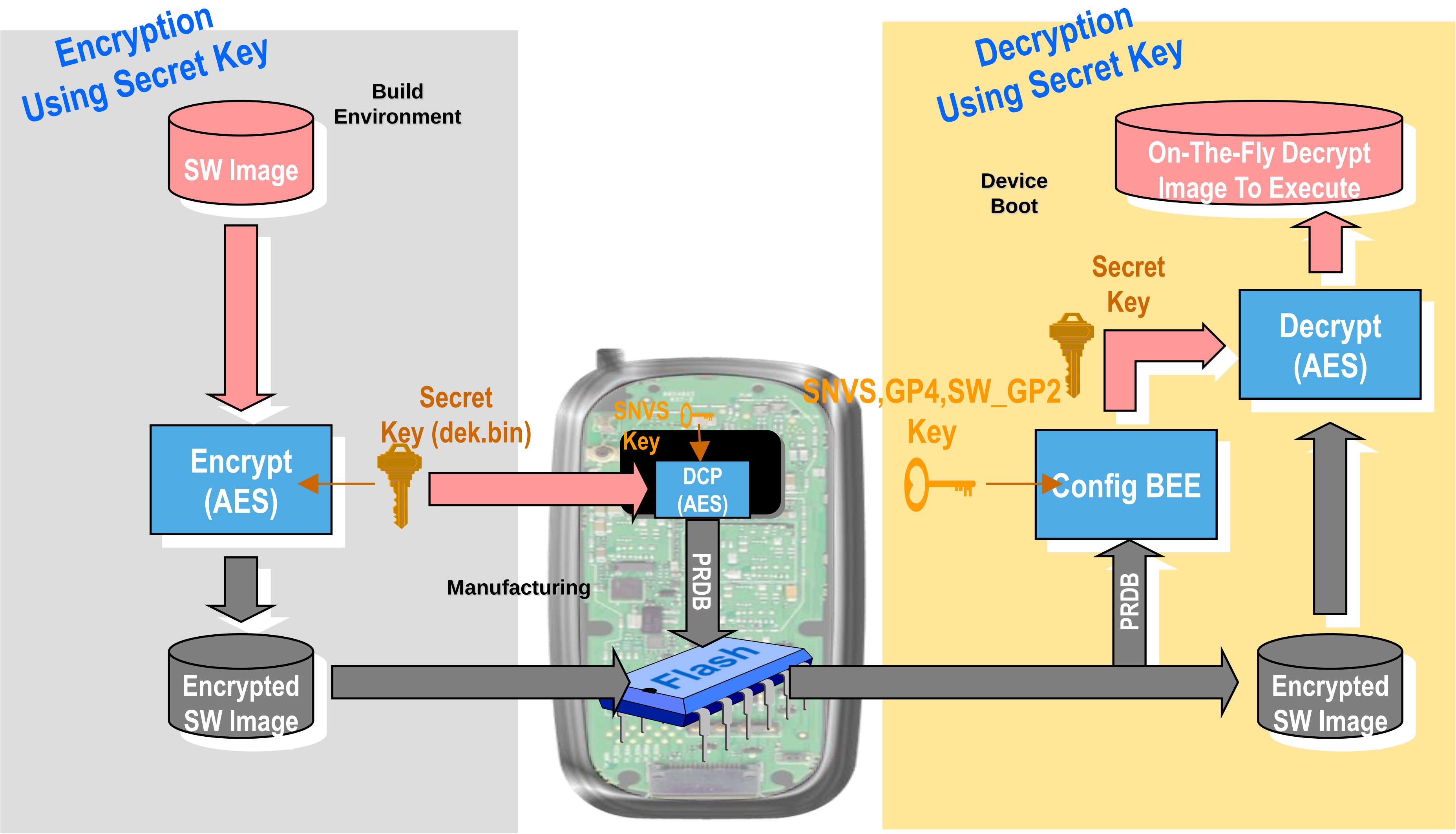
HIGH ASSURANCE BOOT - AUTHENTIC



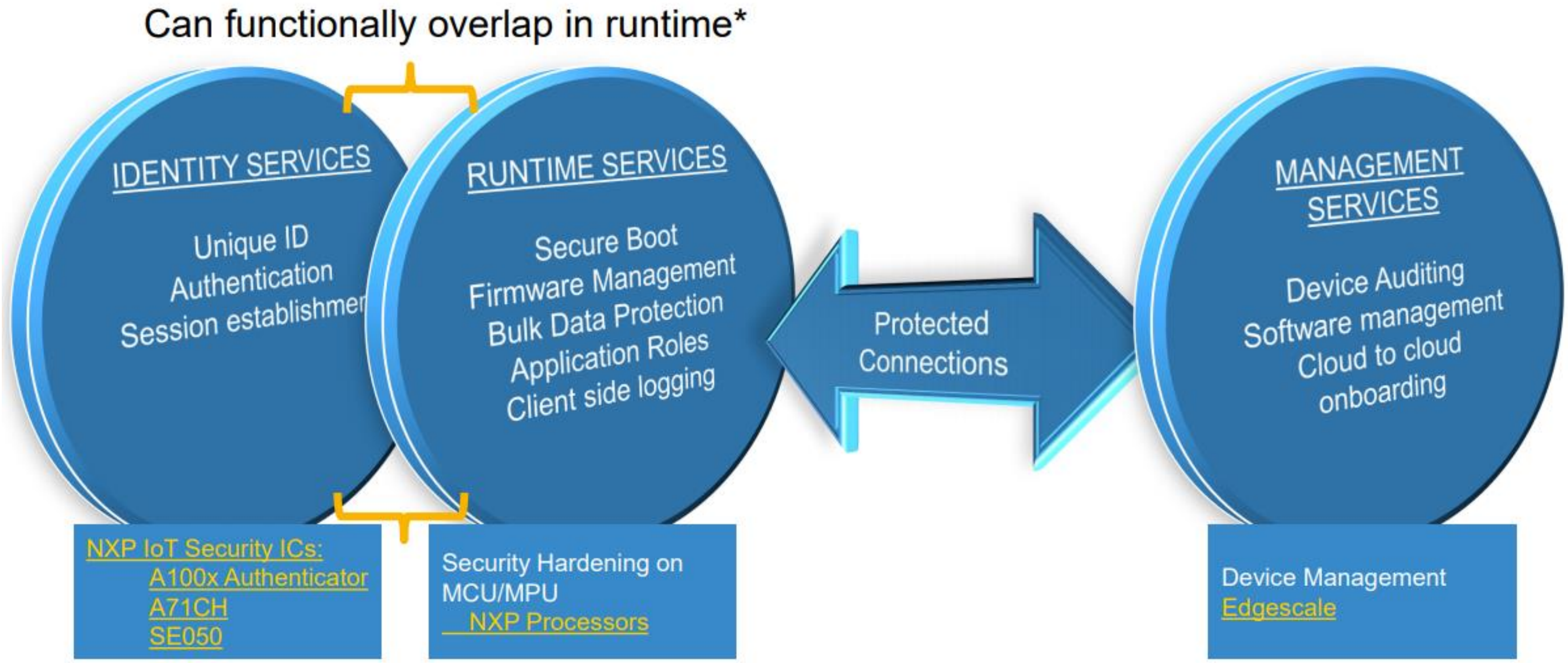
HIGH ASSURANCE BOOT – ENCRYPTED



ON-THE-FLY DECRYPT IMAGE TO EXECUTE



NXP FOR SECURE DEPLOYMENT FROM EDGE TO CLOUD


Can functionally overlap in runtime*

IDENTITY SERVICES

Unique ID
Authentication
Session establishmen

RUNTIME SERVICES

Secure Boot
Firmware Management
Bulk Data Protection
Application Roles
Client side logging

Protected Connections

MANAGEMENT SERVICES

Device Auditing
Software management
Cloud to cloud onboarding

NXP IoT Security ICs:
A100x Authenticator
A71CH
SE050

Security Hardening on MCU/MPU
NXP Processors

Device Management
Edgescale

*With secure provisioning identity can be established in a processor

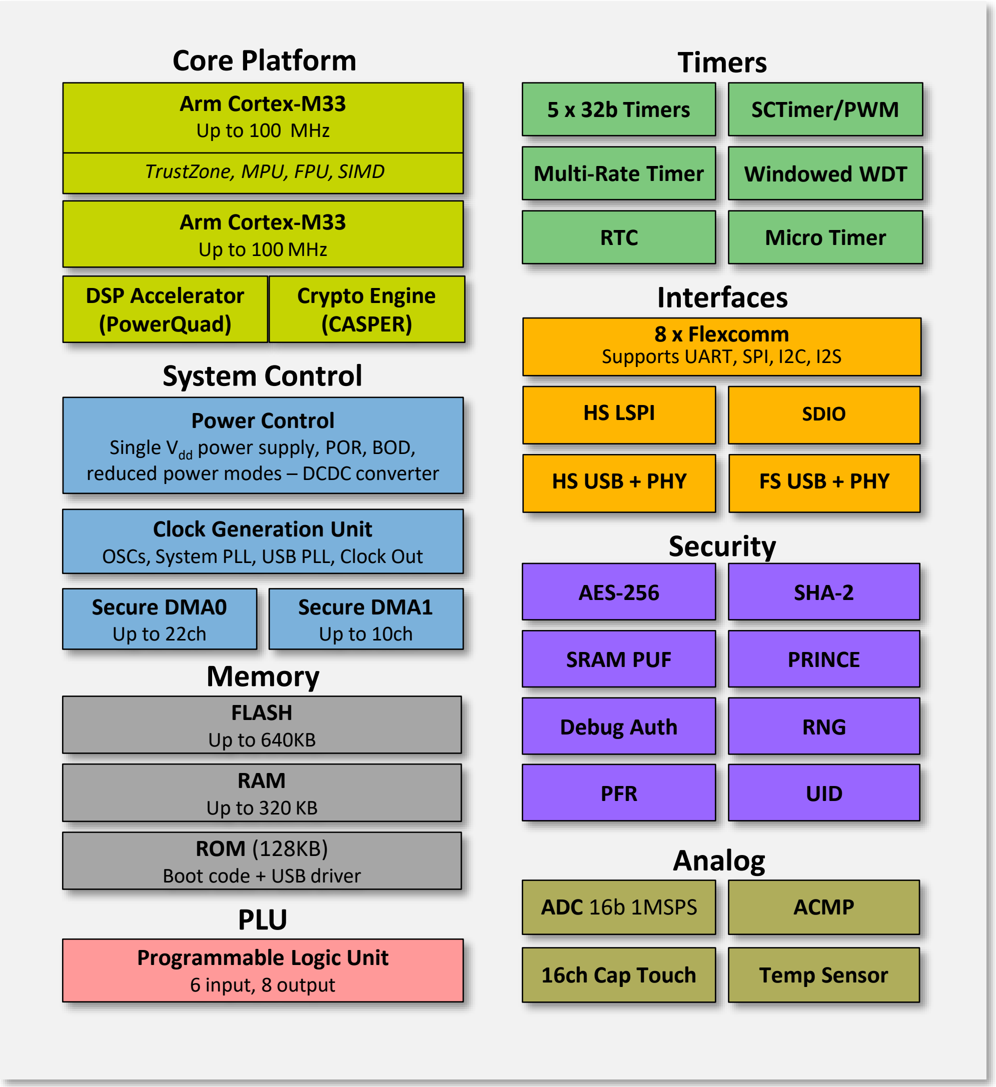
•

LPC55Sxx Security

SECURE CONNECTIONS
FOR A SMARTER WORLD

PUBLIC

LPC5500 PRODUCT OVERVIEW



Core Platform

Arm Cortex-M33
Up to 100 MHz

TrustZone, MPU, FPU, SIMD

Arm Cortex-M33
Up to 100 MHz

DSP Accelerator (PowerQuad)

Crypto Engine (CASPER)

System Control

Power Control
Single V_{dd} power supply, POR, BOD, reduced power modes – DCDC converter

Clock Generation Unit
OSCs, System PLL, USB PLL, Clock Out

Secure DMA0
Up to 22ch

Secure DMA1
Up to 10ch

Memory

FLASH
Up to 640KB

RAM
Up to 320 KB

ROM (128KB)
Boot code + USB driver

PLU

Programmable Logic Unit
6 input, 8 output

Timers

5 x 32b Timers

SCTimer/PWM

Multi-Rate Timer

Windowed WDT

RTC

Micro Timer

Interfaces

8 x Flexcomm
Supports UART, SPI, I2C, I2S

HS LSPI

SDIO

HS USB + PHY

FS USB + PHY

Security

AES-256

SHA-2

SRAM PUF

PRINCE

Debug Auth

RNG

PFR

UID

Analog

ADC 16b 1MSPS

ACMP

16ch Cap Touch

Temp Sensor

Core Platform

- Up to 100MHz Cortex-M33
  - TrustZone, MPU, FPU, SIMD
- Up to 100MHz Cortex-M33
- Coprocessors
  - DSP Accelerator (PowerQUAD)
  - Crypto Engine (CASPER)
- Multilayer Bus Matrix

