April 23, 2021

Post-Quantum Cryptography Hardware

Monolithic Implementations vs. Hardware-Software Co-Design

(Séminaire de cryptographie, Université de Rennes 1)

Dr. Markku-Juhani O. Saarinen
mjos@pqshield.com

Talk Outline

I. **Introduction: Post-Quantum Transition**

II. Monolithic Public-Key Crypto Hardware

III. RISC-V and Hardware-Software Codesign for Post-Quantum

IV. Conclusions: Commercial Post-Quantum Crypto IP

(Recap) Post-Quantum Cryptography Transition

Shor 1994

Algorithms for Quantum Computation:
Discrete Logarithms and Factoring

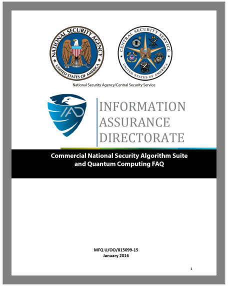
Peter W. Shor
AT&T Bell Labs
Room 2D-149
600 Mountain Ave.
Murray Hill, NJ 07974, USA

Regev 2005

On Lattices, Learning with Errors,
Random Linear Codes, and Cryptography

Oded Regev
Department of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel

NSA 2016



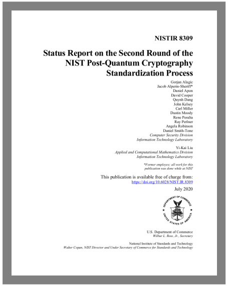
National Security Agency/Central Security Service

INFORMATION
ASSURANCE
DIRECTORATE

Commercial National Security Algorithm Suite
and Quantum Computing FAQ

MFQ U/OO/815099-15
January 2016

NIST 2020



NISTIR 8309

Status Report on the Second Round of the
NIST Post-Quantum Cryptography
Standardization Process

July 2020

- ≈ **25 years ago:** Vulnerabilities in RSA and (ECC) Discrete Logarithm recognized.
- ≈ **15-10 years ago:** Foundations of Post-Quantum Cryptography algorithms.
- ≈ **5 years ago:** Serious post-quantum transition & standardization begins.
- ≈ **2022-2024:** Final standards selected: Engineering, security industry adoption.

NIST Post-Quantum Standardization: Finalists

Pre-Quantum	Post-Quantum
 **(RSA, ELLIPTIC CURVE)** Signatures, Certificates	 **DILITHIUM / FALCON** Signatures, Certificates
 **(ELLIPTIC CURVE)** Key Establishment	 **KYBER / NTRU / SABER** Key Establishment

- → **Lattice schemes** are expected to be prominent (hybrid during transition). Integration path exists into IETF protocols (TLS, IPSec, SSH) and X.509 PKI.
- → **Rainbow** (signature) and **Classic McEliece** (key establishment) use public keys of 100s of kilobytes. These “long term” keys require protocol re-engineering.

Government and Military: National Security Systems (NSS)

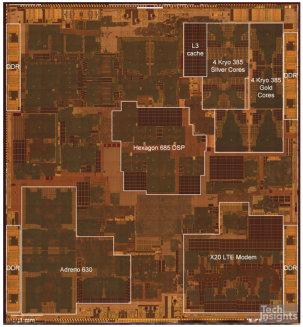
- U.S. **NSS:** Commercial National Security Algorithm Suite (CNSA) and Government-Off-The-Shelf (GOTS) will be updated after NIST Selection.

  October 23, 2020:

  "NSA CSD expects that a NIST-candidate lattice-based signature and a NIST-candidate lattice-based key encapsulation mechanism will be approved for NSS."

- Stateful hash-based signature algorithms (already approved SP 800-208) **XMSS** and **LMS** have found use cases in "root of trust" applications such as firmware updates. (SHAKE acceleration also benefits lattice algorithms.)
- German **BSI** recommends **Classic McEliece** and **FrodoKEM** at Level 3 and 5.
- Note that both BSI and NSS systems are usually certified to Common Criteria protection profiles (PP) – which may include FIPS 140-3 CAVP/CMVP tests.

