life.augmented

Crittografia Post-Quantum su Dispositivi Embedded

Matteo BOCCHI

Senior Cryptography Engineer @ STMicroelectronics

Agenda

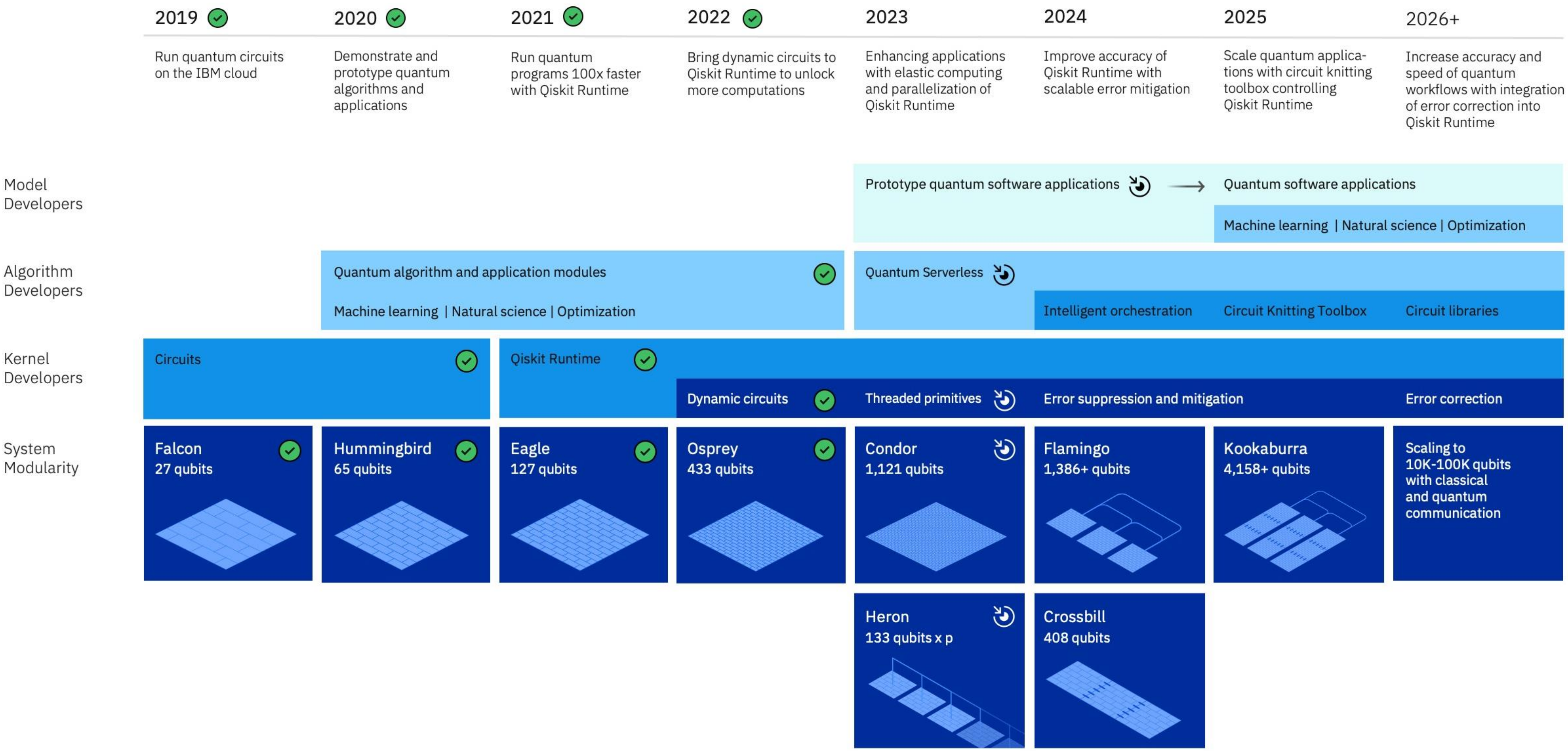
life.augmented

Quantum Computers

Quantum computers and cryptography

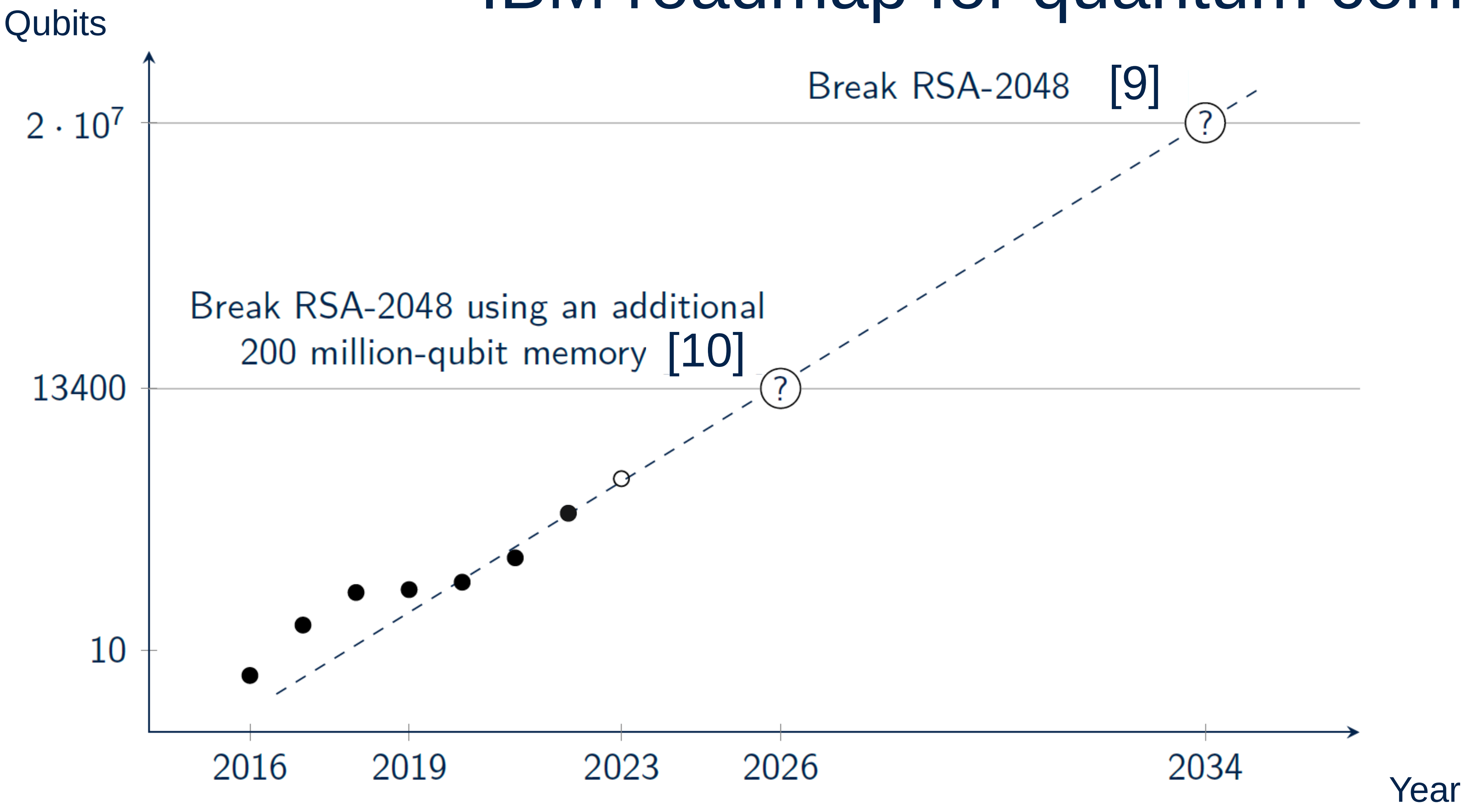
- The security of cryptography relies on intractability of certain problems by modern computers
  - Examples: RSA and integer factorization; ECC and the discrete logarithm problem
- Quantum computers would give a significative speed-up over classical computers:
  - Shor's algorithms solve in polynomial time:
    - Integer factorization (on which RSA is based)
    - The discrete-logarithm problem on elliptic curves (on which ECDSA and ECDH are based)
  - Grover's algorithm speeds up brute-force search
    - 2^{64} quantum operations to break AES-128
    - Actually, there's no global consensus on the real threat to AES by this algorithm [15]