Memory

- Up to 640KB FLASH
- Up to 320KB RAM
- 128KB ROM

Timers

- 5 x 32b Timers
- SCTimer/PWM
- Muiti-Rate Timer
- Windowed Watchdog Timer
- RTC
- Micro Timer

Interfaces

- USB High-speed (H/D) w/ on-chip HS PHY
- USB Full-speed (H/D), Crystal-less
- SDIO, Support 2 cards
- 1 x High-Speed SPI up to 50MHz (HS LSPI)
- 8 x Flexcomms support up to 8x SPI, 8x I2C, 8x UART, 4x I^2S channels (total 8 instances)

Advanced Security

- AES-256 HW Encryption/Decryption Engine
- SHA-2
- SRAM PUF for Key Generation support
- PRINCE – On-The-Fly Encrypt/Decrypt for flash data
- Debug authentication
- RNG
- Protected Flash Region (PFR)

Analog

- 16b ADC, 16ch, 1MSPS
- Analog Comparator
- 16ch Cap Touch Controller
- Temperature Sensor

Packages

- TFBGA100, 6x6x0.9
- HLQFP100, 14x14x1.4

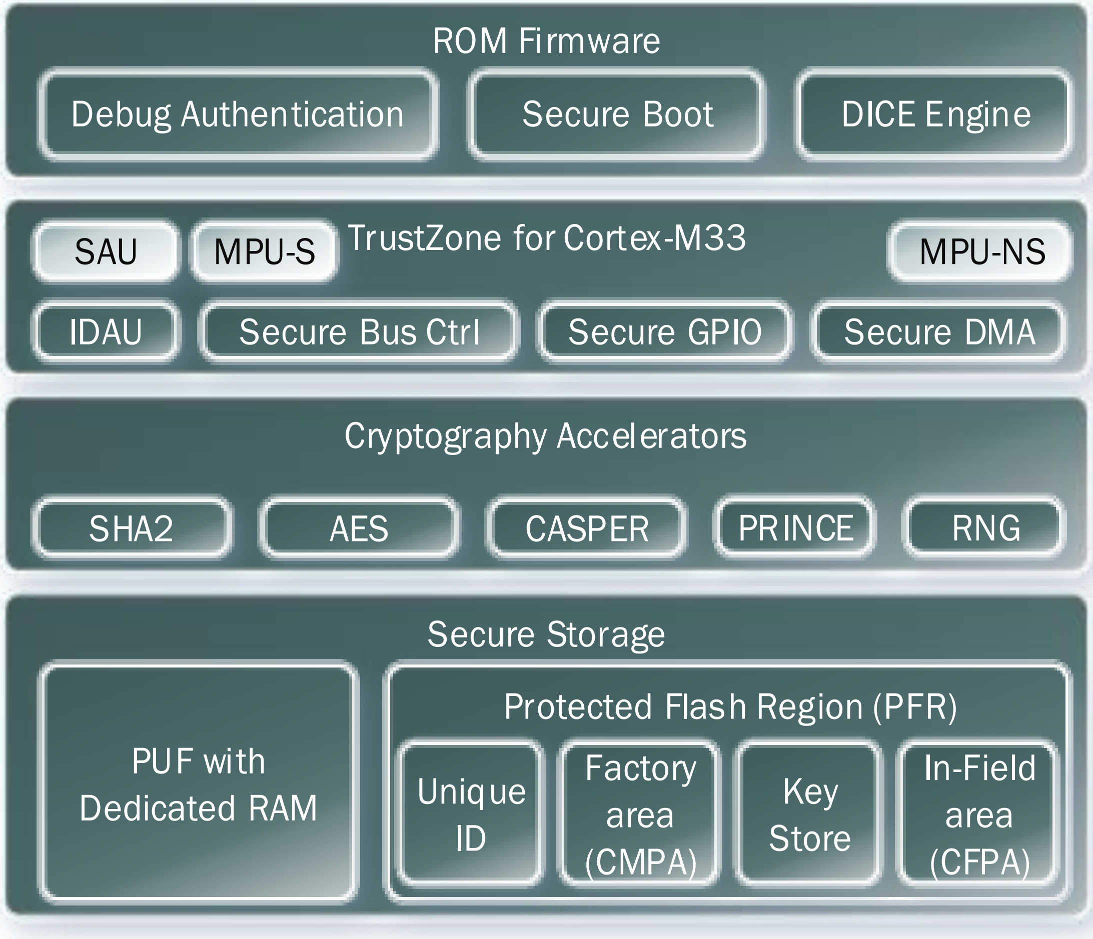
Other

- Programmable Logic Unit
- Buck DC-DC
- Operating voltage: 1.71 to 3.6V
- Temperature range: -40 to 105 °C

* Refer to Datasheet for features available on each package

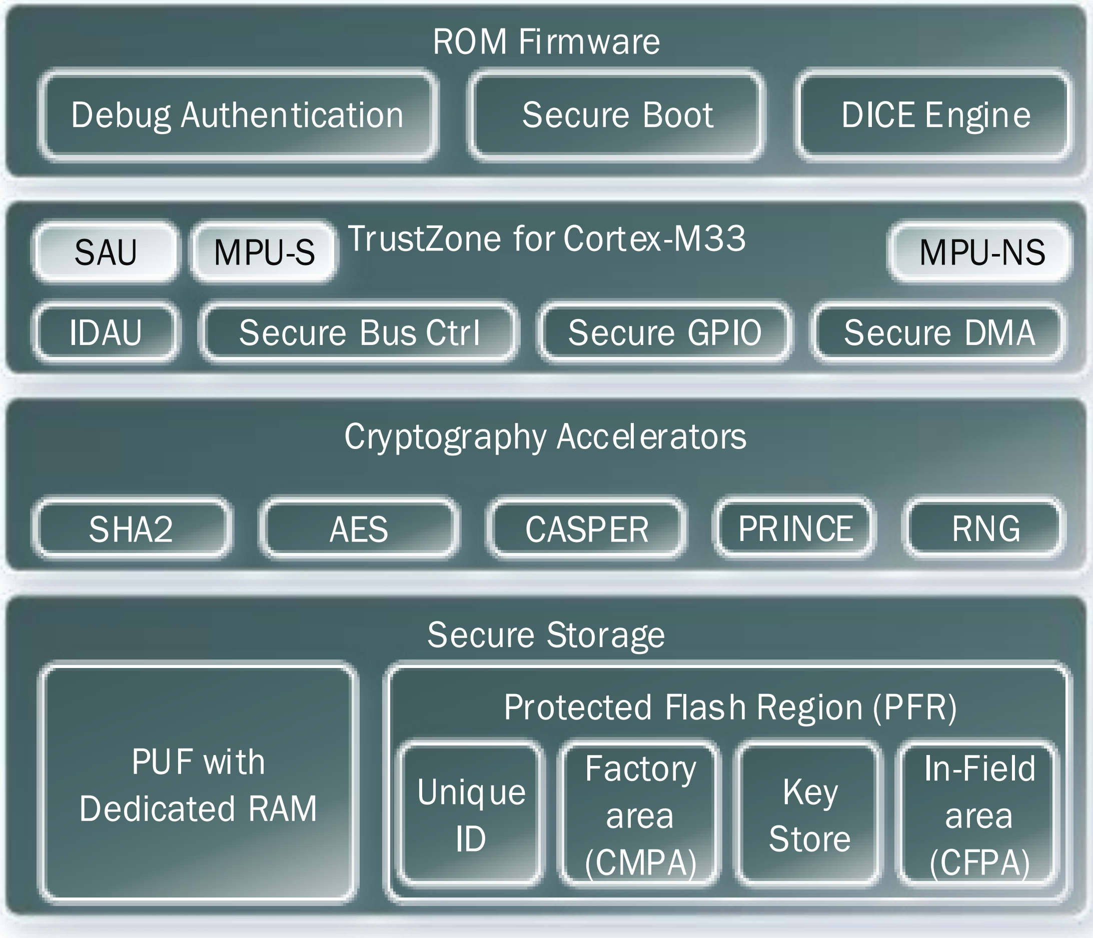
LPC55SXX SECURITY SUB-SYSTEM

- Secure Storage
  - Physically Unclonable Function (PUF)
    - Device unique root key (256 bit strength)
    - Can store key sizes 64 bit to 4096 bit
  - Protected Flash Region
    - RFC4122 compliant 128-bit UUID per device
    - PUF Key Store
      - Activation code, Prince region key codes, FW update key encryption key, Unique Device Secret
    - Customer Factory Programable Area
      - Boot Configuration, RoT key table hash, Debug configuration, Prince configuration
    - Customer In-Field Programable Area
      - Monotonic counter, Prince IV codes



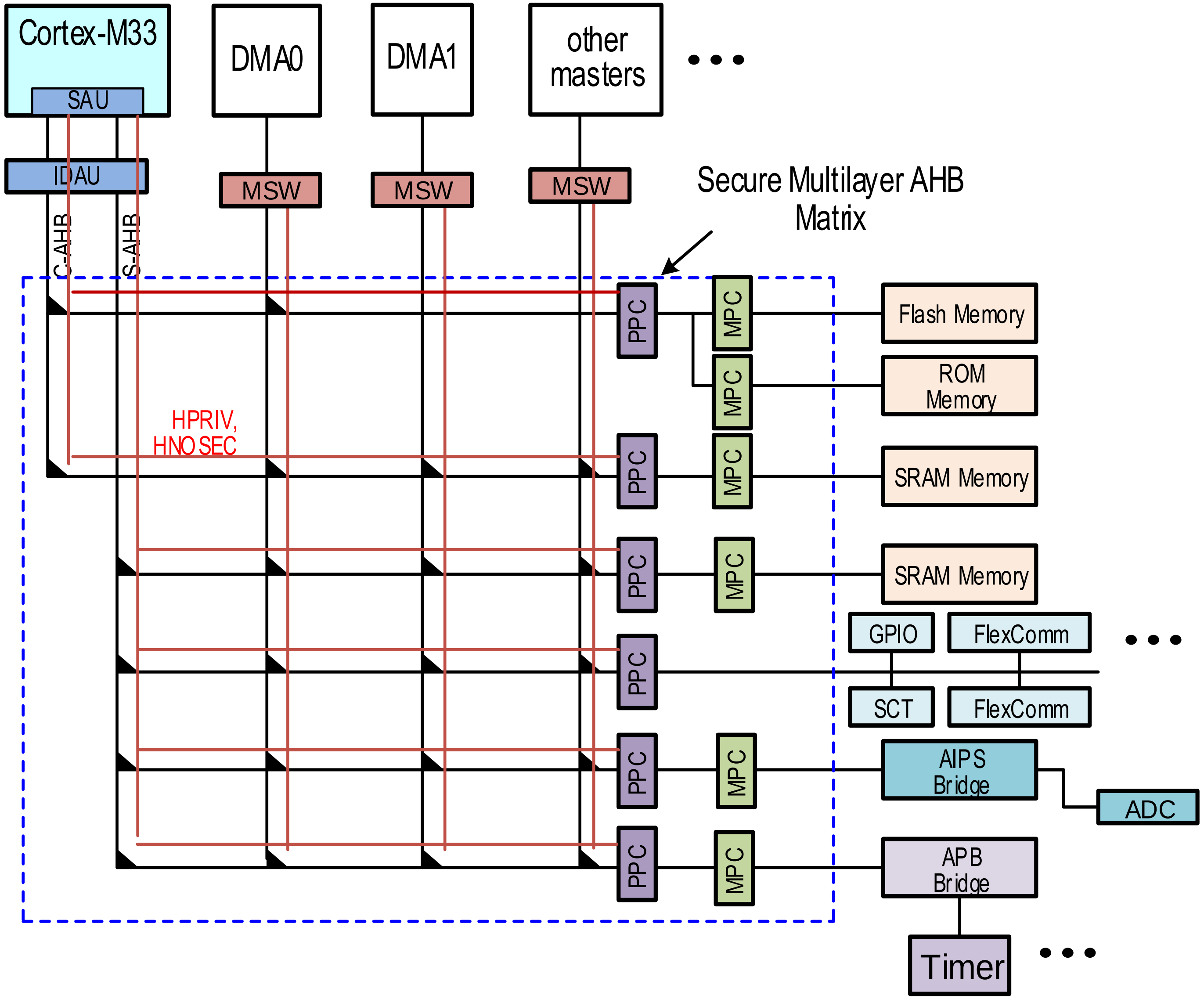
LPC55SXX SECURITY SUB-SYSTEM (CONTD…)