Hardware: Cryptography is Everywhere



Can you identify all subsystems, coprocessors, and controllers in a modern Mobile Device SoC that **use cryptography** *(of any kind) ?*

Storage | Network | Radio | SPI, I2C

Signal Processing | Video Codecs | Audio

Display Engine | Graphics Processing Unit

Motion | Machine Learning | Neural Engine

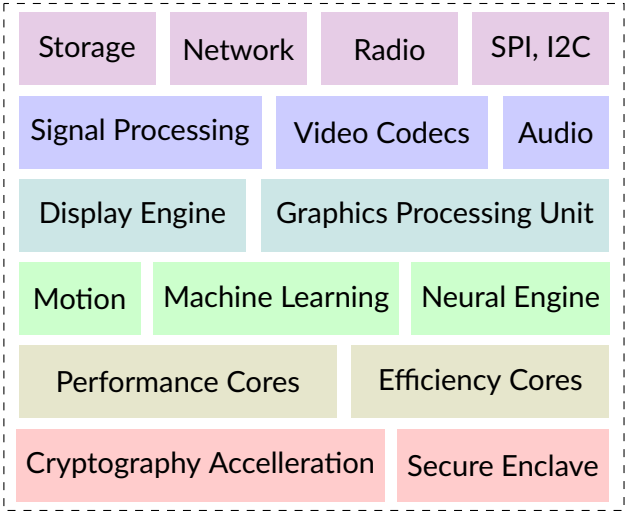
Performance Cores | Efficiency Cores

Cryptography Accelleration | Secure Enclave

Transition: Identify Systems that Need a Rethink

Which are impacted by new Post-Quantum Standards?

1. Secure Enclave / Key Store ?
2. TPM, SIM, other chips ?
3. Processor Instruction Set ?
4. Network interfaces ?
5. 5G, WiFi, and BLE (Radio) ?
6. Storage encryption (disk) ?
7. Firmware update / boot ?
8. User authentication tokens ?
9. Future applications ?



Talk Outline

I Introduction: Post-Quantum Transition

II **Monolithic Public-Key Crypto Hardware**

III RISC-V and Hardware-Software Codesign for Post-Quantum

IV Conclusions: Commercial Post-Quantum Crypto IP

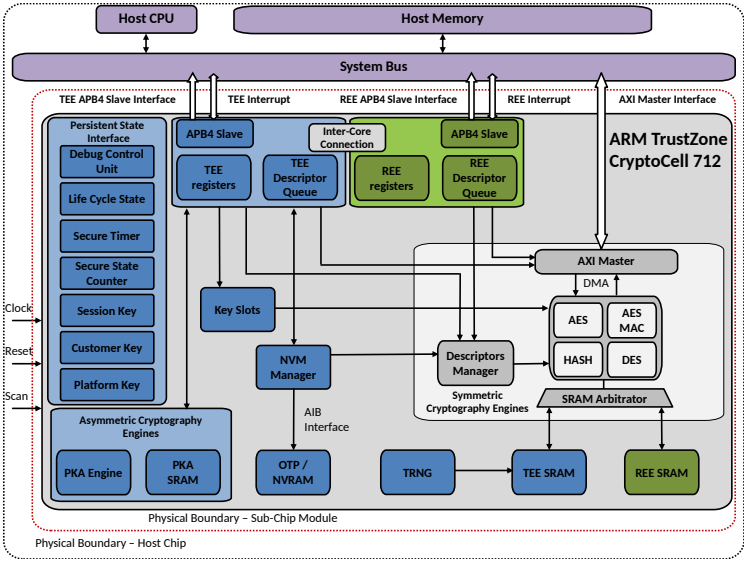
Some Semiconductor Industry Concepts and Acronyms

- **ASIC** (*Application-Specific Integrated Circuit*) a custom chip. Is likely to be a..
- **SoC** (*System-on-Chip*) nowadays; a single chip that has almost all the logic (CPU, Flash, RAM, Accelerators, Interfaces, etc.) that an application needs.
- **FPGA** (*Field-Programmable Gate Array*) is configured with a *bitstream* file to function like a large digital circuit. Can be on the same SoC with CPU cores.
- **Netlist** is a list of gates/components and their connections (like a graph).
- **RTL** (*Register-Transfer Level*) is an abstraction that circuit designers control in..
- **HDL** (*Hardware Description Language*) such as *Verilog* (Industry, Silicon Valley) or *VHDL* (Academics, Europe). VHDL and Verilog are similar apart from syntax.
- **Soft IP** such as a Crypto Accelerator or a RF Unit or CPU is licensed as Verilog source or a Netlist for inclusion in a design. Synthesis can be on FPGA or ASIC.

