Design of secure embedded systems

HSM (Hardware Security Module)

Part B

Prof. Sergio Saponara,
Dip. Ingegneria della Informazione, Università di Pisa
sergio.saponara@unipi.it
+39 3468790937

Outline

- Architectures and components of HSM in automotive:

Analysis of HSM in commercial automotive MCUs (NXP, ST, Renesas)

Analysis of commercial IPs for HSM (by Rambus, Synopsys, …..)

- HW for Secure Cards

- Secure key storage/management and advanced secure features in EPI HSM

- HSM evolution roadmap

    Intrusion Detection System (IDS) support

    Post-quantum cryptography (PQC) and tracking of NIST standardization

- Q&A session

Crypto Secure Engine in NXP/ST

Hardware security features in NXP/ST processor for edge devices

Core function in the CSE (Crypto Secure Engine) – compliant with Basic SHE

On-chip True RNG

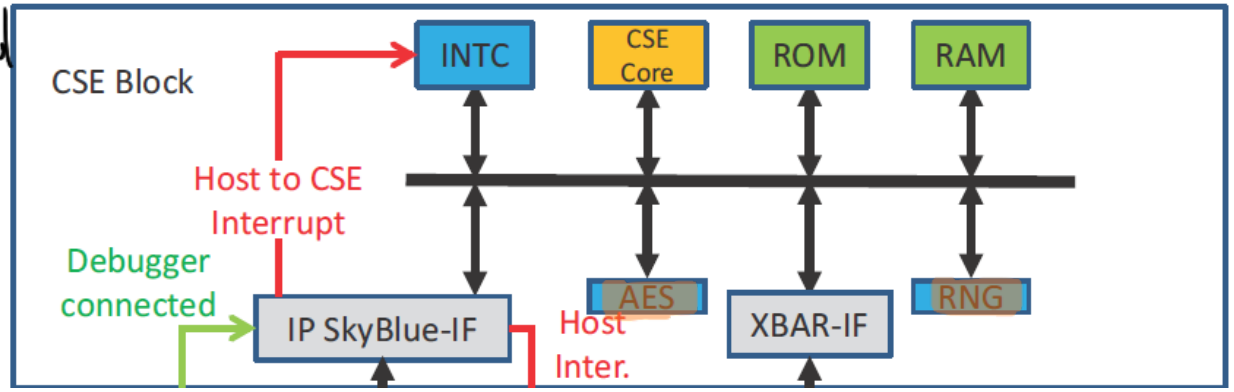
Hardware acceleration for AES-128 (with ECB and CBC modes for encryption, with CMAC mode for authentication)

Dedicated memory for secure key and firmware (boot) storage

Hardware security extra features in NXP processor for edge device (supported via HW or SW)

Hardware or optimized SW library for SHA-256 and for RSA (or ECC) public key cryptography

This is an IP integrated microcontroller



They developed this CSE with basic SHE compliance requirements

- NXP enriched this CSE with additional features (SHA256 and RSA).

This text is talking about **hardware-level security features** built into processors from **NXP** and **STMicroelectronics** (both are big companies that make chips used in cars, smart devices, etc.). These processors are made for **edge devices**, which just means devices that are out in the real world—like a smart sensor in your car, a smart camera, a payment terminal—rather than tucked away in a cloud server.

Now, the **Crypto Secure Engine (CSE)** is a small chunk of the processor that handles **cryptographic tasks**—so encryption, authentication, random number generation, key storage, that kind of stuff. It's like a **tiny vault with a calculator** inside your processor, designed to do security operations safely and quickly.

Here's what that whole thing is saying, simplified:

- **CSE is compliant with Basic SHE**: "SHE" is a security standard for automotive ECUs (Electronic Control Units). So this means the chip meets the basic security expectations for automotive systems.
- **On-chip True RNG**: It has a hardware **Random Number Generator**, not some fake software-based one. This is crucial because secure crypto relies on randomness that can't be predicted.
- **AES-128 hardware acceleration**: AES is a type of encryption, and 128-bit is a decent, widely used level. The chip can do this encryption super fast because it's wired to do it directly (not just through software).
- **Supports ECB and CBC modes**: These are ways AES encryption can be structured. ECB is basic, CBC is more secure. CMAC is for authentication, so it can also check if a message hasn't been tampered with.
- **Dedicated memory for secure storage**: There's a special area in the chip where it stores encryption keys or boot firmware safely, so no one can mess with it.

Then the second part adds even more goodies:

- **SHA-256, RSA, ECC**: These are other cryptographic tools. SHA-256 is a hashing algorithm, RSA and ECC are for public-key cryptography (things like verifying signatures, encrypting stuff). Whether they're done in hardware or optimized software, they're fast and secure.

So in short:
This chip is **loaded with built-in tools** to do **secure computing** right on the device itself, without having to rely too much on external protections. That's perfect for things like **cars**, **IoT devices**, or any gadget that might be exposed to hackers but still needs to keep your data safe.

Safety and security in automotive NXP MCUs

S32K1xx
Single-core MCUs with 128 kB to 2 MB, ASIL B enabled and 32-176 pin, AEC-Q100 Grade 0, Grade 1 and Grade 2 qualified

S32K3xx
Single-, dual- and lockstep-core MCUs with 512 kB to 8 MB, ASIL B/D enabled and 48-289 pin.

different type of memory

Autonomous safety integrity level: high safety level: low error rate

To understand complexity, you can also look at number of pins. From one family to another, big difference.

Safety and security in automotive NXP MCUs

S32K1	Basic set of cryptographic functions for SHE support	20 keys SHE update key protocol	SHE memory authenticity checks during start-up (CMAC)
S32K3	Comprehensive cipher suite SHA-2, SHA-3, RSA and ECC support	Configurable set of keys Extensive key management (import, export, derive)	Extended memory authenticity checks during boot & run-time

→ support

→ more memory in this family

→ NBT only at startup

Monotonic counters
Secure tick

ISO 21434: NXP cybersecurity engineering processes are now certified as compliant with the new automotive cybersecurity standard ISO/SAE 21434. (Certified by TÜV SÜD)

→ They are now standardised by TUV SUD

Product Security Incident Response Team (since 2008 in NXP, response in 24h)

→ NXP has designed a PSIRT.

- TUV SUD is the go to for CS certification

That statement means that **NXP**, which is a big company making chips and tech for cars and other electronics, has made sure its **cybersecurity processes** for car-related stuff now **follow the official international rules** set by the new standard called **ISO/SAE 21434**.

This standard was made specifically for the **automotive world**, where cars are becoming more like rolling computers. It defines how to make sure everything—from design to production to updates—is done in a way that keeps the system safe from cyberattacks.

So when NXP says it's certified, it's basically flexing:
"Hey, we build our car tech in a way that's officially recognized as secure, according to the latest industry rules."
Which is a big deal because car manufacturers and suppliers need to prove that they're not just winging cybersecurity anymore.

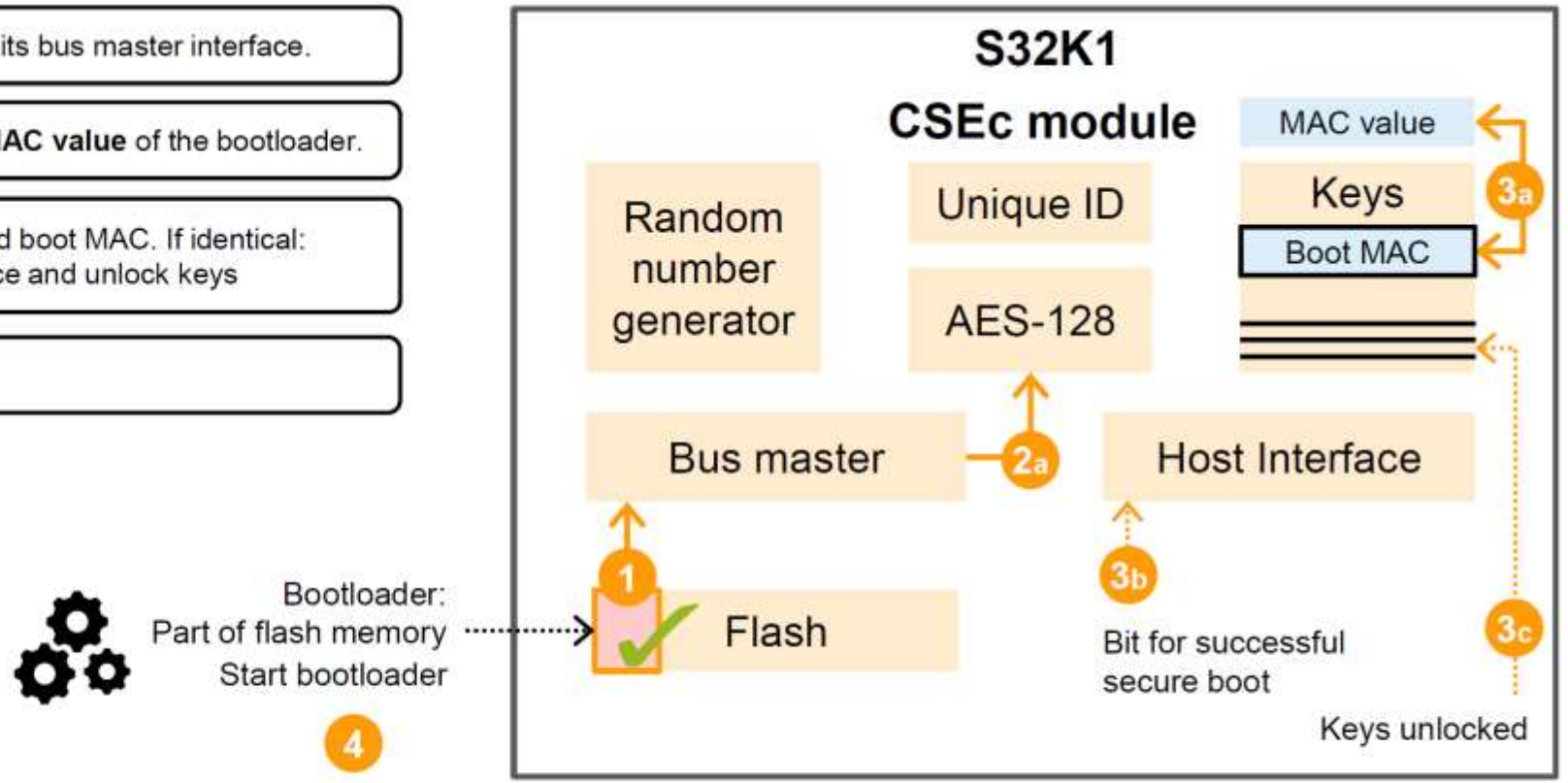
Bootloader authenticity check in S32K1xx NXP MCUs

Step 1: After power on: CSE module reads bootloader via its bus master interface.

Step 2: CSE module uses the boot key to calculates the **MAC value** of the bootloader.

Step 3: CSE module compares calculated MAC with stored boot MAC. If identical: successful secure boot → set respective bit in host interface and unlock keys

Step 4: MCU always starts bootloader.



- **MAC** protects against modification of bootloader and depends on the (secret) boot key → integrity and authenticity of bootloader.
- Only if calculated MAC value matches stored boot MAC value: successful secure boot → set respective bit in host interface and unlock keys for further usage (see next demos)

At power on, first part activated is Secure Boot Engine. Use the boot key to evaluate MAC value of bootloader to verify its authenticity. If identical you can start boot and MCU starts bootloader.

So S32K1 supports secure boot through these steps.

You save multiple versions of FW and only when needed you do the SWAP.
K1 key management is what we saw in S1DE, in K2 it is much more complicated.

Bootloader in S32K3xx NXP MCUs

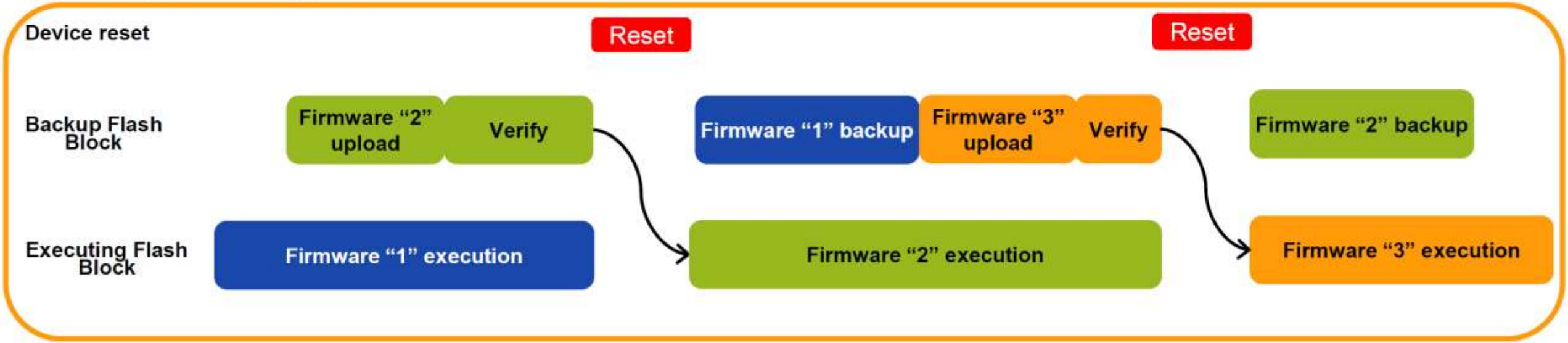
Use case: A/B swap in internal flash

- Current firmware executes and simultaneously uploads new firmware image into backup flash block
- After new firmware upload and verification. On the next reset new firmware will be executed

S32K3 Value

- Zero downtime, instant A/B swap after reset
- Download while application running
- Automatic address translation
- Backup firmware available

S32K3xx Firmware Update



S32K3xx NXP secure features

KEY MANAGEMENT

- Key file management
- Key import
- Key export
- Key generation
- Key derivation
- Key exchange

AES key up to 256 bits
RSA key up to 4096 bits

CRYPTO OPERATIONS

- AES
  Encryption & decryption
- CMAC / HMAC
  Generation & verification
- Hashing (SHA)
- RSA / ECC signature
  Generation & verification
- RSA OAEP / ECIES
  Encryption & decryption
- Random generation
  TRNG & PRNG

All operations
hardware accelerated

PLATFORM SECURITY

Advanced services, like secure boot optimized for latency

- Strict secure boot
  Verify-then-start
  [1st all checks and then boot]
- Parallel secure boot
  Verify-and-start
  (or check and boot together, less secure)
- On-demand verification
  Secure boot control in app.
- Configurable sanctions
  E.g. key usage restrictions
  Maybe IDS that can reduce privileges

Secure boot
optimized for latency

STM automotive secure portfolio

https://www.st.com/content/st_com/en/products/embedded-software/mcu-mpu-embedded-software/spc5-embedded-software/spc5-security-pack.html

STMicroelectronics (STM) is proposing a full security offer based on
- SPC56 4B Line and SPC56 EC Line automotive microcontrollers include a Cryptographic Services Engine (CSE) featuring AES-128, CMAC authentication and secured device boot mode: all the secure functionalities are compliant with SHE 1.1 standard.
- embedded Hardware Security Module (eHSM) supporting the EVITA Medium (SPC58 MCU E/N lines) and the EVITA full (SPC58 H line)
- embedded Secure Element (eSE) tamper proof compliant with EVITA Full profile
- embedded SIM (eSIM) offer designed for Automotive market for secure cellular connectivity
- Dedicated solution to NFC (near field communication) car access

While SPC58, they support EVITA medium and EVITA full, NOT everyone have an HSM inside (SPC58).

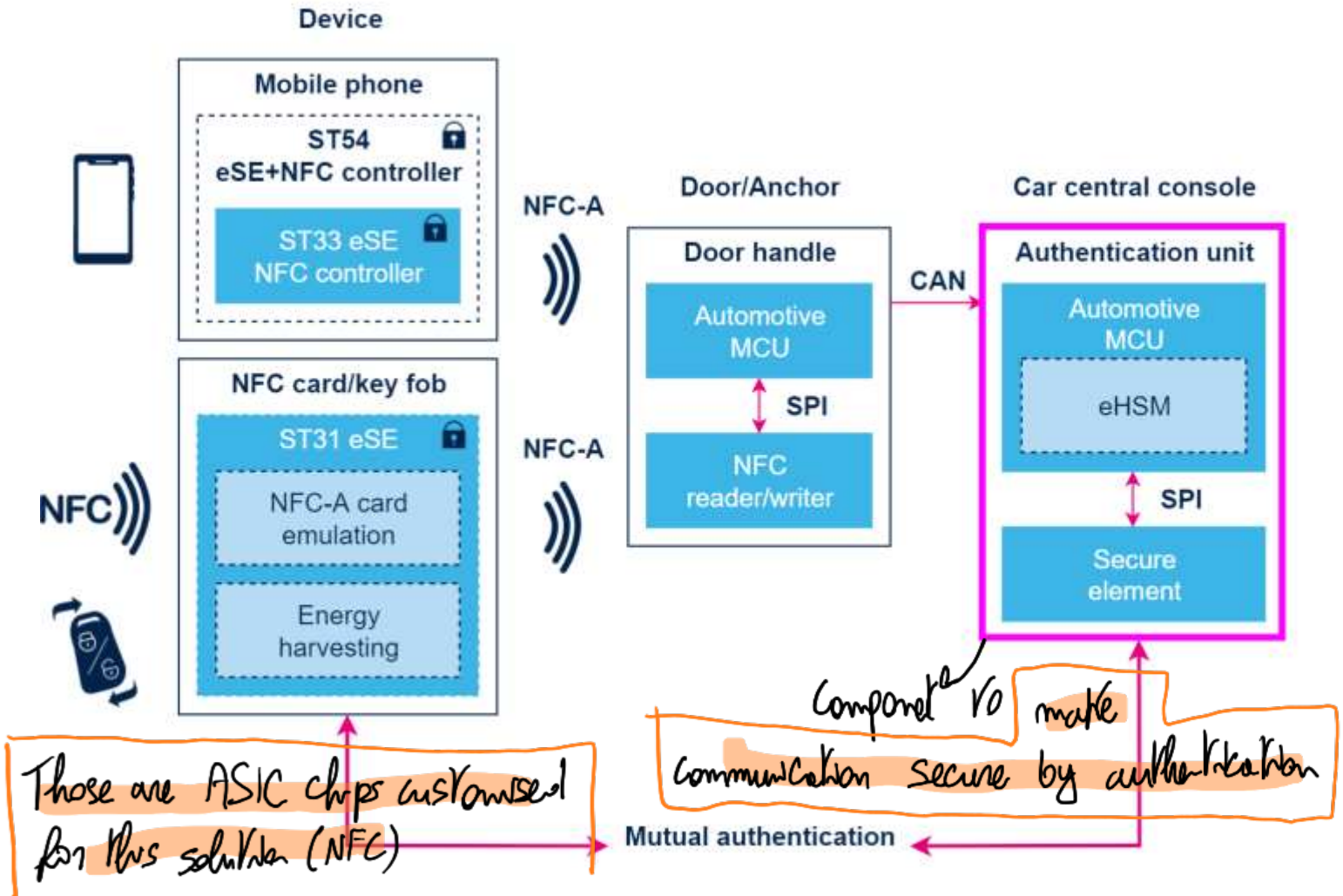
There is a portfolio of components optimized for specific solutions such as NFC.