- ROM supporting
  - Secure Boot
  - Debug Authentication
  - DICE Engine
- TrustZone for Cortex-M33
  - Security Attribution Unit (SAU)
  - Memory Protection Unit (MPU): Secure & Non-Secure
  - NXP IP
    - Defined Attribution Unit (using IDAU interface)
    - Secure Bus Control
    - Secure GPIO Controller
    - Secure DMA Controller
- Cryptography Accelerators
  - HashCrypt engine: AES and SHA
  - PRINCE on-the-fly flash encryption/decryption engine
  - CASPER: Asymmetric cryptography accelerator
  - Random Number Generator (RNG)



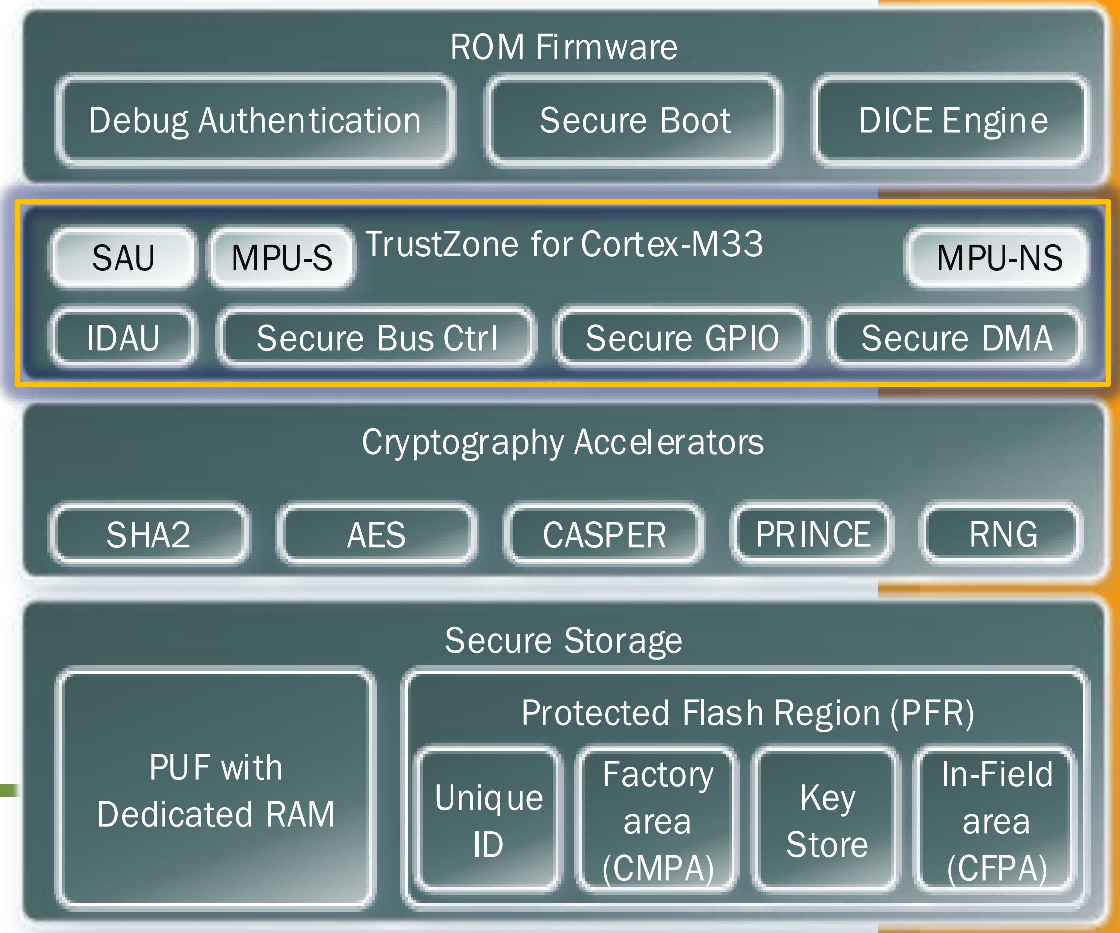
SECURE AHB BUS MATRIX

- Has Security side band signals
  - HPRIV, HNONSEC
    - Pole and anti-pole version of signals used for tamper detection
- PPC per AHB slave port
- Each master has separate security wrapper (MSW)
- For memories and bridges MPCs are used



TrustZone-M Sub-system

- Secure Bus Controller
- Device Attribution Unit (IDAU)
- Secure GPIO
- Secure DMA

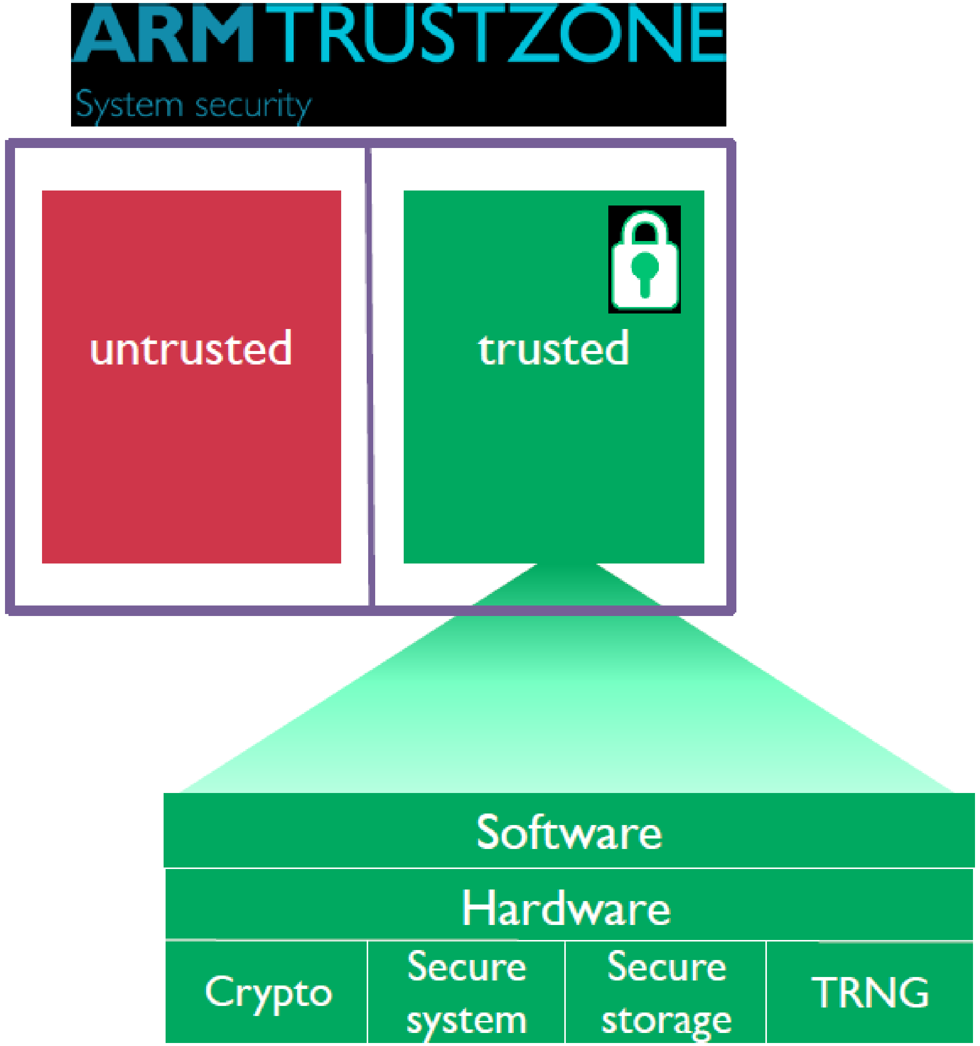


SECURE CONNECTIONS
FOR A SMARTER WORLD

PUBLIC

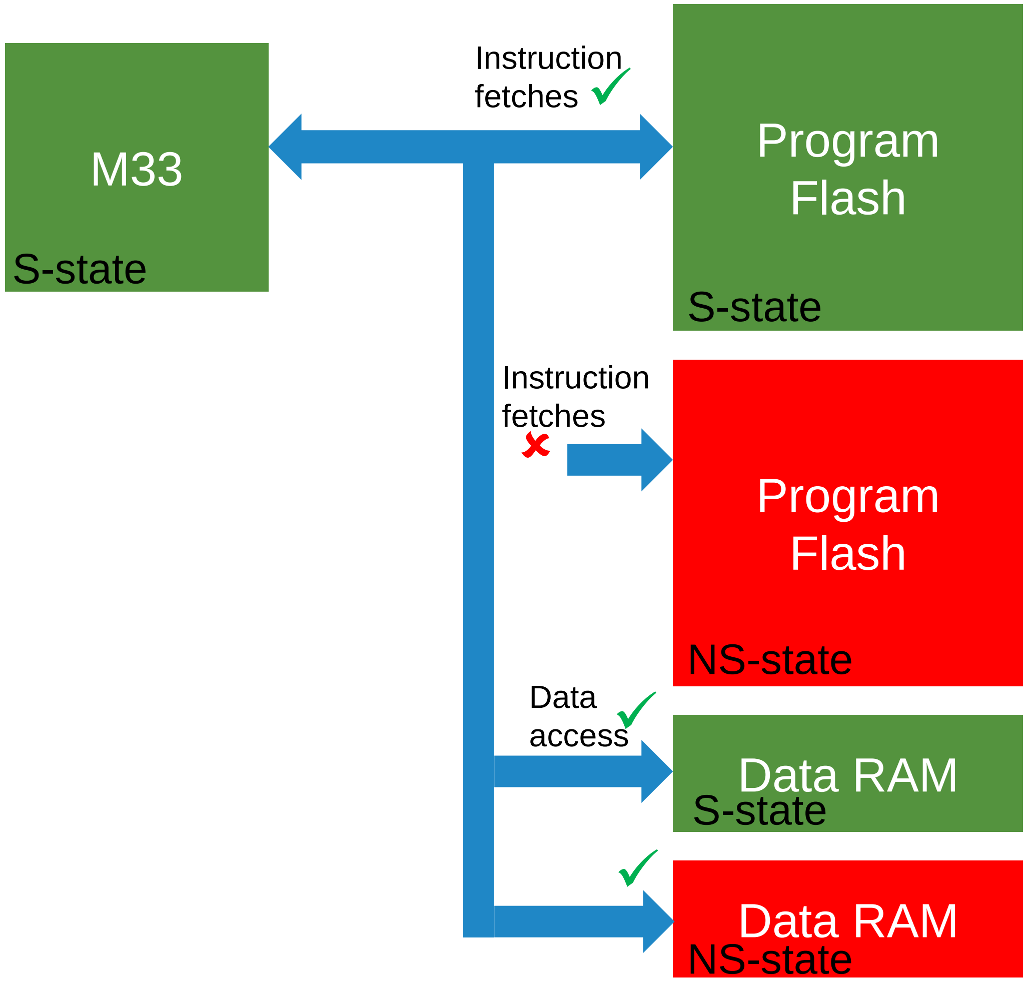
TRUSTZONE FOR ARMV8-M

- Separation and access control Isolate trusted software and resources
  - Reduce attack surface of key components
- Trusted software
  - Provision of security services
  - Small, well-reviewed code
- Trusted hardware
  - Hardware assist for cryptography
  - Secure-access validation built into SoC