  (A typical car or electronic device has hundreds of separate, licensed IP cores.)

Current Example: FIPS 140-2 ARM CryptoCell 712

- IP Core from ARM.
- Licensed essentially as Verilog source.
- Memory mapped to TEE & REE Regs.
- Documentation helps with FIPS/CC.
- FIPS 140-2 (..2020) $\neq$ 140-3 (2021..).
- On-chip “physical” security boundaries.
- Note “**PKA Engine**” (lower left corner).



“Asymmetric Cryptography” used to mean Big Integers

- CryptoCell PKA Engine was a large integer arithmetic logic unit, nothing else: *“Supports integers in the range of 128 bits and 4K bits in size (in steps of 32 bits).”*
- Useful for RSA, DL, and elliptic curves over prime fields. The core functions (exponentiation and scalar multiplication) are algorithmically very simple.
- Apart from Isogeny (SIKE, SIDH) cryptography, big integer arithmetic is **not** directly useful for Post-Quantum Cryptography (none of the finalists).
- No specific side-channel security *claims* (even though mitigations do exist).

PQC Architectural Change 1

Big-Integer Arithmetic Units are not directly useful for Post-Quantum Crypto.

Post-Quantum Arithmetic Compute

PQC Architectural Change 2

PQC algorithms have many **more distinct steps** than core RSA or EC crypto. Need transformations, samplers, and symmetric components *"mid-flight"*.

- **(R)LWE** and **NTRU**: Polynomial rings / vectors over $\mathbb{Z}_q$, typically $q \leq 2^{16}$.
- **Vector ops**: NTT butterfly arrangement/shuffle, polynomial inversion.
- **FIPS 202.** A lot of SHAKE Extendable-Output Function (XOF) output.
- **Samplers**: Binomial, Uniform, Gaussian, Rejection, "constant weight."
- **McEliece**, **Rainbow**: Vec/mat or poly over small binary fields $\mathrm{GF}(2^n), n < 16$.

Example: Kyber or Saber on a Cortex M4 may be 50% cycles SHAKE, 40 % ring arithmetic, 10 % rest. Speeding up a single thing can only yield $2\times$ speedup.

Other Simultaneous Changes; mostly Modernization

- **Non-invasive security.** Timing side channels must be be gone (remote exploit). Many common applications (e.g. Smartcards) need DPA resistance too.
- **Classical TRNGs are fine**. However simultaneous FIPS 140-3 and SP 800-90 changes closer to CC / BSI AIS-31. Cryptography has moved from black-box pass/fail statistical testing to entropy validation and source stochastic models.

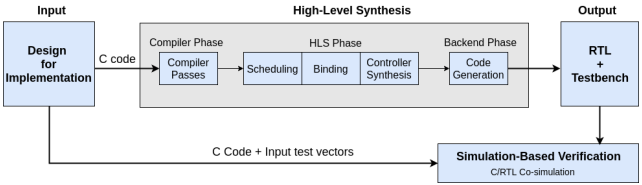
M.-J. O. Saarinen, G.R. Newell, and B. Marshall. *"Building a Modern TRNG: An Entropy Source Interface for RISC-V."* ASHES '20 (2020). `https://ia.cr/2020/866`

UK & USA government signals are very **positive to PQC** and **negative to QRNGs**:

> *"The NCSC believes that classical RNGs will continue to meet our needs for government and military applications for the foreseeable future. [..]" (March 2020)*

Reasons: What is the problem that QRNG solves? Is it physically secure? etc.
`https://www.ncsc.gov.uk/whitepaper/quantum-security-technologies`

HLS: Early Post-Quantum Hardware



- Some early reports based on HLS (High-Level Synthesis), taking C reference implementations trough automated RTL translation. Various rationale given:
  - **Skills shortage**: HLS requires less understanding of HDL or RTL.
  - **Cryptography is hard**: Less understanding of algorithms is needed.
  - **Time to Market**: Just apply the tool, tweak things, get some kind of result.
- HLS results are often reflect on the HLS tools rather than on the algorithm or hardware – for reasons given above. ¯_(ツ)_/¯
- **Monolithic**: Basic HLS implementations typically can do only one function; some single-purpose designs have been the size of half-dozen RISC-V CPUs.