For ST we have different types of microcontrollers that are compliant with different standards (SHE/EVITA)

STM automotive secure portfolio

https://www.st.com/content/st_com/en/products/embedded-software/mcu-mpu-embedded-software/spc5-embedded-software/spc5-security-pack.html

STM has dedicated solution to NFC (near field communication) based secure car access



Renesas automotive secure portfolio

Devices	R-Car (SoCs)	RH850 (MCU)	RL78/F24, RL78/F23 (MCU)
Description	Designed for intelligent and automated driving vehicles and embedded HSM provide advanced security protection against cyber attacks.	32-bit automotive MCUs offers high performance balanced with very low power consumption, rich functional safety and security features embedded HSM.	16-bit ultra low power MCU designed for actuator and sensor application embedded with hardware security features optimized for EVITA-Light security.
HSM	Yes	Yes	Yes
EVITA Targeted Use Cases	**EVITA Full** focuses on securing communication between the vehicle and the outside world	**EVITA Medium** focuses on securing in-vehicle communications between ECUs	**EVITA Light** focuses on securing communication to small low power, low-cost sensors and actuator

Also RENESAS uses evita as a reference on different devices equipped with HSM that are compliant with EVITA Light-Medium-Full.

Renesas automotive secure portfolio

Renesas PSIRT (Renesas Product Security Incident Response Team)

CS Management System

Renesas' automotive CSMS will assign not only leaders responsible for hardware and software security development, but also system security leaders to address system use cases, aiming to provide not only compliance with standards but also optimal solution

Renesas automotive MCUs and SoCs will sequentially follow the ISO/SAE 21434 standard with development starting from January 2022.
This includes the company's 16-bit RL78 and 32-bit RH850 MCUs as well as Renesas' popular R-Car SoC Family.

https://www.renesas.com/eu/en/application/automotive/common-automotive-technologies/automotive-security#overview

Outline

- Architectures and components of HSM in automotive:

Analysis of HSM in commercial automotive MCUs (NXP, ST, Renesas)

Analysis of commercial IPs for HSM (by Rambus, Synopsys, …..)

- HW for Secure Cards
- Secure key storage/management and advanced secure features in EPI HSM
- HSM evolution roadmap
    - Intrusion Detection System (IDS) support
    - Post-quantum cryptography (PQC) and tracking of NIST standardization
- Q&A session

Security IP vendor market

Solutions not for chips but for IP directly. There can be Root of Trust HW solutions or even acceleration solutions. IPs can be soft IPs or even hard IPs customised on a specific technology.

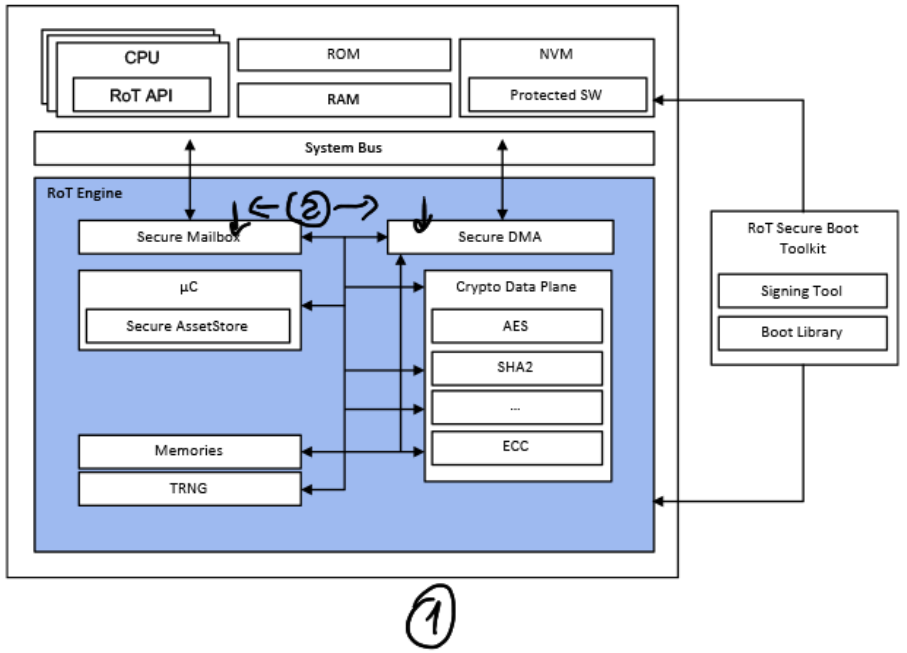
Many security IP vendors prove increase interest in HW acceleration of security (both complete RoT programmable solutions or library of acceleration of several basic crypto functions)

- ARM [Arm Confidential Compute Architecture – Arm®]
- Inside secure (Verimatix) recently acquired by Rambus
- Secure RF (now Veridify Security)
- SecureIC
- Synopsys Security IPs
- Athena group (INTEL partner)
- Cast Silicon IP cores (INTEL partner)
- SiFive (IPs for RISC-V…[Shield SoC Security - Securing the RISC-V Revolution – SiFive])
- Ingeniars (Pisa, partner of Xilinx and Microsemi) [Cyber Security - IngeniArs]
- …..

- Most famous IP vendor is ARM but this is a vibrant market full of acquisitions etc.

Security IP RoT solution example (1/2)

Example Inside Secure programmable Root-of-Trust solution IP



Dedicated IP secure co-processor on the system bus (e.g. AXI) including:

Secure DMA

Secure core to run SW library (not all possible versions/modes of crypto functions may be implemented in HW)

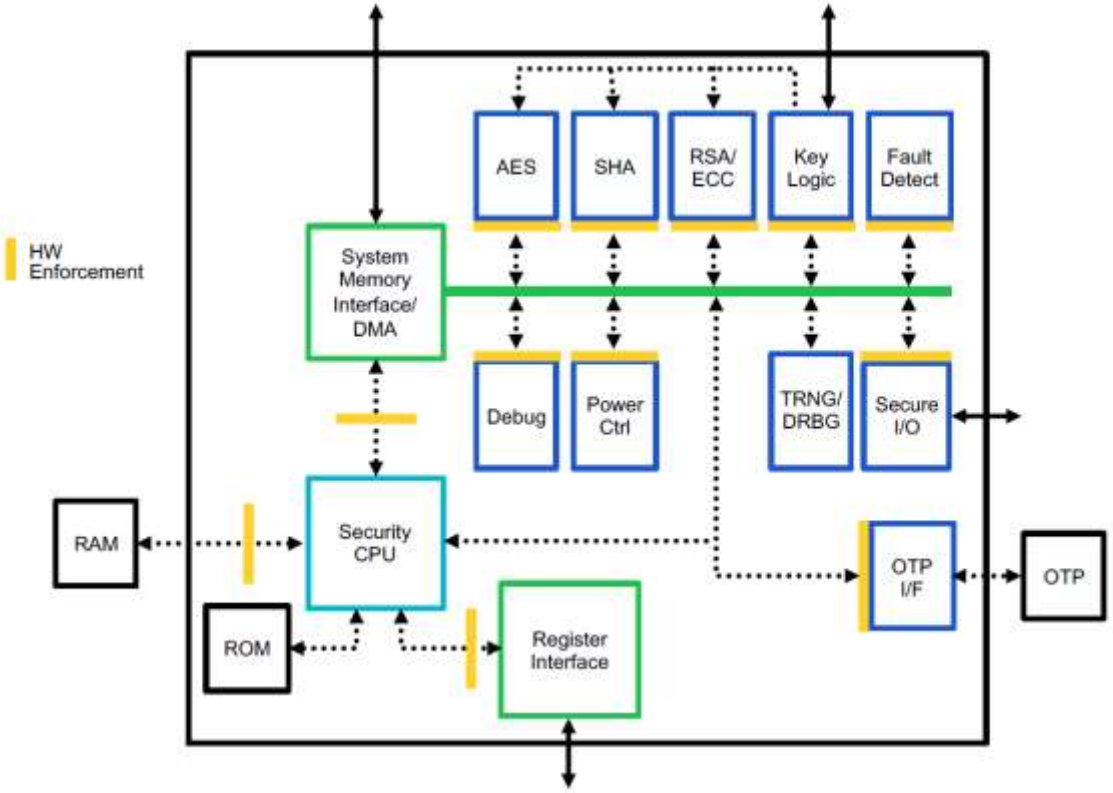
Some dedicated HW engines (ECC, AES, SHA2, RNG)

dedicated secure memory for key and configuration firmware storage

① This is the IP you can buy. ② They work like the firewall interface: mailbox you pass messages for security services request (ex. signature verif.). DMA you might need to access the memory. You have accelerators (Crypto data plane), then Memory and TRNG. The blue piece is the IP.

Security IP RoT solution example (2/2)

- Inside Secure (Rambus) specific RT-730 IP:
- Secure and fast DMA
- 32 bit RISC-V CPU Secure core
- high-capability cryptographic accelerators like AES-128, HMAC 512 and SHA-2, RSA 4K, ECC 521, Fast DMA, NIST compliant Random Bit Generator
- Test and debug interface protected against power/EM side –channel analysis and timing analysis.
- Support of SPA/DPA (Simple and Differential Power Analysis)



This is another version: This model supports protection against side channel attack. Power/Electromagnetic side channel analysis. This might reduce efficacy of chip.

Security IP basic functions, example (1/2)

Entire SoC or only pieces here: only HW accelerators for example

AES accelerators* *Inside Secure example but similar available from other vendors

- AES-IP-36 supports 5 modes ECB, CBC, CTR, OFB, CFB at 128,192 and 256 keys,
- AES-IP-38 supports AES XTS/GCM accelerators plus non feedback mode ECB and CTR, and feedback modes CBC, OFB, CFB
- AES-IP-39 / EIP-39 AES supports ECB/CBC/CTR/OFB-128/CFB-128, CCM/GCM/CMAC/XCBC MAC/ plus optional XTS accelerators. Throughput in the order of few Gbps → Some may be optimized in terms of speed
- AES-IP-61 supports GMAC, CTR, and ECB at 156 bits Keys but up to 10 channels in parallel to sustain up to 100 Gbps encryption

NOTE: hashing engine that can reach gigabits per second in ASIC is standard. FPGA is slower → hundreds of Mb

HASH accelerators*

- CRYPT-IP-120f: AES with all key sizes and modes (128, 192 and 256-bit key support; AES-ECB/CBC/CTR; AES-CBC-MAC; AES-CCM; AES-GCM Optional, by default available) + Local key store (Local 8 x 128-bit (or 4 x 192-bit/256-bit) encryption key storage, SHA-256 / SHA-224. 0.8 GBps for AES 256 to 4Gbps for SHA256
- HASH-IP-57/59 supports MD5 (RFC1231), SHA-1 (FIPS-180-2), SHA-2 (FIPS-180-3/4, 224/256/384/512) and SHA-3 (FIPS-202, 224/256/384/512)
  Throughput in the order of few Gbps to tens of Gbps

TRNG on-chip*

- TRNG-IP-76 (EIP-76): four to eight Free Running Oscillators (FROs) plus DBRG with AES and SHA

Security IP basic functions, example (2/2)

All the market for PKE is there

Public Key Cryprography accelerators (PKA)* *Inside Secure example but similar available from other vendors

- PKA-IP-150 combines PKA-IP-28, TRNG-IP-76 with an AMBA interface such as AXI or AHB
- PKA-IP-28 (The PKA-IP-28 is available in nine different performance configurations ranging from 19K to 515K gates designs, each providing the full set of PKA operations with up to 4160-bit modulus size for modular exponentiations and 768-bit modulus for prime field ECC operations. Smaller and slower versions available: example range 103…3,500 1K bit RSA/CRT ops/s
- The PKI-IP-154 public key accelerator combines an array of PKA-IP-28, TRNG-IP-76 with an AMBA interface such as AXI or AHB (it also supports ECDSA 521-bit sign: 2,310 ops/s & verify: 1,190 ops/s)
- PKI-IP-154 is available in 2 different performance configurations ranging from 350K to 1000K gates designs, each providing the full set of TRNG and PKA operations with up to 4160-bit modulus size for modular exponentiations and 768-bit modulus for prime field ECC operations

Ko. Usually RSA, ECC...

Metric here is # signatures you can do per second

Performance @400MHz

- ECC DH 180/1K-bit exp/mod negotiate: 10,500 ops/s; RSA 1K-bit sign (no CRT): 2,000 ops/s; RSA 1K-bit sign (with CRT): 3,500 ops/s; RSA 1K-bit verify (17 bits exp): 70,000 ops/s
- DSA 160/512-bit exp/mod sign: 16,000 ops/s & verify: 8,900 ops/s
- ECDSA 192-bit sign: 2,950 ops/s, verify: 1,650 ops/s
- ECDSA 384-bit sign: 900 ops/s; verify: 490 ops/s

INTEL dedicated secure solutions for edges (including vehicles)

INTEL FPGAs/SoCs embedded solutions for secure device management

INTEL QAT (Quick Assistant Technology) in ATOM processors

INTEL partners for IP core acceleration

- INTEL has some dedicated solutions too.

INTEL secure device manager in HW

Intel FPGAs/SoCs embedded solutions for secure device management

The Intel® Stratix® 10 device family (FPGAs and SoCs) introduces a full-feature Secure Device Manager (SDM). Present also in its successor Intel Agilex

Serving as the central command center for the entire FPGA, the Secure Device Manager controls, such as configuration, device security, single event upset (SEU) response key operations, and power management

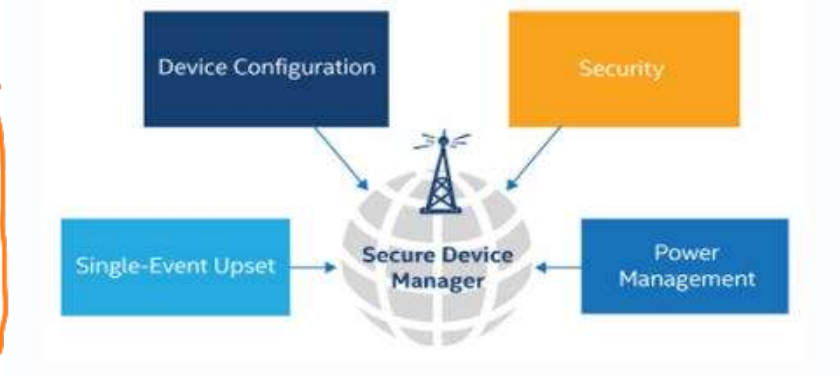
Booth strategy: first the SDM is configured, then the SDM configures the FPGAs/SOC

Secure Boot works the same way here too, Note that if FPGAs are RAM based, you need to reconfigure them at boot up.

Support of secure GPGA sector-base configurations

INTEL secure device manager in HW

once you check for integrity, you check also for Single-Event Upset



Key Operation	Description
Configuration	• Manages device startup in user mode. • Supports loading user configuration data. • Configuration bitstream decompression.
Security	• Provides security services to other modules. • Key encryption and authentication. • Bitstream decryption. • Tamper monitoring.
Single-Event Upset (SEU)	• SEU detection and correction.
Power Management	• Manages smart voltage ID operations. • Monitors critical power supplies.

Attacks can also show in Power consumption so check

INTEL secure device manager in HW

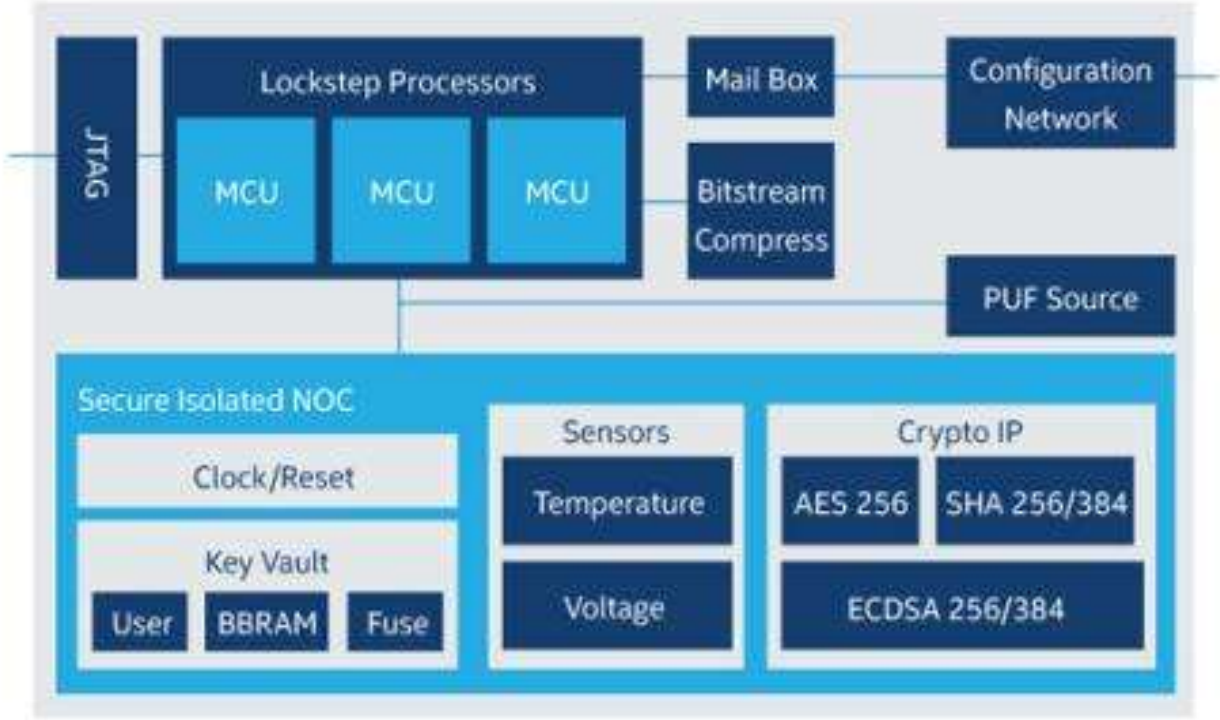
AES-256/SHA-256 for encryption/authentication, physically unclonable function (PUF)
ECDSA 256/384 boot code authentication,
side channel attack protection
(full features in Intel Agilex-10nm/Stratix10-14nm)

Advanced key management schemes enabled by secure processor

specifically elliptic curve Diffie–Hellman (ECDH) for key exchange and Elliptic Curve Digital Signature Algorithm (ECDSA) for digital signature

Elliptic curve algorithms significantly reduce key length versus signature strength over other asymmetric algorithms such as RSA

Structure with three MCU (triple redundancy), a PUF source to identify chip in a unique way.