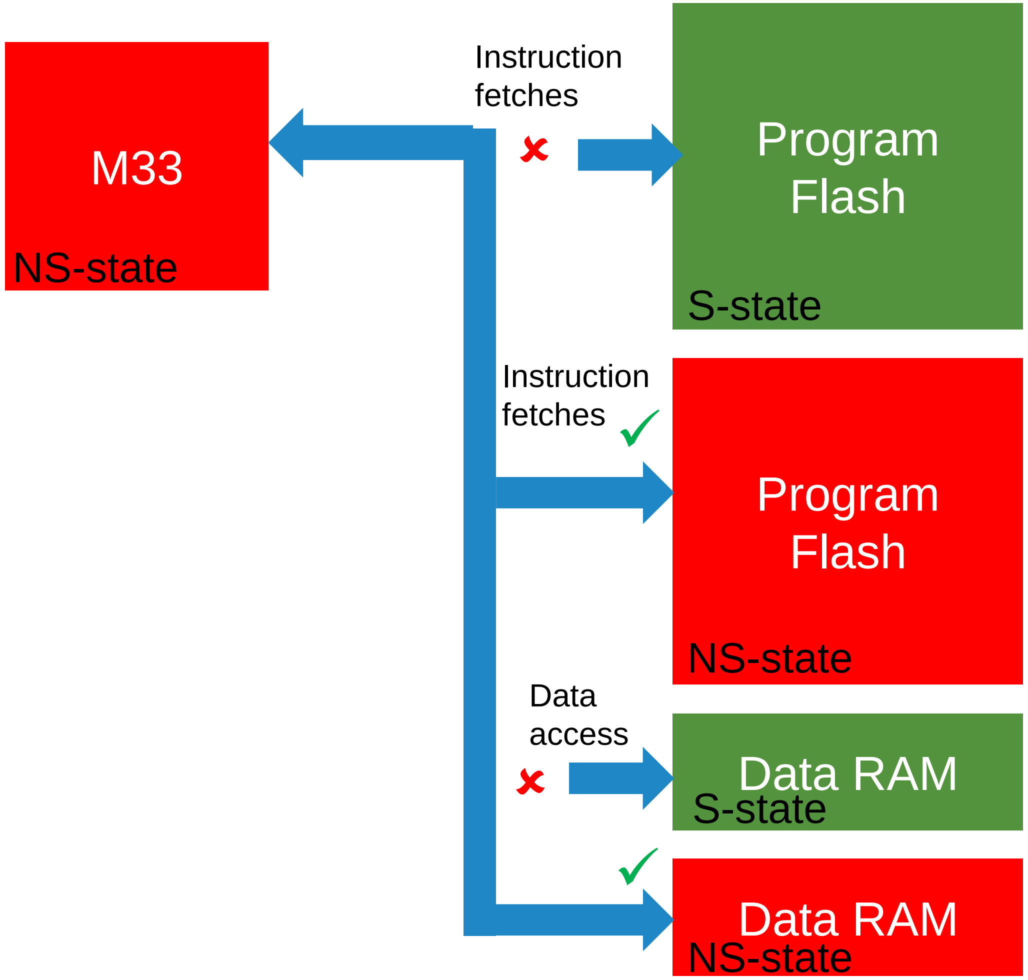
SECURE CPU STATE

- CPU in secure state can only execute from Secure Program memory.
- CPU in secure state can access data from both secure and NS memory.



NON-SECURE CPU STATE

- CPU in non-secure state can only execute from non-secure program memory.
- CPU in non-secure state can access data from both NS memory only.

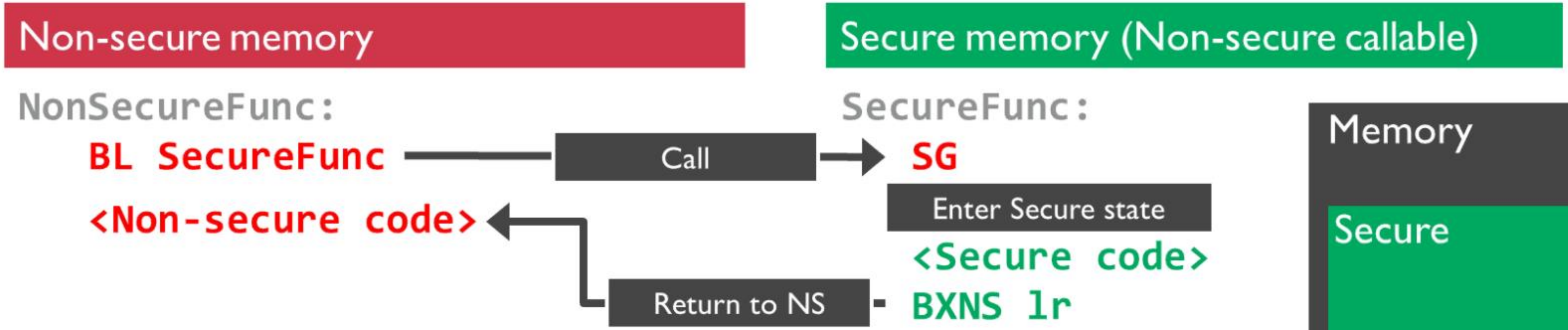


NON-SECURE CALLABLE (NSC) MEMORY

- Certain portion of Secure memory should be marked as Non-Secure Callable (NSC) memory for cross-domain calls.
- NSC memory regions contain tables of small branch veneers (entry points).
  - The first instruction in API must be SG instruction
  - NSC memory is to prevent hackers to use binary data matching SG opcode value

Cross-domain function calls

An assembly code level example



Non-secure memory

Secure memory (Non-secure callable)

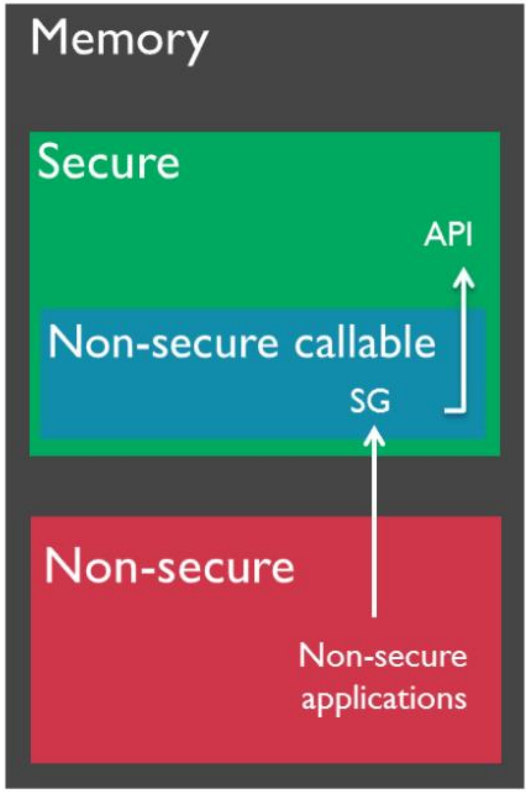
```
NonSecureFunc:
    BL SecureFunc
    <Non-secure code>
```

```
SecureFunc:
    SG
    <Secure code>
    BXNS lr
```

Call

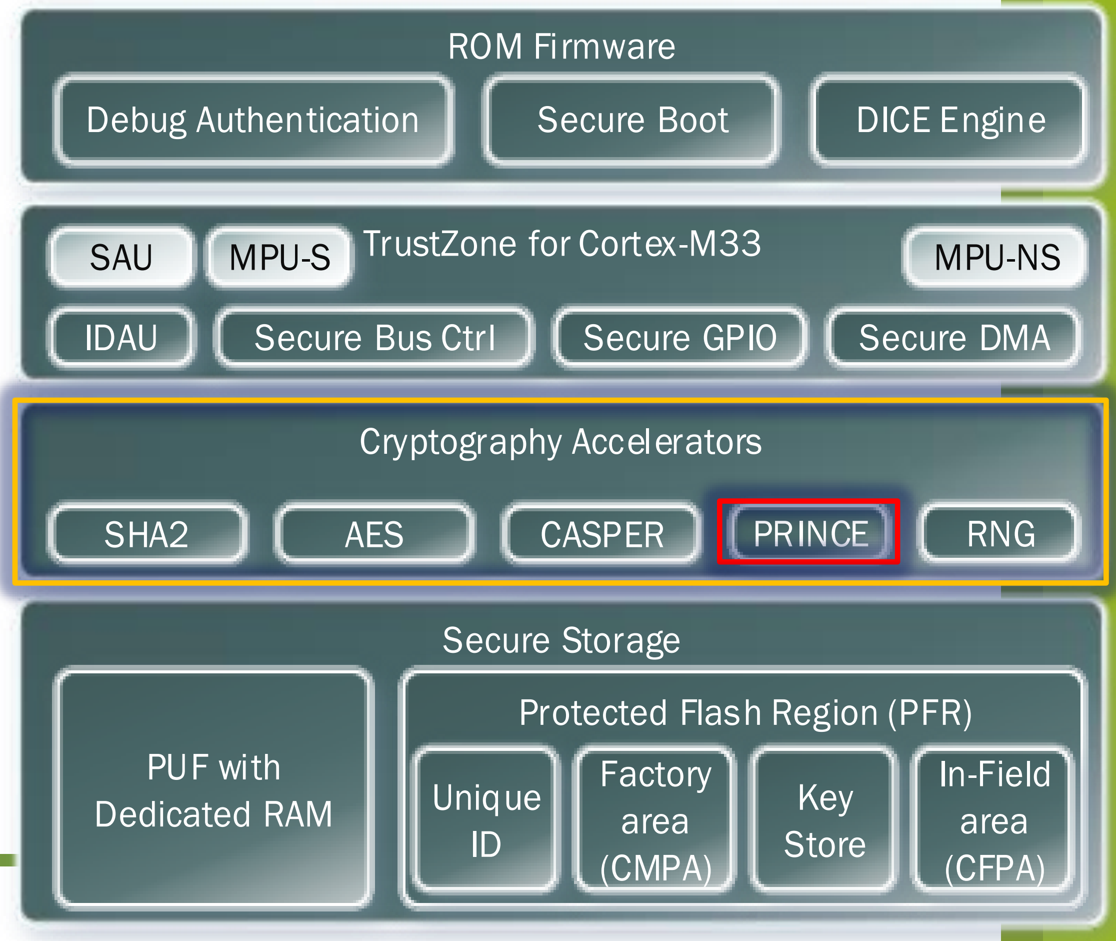
Enter Secure state

Return to NS



- Secure Gateway (SG) polices entry point
  - Placed at the start of Secure function callable from Non-secure code
- Non-secure → Secure branch faults if SG isn't at target address
  - Branch into the middle of functions is not allowed
  - Calling internal functions is not allowed
- Code on Non-secure side identical to existing code

PRINCE



SECURE CONNECTIONS
FOR A SMARTER WORLD

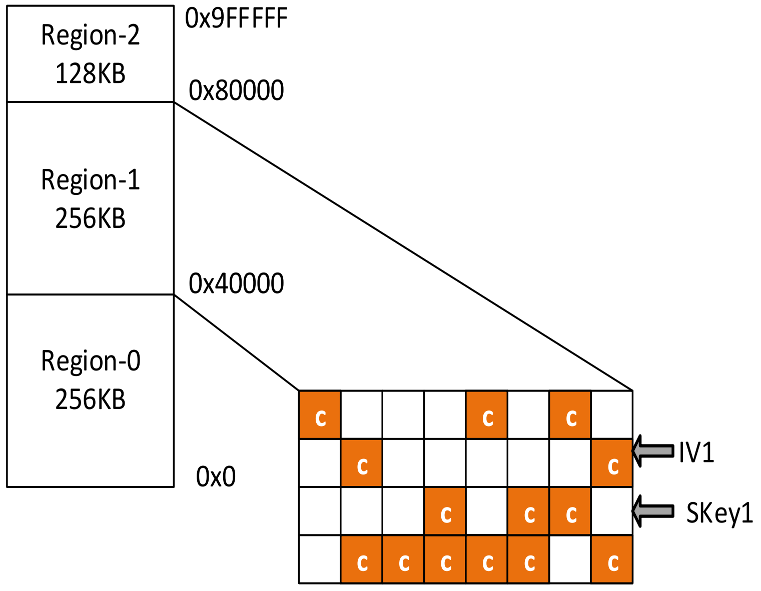
CHALLENGE: ASSET PROTECTION

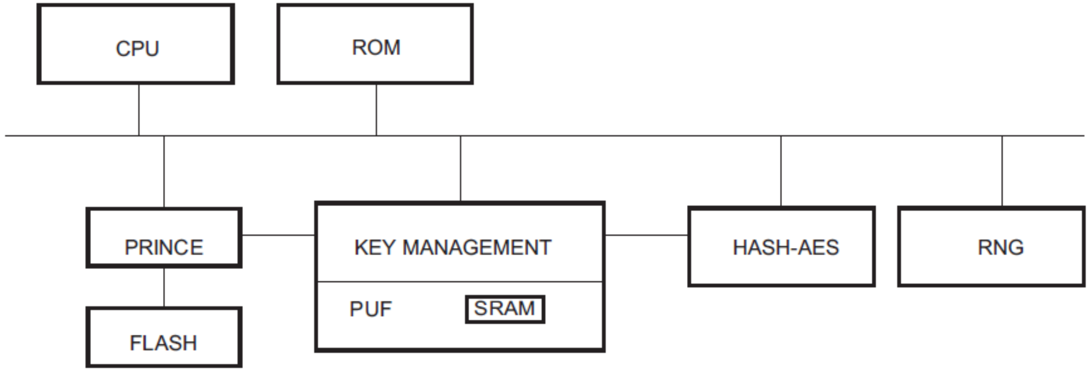
- On-chip non-volatile storage is used for storing important assets
  - Secret keys
  - Proprietary SW from OEM and Silicon Manufacturer
  - Application code
  - Other sensitive information
- Prone to attacks with malicious intent
  - Reading the code for cloning
  - Tampering for
    - Illegally gaining trust
    - Changing execution sequence
    - Changing programming value
  - Stealing keys
- Solution:
  - Encrypt the code stored in Flash
    - System performance cannot be compromised