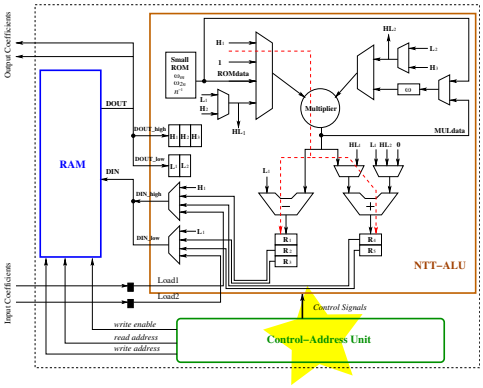
HLS.. When You Actually Know What You're Doing

Later, the GMU team compared those HLS results to their RTL engineered designs. From their Round 2 report (*recommended*): `https://eprint.iacr.org/2020/795`

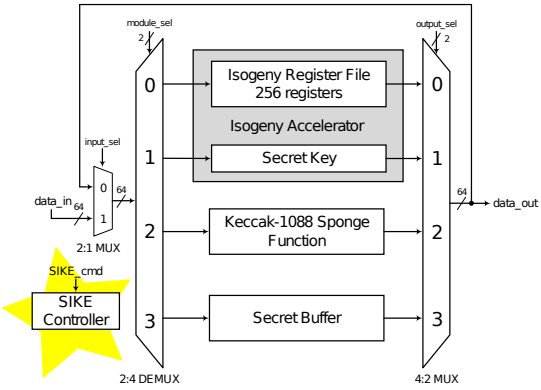
> *"The differences in obtained results are huge, although probably not that surprising, taking into account the almost complete reliance on tools in [14]. [..] Overall in terms of the latency times area product, the HLS-based designs are* **three orders of magnitude worse**."

- → Also observed was the unpredictability; NewHope and Kyber are very similar on RTL (and as algorithms) but had a huge difference in Early HLS.
- → However, one can use HLS **C-language pragmas** to guide the synthesis process in order to archive results very close to RTL. Requires more time & skill.
- → Maintenance: Non-portable HLS tool lock-in. Bad for commercial design reuse.

Monolithic PQC Blocks Have Custom "Controllers."



Roy et al. (2014) "Compact Ring-LWE Cryptoprocessor." CHES 2014



Koziel et al. (2019) "SIKE'd Up: [..]" (Isogeny) `https://ia.cr/2019/711`

In addition to arithmetic & symmetric, a **controller**: *"For the SIKE controller, we fit all SIKE functionality in 107 32-bit instructions, thus fitting well within a 1KB ROM block."*

→ Custom execution unit, Control ISA & Assembler; **Effectively a mini-CPU.**

Talk Outline

I Introduction: Post-Quantum Transition

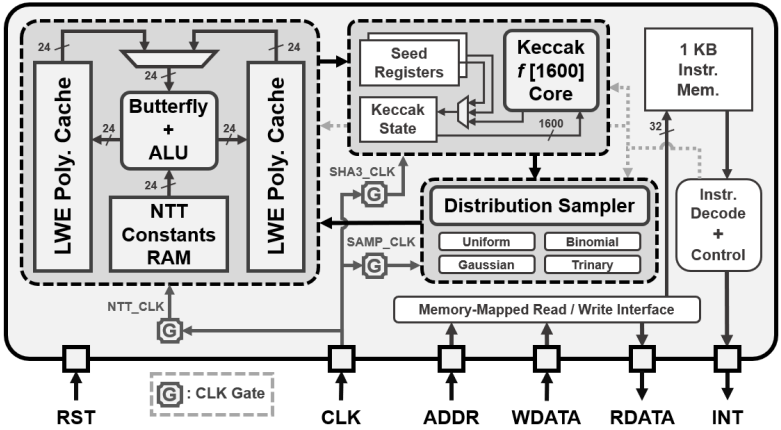
II Monolithic Public-Key Crypto Hardware