Embedded secure services in Intel SoCs/FPGAs

Device Security Features	Cyclone® III LS FPGA	Stratix V FPGA Arria V FPGA/SoC Cyclone V FPGA/SoC	Intel MAX® 10 FPGA	Intel Arria 10 FPGA/SoC	Intel Stratix 10 FPGA/SoC
Bitstream Encryption and Authentication (AES-256/SHA-256)	Encryption	Encryption	Encryption	Encryption/ Authentication	Encryption/ Authentication
Volatile and Non-Volatile Key Storage	Volatile	Both	Non-Volatile	Both	Both plus PUF
Boot Code Authentication (SoC)	N/A	No	N/A	ECDSA256	ECDSA 256/384
Side Channel Attack Protection	No	No	No	Yes	Yes
Readback, JTAG, and Test Mode Disable	No Readback, JTAG Disable	No Readback, Fuse JTAG Disable	No Readback, Fuse JTAG Disable	No Readback, Fuse JTAG Disable	Readback and JTAG Fuse Disable
Dedicated Secure Configuration Processor	No MISSING	No	No	No	Yes, see Table 2

→ Different families of products

Table 1. Overview of Security Features Offered by Intel FPGA Products

Intel Stratix 10 Device New Security Features	Feature Value		
	IP Protection	Glitch/Malware Protection	Tools for Secure System Design
Sector-Based Security	Allows Multi-Key IP Protection	SEU/Glitches Detected at Sector Level	–
SRAM PUF for Key Protection and Hardware Identity	Additional Source of Key Protection	–	Unique ID Available for Hardware Authentication
Flexible Key Management and Generation Capability	Broader Source of Key Material	Device/Key Updates After Compromise	Tools for Key Management Applications
Hard Encryption/Authentication Engines	–	Hashes and Signatures to Detect Modification	Hard IP Blocks Available to FPGA Designers
Programmable and Updateable Configuration Process	Update Configuration After Compromise	Update Configuration After Compromise	–
Command Authentication for Remote Update and Key Management	Secure Update of Device	Active Prevention of Malicious Updates	Field Upgrades
Scripted Responses to Sensor Inputs, including Tailored Zeroization	Zeroize keys when attack detected	–	Part of Larger Security and Tamper Resistance Solution

Table 2. New Security Features and Use Cases

→ Pawe mode: cancel the key if attack is detected

Intel® QuickAssist Technology (Intel® QAT)

↳ For ATOM they use this.

- Used in ATOM edge processors
- Includes symmetric encryption and authentication, asymmetric encryption, digital signatures, RSA, DH, and ECC, and lossless data compression
- Intel® QAT engine for OpenSSL 1.1.0 improves the performance of secure applications by offloading the computation of cryptographic operation

↳ wide used library for security. Taking openssl and we accelerate it.

- Asymmetric Public Key Encryption Offload:

RSA Support with PKCS1 Padding for Key Sizes 1024/2048/4096

DH Support for Key Sizes: 768/1024/1536/2048/3072/4096

DSA Support for Key Sizes: 160/1024, 224/2048, 256/2048, 256/3072

ECDH Support for the curves: NIST Prime Curves P-192/P-224/P-256/P-384/P-521. NIST Binary Curves B-163/B-233/B-283/B-409/B-571. NIST Koblitz Curves K-163/K-233/K-283/K409/K-571.

ECDSA Support for the curves: NIST Prime Curves P-192/P-224/P-256/P-384/P-521. NIST Binary Curves B-163/B-233/B-283/B-409/B-571. NIST Koblitz Curves K-163/K-233/K-283/K409/K-571.

Symmetric Chained Cipher Offload: AES128-CBC-HMAC-SHA1/AES256-CBC-HMACSHA1. AES128-CBC-HMAC-SHA256/AES256-CBC-HMACSHA256.

Pseudo Random Function (PRF) offload while freeing up the processor to perform other tasks.

Athena IP solutions for INTEL embedded security

Athena Terafire IP catalougue for INTEL FPGAs/SoCs

Model	Description
InCipher-DPA	Inline Memory Encryptor
EXP-F5200	Embedded Cryptography Microprocessor
EXP-F5200B	Embedded Suite B Cryptography Microprocessor
EXP-E5200	32-bit Cryptography Microprocessor
EC Ultra	High performance Elliptic Curve Accelerator
EC Ultra HP	Ultimate Performance Elliptic Curve Accelerator
AES-A500	Ultra Performance AES
AES-A100	High Performance AES
AES-A200	Standard Performance AES
SHA2-A300	SHA-1 plus SHA-2 (224/256/384/512)
SHA2-A400	Double Performance SHA2-A300
RNG-A100	SP800-22 True Random Number Generator
RNG-A200	SP800-90 True Random Number Generator
KAS-A100	Kasumi 3GPP Cipher
SNOW-A100	SNOW 3GPP Cipher
ZUC-A100	ZUC 3GPP Cipher
3DES-A100	Data Encryption Standard (DES/3DES)
ARC4-A100	ARC4 Stream Cipher
MD5-A100	Message Digest 5 (MD5)

Model	Description
InCipher-SCA	Inline Memory Encryptor
EXP-F5200-SCA	Embedded Cryptography Microprocessor
EXP-F5200B-SCA	Embedded Suite B Cryptography Microprocessor
EXP-E5200-SCA	32-bit Cryptography Microprocessor
EXP-E6400-SCA	64-bit Cryptography Microprocessor
EC Ultra-SCA	High Performance Elliptic Curve Accelerator
EC Ultra HP-SCA	Ultimate Performance Elliptic Curve Accelerator
AES-A500-SCA	Ultra Performance AES
AES-A100-SCA	High Performance AES
AES-A200-SCA	Standard Performance AES
AES-A300-SCA	Compact AES
SHA2-A300-SCA	SHA-1 plus SHA-2 (224/256/384/512)
SHA2-A400-SCA	Double Performance SHA2-A300
RNG-A200-SCA	SP800-90 True Random Number Generator

Athena IPs for INTEL embedded security-performance for public key cryptography

Table 1: TeraFire® 5200 Series Performance[a]

Operation	Cyclone® 10	Arria® 10	Stratix® 10
	op/s	op/s	op/s
RSA-1024 Private Key	1637	1637	1731
RSA-1024 Private Key (Paired Cores)	3274	3274	3462
1024-bit Expo w/ 1024-bit Expo.	324	324	343
1024-bit DSA Sign	1860	1860	1966
1024-bit DSA Verify	944	944	998
RSA-2048 Private Key	164	164	174
RSA-2048 Private Key (Paired Cores)	328	328	347
2048-bit Expo w/ 2048-bit Expo.	55.5	55.5	58.7
2048-bit DSA Sign	469	469	496
2048-bit DSA Verify	246	246	260
RSA-3072 Private Key	59	59	62.4
RSA-3072 Private Key (Paired Cores)	118	118	125
NIST P-256 EC Point Multiply	724	724	765
NIST P-256 ECDSA Sign	659	659	697
NIST P-256 ECDSA Verify	568	568	601
NIST P-384 EC Point Multiply	363	363	384
NIST P-384 ECDSA Sign	335	335	354
NIST P-384 ECDSA Verify	283	283	299
521-bit EC Point Multiply	73.9	73.9	78.1
Area (LUTs)	8470	8550	9020
[redacted]	175 [redacted]	[redacted]	[redacted] MHz

Table 1: TeraFire F5200 Performance[a]

Operation	Cyclone® 10	Arria® 10	Stratix® 10
	op/s	op/s	op/s
RSA-1024 Private Key	93	96	100
RSA-1024 Private Key (Paired Cores)	186	192	199
1024-bit Expo w/ 1024-bit Exponent	26.2	27.1	28.1
1024-bit DSA Sign	172	177	184
1024-bit DSA Verify	82	84	87
RSA-2048 Private Key	13.4	13.8	14.3
RSA-2048 Private Key (Paired Cores)	26.8	27.6	28.7
2048-bit Expo w/ 2048-bit Exponent	3.5	3.6	3.8
2048-bit DSA Sign	30.7	31.7	32.9
2048-bit DSA Verify	15.8	16.3	16.9
RSA-3072 Private Key	4.1	4.2	4.4
RSA-3072 Private Key w/ Paired Cores	8.2	8.4	8.8
256-bit EC Point Multiply	78	81	84
256-bit ECDSA Sign	72	75	78
256-bit ECDSA Verify	63	65	67
384-bit EC Point Multiply	28	29	30
384-bit ECDSA Sign	25.8	26.6	27.6
384-bit ECDSA Verify	22.1	22.8	23.7
521-bit EC Point Multiply	11.3	11.6	12.1
Area (LUTs)	2770	2720	2900
Frequency	155 MHz	160 MHz	166 MHz

a. Other Intel device families supported. Contact Athena for more information.

Athena IPs for INTEL embedded security-performance for HASH and symmetric key cryptography

Table 1: SHA Cyclone® 10 Specifications[a]

	SHA2-A300	SHA2-A400
SHA1	1.4 Gbps	3.0 Gbps
SHA2-224/256	1.7 Gbps	3.7 Gbps
SHA2-384/512	2.7 Gbps	3.1 Gbps
Area (LUTs)	1550	TBD
Frequency	250 MHz	250 MHz

Athena supports all AES modes, including ECB, CBC, CFB, OFB, CTR, CMAC, CCM, GCM, and GHASH, and even XTS mode (SP800-38E). Any modes and/or key sizes not required can be omitted to reduce area.

Table 2: TeraFire F5200 Options[a]

Operation	Cyclone 10		Arria 10		Stratix 10	
	Perf.	Area	Perf.	Area	Perf.	Area
AES	211 Mbps	280	218 Mbps	325	226 Mbps	237
GCM	211 Mbps	737	218 Mbps	756	226 Mbps	812
SHA	275 Mbps	463	284 Mbps	507	294 Mbps	535
SP800-90 DRBG	70 Mbps	1345	73 Mbps	1306	75 Mbps	1296

a. Area (LUTs) in addition to base F5200 core, for each feature. Other Intel device families supported. Contact Athena for more information.

Table 3: SHA Stratix® 10 Specifications[a]

	SHA2-A300	SHA2-A400
SHA1	1.2 Gbps	2.5 Gbps
SHA2-224/256	1.4 Gbps	3.2 Gbps
SHA2-384/512	2.3 Gbps	2.7 Gbps
Area (LUTs)	1780	TBD
Frequency	216 MHz	216 MHz

Table 6: AES-GCM Stratix® 10 Specifications[a]

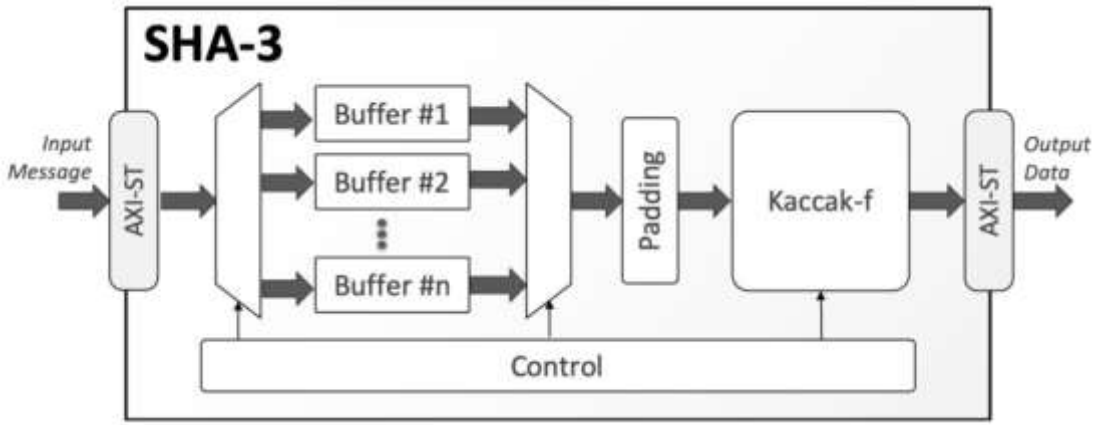
	AES-A100-GCM	AES-A200-GCM
Throughput	2.4 Gbps	700 Mbps
Area (LUTs)	3290	3100
Frequency	223 MHz	250 MHz

CAST IP SHA-3 for INTEL

- https://www.cast-inc.com/pdfs/cast_sha-3-brief-intel.pdf

Arria10 (speed grade -2)

Area (ALMs)						Freq. (MHz)	Number of In. Buffers
SHA3-224	SHA-256	SHA3-384	SHA3-512	SHAKE-128	SHAKE-256		
3,350	3,256	2,907	2,698	3,605	3,311	250	0
3,869	3,748	3,328	3,135	4,315	3,819	275	2



Conclusions on State-of-art HSM

- HW embedded security at the state of art used to enable security services in real-time and with limited power/area complexity
- Different architectures available and also security IP cores available for both complete programable root-ot-trust solutions or basic crypto functions

Usually proposed solutions include

- Symmetric crypto core: min AES 128 (basic modes) up to AES-128/256 full modes
- HASH: basic SHA1, up to SHA-2 with 512 (SHA3 still not with widespread adoption)
- On-chip True RNG (and optional DRBG to increase data-rate)
- Public Key crypto cores: from basic RSA to ECC with support of ECDSA, ECDH
- On-chip dedicated memory for key storage and configuration storage
- Optional Programmable 32b core

Outline

- Architectures and components of HSM in automotive:

Analysis of HSM in commercial automotive MCUs (NXP, ST, Renesas)

Analysis of commercial IPs for HSM (by Rambus, Synopsys, …..)

- HW for Secure Cards
- Secure key storage/management and advanced secure features in EPI HSM
- HSM evolution roadmap

    Intrusion Detection System (IDS) support

    Post-quantum cryptography (PQC) and tracking of NIST standardization

- Q&A session

HW for Secure Cards

HW for specific memory cards

- SD (Secure Digital)
- SIM (Subscriber Identity Module)

Both are standardized

Memory cards have embedded security features for ID of the contract subscriber

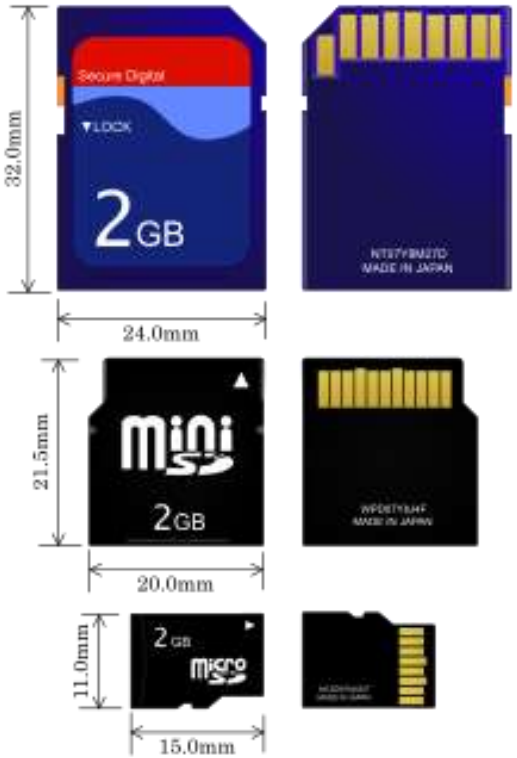
HW for Secure Cards

Reprogrammable memory card

Secure Digital, abbreviated as SD, is a proprietary, non-volatile, flash memory card format of the SD Association (SDA, involving people from SanDisk, Panasonic, Matsushita, Toshiba developed for use in portable devices → it is an industrial standard created

like EVITA

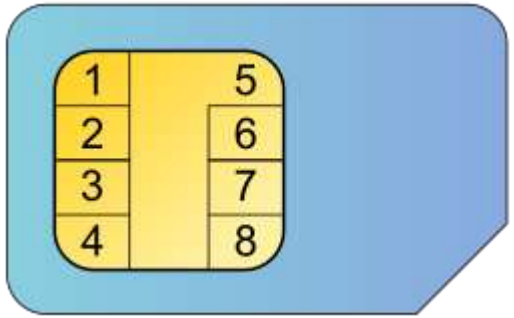
SD cards have a write lock. This is a switch on the side of the card. The DOWN position is write protect ON and the UP position is write protect OFF



Form Factor		SD	microSD
Dimension			
Card Capacity Type		SD, SDHC, SDXC and SDUC	
Physical	Number of pins	High Speed and UHS-I : 9 pins UHS-II: 17 pins SD Express 1-lane: 17-19 pins SD Express 2-lane: 25-27 pins	High Speed and UHS-I : 8 pins UHS-II: 16 pins SD Express 1-lane: 16-17 pins
	Operating Voltage	3.3V VDD range in the first-row: 2.7V – 3.6V 1.8V VDD range in the second-row: 1.70V-1.95V	
	Write-protect Switch	YES	NO

- Standard defines the size and pin of cards.
  For security, cards can be write locked (only SD)
- Also the creations of smartphones and others adapted to this standard.

HW for SIM (Subscriber Identity Module)



Electric contacts of a SIM card (8 contacts version)

1 - **V_{cc}** (voltage supply)
2 – **RST** (reset)
3 – **CLK** (clock)
4 – **reserved** (absent in SIM 6 contacts)
5 – **GND** (ground)
6 – **V_{pp}** (voltage used to program/erase the NonVolatile memory inside the SIM card)
7 – **I/O** (dati input/output)
8 – **reserved** (absent in SIM 6 contacts)

SIM card (Subscriber Identity Module or Subscriber Identification Module) is an integrated circuit (IC) intended to securely store an international mobile subscriber identity (IMSI) number and its related key, which are used to identify and authenticate subscribers on mobile telephone devices

A SIM contains a unique serial number, integrated circuit card identification (ICCID), international mobile subscriber identity (IMSI) number, security authentication and ciphering information, temporary information related to the local network, a list of the services the user has access to, and four passwords: a personal identification number (PIN) for ordinary use, and a personal unblocking key (PUK) for PIN unlocking as well as a second pair (called PIN2 and PUK2 respectively) which are used for managing fixed dialing number and some other functionality.

The chip in smartphone card is standardised. Chip is smaller than card itself (Micro SIM etc.). Chip is semiconductor part + plastic support, you can have different sizes. Prim you see are standardised. There's a 6 or 8 contact versions.

① SIM contains NVM that you can program. Program requires voltage level higher than the one you need to read.

SIM de facto is a non volatile memory. Inside SIM you store secrets (separated from memory of the smartphone), and SIM should be registered to your name in a national registry. SIM contains the IMSI number and its key used to identify and authenticate subscribers on mobile devices. Provider is like a certification authority. The IMSI can be used to verify that you are authentic, and can be also used for geolocalization.

PUK is a PW to unlock PIN. PUK is longer, PIN is shorter.

- **IMSI (International Mobile Subscriber Identity):** This is like your phone's secret name in the mobile network's eyes. It helps identify you as a subscriber. Think of it as your phone's true name, the one it whispers only to the mobile gods when it wants access.
- **ICCID (Integrated Circuit Card Identifier):** This one is the ID of the SIM card itself—not the user. It's useful for logistics, like knowing which card is which in a stack of a thousand.
- **Authentication key (a secret tied to IMSI):** The SIM uses this to prove to the network that it's the real deal. Like a secret handshake. Without this, the network won't trust you.
- **Ciphering info:** That's the stuff used to encrypt your calls and texts so random folks can't just listen in.
- **Temporary info related to the local network:** This is just your SIM remembering where you are, which towers you're near, and other stuff to make reconnecting faster.
- **Service list:** This one tells the phone what features your carrier allows—like voicemail, data, or whether you can make international calls.

ICCI identifies HW, IMSI identifies the owner.

Now, about those passwords:

- **PIN:** The basic lock to stop someone from using your SIM if they steal it.
- **PUK:** If you mess up the PIN too many times, this guy comes in to unlock it. It's like calling the bank after too many wrong tries.
- **PIN2 and PUK2:** These are extra codes for more specific stuff, like limiting which numbers can be dialed (useful if you're giving a phone to a kid or using it for a work thing).

② See it as a NVM plus a small controller (Finite State Machine) to manage functionality (ex: PIN inputting)

Example of security features embedded in HW.