PRINCE

- Is a cryptographic algorithm developed by NXP + 2 Universities
  - https://eprint.iacr.org/eprint-bin/getfile.pl?entry=2012/529&version=20140612:115014&file=529.pdf
- A light-weight symmetric block cryptography algorithm
  - 64b block cipher, with 128b crypto key
  - Same HW block supports encrypt and decrypt
- Real-time
  - Low latency decryption, no additional cycles added to read path (compared to 10-14 cycles in AES)
  - No initialization time
  - Combinatorial logic
- Efficient
  - Low cost (Si area)
  - Power efficient
  - No RAM buffers needed

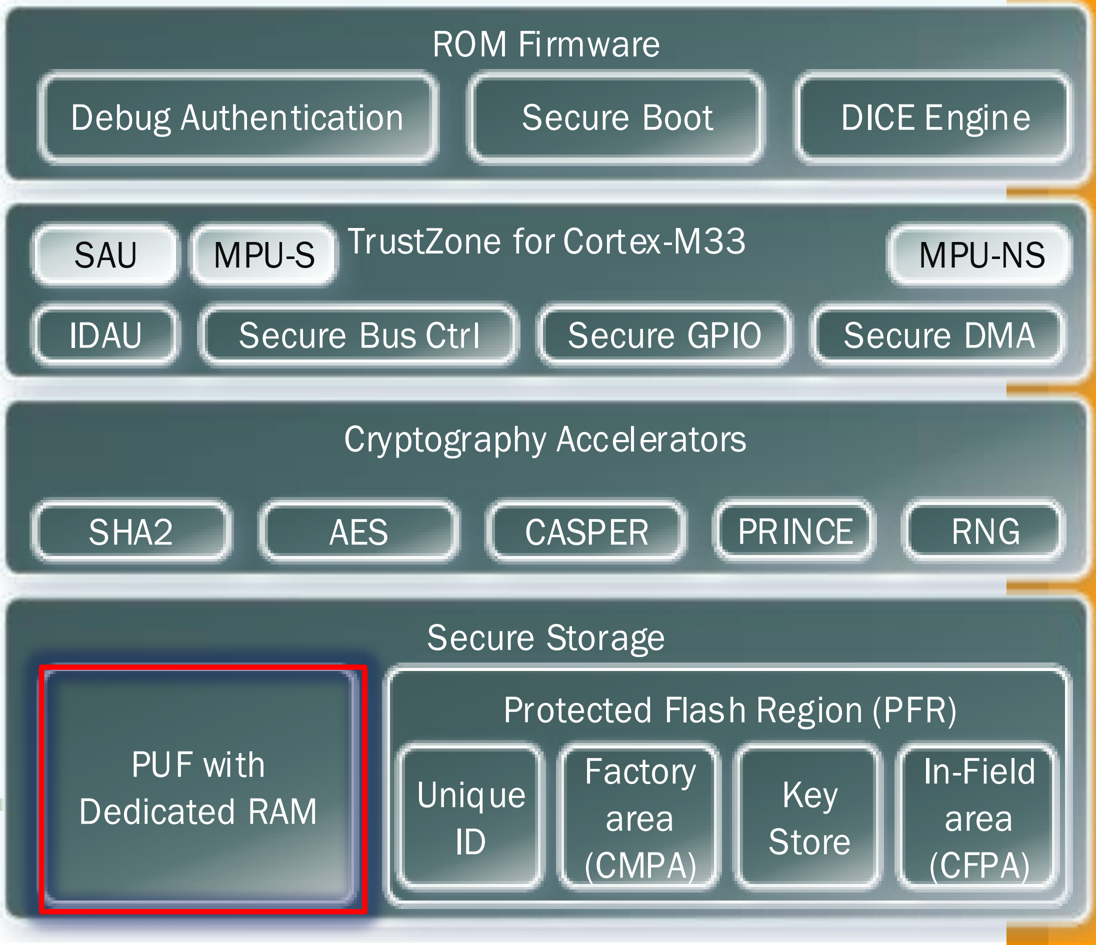
LPC55SX ADOPTION OF PRINCE





- Data stored in Flash is encrypted version
  - On-the-fly encrypt/decrypt for Flash contents
    - No additional latency compared to existing Flash access latency
- 128b Secret keys are supplied from PUF via secret-bus interface (not accessible by SW)
  - Key codes for Keys and IV are stored in Protected Flash Region
- Supports 3 regions in 640KB Flash
  - Each region is be at 256KB Address boundary
  - Allows multiple code images from independent source to co-exist
  - Secret-Key and IV Pair per region
- Register programmable crypto-enable bit per sub-region
  - One register per region
  - Each sub-region has 8kB granularity
  - Settings can be stored in PFR and be applied by ROM
- Cached data in FMC (cache) is obscured further using XOR mask with random number

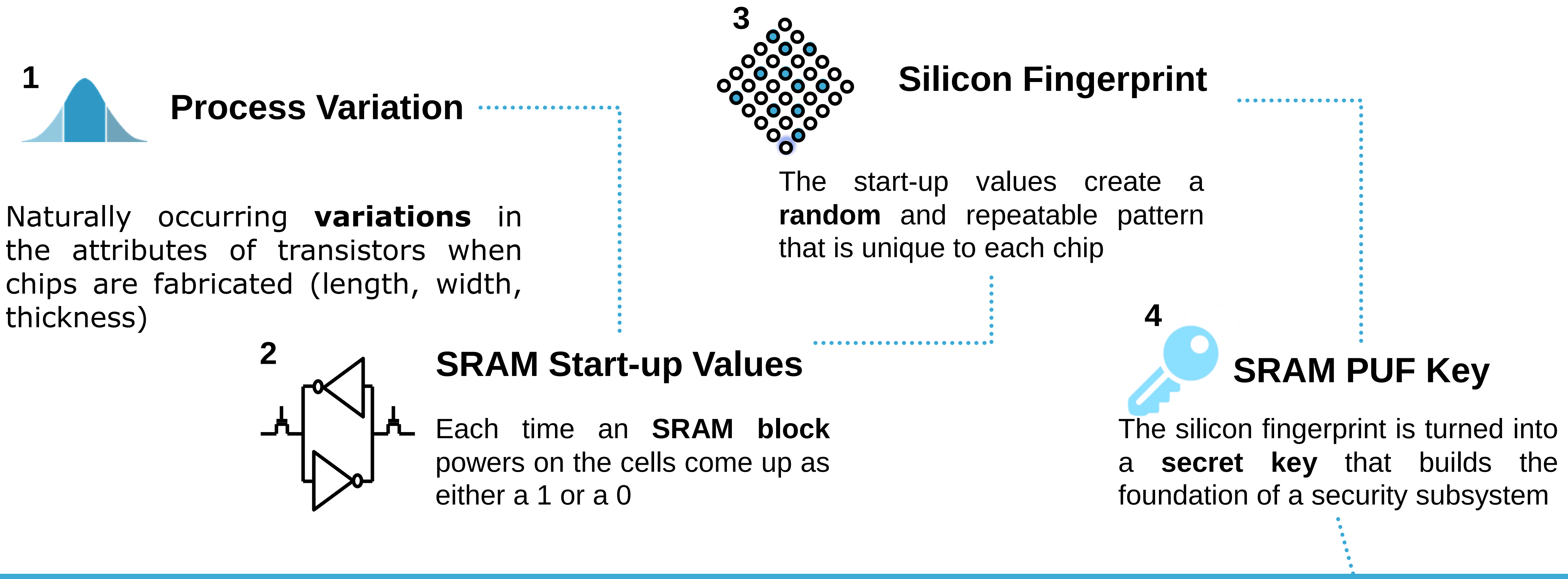
PUF



SECURE CONNECTIONS
FOR A SMARTER WORLD

PUBLIC

SRAM PUF TECHNOLOGY



1

Process Variation

Naturally occurring **variations** in the attributes of transistors when chips are fabricated (length, width, thickness)

2

SRAM Start-up Values

Each time an **SRAM block** powers on the cells come up as either a 1 or a 0

3

Silicon Fingerprint

The start-up values create a **random** and repeatable pattern that is unique to each chip

4

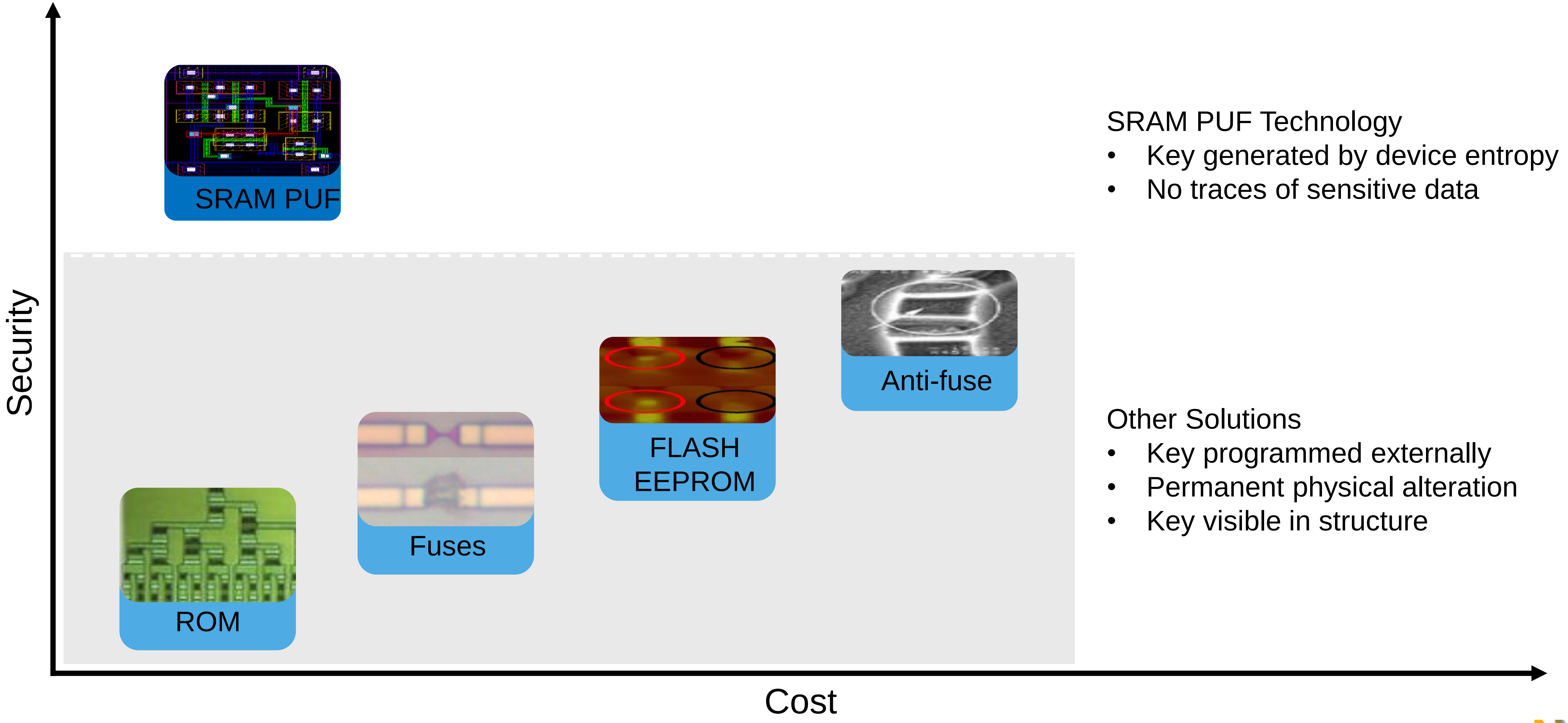
SRAM PUF Key

The silicon fingerprint is turned into a **secret key** that builds the foundation of a security subsystem

SRAM PUF Benefits

- Device-unique, unclonable fingerprint
- Leverages entropy of mfg. process
- No key material programmed