IBM roadmap for quantum computing



Source: *The IBM Quantum Development Roadmap*, https://www.ibm.com/quantum/roadmap, May 17th, 2023.

IBM roadmap for quantum computing



Impact of large quantum computers

- Public key cryptography would be broken
  - ~~RSA~~
  - ~~ECDSA (and Elliptic Curve Cryptography)~~
  - ~~DSA (and Finite Field Cryptography)~~
  - ~~Diffie-Hellman key exchange~~

- Symmetric key-based cryptography would be affected, but not broken
  - Keys size may have to be doubled

Vulnerable NIST standards

- FIPS 186 - Digital Signature Standard
  - Digital Signatures: RSA, DSA, ECDSA

- SP 800-56A/B - Recommendation for Pair-Wise Key Establishment Schemes
  - Discrete Logs: DH, MQV
  - Factorization based: RSA key transport

Post-Quantum Cryptography

life.augmented

Post-Quantum Cryptography (PQC)

- Also known as “quantum-safe” or “quantum-resistant”
- Cryptosystems which run on classical computers, and are considered to be resistant to quantum attacks
  - Lattice-based (e.g. Kyber, Dilithium)
  - Code-based (e.g. McEliece, HQC)
  - Hash-based (e.g. SPHINCS+, LMS, XMSS)
  - Others
    - Multivariate
    - Isogeny-based
    - Etc.…

life.augmented

NIST PQC project

- Started in 2012 to monitor progress in post-quantum cryptography
- On February 2016, NIST announced its plan to find and standardize quantum-resistant public-key algorithms
- A public competition as it was for AES and SHA3
  - To ensure transparency of the process and legitimacy of the outcome
  - The goal is to achieve community consensus
- On July 2022 NIST announced the winners
  - Plus some “extra” algorithms needing further evaluations
- Standards draft expected by 2023 and final versions by 2024

Winners and 4th round candidates to NIST PQC

Winners

Key Encapsulation (KEM)

- Kyber

Digital Signature

- Dilithium
- Falcon
- SPHINCS+

4th round candidates

Key Encapsulation (KEM)

- BIKE
- Classic McEliece
- HQC
- SIKE (*)