III RISC-V and Hardware-Software Codesign for Post-Quantum

IV Conclusions: Commercial Post-Quantum Crypto IP

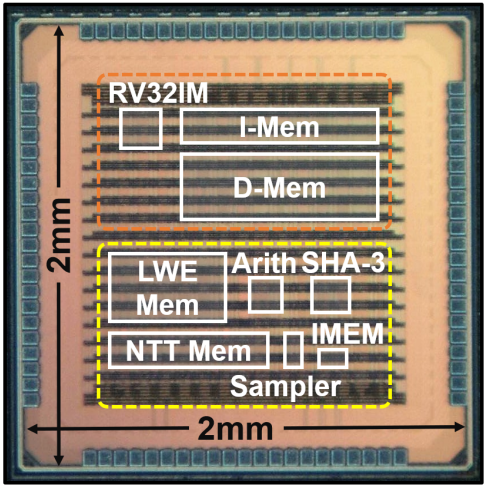
Sapphire (2019)

U. Banerjee, T. S. Ukyab, and A. P. Chandrakasan. "**Sapphire**: *A Configurable Crypto-Processor for Post-Quantum Lattice-based Protocols.*" TCHES 2019(4).



(NIST PQC Round2) Frodo, NewHope, qTESLA, Kyber, and Dilithium.

Sapphire (2019) – Hello RISC-V



Chip Specifications	
Technology	TSMC 40 nm LP CMOS
Supply voltage	0.68 – 1.1 V
Package	64-pin QFN
Die size	2 mm x 2 mm

Lattice Cryptography Processor (Skywalker)	
Total area	0.28 mm^2
Logic gates	106k (NAND2 equiv.)
SRAM	40.25 KB
Max. frequency	12 MHz at 0.68 V and 72 MHz at 1.1 V
Lattice parameters	N: 64 to 2048 and q: 24-bit configurable
CS-PRNG	SHAKE - 128 / 256
Hash function	SHA-3 - 256 / 512

Processors are not Magical..

- There is only a small step from a Finite State Machine (**FSM**) to a "processor." (The 1981 PC Keyboard had a 8048 microcontroller, every toaster since, etc.)
- RISC-V instruction words have 4×2^{25} blocks reserved for **custom extensions**.
- Why not encode the control into those bits and have a tiny 1-CPI RISC-V core?
- PQC algorithms are complex; you'll need a well-tested controller anyway.
- My Keccak (SHA-3 / SHAKE) unit is also about the same logic size as my RV32I.

Custom-0	`xxxxxxxx_xxxxxxxx_xxxxxxxx_x0001011`
Custom-1	`xxxxxxxx_xxxxxxxx_xxxxxxxx_x0101011`
Custom-2	`xxxxxxxx_xxxxxxxx_xxxxxxxx_x1011011`
Custom-3	`xxxxxxxx_xxxxxxxx_xxxxxxxx_x1111011`

Note: Custom-0 and Custom-1 will never be used by **standard extensions**.
(Custom-2 and Custom-3 may not be available on RV128; RV32 & RV64 okay.)

My Experience with Modern SoC Architectures

A processor has a processor has a processor.. *(and after NDA you'll learn of 6 more)*

RISC-V Standard Extensions for Cryptography

The standard extensions are supported by compiler toolchains, and can be used in standard middleware such as the Kernel, OpenSSL, etc. **Permissive BSD license.**

- **Scalar Crypto K** extensions provide standards-compliant AES, SHA, DRBG, Entropy Source if needed. Status: `https://wiki.riscv.org/x/MVcF`

  B. Marshall, G. R. Newell, D. Page, M.-J. O. Saarinen, and C. Wolf. *"The design of scalar AES Instruction Set Extensions for RISC-V."* TCHES 2021(1). `https://doi.org/10.46586/tches.v2021.i1.109-136`
- **Bitmanip B** has additional arithmetic instructions that may be helpful.
- **Zkt "timing-safe list"** attests that the machine has data-independent latency for a set of instructions. `https://github.com/rvkrypto/riscv-zkt-list`
- **Vector V** extension will be very helpful for Post-Quantum Cryptography. We're also building a **Vector Krypto K** in the RISC-V CETG (a draft exists).

Tightly Coupled vs Loosely Coupled

1. **Crypto K** and **Bitmanip B** standard extensions (optional – if needed!).
2. We may add **Custom X** register-register instructions too, e.g. $\mathrm{GF}(q)$ or $\mathrm{GF}(2^n)$ small-field arithmetic, sampler/rounding functions, constant-time tables.
3. **Complex peripherals** usually best controlled via traditional memory mapping. (Due to scalability, place-route & timing constraints, electrical crosstalk, etc.)
4. Slightly faster control may be achieved directly from decoder / instruction flow.

We've been doing PQC hardware commercially for a few years and use all 1–4.