Outline

- Architectures and components of HSM in automotive:

Analysis of HSM in commercial automotive MCUs (NXP, ST, Renesas)

Analysis of commercial IPs for HSM (by Rambus, Synopsys, …..)

- HW for Secure Cards

- Secure key storage/management and advanced secure features in EPI HSM

- HSM evolution roadmap

    Intrusion Detection System (IDS) support

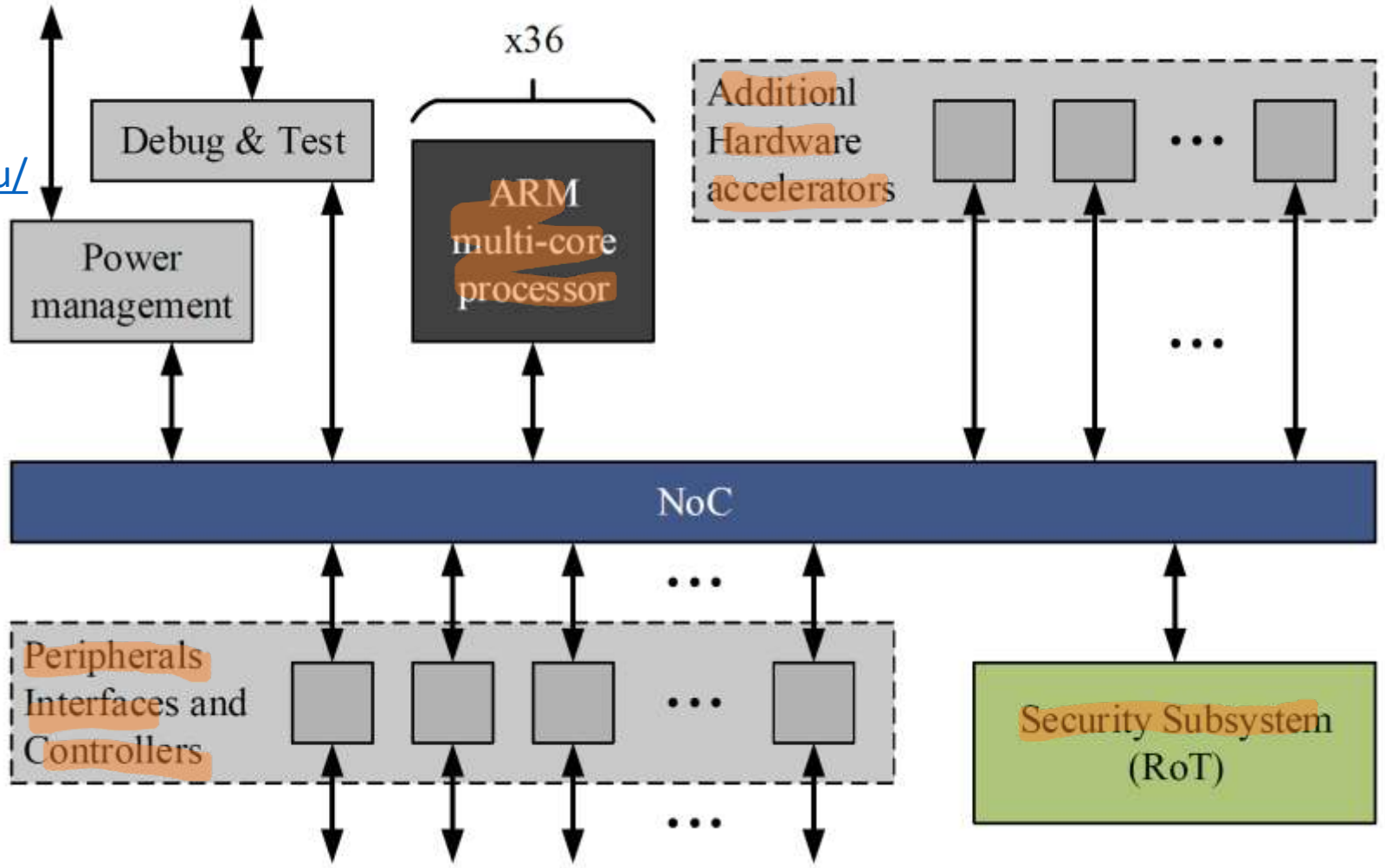
    Post-quantum cryptography (PQC) and tracking of NIST standardization

- Q&A session

European Processor Initiative (EPI) chip

Designed-in-EU chips based on ARM multi-core in TSMC technology
Customized secure solutions
Mixing industrial partners (e.g. Sipearl, BMW, Elektrobit, Infineon, BMW) and academic ones (e.g. University of Pisa)

https://www.european-processor-initiative.eu/



In the designed chip you have an 80 core processor along with periph., HW accelerators and a sec. subsystem all communicating on a NoC

- Founded by Europe this EPI, high end processor to be designed in Europe to face the fact that many solutions for high end processors come from AMD and INTEL.

This is the European Processor Initiative.

We will discuss solutions used in CS. Multi-core processor, IP acquired from ARM, 64 bit processor. 36 multi core processor at the beginning, now we shifted to an 80 core. Components are connected through "Network on Chip", not a Bus: bus stops at level 2. When you have a lot of parallel working cores, you work on higher layers too, considering routing etc. Additional components include security solution.

- Entire chip divided in 4 quadrants (trusted zone) + a security subsystem, with a secure MCU in RISC V to add diversity from ARM.*

In each quadrant you have this security subsystem (or kile).

You have dedicated memory for sub system and crypto kile (an accelerator): so 20 arm cores per quadrant that can interact with security sub system. Each quadrant considers the other untrusted.

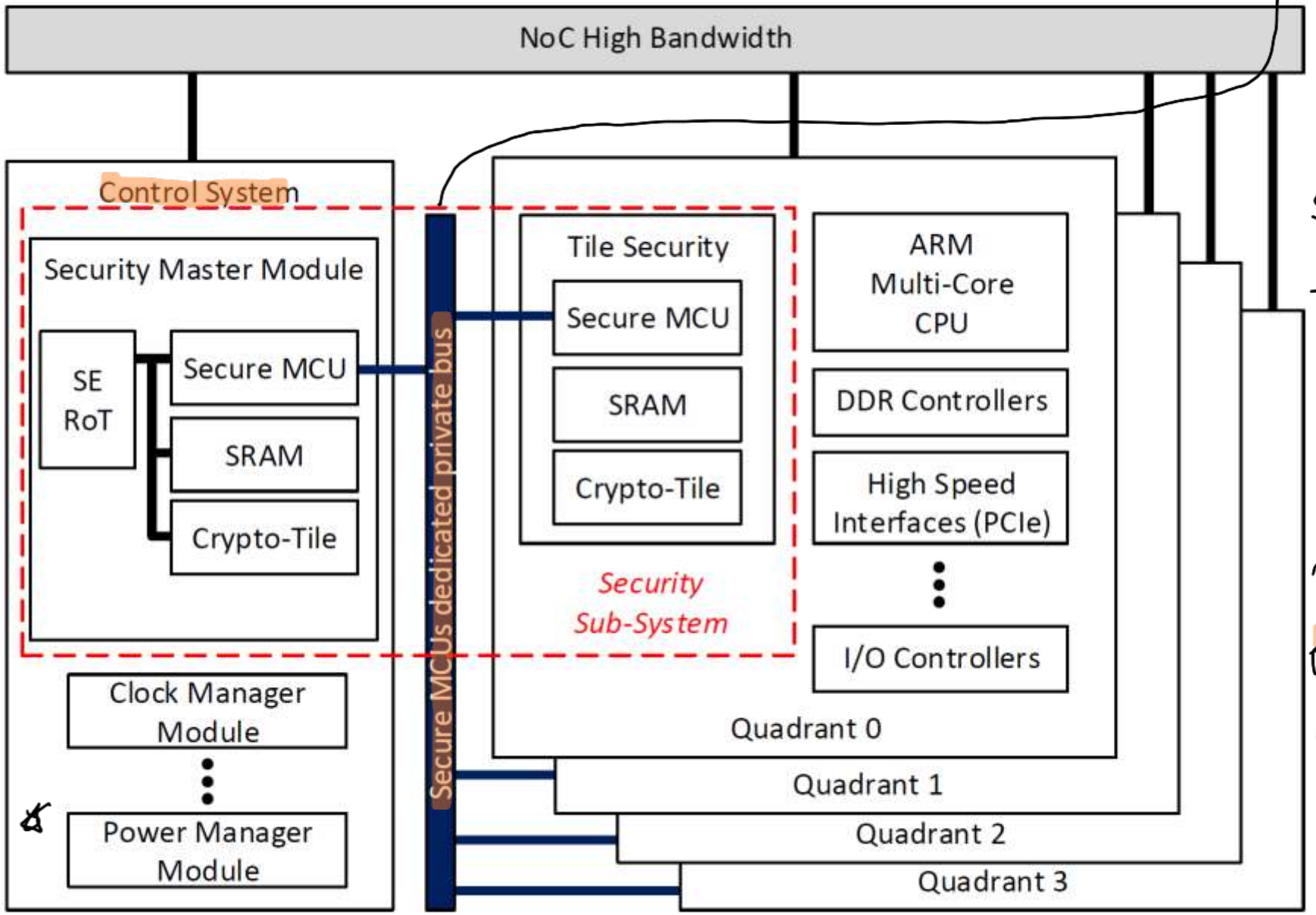
* Another instruction set in case ARM instr. set is bugged: separate controller IP and controlled.

The European Processor Initiative (EPI) is a project launched by the European Union to develop high-performance, energy-efficient processors that are made in Europe. The main idea behind it is to reduce Europe's dependence on foreign chip technologies—especially from the US and Asia—and strengthen its sovereignty in areas like supercomputing, data centers, and even AI and automotive industries.

At its core, the EPI is trying to build a European-made CPU architecture that can power future exascale supercomputers (basically insanely powerful machines used for things like climate modeling, space research, or simulating nuclear fusion). They're working on both general-purpose CPUs (based on Arm architecture) and RISC-V-based accelerators, which are more specialized for things like AI and machine learning.

TRUSTED ZONE : quadrant division

EPI chip security



Connects the security sub-systems privately to a security Master Module that controls security elements.

So 5 security elements, 1 master and 4 for quadrants.

Master has additional memory and a crypto-tile for PUF, secure boot

connected to a Secure Element Root of Trust.

The master manages PUF, secure boot, etc. through SE-RoT.
* In the same part of the system we have a clock manager and power manager. Each quadrant and processor can go at its frequency to optimize power consumption.

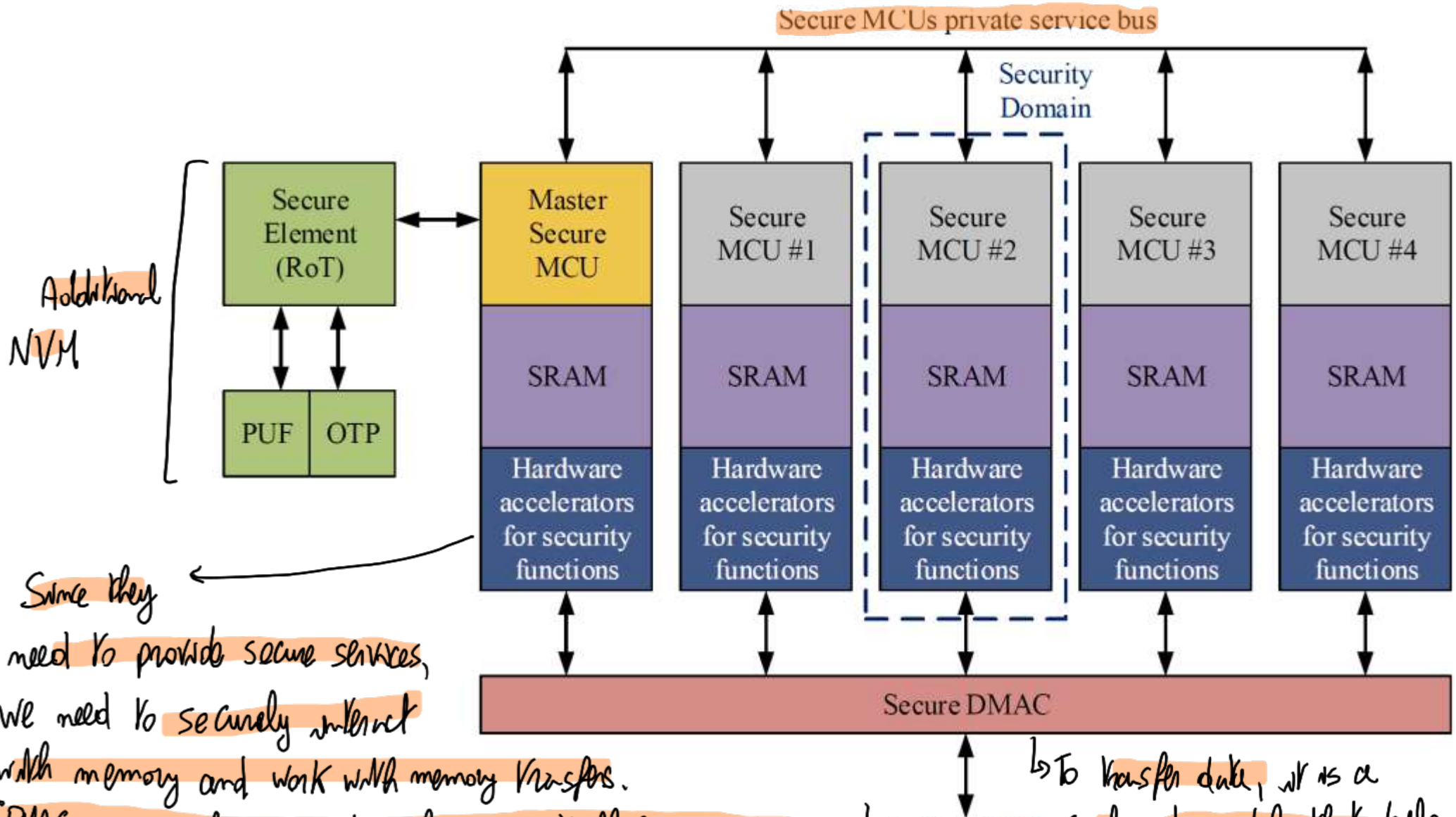
3. Crypto-Tile:
This is a dedicated hardware block for cryptographic operations—like encryption, decryption, hashing, digital signatures, and key generation. By offloading these to a specialized block, it speeds up operations and reduces attack surfaces, since the main CPU doesn't have to deal with sensitive computations directly.

4. SE RoT (Secure Element – Root of Trust):
This is the most sacred part of the chip. It holds the root cryptographic keys and anchors the whole system's trust chain. The SE RoT is often immutable or at least extremely hard to modify—it's the thing that says, "Yes, this chip is genuine, this code is trusted, and we're booting safely."

PUF

Secure Boot

EPI chip security



Additional NVM

Since they need to provide secure services, we need to securely interact with memory and work with memory transfers.

SDMAC can implement security rules: ex: not all SO processors can read

↳ To transfer data, we use a configured module that deals with secure memory transfer

Secure Direct Memory Access Controller

Crypto accelerator in the EPI chip

The restricted access to the OTP memory (SE and Master Secure MCU), storing boot-code and sensitive data, strengthen the security assets of the Rhea chip (secret, certificate, configuration, security policies, life-cycle, ...) against impact of bug, applicative malicious code, and external attacks

The PUF module is a reliable method to derive unique physical fingerprint that is extremely difficult to be cloned onto another same hardware component, and it can be integrated to assist the OTP and the SE in the secure boot sequence (making different chip by chip and hence reducing the risk that a secret stolen in one chip can be used for all the others in millions of vehicles)

guarantees secure boot sequence too. PUF also used as a sort of generator of the master key so that each component is different from the other.

			Class of cryptographic algorithms			
			Symmetric-key algorithms	Public-key algorithms	Hash functions	RBGs
Cybersecurity services	Confidentiality		✓	✓	×	×
	Integrity		✓	×	✓	×
	Authentication	Authentication of integrity (MAC)	✓	✓	✓	×
		Authentication of source (digital signature)	×	✓	×	×
	Non-repudiation (digital signature)		×	✓	×	×
	Support services	Key generation	×	✓	×	✓
		Key exchange/distribution	×	✓	×	×
		Key derivation	✓	×	×	×
		Generation of random and non-repeating values	✓	×	×	✓

OTP memory stores boot-code and sensitive data. Depending on use OTP can be part OTP part non volatile.

Crypto accelerator in the EPI chip

Minimum security strength is 128-

Security Strength (bits)	Process	Through 2030	2031 and Beyond
< 112	Applying protection	Disallowed	
	Processing	Legacy use	
112	Applying protection	Acceptable	Disallowed
	Processing		Legacy use
128	Applying protection and processing information already protected	Acceptable	Acceptable
192		Acceptable	Acceptable
256		Acceptable	Acceptable

Security Strength	Symmetric-key algorithms	Public-key schemes		Hash functions	MACs generation, RBGs
		IFC	ECC		
≤ 80	2TDEA	1024	160-223	SHA-1	-
112	3TDEA	2048	224-255	SHA2-224, SHA-3-224	-
128	AES-128	3072	256-383	SHA2-256, SHA-3-256	SHA-1
192	AES-192	7680	384-511	SHA2-384, SHA-3-384	SHA2-224, SHA-3-224
≥ 256	AES-256	15360	512+	SHA2-512, SHA-3-512	SHA2-256, SHA-3-256, SHA2-384, SHA-3-384, SHA2-512, SHA-3-512

Given that we want a security strength of 128 bits, and given the feature that we want, we choose the following algorithms: each standardised by an algorithm.

Crypto accelerator in the EPI chip

Very few use AES-192, to save area.