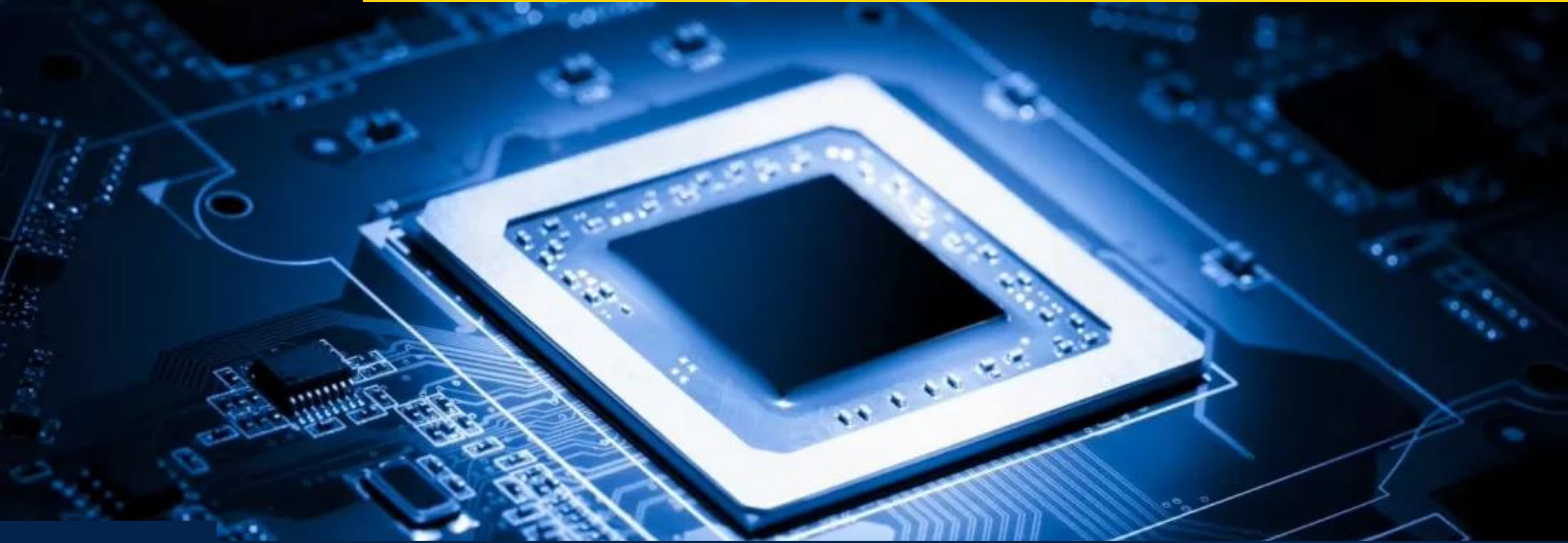
(*) Heavily attacked after the 4th-round call

Code-based
Elliptic-curve isogenies
Lattice-based
Stateless Hash-based Signature

*“NIST also [performed] a **new Call for Proposals** for public-key (quantum-resistant) **digital signature algorithms** […]. NIST is primarily looking to diversify its signature portfolio, so signature schemes that are **not based on structured lattices** are of greatest interest. NIST would like submissions for signature schemes that have **short signatures and fast verification**.” → Submission deadline: June 1st, 2023*

life.augmented

PQC HW Acceleration



Kyber and Dilithium

- Among the NIST PQC Competition winners
  - Respectively for Key Encapsulation and Digital Signature categories
- Both are based on Lattices
  - Heavily rely on polynomial arithmetic and SHA-3 hashing
- Investigations on hardware components to accelerate the two algorithms have been done
- Several flavours to fit different scenarios and certifications requirements
  - Design and develop solutions, both protected and unprotected against physical attacks
- Studying countermeasures against side-channel attacks
  - Recent paper written by colleagues in our team [16]

Some considerations

- Dilithium uses larger resources (3-30× RAM, 4-20× runtime) for comparable security, with respect to Kyber
  - Moreover, timing of signature generation varies across executions, ¼× to 6× in 99% of cases
- Protected implementations are more costly
  - Both Kyber and Dilithium require a protected SHA-3
  - Latency at least doubles with strong physical attacks protections
  - RAM increases
  - In general, Dilithium is harder to protect against side-channel attacks
    - Would some actors prefer to wait for new digital signatures algorithms coming from NIST new competition?

life.augmented

Stateful Hash-Based Signatures (HBS)

Stateful Hash-Based Signatures

- Well known techniques based on cryptographic hash functions
  - Are gaining interest because of their resistance against quantum attacks
- Signature generation involves the private key, which:
  - is a set of one-time signature (OTS) keys
    - Using the same OTS key twice will break the scheme's security
    - This means we have a limited number of signatures per secret key
  - must be written in a persistent RW storage location
    - The state must change at every signature
    - Storing it in RAM is dangerous because a hard reset will cancel the state and the same OTS key can be used twice
- Signature verification needs public key only, therefore not having these limitations