Many, many academics too (too many to mention all!). Interesting current work:

T. Fritzmann et al, *"Masked Accelerators and Instruction Set Extensions for Post-Quantum Cryptography"* (April 2021) `https://ia.cr/2021/479`

My Co-Design Exploration Flow

An ISA does not define *how* to implement the instruction set. We do it two ways:

PQSoC: RTL Implementation

Modular RV32IMCK, pipelined 3-Stage Harvard Architecture.

Verilog HDL suitable for FPGA and ASIC hardware synthesis.

Small footprint (**energy**), mostly single-cycle execution (**fast**).

PQSE: Full System Emulator

Interprets RISC-V instructions on a PC, near real-time speed.

Also emulates PQSoC system peripherals, crypto hardware.

Advanced performance profiling, security analysis features.

Flow: Co-design of a new cryptographic *accelerator* and/or *custom instruction*.

Conceptualize How about X?

Add to Emulator Easier than HW.

Develop Drivers Does this work?

Implement RTL Synth, Verify.

Talk Outline

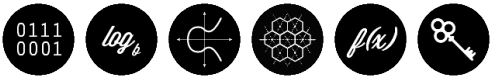
I Introduction: Post-Quantum Transition

II Monolithic Public-Key Crypto Hardware

III RISC-V and Hardware-Software Codesign for Post-Quantum

IV Conclusions: Commercial Post-Quantum Crypto IP

Academic and Real-Life Cryptography: Differences



KYBER, NTRU, SABER, Classic McEliece, SIKE, FrodoKEM
DILITHIUM, FALCON, RAINBOW, SPHINCS+, XMSS, XMSS^{MT}

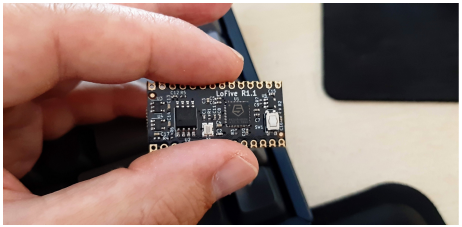
- Real-life cryptography is designed to support an application or system.
- Use case determines the algorithm and its clock, area/speed/power tradeoff.
- Much (most) of the work in real-life implementations goes into interfacing the bus interface, registers, (kernel) drivers, libraries, testing, and documentation.
- Portability, (formal) verification, algorithm test suites, compliance, flexibility.

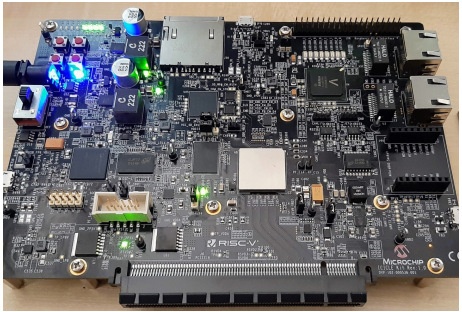
“No FIPS, No Sales.” – A Note on Testing

- A commercial customer expects that you can give them a lot of assurance.
- Hardware components need to have a level of **formal verification**; SystemVerilog Formal Assertions (SVA) is typical offering in the industry.
- Processor formal verification and RISC-V architectural tests (of course).
- PQC Algorithms not yet in **FIPS 140-3** certifiable, but the “helper components” like SHA-3/SHAKE (FIPS 202), AES (FIPS 197), DRBG (NIST SP 800-90A) are.
- Higher level PQC APIs specifically need hooks to turn *{ KeyGen, Encaps, Decaps, Sign, Verify }* into deterministic ones for **Known Answer Tests** (**KAT**s).
- Also test **failures**; Signature and KEM algorithms must fail in very *specific ways*.
- **TVLA** (ISO/IEC 17825) with (ISO/IEC 20085), others for **side-channel** profiling.
- Entropy Source against SP 800-90B (for FIPS 140-3) or BSI AIS-31 (for CC).
- **.. and your proprietary security features on top!**

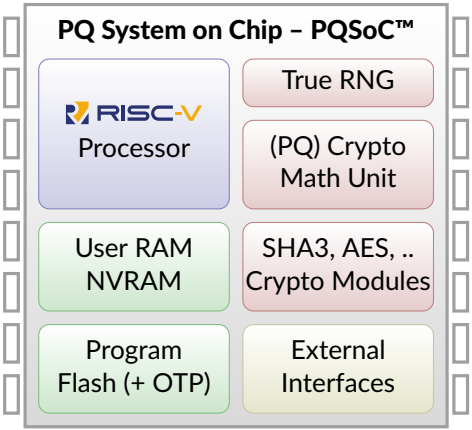
PQShield – RISC-V Security Standardization