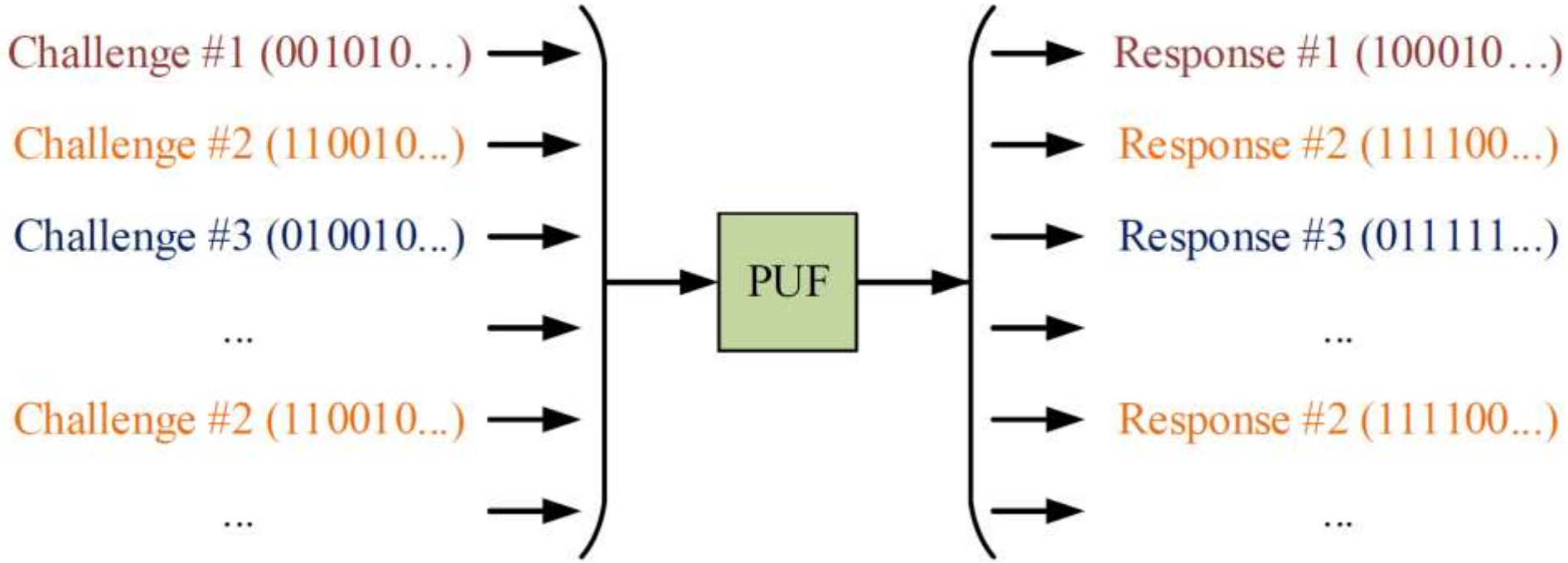
Cryptographic algorithm	Reference standard	Revision
AES-128, AES-256	NIST FIPS 197 [115]	Final revision, November 2001
ECB, CBC, CFB, OFB, CTR	NIST SP 800-38A [116]	Final revision, December 2001
CMAC	NIST SP 800-38B [117]	Final revision, June 2016
CCM	NIST SP 800-38C [118]	Final revision, July 2007
GCM (New methods)	NIST SP 800-38D [119]	Final revision, November 2007
XTS (New methods)	NIST SP 800-38E [120]	Final revision, January 2010
ECC functions	NIST FIPS 186-4 [121]	Final revision, July 2013
	SECG SEC 1 [122]	Version 2, May 2009
	ANSI X9.62 [123]	November 2005
SHA2	NIST SP 800-38C [124]	Final revision, August 2005
SHA-3	NIST SP 800-38D [125]	Final revision, August 2005
HMAC	NIST SP 800-38E [126]	Final revision, July 2008
Entropy source module (of CSPRNG)	NIST SP 800-38C [127]	Final revision, January 2018
DRBG mechanism (of CSPRNG)	NIST SP 800-38D [128]	Revision 1, June 2015
CSPRNG	NIST SP 800-38E [129]	Latest draft, April 2016

If you need AES-192 you can run it on SW on the secure MCU.

Crypto accelerator in the EPI chip

The PUF is based on the challenge-response protocol, i.e. when a stimulus is applied to it (challenge, C), it reacts in an certain way (response, R)

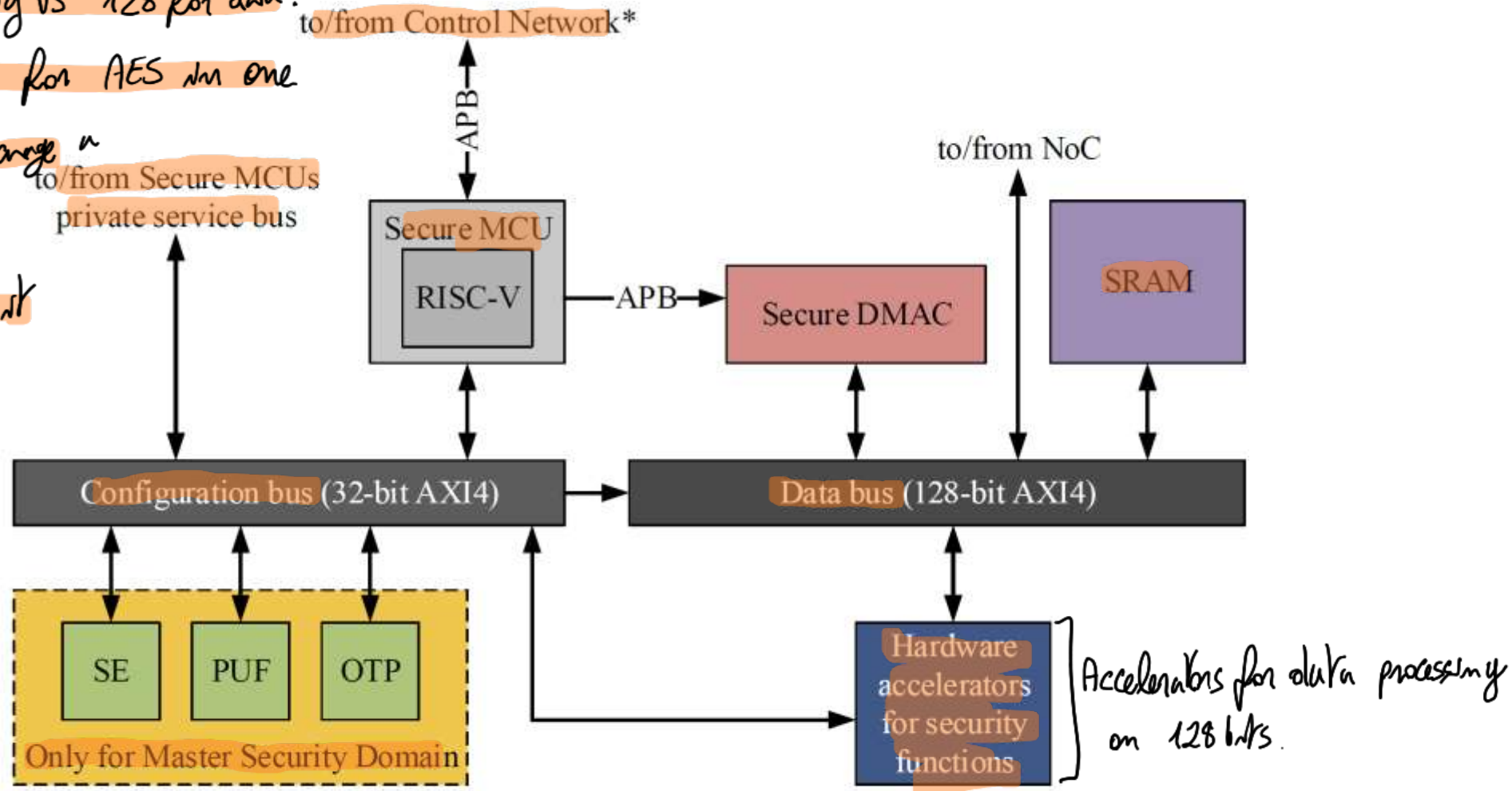
The reaction of the PUF is unpredictable, because it depends on the internal device structure, and this last one is determined by intrinsic random fluctuations that necessarily occur during the physical manufacturing processes. Anyway, even if the reaction of a PUF module is unpredictable, on one hand, on the other one it is also repeatable, hence each challenge is coupled to a specific response (challenge-response pair). Due to the unavoidable variations of chip manufacturing processes, each instance on silicon of the same PUF implementation brings to a different physical structure, thus to a different reaction when the same stimulus is applied to all of them.



- If you take one MB of SRAM at power on, due to process variation, you can kind of expect a signature to be preset. Challenge: addresses of the rows, for example. That RAM will give the answer that will be different for each chip. Pairs (addresses, answers) is the fingerprint.

Crypto accelerator in the EPI chip

Why 32 bits for config vs 128 for data?
32 bits is enough while for AES in one transaction you can manage a single block.
RISC V is a 32 bit microcontrollers.

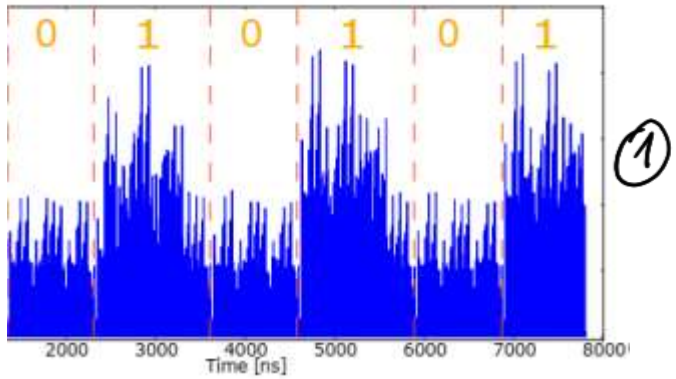


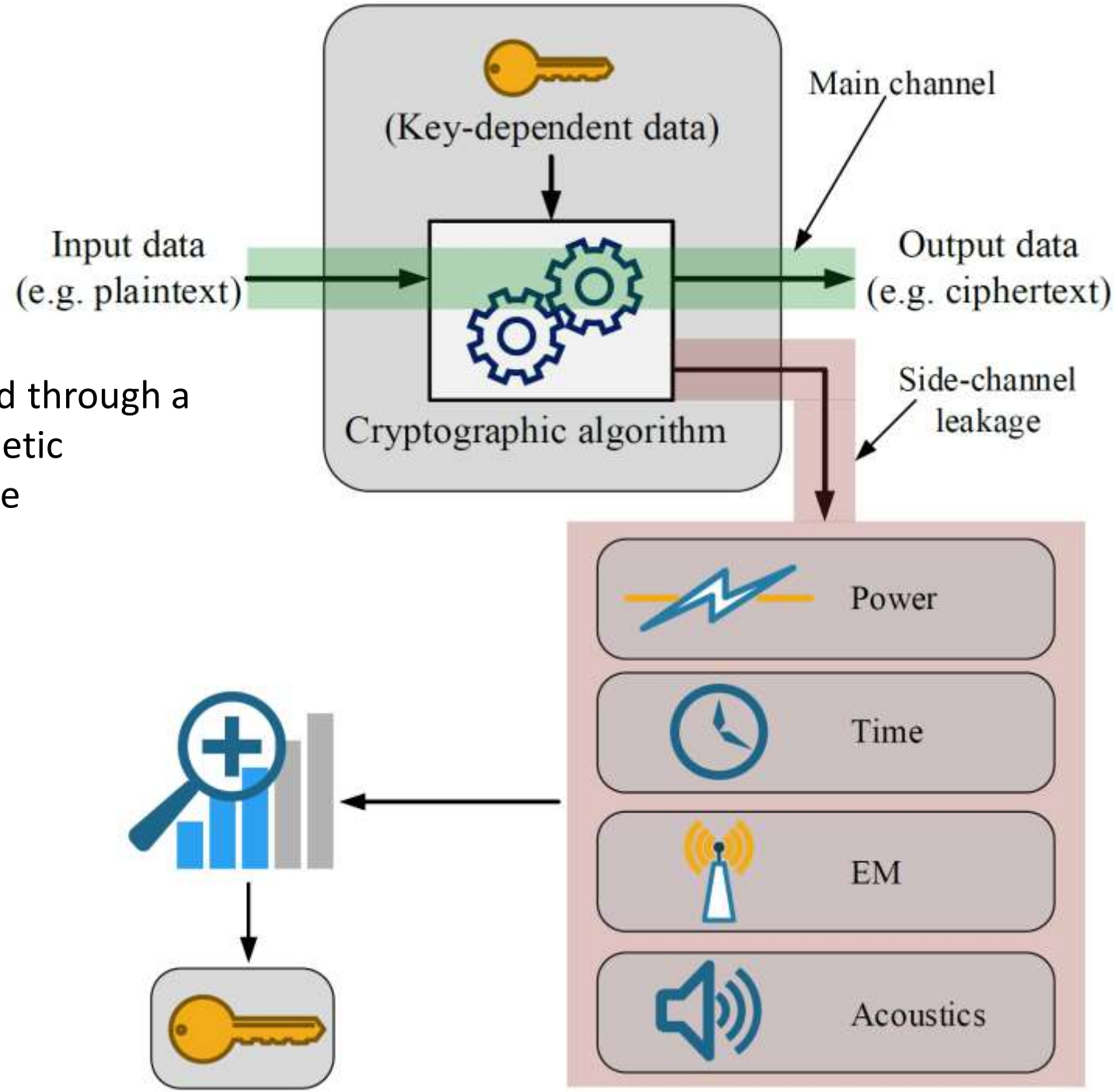
Note: if you work with SW implementation on secure MCU, you have an additional 4 times slowdown because you work on 32 bits.

* Only for Security Domain #1, #2, #3 and #4; not included in Master Security Domain

Side Channel Attack (SCA)

Scheme of operations of SCAs.
By analysing the leakage of information emitted through a side-channel such as power, time, electromagnetic emissions(EM) or acoustics, an adversary is able to retrieve the secret key.



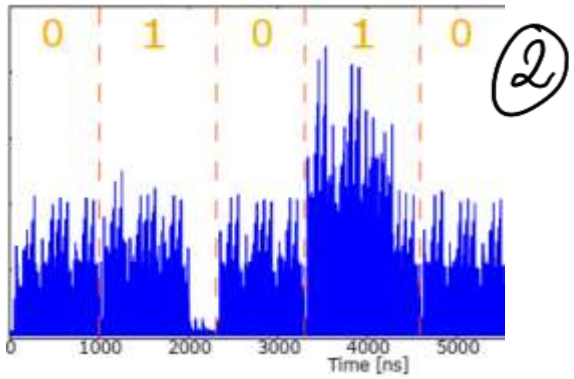


[In processor we also put protections against SCA.]

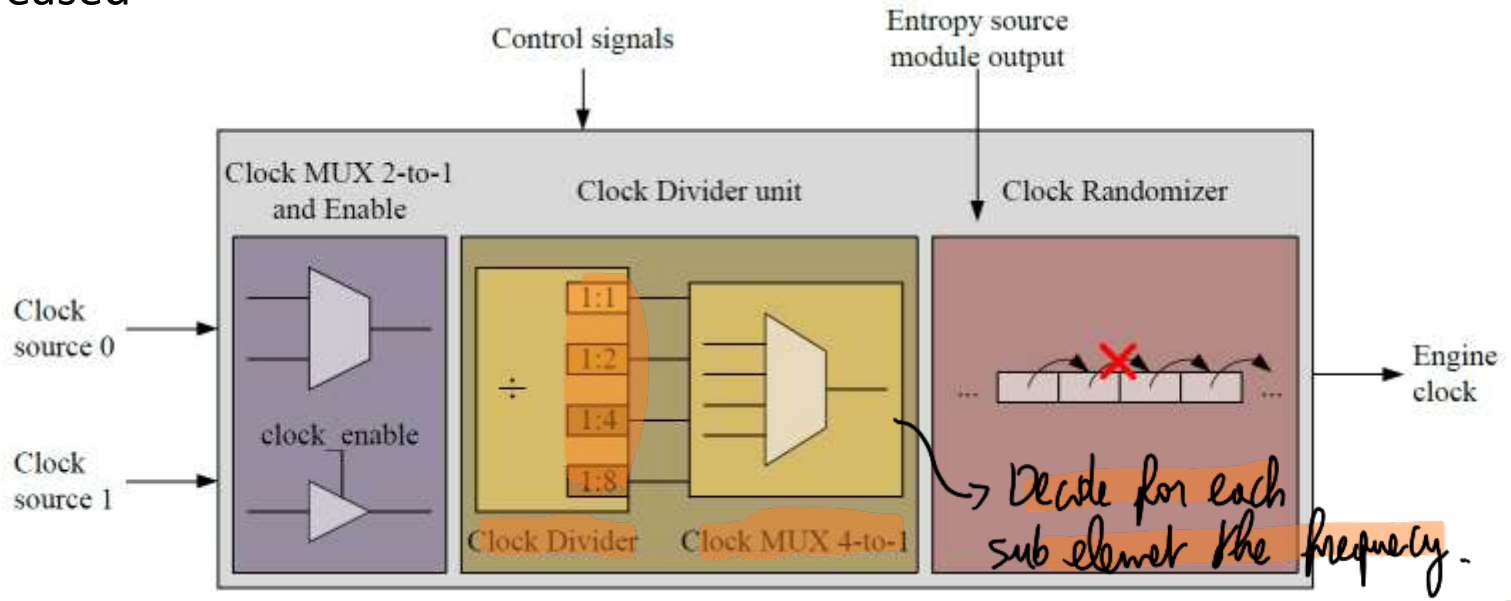
① If you design a chip in a way that it blatantly shows what you are doing you can be subject to SCA.
Idea is you infer what a processor or a device is doing, you can analyse a side channel leakage related to power consumption, computation time, electromagnetic waves and acoustics. This might give enough info for an attack.
- Idea is uncorrelate info with leakage. Look at ②

Side Channel Attack (SCA)

Thanks to circuit design tricks (like randomizations of clock sources) SCA-resistance can be increased



② ③



You could randomize division

You spread frequency

Dedicated clock sub-system: A dedicated clock sub-system allows to vary the clock used for cryptographic operations (including key management), in order to enhance the robustness with respect to time based and statistical attacks:

- Every cryptographic engine (i.e. AES, SHA, RNG and ECC engine) inside the Crypto-Tile has a clock divider configurable by software;
- Each engine contains a pseudo random number generator to skip randomly clock steps used for cryptographic operations;

- You work at same clock frequency but you have a clock divider that creates decorrelation / background noise. Ex: one component works at 1GHz, another at 2 GHz, etc.

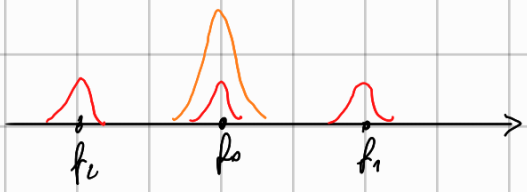
- Why does dividing frequency work? 1st. it increases resistance to jamming attacks: if someone is jamming 1GHz you can change frequency. Look down! ↓ •

③ To sort of randomize emission more, you can skip clock steps in a random way.

- HERE: 011011100

If you have a sequence of bits and are working at a specific frequency, you have a spectrum peak at that specific frequency. I acquire signal and transform in time domain. If you randomize the clock, you jump to different places with same energy.