SRAM PUF ADVANTAGES



SRAM PUF Technology

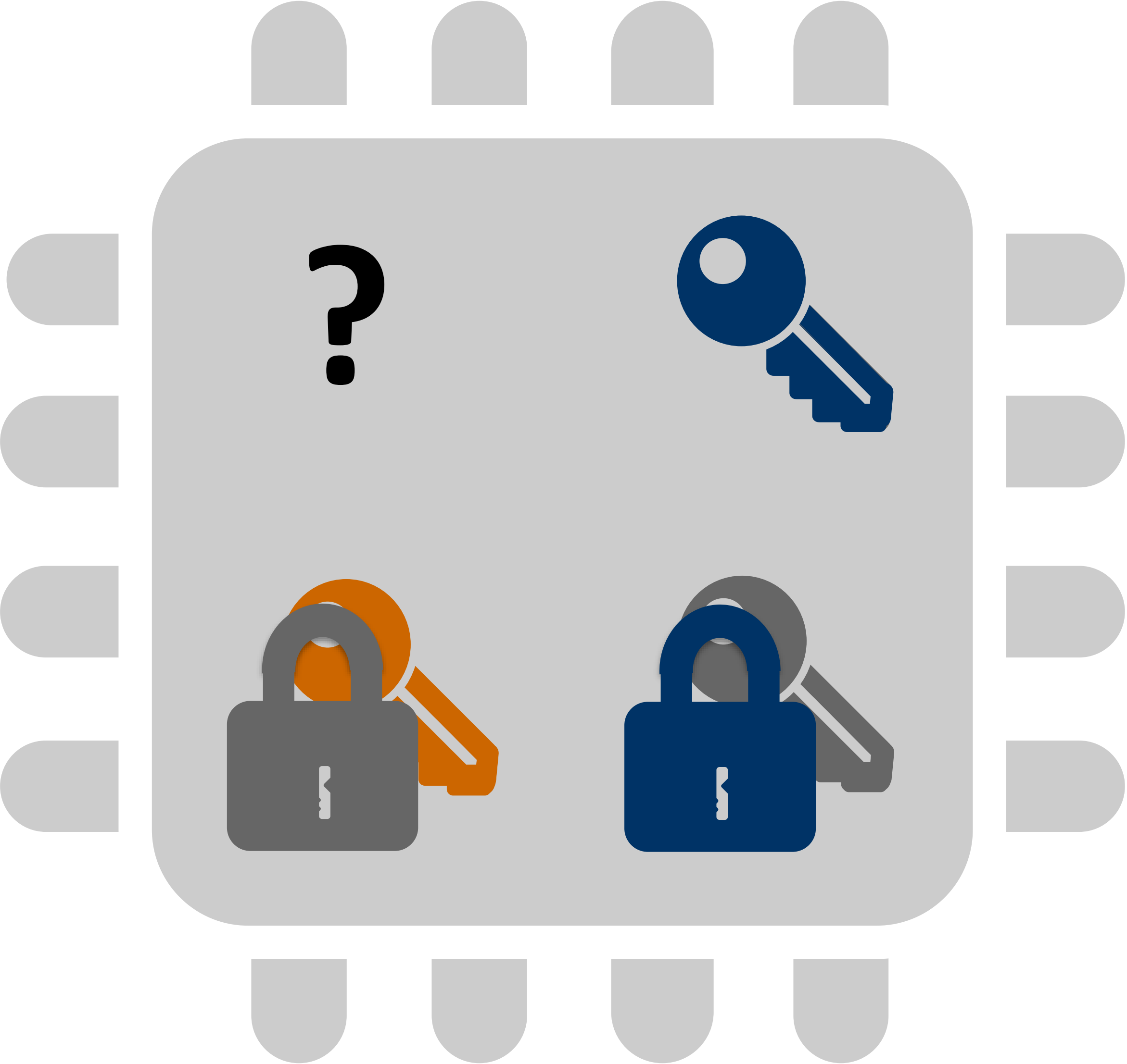
- Key generated by device entropy
- No traces of sensitive data

Other Solutions

- Key programmed externally
- Permanent physical alteration
- Key visible in structure

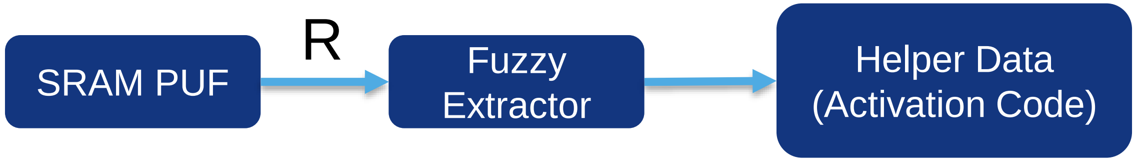
USE CASE: ROOT KEY STORAGE

- Bootstrapping the cryptographic system of a device requires a **Root Key (KPUF)**.
  - From One Key, Many
- Root Key Provisioning (RKP)
- Tamper resistant secure storage

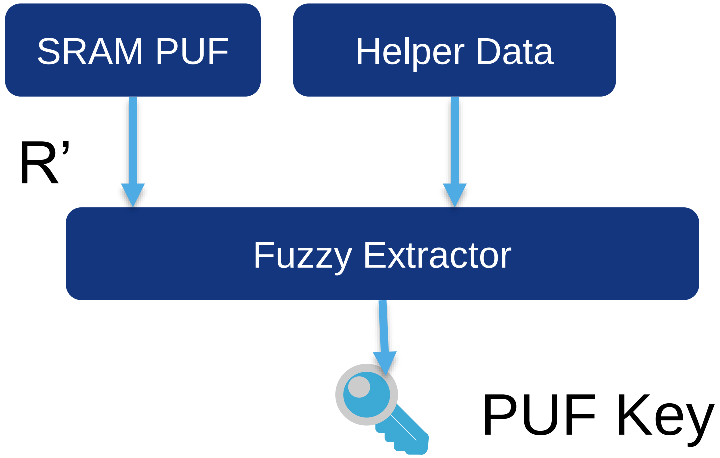


KEY PROVISIONING BASED ON SRAM PUF

1. Enrollment Mode

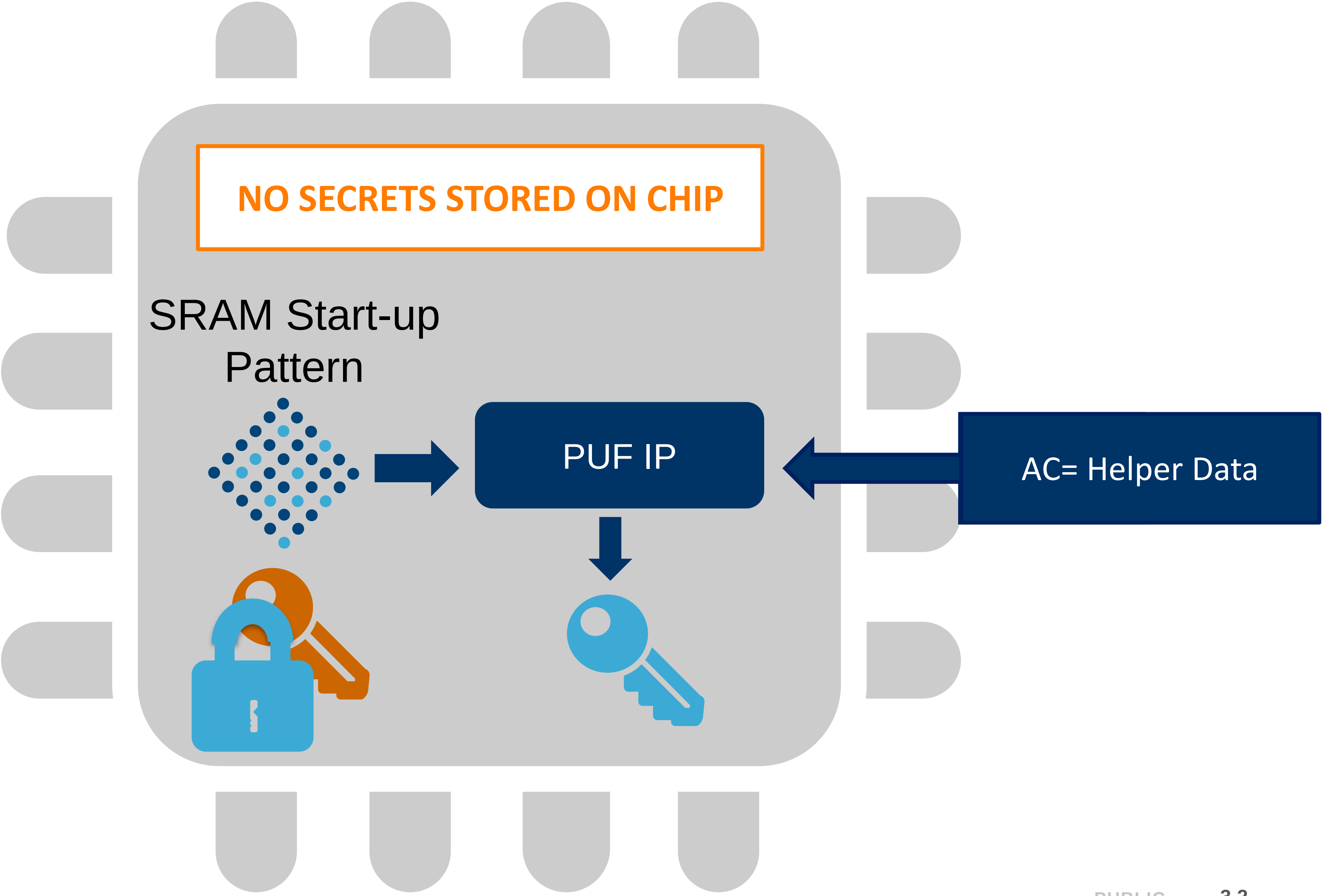


2. Key Reconstruction Mode



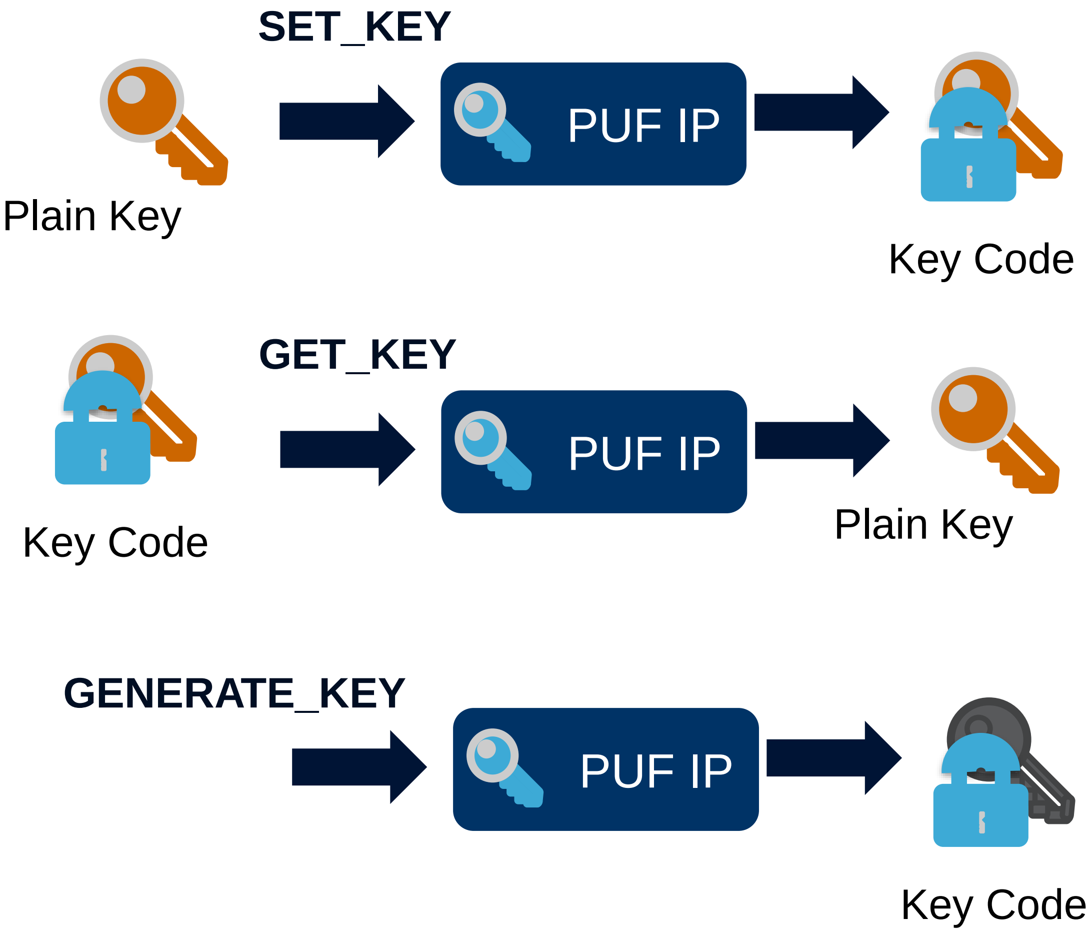
- SRAM PUF response (R) is a noisy fingerprint of the chip.
- PUF IP implements the Fuzzy Extractor or Helper Data Algorithm.
  - Error correction
  - Privacy Amplification
- Two operation modes:
  - Enrollment mode
  - Key Reconstruction Mode