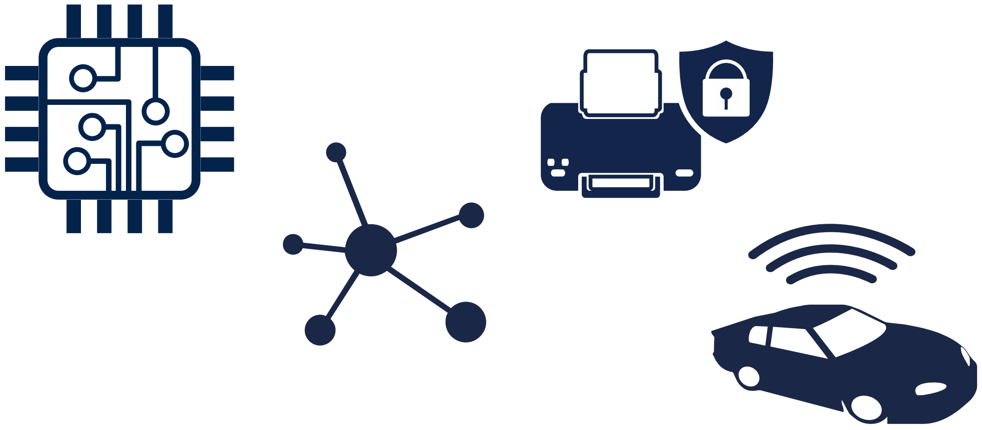
Standardization of Stateful HBS

- Stateful HBS is not part of the NIST PQC context
  - Its API is different from classical digital signature algorithms
- But there is significant interest in the standardization of such schemes
  - Well understood underlying technology
  - Already deployed in some systems (e.g. Git, Bitcoin and Ethereum peer-to-peer networks)
  - Security based on hash function's security, better known with respect to other PQC schemes
- NIST established a sub-project for approving stateful HBS schemes.
  Two schemes (developed through the IETF) are actively considered:
  - LMS, specified in RFC 8554
  - XMSS, specified in RFC 8391

Stateful Hash-Based Signature Verification on a Cortex®-M4

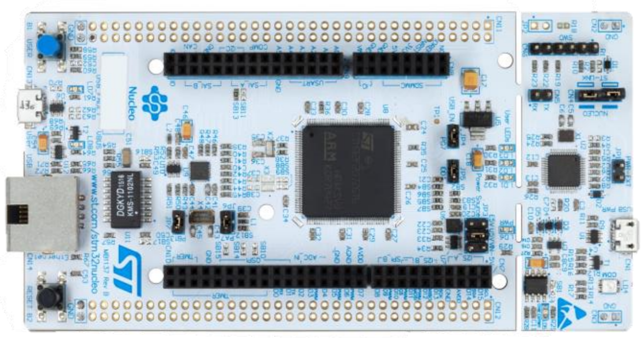
Stateful HBS for MCU

- Digital signatures are widely used in scenarios where MCUs are involved
  - MCU firmware upgrade
  - Secure communication between IoT devices
  - Car-2-Car communication for traffic monitoring
- Most used algorithms are RSA and ECC
  - They will be broken by large quantum computers
  - Not all MCUs provide Public Key Accelerators (PKA), and SW implementations are slow
- There exist very important scenarios perfect for Stateful HBS signature verification
  - e.g., secure firmware upgrade
  - It is possible to leverage on HW accelerated hash functions (wider usage than PKA)



LMS/XMSS optimized implementations

- Starting from reference code (XMSS | LMS)
  - Not exactly embedded devices “friendly”
    - E.g.: XMSS relied on OpenSSL
- NUCLEO-F439ZI board (Cortex®-M4 CPU @180MHz)
- Added support for Hash HW peripheral (not trivial for LMS)
- Code refactoring and optimizations
  - To speed up the performance
  - And reduce the code size, avoiding inclusion of unused component



LMS/XMSS: some results (Signature Verification)