- No random - Yes random



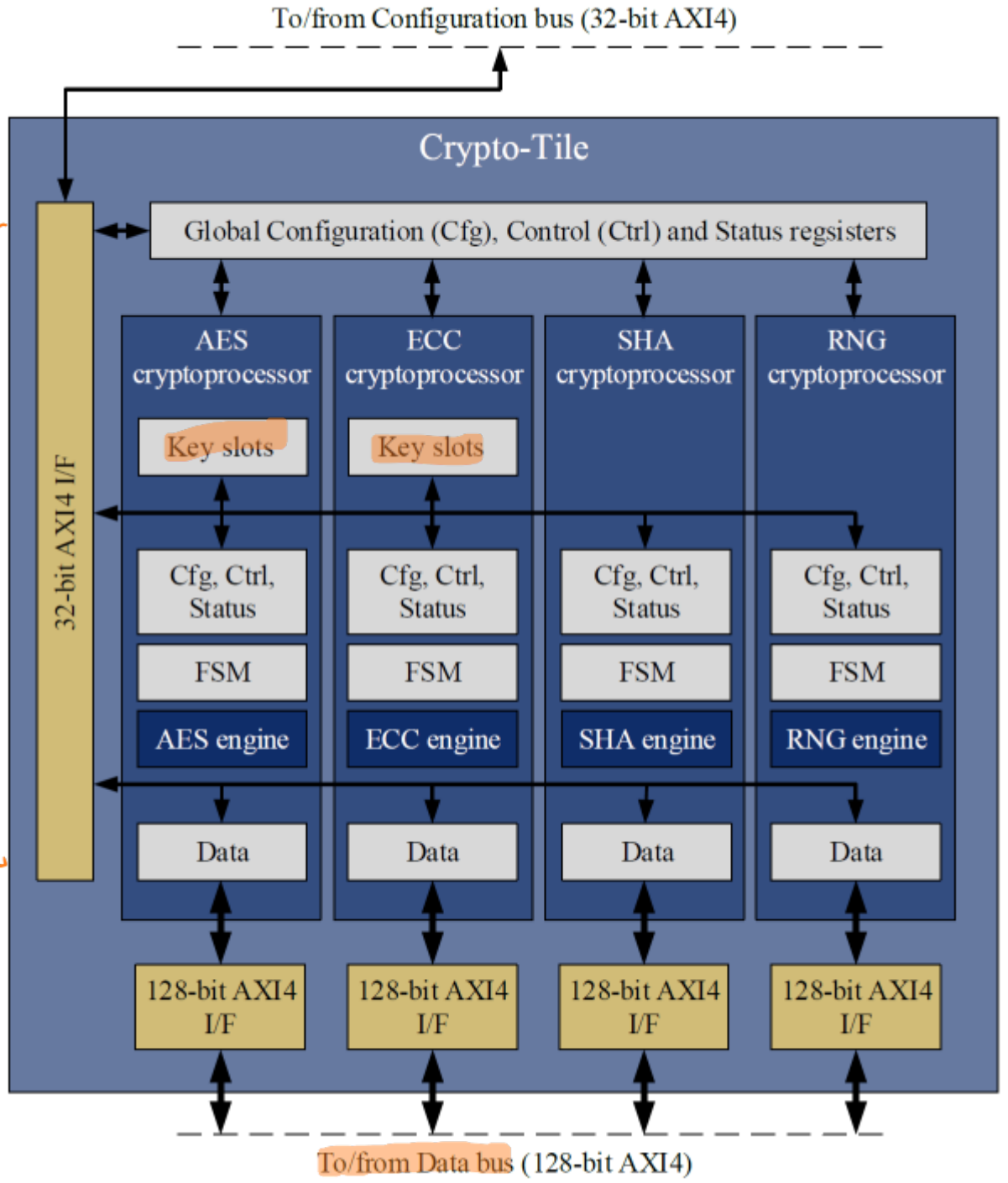
Plus, skipping a clock modifies the shape of the spectrum. Idea is don't give enough time to the attacker to understand what you are doing.

Note that decorrelating decreases efficiency.

Crypto Accelerator

Blue module: HW accelerators for security functions

- Different cryptoprocessors for the operations. Interesting to see that you can parallelize: if on same data you want to do AES and ECC you can also that.
- Note that AXI4 buses we use are kind of standards in the scene, by ARM. And ARM solutions are already compliant with ARM buses, so win win for ARM and producers.



EPI Crypto accelerator performance

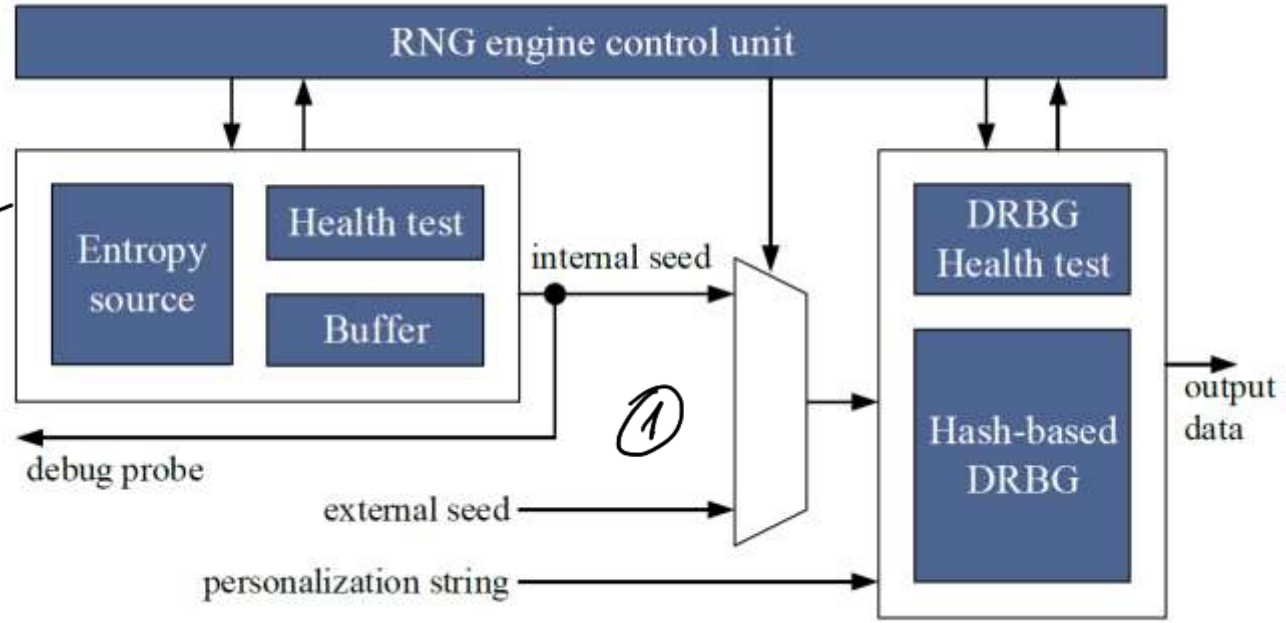
ECDSA CO		Execution time		Ratio
		SW-only (T_S)	HW accel. (T_H)	(T_S/T_H)
NIST P256, SHA2-224	Generation	168.46 ms	0.41 ms	410.88
	Verification	221.68 ms	0.68 ms	326.00
NIST P256, SHA2-256	Generation	167.96 ms	0.41 ms	409.66
	Verification	220.84 ms	0.68 ms	324.76
NIST P256, SHA2-384	Generation	169.67 ms	0.41 ms	413.83
	Verification	222.50 ms	0.68 ms	327.21
NIST P256, SHA2-512	Generation	167.23 ms	0.41 ms	407.88
	Verification	221.21 ms	0.68 ms	325.31
NIST P521, SHA2-224	Generation	694.89 ms	2.75 ms	252.69
	Verification	920.55 ms	4.56 ms	201.88
NIST P521, SHA2-256	Generation	698.12 ms	2.75 ms	253.86
	Verification	919.66 ms	4.56 ms	201.68
NIST P521, SHA2-384	Generation	692.68 ms	2.75 ms	251.88
	Verification	915.03 ms	4.56 ms	200.66
NIST P521, SHA2-512	Generation	695.41 ms	2.75 ms	252.88
	Verification	917.65 ms	4.57 ms	200.80

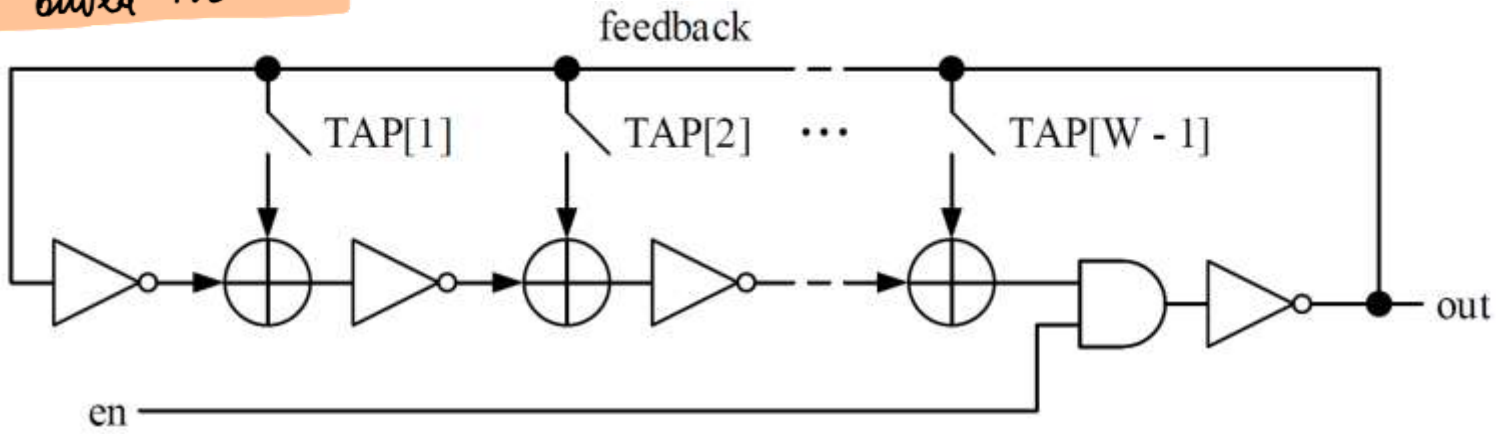
Tests: SW vs HW accelerators on RISC V

On-chip RNG

Implementation: true RNG + a buffer that collects the produced seed. Health test is to check whether or not what you produce is really random or not (finite state machine) according to standard by which we build the RNG.

But you can also work with an external seed provided ①. Of course this can be a potential source of attacks.





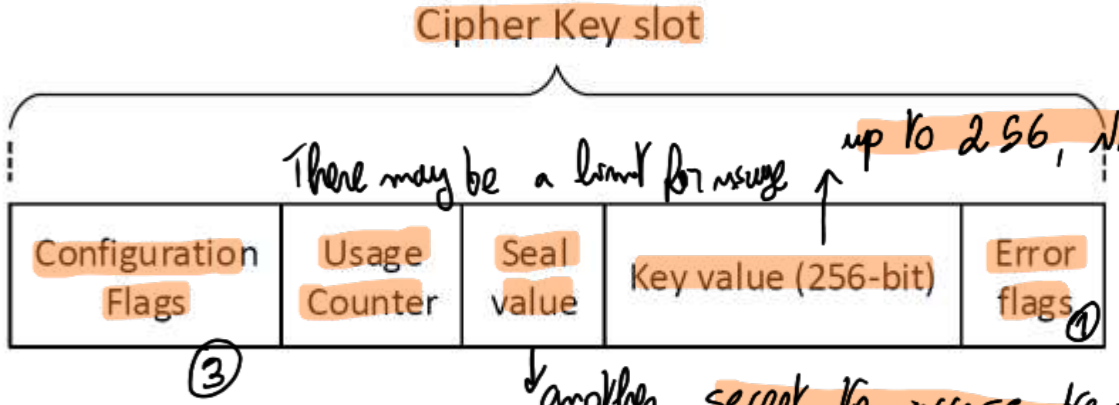
Schematic of a GaRO with enable and W *inverting elements. The inverting elements are* NOT *gate and the global enable signal of the circuit is* en*, while the* TAP[i] *elements are the coefficient of the polynomial associated to the feedback net: if* TAP[i] = *1, the corresponding connection is a short-circuit, hence the corresponding* XOR *gate is instantiated within the circuit, otherwise (*TAP[i] = *0) the corresponding connection is an open circuit and the associate* XOR *gate is not instantiated.*

One can force the TRNG to stop working so that it switches to external seed. The second part is a Deterministic Random Bit Generator that works on random seed for higher data rate.

NOTE: Basic function of entropy source is collection of ring oscillators in which you force instability. You have a sort of generation of glitches, a violation of synchronous clock rule, to make values commute randomly. The more elements you put, the more you increase process variation dependency.

Key management and access restriction

This related to key management policy.



(2) Apply the seal, if okay you can use it

(3) Extra rule: who can update, what is the policy on update etc.

(1) Signs if there are errors in case of non respecting rules (so SDMA can stop operation if needed)

Access restriction: In order to implement full isolation rule of each Security Domain, and to minimize complexity of Control System bus configuration the Crypto-Tile interface shall implement itself limited access in hardware (Figure 2.1, the Control System contains the Security Master Module, but also sensitive hardware modules of the EPI GPP as power and clock management):

- The Crypto-Tile interface access on MCU side shall be allowed if the access is issued from the Secure Core. Instead, accesses on DMA channels can be performed only by Secure DMA. Any other access issued shall be rejected with a bus error;

Debug and panic modes

Debug capabilities may be used to circumvent Crypto-Tile configuration and assets confidentiality:

- The debug capabilities of the Crypto-Tile can be fully disabled;
- Once one of the debug capabilities is disabled, the Crypto-Tile remains disabled until the next reset;
- The debug capabilities are disabled by default, they can be enabled by means of a dedicated register accessible by the software application during the boot sequence;
- The Crypto-Tile configuration relative to its internal debug capabilities shall be locked, writing in a dedicated register, until next hardware reset before allowing any cryptographic operation (to prevent malicious software re-opening debug capabilities to get access to sensitive data);

NO DEBUG! You have no reprogram capability!
You have to decide kudeoff

Panic mechanism: A panic mechanism is implemented in hardware to react immediately and with minimal assistance of software to manage events that may lead to a critical security breach: elimination of sensitive data (1)

This can also be used at end of lifecycle to prevent software copying.

You have to foresee diagnostic and debug mode, but this can be a security issue.

① Custom policy: in case a security breach is detected you take a particular action (running a routine, flushing devices).

Outline

- Architectures and components of HSM in automotive:

Analysis of HSM in commercial automotive MCUs (NXP, ST, Renesas)

Analysis of commercial IPs for HSM (by Rambus, Synopsys, …..)

- HW for Secure Cards

- Secure key storage/management and advanced secure features in EPI HSM

- HSM evolution roadmap

    Intrusion Detection System (IDS) support

    Post-quantum cryptography (PQC) and tracking of NIST standardization

- Q&A session

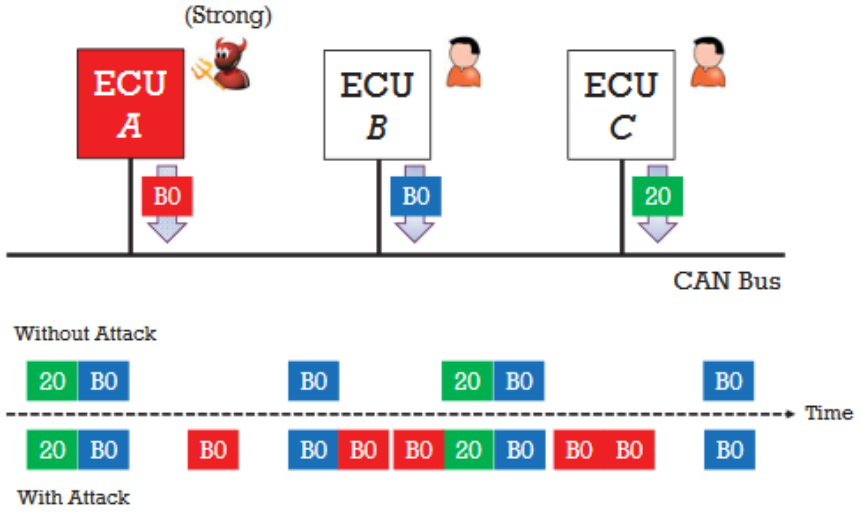
ECU IDS and fingerprinting

Intrusion Detection Systems (IDS) required by 21434. the IDS detects external attacks on ECUs and networks in the automobile, collects them and sends them to a backend at the vehicle manufacturer - also called a Security Operations Center (SOC). The OEM evaluates the data and then decides how to respond to the attempted attacks.

In the CAN protocol there isn't MAC address (Message Authentication Code) → needs of algorithms for anomaly detection and ECU fingeprinting analyzing message contents and physical characteristics (time and voltage)

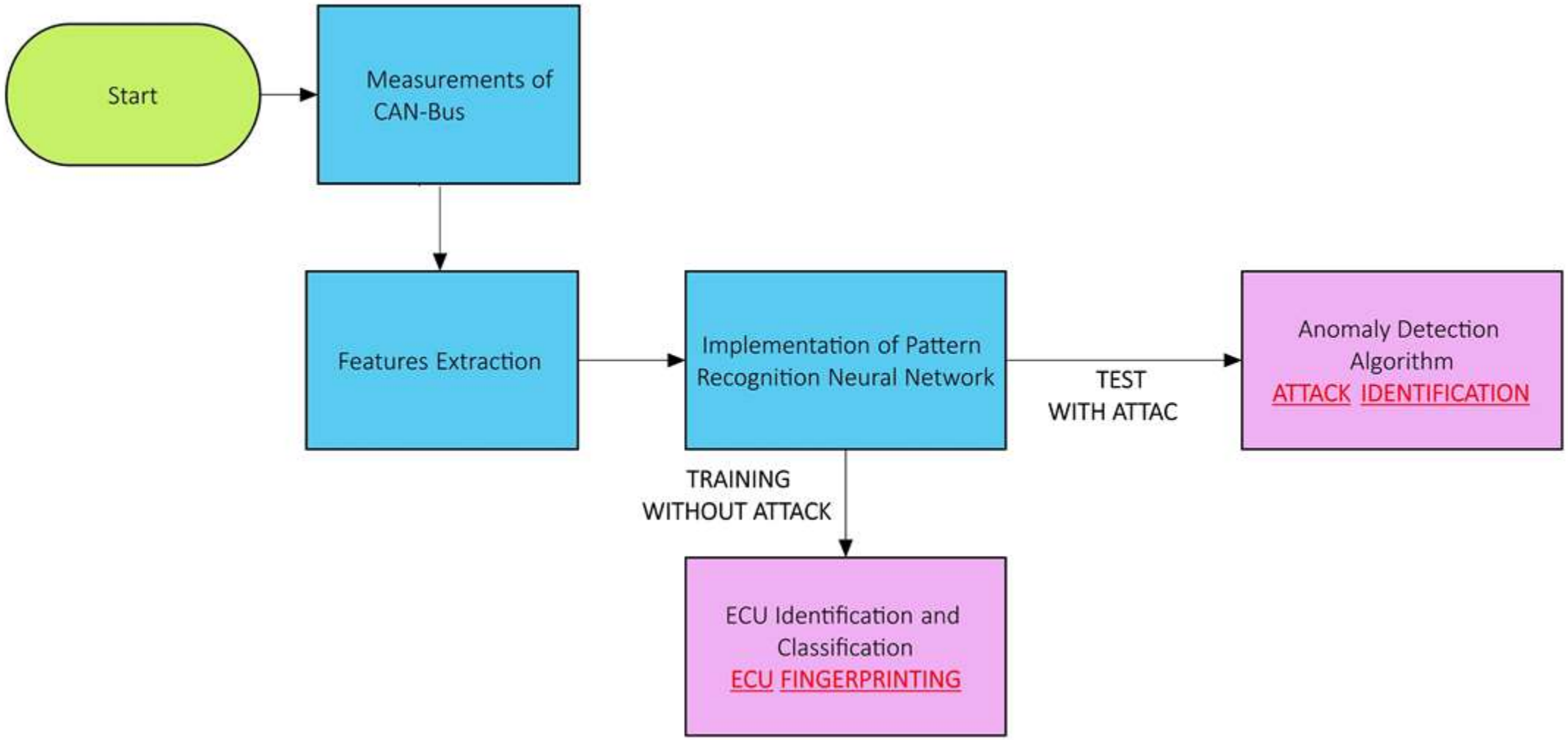
Most solutions in the market are now still SW based.