USING PUF IP

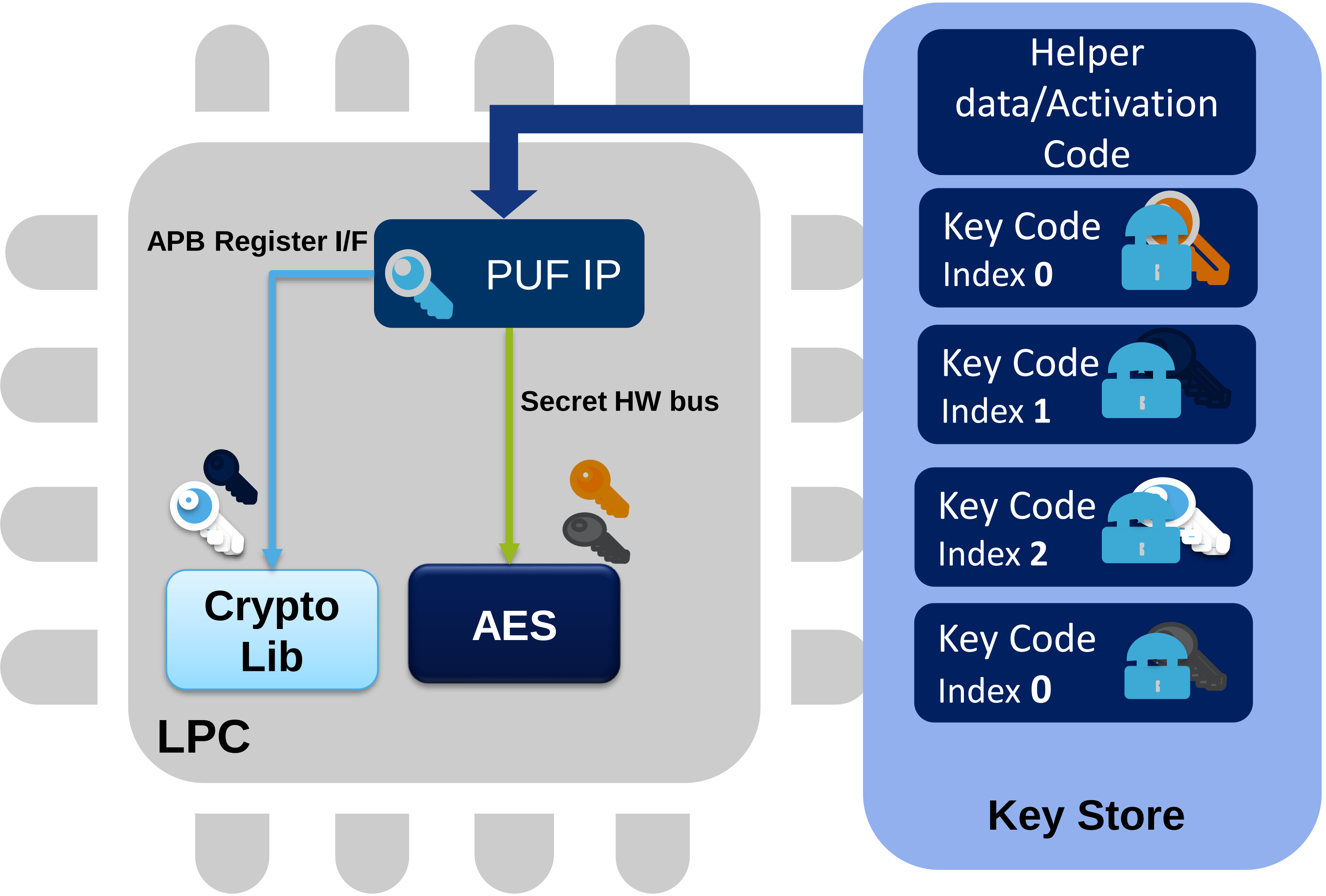

NO SECRETS STORED ON CHIP
SRAM Start-up Pattern
PUF IP
AC= Helper Data

PUF KEY STORE – KEY GENERATION

- Keys generated externally can be stored through PUF using SET_KEY operation.
- PUF controller provides generation of device unique cryptographic strength keys (64 to 4096 bits) using GENERATE_KEY operation.
  - If key index parameter is set to 0 then key is not known to anybody.
  - Any other key index are accessible through register interface using GET_KEY operation.



LPC PUF FEATURES



- 256 bit strength Root key
- Supports wrapping of keys
  - 64 to 4096 bits keys
  - Generation of Intrinsic keys (random key)
  - Index 0 accessible through HW secret bus
  - Other indexes through register I/F
  - Locking of indexes (DICE – UDS)

LPC55S SECURITY ARCHITECTURE SUMMARY

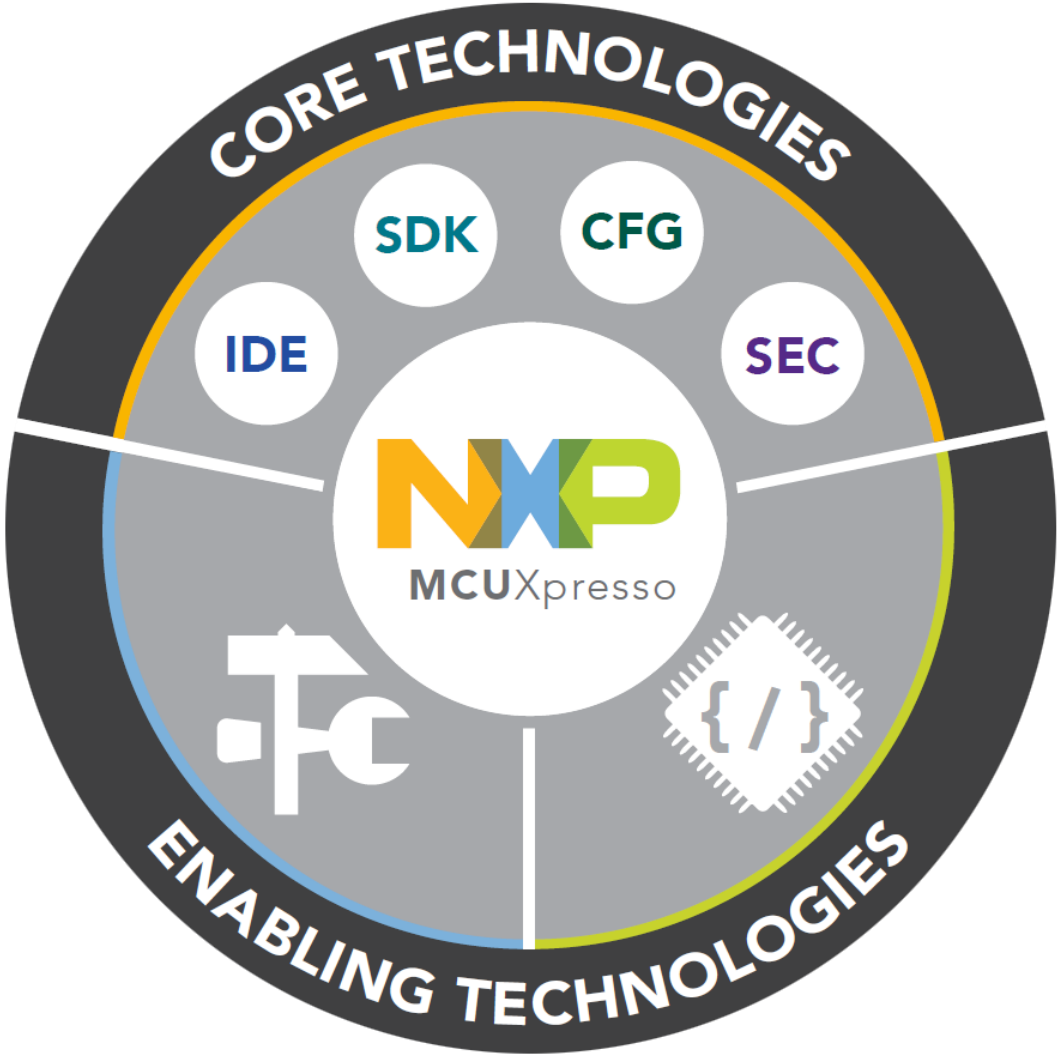
Security Blocks on LPC55Sxx	Protection Domain	Examples
SHA Engine(SHA1, SHA2), AES 256 Engine(ECB, CTR, CBC), RNG (FIPS 140-1), CASPER	Communication	Cryptography, Signature validation
SRAM PUF, AES-256 Engine, CASPER	Data	Secrets, keys, personal information
PRINCE (Encrypted flash)	Firmware	IP theft, reverse engineering
UUID (RFC4122), Chip unique Root Key(PUF), DICE (unique tracking)	Operational integrity	Maintaining service and revenue
Pole/anti-pole checks, Secure boot, Measured boot (via DICE),	Anti-tamper	Physical attacks
Secure boot ROM, PFR (For RoTKH*), DICE *public key hash	Root of trust	Secure boot ROM, PFR (For RoTKH), DICE

MCUXpresso ENABLEMENT

SECURE CONNECTIONS
FOR A SMARTER WORLD

PUBLIC

THE MCUXPRESSO ECOSYSTEM



Core Technologies from NXP

- MCUXpresso IDE
- MCUXpresso SDK
- MCUXpresso Config Tools
- MCUXpresso Secure Provisioning Tool

Enabling Software Technologies

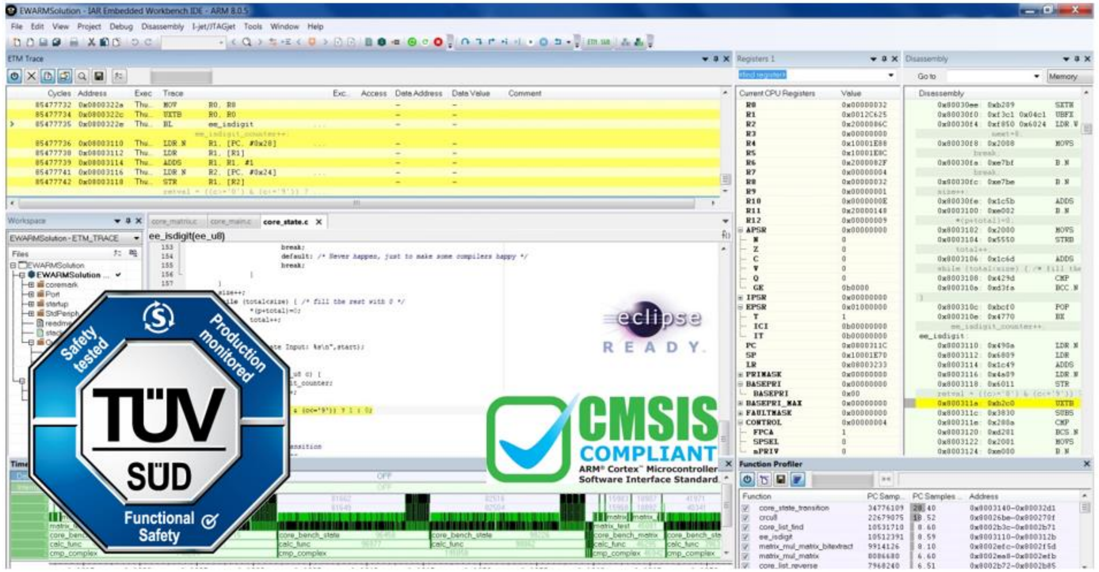
- Run time software libraries and middleware
- Enable customers to focus on differentiation
- From NXP and partners