ALGO		# Sig	Signature size (B)	Pubkey size (B)	Original timings (10^3 cycles)	HW hash & SW opt. (10^3 cycles)	Speedup factor	Code size (kB)
LMS		1 024	4 624	60	1 890	367	**5.2x**	5.3
		1 024	2 512	60	3 138	433	**7.3x**	
		1 048 576	2 832	60	3 239	444	**7.3x**	
XMSS		1 024	2 500	64	19 149	2 942	**6.5x**	5.6
		1 048 576	2 820	64	19 819	2 983	**6.6x**	
ECDSA P-256	mbedTLS	∞	64	64	69 279	-	-	25.9
	X-Cube-CryptoLib (*)				5 382	-	-	17.7
	HW acc. (**)				-	5 249	-	1.7

(*) STM32 Cryptographic Library, https://www.st.com/en/embedded-software/x-cube-cryptolib.html
(**) STM32 Public Key Accelerator, STM32WB

LMS/XMSS: some results (Signature Verification)

ALGO		# Sig	Signature size (B)	Pubkey size (B)	Original timings (10^3 cycles)	HW hash & SW opt. (10^3 cycles)	Speedup factor	Code size (kB)
LMS		1 024	4 624	60	1 890	367	**5.2x**	5.3
		1 024	2 512	60	3 138	433	**7.3x**	
		1 048 576	2 832	60	3 239	444	**7.3x**	
XMSS		1 024	2 500	64	19 149	2 942	**6.5x**	5.6
		1 048 576	2 820	64	19 819	2 983	**6.6x**	
ECDSA P-256	mbedTLS	∞	64	64	69 279	-	-	25.9
	X-Cube-CryptoLib (*)				5 382	-	-	17.7
	HW acc. (**)				-	5 249	-	1.7

(*) STM32 Cryptographic Library, https://www.st.com/en/embedded-software/x-cube-cryptolib.html
(**) STM32 Public Key Accelerator, STM32WB

LMS/XMSS: some results (Signature Verification)

ALGO		# Sig	Signature size (B)	Pubkey size (B)	Original timings (10^3 cycles)	HW hash & SW opt. (10^3 cycles)	Speedup factor	Code size (kB)
LMS		1 024	4 624	60	1 890	367	**5.2x**	5.3
		1 024	2 512	60	3 138	433	**7.3x**	
		1 048 576	2 832	60	3 239	444	**7.3x**	
XMSS		1 024	2 500	64	19 149	2 942	**6.5x**	5.6
		1 048 576	2 820	64	19 819	2 983	**6.6x**	
ECDSA P-256	mbedTLS	∞	64	64	69 279	-	-	25.9
	X-Cube-CryptoLib (*)				5 382	-	-	17.7
	HW acc. (**)				-	5 249	-	1.7

(*) STM32 Cryptographic Library, https://www.st.com/en/embedded-software/x-cube-cryptolib.html
(**) STM32 Public Key Accelerator, STM32WB

LMS/XMSS: some results (Signature Verification)

ALGO		# Sig	Signature size (B)	Pubkey size (B)	Original timings (10^3 cycles)	HW hash & SW opt. (10^3 cycles)	Speedup factor	Code size (kB)
LMS		1 024	4 624	60	1 890	367	**5.2x**	5.3
		1 024	2 512	60	3 138	433	**7.3x**	
		1 048 576	2 832	60	3 239	444	**7.3x**	
XMSS		1 024	2 500	64	19 149	2 942	**6.5x**	5.6
		1 048 576	2 820	64	19 819	2 983	**6.6x**	
ECDSA P-256	mbedTLS	∞	64	64	69 279	-	-	25.9
	X-Cube-CryptoLib (*)				5 382	-	-	17.7
	HW acc. (**)				-	5 249	-	1.7

(*) STM32 Cryptographic Library, https://www.st.com/en/embedded-software/x-cube-cryptolib.html
(**) STM32 Public Key Accelerator, STM32WB

life.augmented

LMS/XMSS: some results (Signature Verification)