Evolution towards HW-SW solutions for
a real-time and energy efficient IDS



New trends: implement IDS at HW level. ISO 21434 requires IDS for compliancy. A lot of systems are distributed and interact with each other. You might have controls being sent on a bus, or even info shared wirelessly. A machine should be able to detect an intruder. Usually, on SW, you have some control rule to understand if you follow that rule with communication. They analyze traffic on Network and reason based on rules. Problem is you only check logical level, not what is happening at HW level. Another problem is that a lot of IDS have been created to run on PCs with OS ok. You here might have a network of microcontrollers and have to reason on that. Now we are moving to mixed HW-SW IDS that can also run on microcontrollers. Idea is also analyzing HW fingerprinting. Imagine someone took control of ECU A and they want to take control of ECU D by impersonating it, Idea is work with biometrics of the HW: this can be "reading puf code" for example. ECU A can create packet pretending to be D but it cannot send PUF of D, Plus, based on position, temperature etc., different components experiment different noise profiles (ECU A close to engine, higher temperature). Of course you need to elaborate profiles before to compare.

ECU IDS and fingerprinting



So you make measurement on bus, extract feature, implement pattern recognition and check with the logical ID (logical ID: "I'm ECU D", physical ID: "I'm ECU A")
So extend IDS to level 1: you also have HW part and measure hw related info (PUF, noise level).

Outline

- Architectures and components of HSM in automotive:

Analysis of HSM in commercial automotive MCUs (NXP, ST, Renesas)

Analysis of commercial IPs for HSM (by Rambus, Synopsys, …..)

- HW for Secure Cards
- Secure key storage/management and advanced secure features in EPI HSM
- HSM evolution roadmap

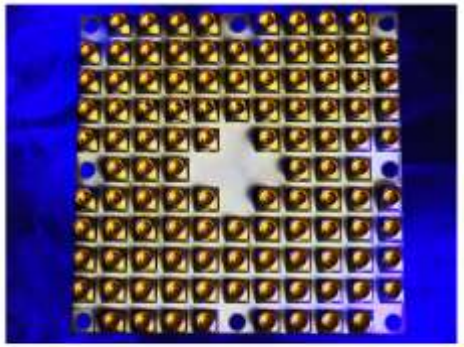
    Intrusion Detection System (IDS) support

    Post-quantum cryptography (PQC) and tracking of NIST standardization

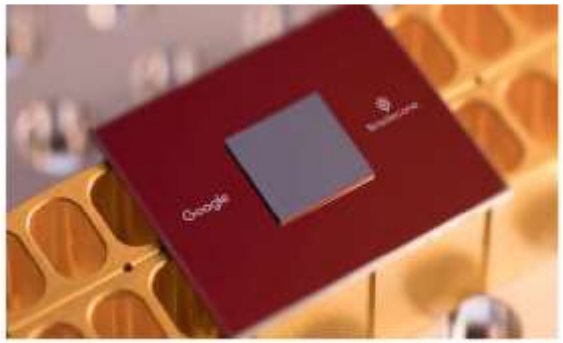
- Q&A session

What’s next

Increase in Data center computational power and appearing of quantum computers is reducing the security level of some cryptofunctions



Intel’s Tangle lake 49 Qubits



Google’s Bristlecone – 72 Qubits

https://www.intel.com/content/www/us/en/research/quantum-computing.html (commercial phase expected from 2025)

What is no more secure in state of art ?

Algorithm	Secure in Post-quantum Era?
RSA-1024, -2048, -4096	No No matter the extension
Elliptic Curve Crypto (ECC)-256, -521	No
Diffie-Hellman	No
ECC Diffie-Hellman	No
AES-128, -192	No

Security degradation in quantum era

Attack Platform	Symmetric Encryption			Asymmetric (Public-key) Encryption		
	Algorithm	Key Size	Security Level	Algorithm	Key Size	Security Level
Classic Computers	AES-128	128	128	RSA-2048	2,048	112
	AES-256	256	256	RSA-15360	15,360	256
Quantum Computers	AES-128	128	64	RSA-2048	2,048	25
	AES-256	256	128	RSA-15360	15,360	31

Grover's algorithm

Shor's algorithm

AES decrease is by a factor of 2 in security level

Solution? Computer-based quantum-proof cryptography

Computer-based quantum-proof cryptography

Increasing the key size of some techniques

Proposing new hard mathematic problems,

supported by new algorithms for other techniques

In all cases HW acceleration still needed

AES 256 still secure enough

SHA2/SHA-3 up to 512 still secure enough

From PKE you redesign from scratch

Symmetric Cryptography:

- **Issue:** Grover's algorithm [Gro'96] is expected to break AES128 and SHA256
- **Mitigation:** Increase keys/parameters of algorithms (Ex: AES128 → AES256)

Post-Quantum Cryptography – Challenges and Opportunities

Rafael Misoczki, PhD

Intel Labs – Security & Privacy Research Lab

Computer-based quantum-proof cryptography

Increasing the key size of some techniques

Proposing new hard mathematic problems, supported by new algorithms for other techniques

In all cases HW acceleration still needed

Open research on public key cryptography

for encryption, key generation/management and signature-verification

to replace RSA and ECC-based families like ECSIE, ECDH, ECDSA

- **Issue**: Shor's algorithm [Shor'94] is expected to completely break RSA and ECC
- **Mitigation**: Replace all digital signature, key exchange and asymmetric encryption algorithms

Standardization efforts and timing

Standardization timing for state-of-art techniques (AES, ECC, SHA)

- ECC: proposed by mid-1980's + 2 decades to gain some adoption
- AES: 4 years of competition + more than a decade to gain wide adoption
- SHA-3: 5 years of competition + 6 years since publication. No wide adoption (yet?)

NIST and ISO/IEC standardization efforts on Post-quantum Cryptography

↳ are already working on standardization

NIST:

- **Scope**: Key Encapsulation, Encryption and Stateless Signatures
- **Candidates**: 82 submissions
- **Status**: between Round-1 and Round-2
- **Timeline**: expected by 2022-2024

ISO/IEC (JTC1 SC27 WG2):

- **Scope**: Key Encapsulation, Encryption and Signatures
- **Status**: Early stage
- **Timeline**: Under discussion

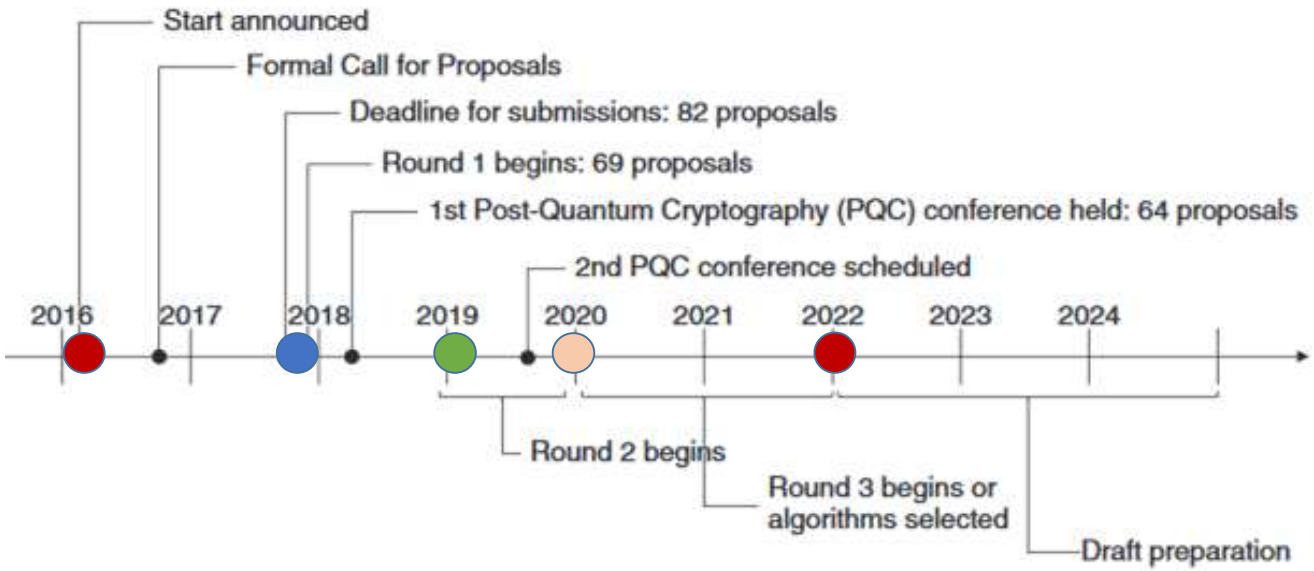
NIST standardization effort

works on different rounds

- NIST (in US) is working towards PQC standardization alternatives
- https://csrc.nist.gov/projects/post-quantum-cryptography
- Already on-going: round1 and round 2 done, round3 in place
- Round1: Jan 2017-Dec2018: 69 submissions, in 4 main domains
  Lattix-based, code-based, Hash-based, others
- Round2: NIST Jan 2019 published round 2 selected candidates

Round 3 timing: started end 2021

Standard finalized: 2022/2024



NIST Second round

26 candidates of which:

9 specific for **digital signature** (CRYSTALS-Dilithium, Falcon, GeMSS, LUOV, MQDSS, Picnic, qTESLA, Rainbow, SPHINCS+)

17 specific to **Public-key Encryption** and **Key-establishment Algorithms**

Bike, Classic Mcliece, CRYSTALS-Kyber, FrodoKEM, *HQC*, LAC, *Ledacrypt (LDPC)*, NewHope, NTRU, NTRUprime, *NTS-KEM, ROLLO,* Round5, *RQC,* SABER, SIKE, Three bears)

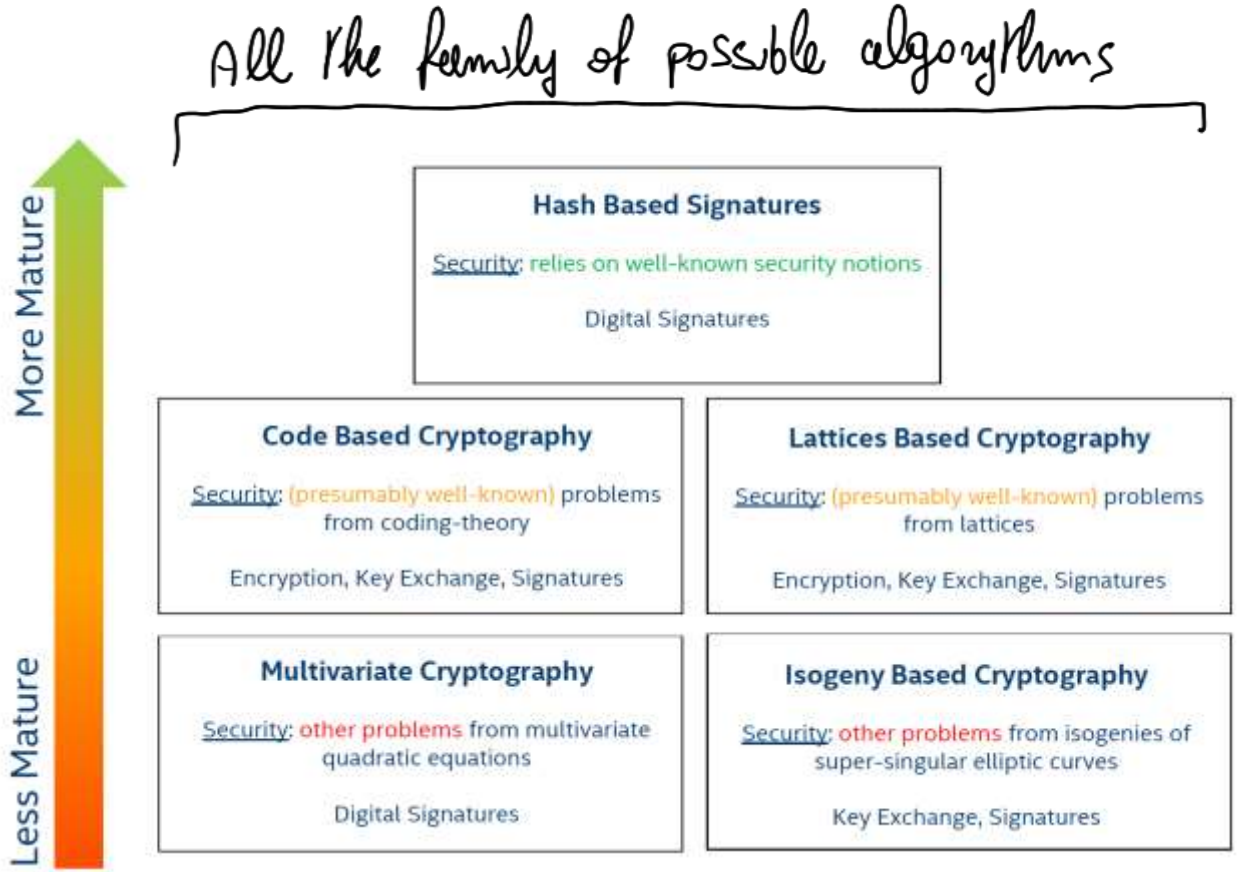
You reuse techniques in signal theory for security.

12 are **lattice based** (for both hash and encryption and key-establishment),

7 are code based (i.e. applying channel code techniques for security),

7 are belonging to other types (mainly for hash-based signature)

Candidate classes for PQC



Family	Signature scheme	Key encapsulation method or key exchange scheme
Hash	Sphincs++	
Multivariate	GeMSS LUOV MQDSS Rainbow	
Code		BIKE Classic McEliece HQC LEDAcrypt NTS-KEM ROLLO RQC
Isogeny		SIKE
Lattice	CRYSTALS-DILITHIUM qTESLA FALCCON	CRYSTALS-KYBER FrodoKEM LAC NewHope NTRU PRIME Round5 SABER Three Bears
Others		Picnic

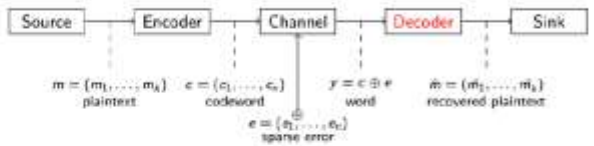
Table 1. NIST candidates of 2nd round.

Each of the classes is based on a different mathematical problem that is hard to be solved by both modern computers and quantum computers

Code-base cryptography – McEliece (1/2)

Coding-Theory

Techniques to efficiently transmit data through channels subject to noise.



Desirable features:

- Error detection.
- Error correction.

- Since [Pra62], the cost of correcting errors in a random code has remained essentially the same:

$$WF_A(n,k,t) = 2^{ct(1+o(1))}$$

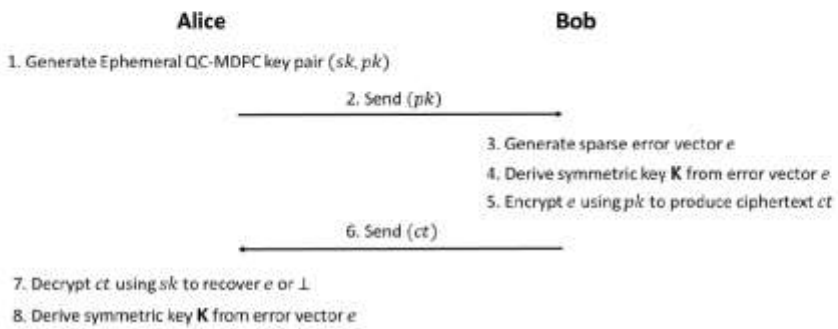
McEliece-like Schemes

- **McEliece Cryptosystem:** besides decoding, security is also based on hardness of distinguishing public key from random. It can be instantiated with different code families.
- **Challenges:**
  - **Efficiency:** unstructured codes suffer from huge public keys. Efficiency of key-generation, encryption and decryption strongly depend on the selected code family
  - **Indistinguishability**: Strongly depends on the code family. Quasi-cyclic codes do not seem to offer great advantage to adversaries.

Code-based cryptography – Bike (2/2)

Overview

- Submission to NIST "Competition" on PQC
- Joint work between Industry & Academic professionals
- **Security**:
  - Based on two well-known coding-theory problems: Codeword-Finding & Syndrome Decoding Problem
  - Ephemeral Keys: Forward Secrecy



Performance

- Message Size:
  - From ~1kB up to ~8kB
  - It varies depending on variant & security level
- Latency:
  - From 100's of thousands to a few million cycles
  - It varies depending on variant & security level

http://bikesuite.org

Key-encapsulation & encryption: code-based vs Lattix-based

Sr. No.	Key Establishment/Encapsulation	
	Lattice-based/R-LWE	Code-based
1	NewHope (R-LWE)	BIKE (MDPC)
2	NTRU (R-lattice)	NTS-KEM (Binary Goppa)
3	FrodoKEM (R-LWE)	LEDA (LDPC)
4	CRYSTALS (R-LWE)	ROLLO (LRPC) (LAKE & LOCKER)
5	SABER (Mod-LWR)	
6	Three Bears (Mod-LWR)	

Sr. No.	Public-Key Encryption	
	Lattice-based/R-LWE	Code-based
1	NTRU Prime (R-lattice)	Classic McEliece (Binary Goppa)
2	NTRU (R-lattice)	HQC (BCH & Cyclic)
3	LAC (R-LWE)	RQC (Cyclic)
4	SABER (Mod-LWR)	LEDA (LDPC)
5	Round5 (R-LWR)	ROLLO (LAKE & LOCKER) (LRPC)

SIKE is an isogeny-based key encapsulation

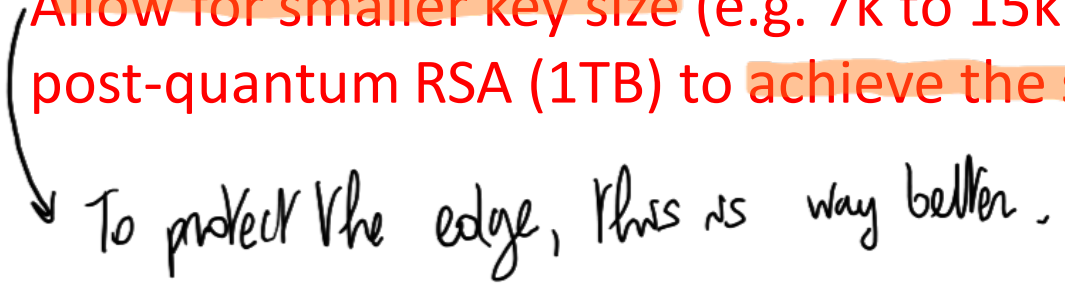
Digital signatures: Lattix-based vs others

Sr. No.	Digital Signature		
	Lattice-based/R-LWE	Multivariate-based	Others
1	FALCON (NTRU R-lattice)	GeMSS	Picnic
2	qTESLA (R-LWE)	MQDSS	SPHINCS
3	CRYSTALS (R-LWE)	LUOV	
4		Rainbow	

Lattix-based as most promising PQC

LWE (Learning With Error) and particularly RLWE (Ring Learning With Error) lattix-codes are very promising since:

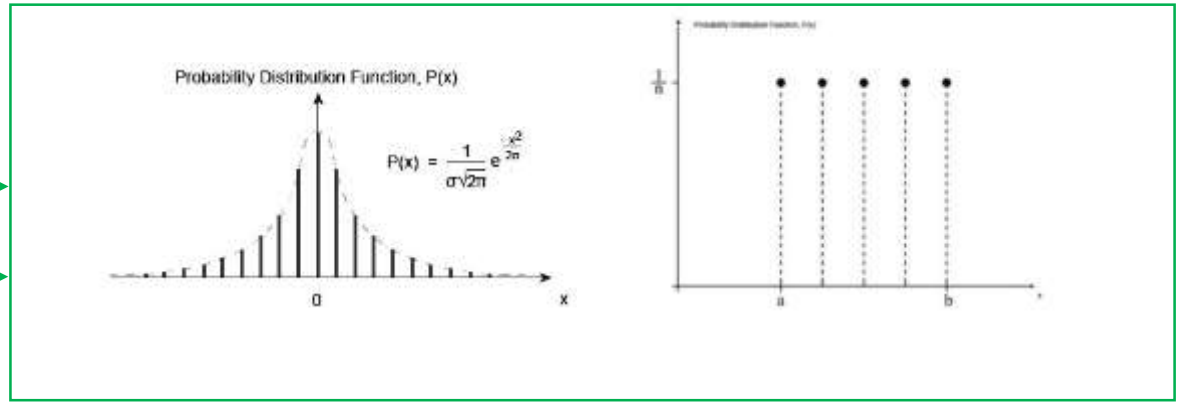
- Are covering all problems of encryption, key encapsulation and signature
- Can extend to homomorphic encryption
- Allow for smaller key size (e.g. 7k to 15k bits) vs code based crypto (e.g. 1 MB) and post-quantum RSA (1TB) to achieve the same security strength



RLWE (Ring Learning With Errors)

Algorithm 1 NewHope-CPA-PKE Key Generation

1: **function** NewHope-CPA-PKE.Gen()
2: $seed \xleftarrow{\$} \{0,\ldots,255\}^{32}$
3: $z \leftarrow \mathsf{SHAKE256}(64, seed)$
4: $publicseed \leftarrow z[0{:}31]$
5: $noiseseed \leftarrow z[32{:}63]$
6: $\hat{a} \leftarrow \mathsf{GenA}(publicseed)$
7: $s \leftarrow \mathsf{PolyBitRev}(\mathsf{Sample}(noiseseed, 0))$
8: $\hat{s} \leftarrow \mathsf{NTT}(s)$
9: $e \leftarrow \mathsf{PolyBitRev}(\mathsf{Sample}(noiseseed, 1))$
10: $\hat{e} \leftarrow \mathsf{NTT}(e)$
11: $\hat{b} \leftarrow \hat{a} \circ \hat{s} + \hat{e}$
12: **return** $(pk = \mathsf{EncodePK}(\hat{b}, publicseed), sk = \mathsf{EncodePolynomial}(\hat{s}))$



$\mathbb{Z}_{13}[x]/\langle x^4+1\rangle$

	$4 + 1x + 11x^2 + 10x^3$	a	random
×	$6 + 9x + 11x^2 + 11x^3$	s	secret
+	$0 - 1x + 1x^2 + 1x^3$	e	small noise
=	$10 + 5x + 10x^2 + 7x^3$	b = a s + e	

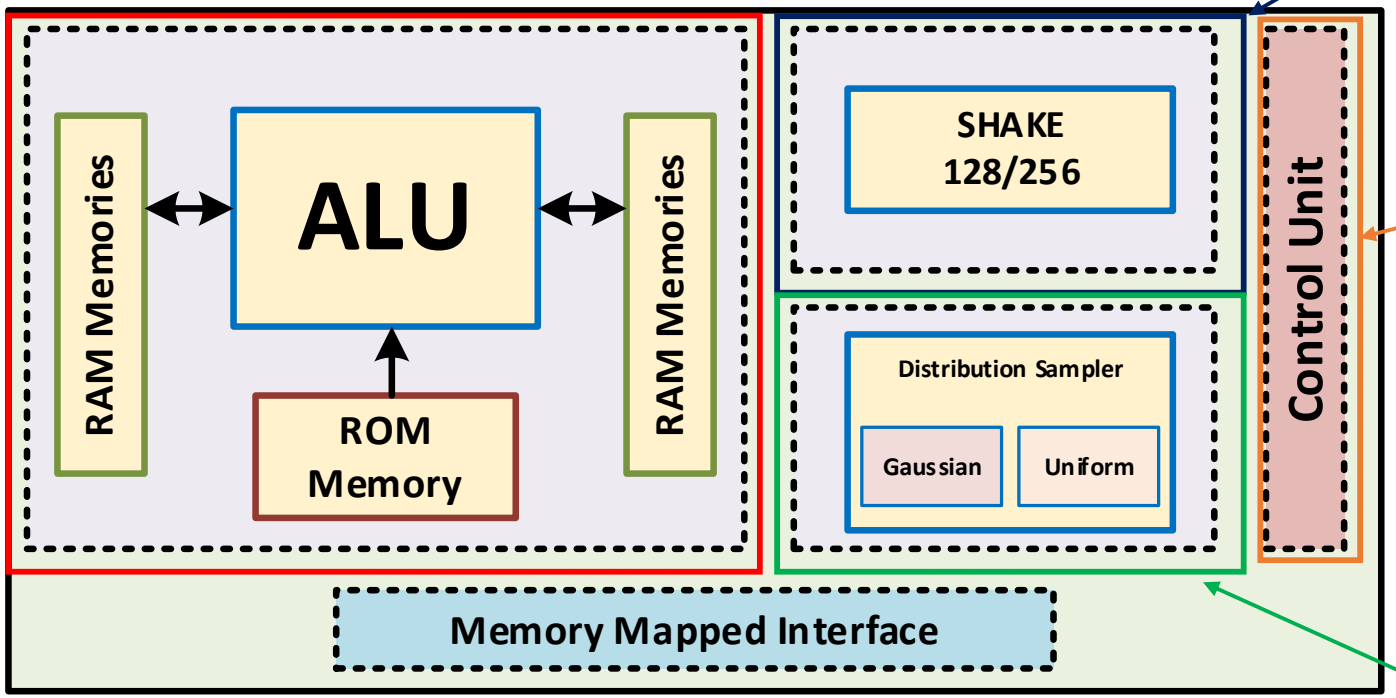
RLWE HW accelerator

HASH Core

SHAKE 128/256

Arithmetic Logic Unit and Memories

- Polynomial multiplication (NTT)
- Polynomial Add/Sub
- Coefficient-wise multiplication



Control Unit

Customizable for each algorithm

Noise Sampler

High industrial interest on RLWE (1/3)

Work by Continental/Infineon on Post-Quantum Secure Architectures for Automotive Hardware Secure Modules

Result analysis on Code-based PQC

- One main issue in a McEliece cryptosystem is the large size of the public key
- Some research efforts have been focused on reducing the size of the keys by incorporating structures into the code, e.g., QC-MDPC codes. However, these structured codes can potentially lead to attacks
- Code based encryption is only suited for message encryption or key encapsulation
- The big disadvantage of code-based schemes for embedded solutions or solutions with constraints on memory footprint is the large key size of the key pair, which is several megabytes. So Code-base schemes useful for server rather then IoT or edge devices

Result analysis on Isogeny

- underlying arithmetic is similar to those in classical ECC schemes since the scheme is constructed based on the isogenies between elliptic curves.
- Due to the novelty of isogeny-based problems, there is not yet enough confidence in these schemes

- There are already companies implementing solutions for post quantum cryptography
  But if you don't have a winning algorithm, it's a big risk to implement them

High industrial interest on RLWE (2/3)

Result analysis on PQC HASH schemes

- Hash-based Schemes use hash algorithms as an underlying primitive and tree structures to establish so-called few time signatures.
- The security of these schemes is based on the mathematical hardness of finding a collision or a pre-image of a hash value.
- This family of PQC schemes only supports signature operations. The advantage of this approach is that such schemes have been studied for a long time.
- Hence, they are very well understood, leading to a very high-security maturity level.
- The disadvantage of these schemes is that a private-public key pair can only be used for a certain number of signature generations. If the number exceeds, a new private-public key pair needs to be generated

High industrial interest on RLWE (3/3)

Work by Continental/Infineon on Post-Quantum Secure Architectures for Automotive Hardware Secure Modules

https://www.infineon.com/cms/en/product/promopages/post-quantum-cryptography/

Result analysis on Lattice-based cryptography

- Arguably the most popular among different PQC families.
- Its security is based on different types of hard problems defined on a high-dimension lattice, e.g., “learning with errors”
- have much smaller keys and better performance than

- Proposed HSM: AES-256 + RNG + Hash (SHA3-512/Shake-256) + **Gaussian sampler&NTT (number theoretic transform) for Lattix computation**

HW acceleration still needed for RLWE

- NewHope512 via SW on a **RISC-V** 64b: U54 (sifive,u54-mc); 4 cores at 1.4 GHz;

hifiveunleashedriscv, key pair generation + encapsulation + decapsulation need 675215 cycles/key pair generation + 1071604 cycles/encapsulation + 1176217 cycles/decapsulation=2923026 cycles/op → at 1 GHz only 300 op/s

- NewHope1024 via SW needs more than double cycles than NewHope512

https://bench.cr.yp.to/results-kem.html

- NewHope512 via SW on **ARM A53** (410fd034); 4 cores at 1.2 GHz;

NewHope512cca needs 773872 cycles per key pair generation + 1230680 cycles per encapsulation + 1295933 cycles per decapsulation =

3300485 cycles/op (+13% cycles vs. a sifive RISC-V 64b 4 cores)

HW acceleration still needed for RLWE

- NewHope512 via SW on a **Intel Atom D2500; 2 core at 1.866 GHz**

key pair generation + encapsulation + decapsulation need 489475 cycles/key pair generation + 796908 cycles/encapsulation + 983080 cycles/decapsulation=2269463 cycles/op → at 1 GHz only 440 op/s

- NewHope1024 via SW needs on a **Intel Atom D2500; 2 core at 1.866 GHz**

key pair generation + encapsulation + decapsulation need 1047536 cycles/key pair generation + 1685733 cycles/encapsulation + 2099300 cycles/decapsulation=4843569 cycles/op → at 1GHz only 206 op/s

https://bench.cr.yp.to/results-kem.html

TLS using PQC (We also have development of TLS based on PQC) Work has already started.

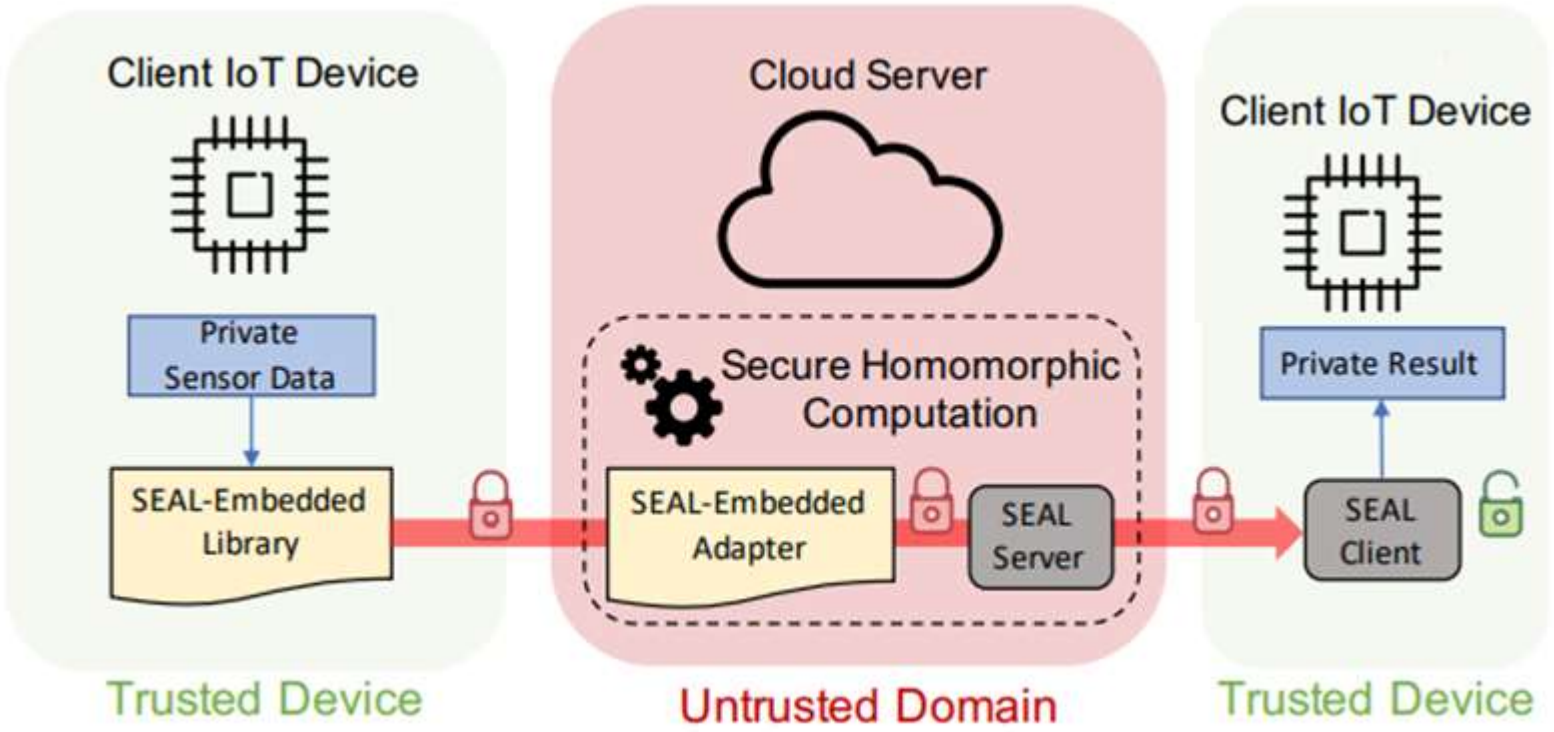
Transport Layer Security (TLS) Integrated with PQ signature algorithms into TLS 1.3 at the state of art

Evaluated the TLS handshake latency observed by a client along with the throughput of a PQ authenticated server by considering realistic network conditions

The signature and certificate chain size impact the total handshake time, especially with relatively fast signing and verification primitives.

Results show that the PQ algorithms with the best performance for time-sensitive applications are Dilithium and Falcon

Homomorphic encryption using Microsoft SEAL library based on RLWE HW acceleration



Embedded devices only have to encrypt data, homomorphic operations are performed by the server (the server can access only encrypted data)
The results of server computation on encrypted data is the same if applied in plaintext data

Post-quantum Blockchains

- Primitives for encryption/decryption (e.g. AES, ECC,…) and hashing (e.g. SHA,..) are the core of **Blockchains** used for secure transactions and traceability in e-commerce, e-voting, smart health, smart factories and where IoT edge devices has to be interact with central servers/cloud
- Blockchain acts as a distributed ledger based on a chain of data blocks linked by **hashes** that allow for sharing information among peers that do not necessarily trust each other.
- Blockchain users interact securely with the blockchain by leveraging **public-key/asymmetric cryptography**, which is essential for the authentication of transactions
- The same concepts discussed for state-of-art HSM, and for next generation (i.e. post-quantum) signature, key generation and encapsulation cryptography, apply to Blockchain
- Research on post-quantum Blockchain started, based on the same new techniques analyzed before (Lattix, Code-based,…)
- Lattix-based techniques are seen as a promising solution for IoT devices that have to interact within a next-generation Blockchain scheme

Ref. T.M. Fernandez-Ceramas et al., Towards Post-Quantum Blockchain: A Review on Blockchain Cryptography Resistant to Quantum Computing, IEEE Access, vol. 8, pp. 21091-21116, 2020

Conclusions

- In quantum-computing era AES and SHA can be saved increasing key size
- In quantum-computing era RSA and ECC-based completely broken
- New public key crypto techniques under design and for key encapsulation, encryption and signature
- Lattice-based PQC is the most promising, particularly for edge devices and IoT
- HW acceleration needed
- It is based on 4 HW blocks: RNG, SHA-3/Shake, Gaussian sampler, NTT
- New block challenge for designs are NTT and Gaussian sampler

Conclusions & other application domains

Blockchains

- The same concepts discussed for state-of-art HSM, and for next generation (i.e. post-quantum) signature, key generation and encapsulation cryptography, apply to Blockchain
- Research on post-quantum Blockchain started, based on the same new techniques analyzed before (Lattix, Code-based,…)
- Lattix-based techniques are seen as a promising solution for IoT devices that have to interact within a next-generation Blockchain scheme

Softerrors (Reliability)

- The concepts discussed for verification of data integrity and authentication can be used also to detect the presence of soft errors (e.g. bit flips due to SEEs or interference rather than cyber attacks)

References

W. Wang, M. Stottinger, Post-Quantum Secure Architectures for Automotive Hardware Secure Modules, https://eprint.iacr.org/2020/026.pdf

M. Dehm, S. Reith, M. Stottinger. Post-Quantum Cryptography: A Moving Target. 6th International VDI Conference - Cyber Security for Vehicles

D. Sikeridis, P. Kampanakisy, M. Devetsikiotis, Post-Quantum Authentication in TLS 1.3: A Performance Study, https://eprint.iacr.org/2020/071.pdf

R. Agrawal, Bu Lake, A. Ehret, M. Kinsy, Open-Source FPGA Implementation of Post-Quantum Cryptographic Hardware Primitives, 2019 29th Int. Conf. on Field Programmable Logic and Applications (FPL)

R. Misoczki, Post-Quantum Cryptography –Challenges and Opportunities, https://crypto.iacr.org/2018/affevents/qsci/medias/Rafael_Misoczki.pdf

Intel® QuickAssist Technology (Intel® QAT) and OpenSSL-1.1.0: Performance White Paper

R. Soja, Automotive Security: From Standards to Implementation, NXP white paper AUTOSECURITYWP REV 1

ST AN4240, https://www.st.com/resource/en/application_note/dm00075575-introduction-to-the-cryptographic-service-engine-cse-module-for-spc56ecxx-and-spc564bxx-devices-stmicroelectronics.pdf

https://www.evita-project.org/Publications/HRSW09.pdf

https://csrc.nist.gov/projects/post-quantum-cryptography

White paper INTEL Trusted Execution Technology, https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/trusted-execution-technology-security-paper.pdf

T.M. Fernandez-Ceramas et al., Towards Post-Quantum Blockchain: A Review on Blockchain Cryptography Resistant to Quantum Computing, IEEE Access, vol. 8, pp. 21091-21116, 2020. doi: 10.1109/ACCESS.2020.2968985

J- M. Mogollon et al., Real Time SEU Detection and Diagnosis for Safety or Mission-Critical ICs Using HASH Library-Based Fault Dictionaries, IEEE RADECS 2011

C.H. Gebotys, Reliable Testable Secure Systems, Chapter 10 in Springer Security in Embedded Devices, pp 263-289, 2009

H. Cam et al., A COMBINED ENCRYPTION AND ERROR CORRECTION SCHEME:AES-TURBO, JOURNAL OF ELECTRICAL & ELECTRONICS ENGINEERING 2009

Part B: Q&A session

HSM (Hardware Security Module)

Prof. Sergio Saponara,
Dip. Ingegneria della Informazione, Università di Pisa
sergio.saponara@unipi.it
+39 3468790937