- Active members in RISC-V CETG *(Cryptographic Extensions Task Group)*.
- PQShield ISA Extensions for AES, SHA, and Entropy Source (TRNG) contributed to RISC-V, now part of K extension.
- We build custom RISC-V CPUs, PQC coprocessors and “minion processors”.
- 100% Ownership of HDL and Firmware Supply Chain for Crypto in FPGA Fabric.





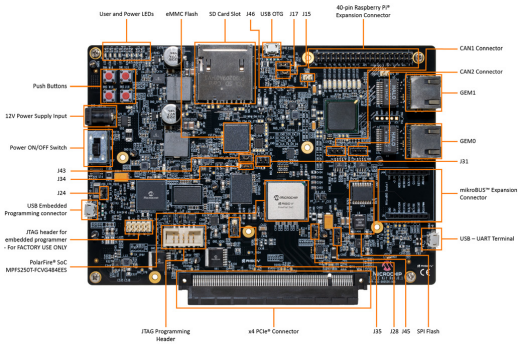
PQShield's PQSoC (2019) for Secure Elements

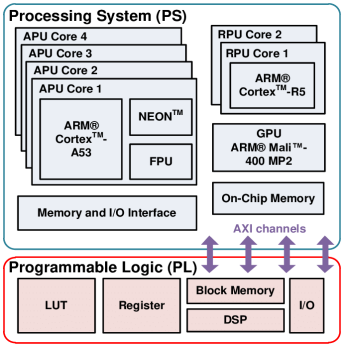


Secure Elements (TPM, Smart Cards, etc.) are little custom, single-chip computers.

- PQC Math and symmetric crypto coprocessors reduce latency, energy.
- **PQC Standards = Different Math**.
- Available from PQShield as modular IP or self-contained solution.
- FIPS 140-3 Ready, we can assist with Common Criteria and others.
- **PQSLIB3:** All NIST PQC finalist algorithms + SP 800-208 supported.
- **It works**: Speedup currently uneven, but we know the trade-offs.

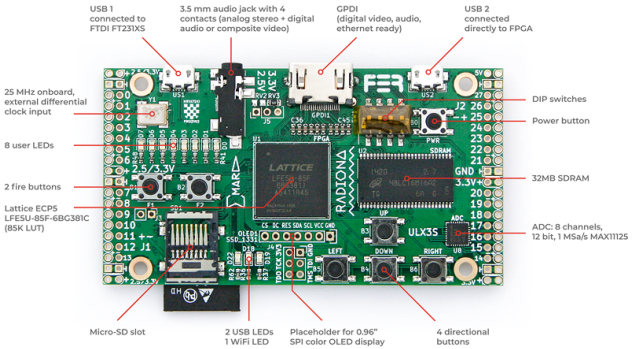
Typical Target 1: Hybrid RISCV+FPGA or ARM+FPGA





- A **single chip** with multiple “hard” CPU cores + configurable FPGA logic fabric.
- Lots of compute power in a small, readily available package. Linux & TCP/IP.
- Post-Quantum Cryptography accelerated or contained (“TPM”) in FPGA.

Typical Target 2: Self-contained ASIC of FPGA Blocks



- Generic Post-Quantum IP: Embedded Library, Crypto blocks, Control CPU.
- Applications: Secure Boot, Trusted Key Store, Authentication, SW Updates.
- Easy integration, high-quality PQC Implementations. FIPS 140-3 targeted.

Conclusions

High-Level Engineering Properties

+ **Coprocessor:** Lattice schemes can't leverage "big integer" much; a next-gen coprocessor can support both without much overall area/energy increase.
+ **CPU:** PQC Instruction Set Extension options in addition to the coprocessor.
+ **Memory:** Finalist lattice schemes don't need much work memory; 64 kB is often sufficient. Non-lattice Rainbow and McEliece many need a couple of MB.
= **Consumer Unit Price:** Will be low. (Even now on cheap FPGAs, low-end ASIC.)

Mature, commercial PQC IP will be available even before standardization.