Enabling Tools Technologies

- Partner IDEs
- Debug Probes
- Development Boards
- From NXP and partners

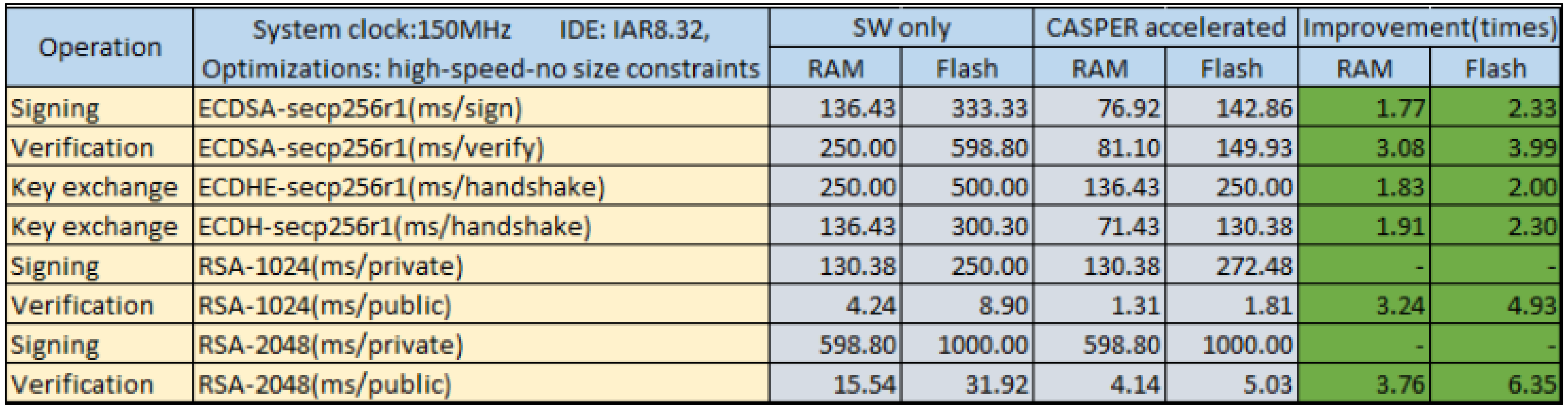
ENABLING TOOLS TECHNOLOGIES
LEAD DEVELOPMENT TOOL TECHNOLOGY PARTNER - IAR SYSTEMS

- IAR is an NXP Platinum Partner
- IAR Embedded Workbench (EWARM) has been integrated in MCUXpresso since its introduction
- MCUXpresso SDK
  - Projects for Embedded Workbench are included in all drivers/examples
  - CMSIS-DAP and J-link options included
- MCUXpresso Config Tools
  - Standalone config tools provide IAR-compatible initialization source and header files
  - Project cloner to create starting point projects from SDK examples


EWARMSolution - IAR Embedded Workbench IDE - ARM 8.0.5
File Edit View Project Debug Disassembly I-jet/JTAGjet Tools Window Help
ETM Trace
Registers 1
Disassembly
Workspace
Function Profiler
Safety tested
Production monitored
TÜV SÜD
Functional Safety
eclipse
READY
CMSIS
COMPLIANT
ARM® Cortex™ Microcontroller Software Interface Standard

MCUXPRESSO SDK SECURITY ENABLEMENT

- Trustzone examples
  - Secure + non-secure applications
  - Secure fault handling
  - Use of secure GPIOs
- Mbed Crypto and Mbed TLS
  - Benchmark test examples included
  - Also used in AWS Cloud Connectivity examples
  - Libraries utilize CASPER hardware acceleration where available for major RAM, energy and processor cycle savings (see https://www.nxp.com/docs/en/application-note/AN12445.pdf)



Operation	System clock:150MHz IDE: IAR8.32,	SW only		CASPER accelerated		Improvement(times)	
	Optimizations: high-speed-no size constraints	RAM	Flash	RAM	Flash	RAM	Flash
Signing	ECDSA-secp256r1(ms/sign)	136.43	333.33	76.92	142.86	1.77	2.33
Verification	ECDSA-secp256r1(ms/verify)	250.00	598.80	81.10	149.93	3.08	3.99
Key exchange	ECDHE-secp256r1(ms/handshake)	250.00	500.00	136.43	250.00	1.83	2.00
Key exchange	ECDH-secp256r1(ms/handshake)	136.43	300.30	71.43	130.38	1.91	2.30
Signing	RSA-1024(ms/private)	130.38	250.00	130.38	272.48	-	-
Verification	RSA-1024(ms/public)	4.24	8.90	1.31	1.81	3.24	4.93
Signing	RSA-2048(ms/private)	598.80	1000.00	598.80	1000.00	-	-
Verification	RSA-2048(ms/public)	15.54	31.92	4.14	5.03	3.76	6.35

MCUXPRESSO CONFIG TOOLS – CONFIGURATION AND CODE GENERATION

SDK Builder packages custom SDKs based on user selections of MCU, evaluation board, and optional software components.

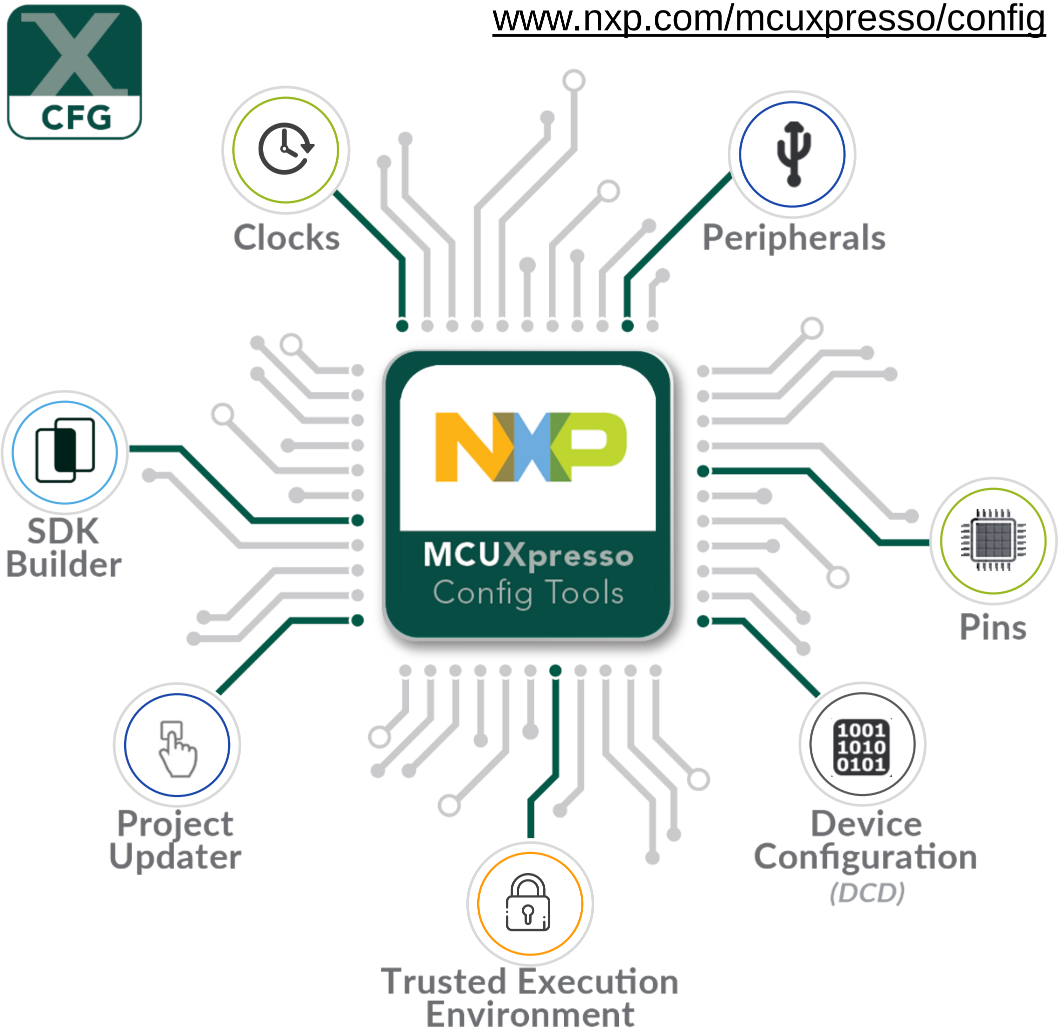
Pins, Clocks, Peripherals and Cloner tools generate initialization for custom board support; cloner creates standalone SDK project based on SDK examples.

Project Updater works directly with existing SDK-based IDE projects with generated Pins, Clocks and Peripherals source files.

Device Configuration tool allows DCD commands sequence config for pre-initialization of devices at boot time.

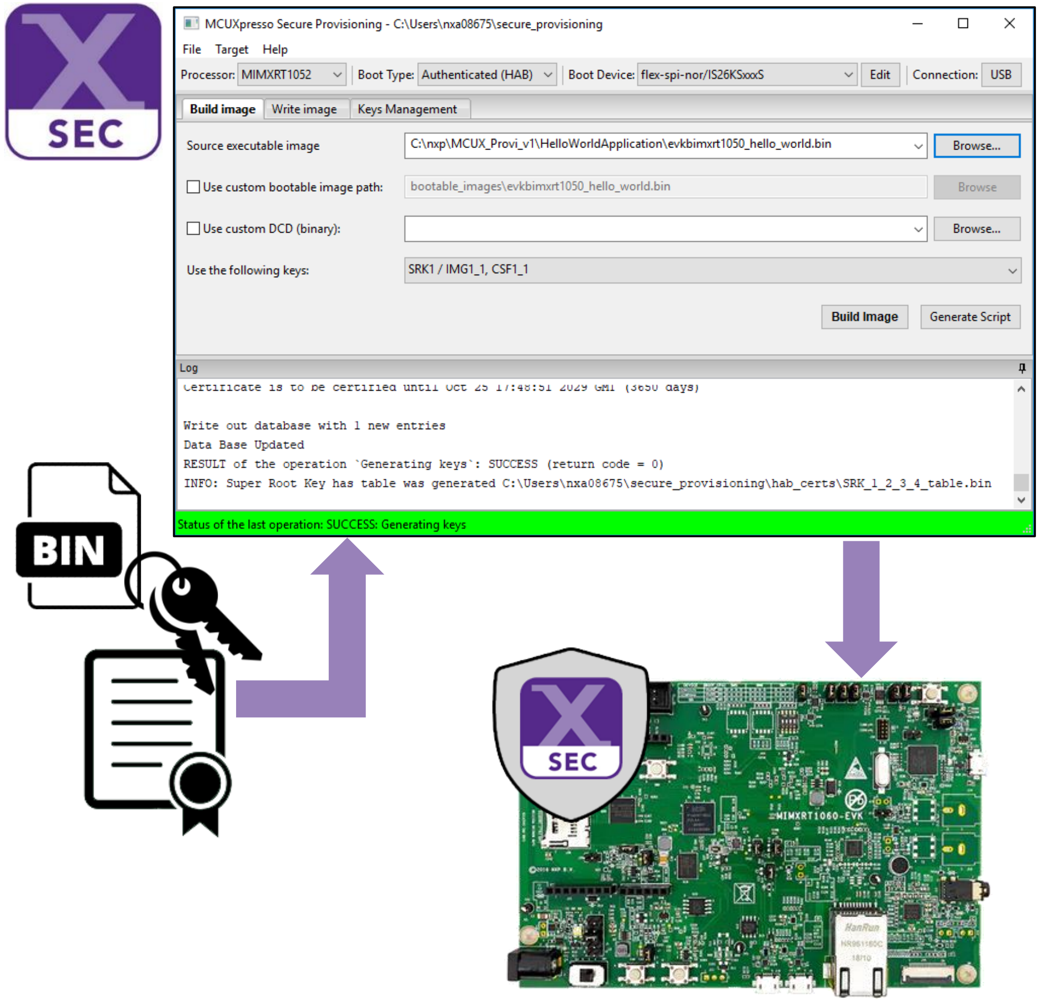
Trusted Execution Environment configures protection and isolation of sensitive parts of the application

www.nxp.com/mcuxpresso/config



MCUXPRESSO SECURE PROV
PROGRAMMING AND SECURE PROVISIONING TOOL

- Primary Functionality
  - Key/Certificate Management and Generation
    - Using integrated OpenSSL or externally specified keys, signatures, and certificates
  - Secure Image Preparation
    - Encrypting and signing of ELF executables, SREC, and raw binaries
  - Device Provisioning and Programming
    - Programming of eFUSEs and One-Time-Programmable flash regions
    - Direct connection to the target via UART, USB-HID for provisioning and programming
- Provides a unified graphical frontend for enabling and configuring **devices with Secure Boot capabilities**
- Features a command-line interface for scriptable execution and advanced configuration



MCUXpresso Secure Provisioning - C:\Users\nxa08675\secure_provisioning

Processor: MIMXRT1052 | Boot Type: Authenticated (HAB) | Boot Device: flex-spi-nor/IS26KSxxxS | Connection: USB

Build image | Write image | Keys Management

Source executable image: C:\nxp\MCUX_Provi_v1\HelloWorldApplication\evkbimxrt1050_hello_world.bin

□ Use custom bootable image path: bootable_images\evkbimxrt1050_hello_world.bin

□ Use custom DCD (binary):

Use the following keys: SRK1 / IMG1_1, CSF1_1

Log

```
Certificate is to be certified until Oct 25 17:48:51 2029 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
RESULT of the operation `Generating keys`: SUCCESS (return code = 0)
INFO: Super Root Key has table was generated C:\Users\nxa08675\secure_provisioning\hab_certs\SRK_1_2_3_4_table.bin
```

Status of the last operation: SUCCESS: Generating keys

SECURE CONNECTIONS
FOR A SMARTER WORLD