ALGO		# Sig	Signature size (B)	Pubkey size (B)	Original timings (10^3 cycles)	HW hash & SW opt. (10^3 cycles)	Speedup factor	Code size (kB)
LMS		1 024	4 624	60	1 890	367	**5.2x**	5.3
		1 024	2 512	60	3 138	433	**7.3x**	
		1 048 576	2 832	60	3 239	444	**7.3x**	
XMSS		1 024	2 500	64	19 149	2 942	**6.5x**	5.6
		1 048 576	2 820	64	19 819	2 983	**6.6x**	
ECDSA P-256	mbedTLS	∞	64	64	69 279	-	-	25.9
	X-Cube-CryptoLib (*)				5 382	-	-	17.7
	HW acc. (**)				-	5 249	-	1.7

(*) STM32 Cryptographic Library, https://www.st.com/en/embedded-software/x-cube-cryptolib.html
(**) STM32 Public Key Accelerator, STM32WB

life.augmented

LMS/XMSS: some results (Signature Verification)

ALGO		# Sig	Signature size (B)	Pubkey size (B)	Original timings (10^3 cycles)	HW hash & SW opt. (10^3 cycles)	Speedup factor	Code size (kB)
LMS		1 024	4 624	60	1 890	367	**5.2x**	5.3
		1 024	2 512	60	3 138	433	**7.3x**	
		1 048 576	2 832	60	3 239	444	**7.3x**	
XMSS		1 024	2 500	64	19 149	2 942	**6.5x**	5.6
		1 048 576	2 820	64	19 819	2 983	**6.6x**	
ECDSA P-256	mbedTLS	∞	64	64	69 279	-	-	25.9
	X-Cube-CryptoLib (*)				5 382	-	-	17.7
	HW acc. (**)				-	5 249	-	1.7

(*) STM32 Cryptographic Library, https://www.st.com/en/embedded-software/x-cube-cryptolib.html
(**) STM32 Public Key Accelerator, STM32WB

life.augmented

LMS/XMSS: some results (Signature Verification)

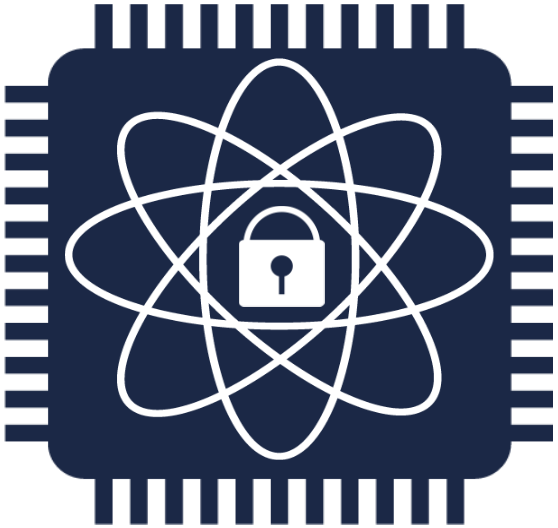
ALGO		# Sig	Signature size (B)	Pubkey size (B)	Original timings (10^3 cycles)	HW hash & SW opt. (10^3 cycles)	Speedup factor	Code size (kB)
LMS		1 024	4 624	60	1 890	367	**5.2x**	5.3
		1 024	2 512	60	3 138	433	**7.3x**	
		1 048 576	2 832	60	3 239	444	**7.3x**	
XMSS		1 024	2 500	64	19 149	2 942	**6.5x**	5.6
		1 048 576	2 820	64	19 819	2 983	**6.6x**	
ECDSA P-256	mbedTLS	∞	64	64	69 279	-	-	25.9
	X-Cube-CryptoLib (*)				5 382	-	-	17.7
	HW acc. (**)				-	5 249	-	1.7

(*) STM32 Cryptographic Library, https://www.st.com/en/embedded-software/x-cube-cryptolib.html
(**) STM32 Public Key Accelerator, STM32WB

life.augmented

Conclusions

Conclusions



- PQC HW Acceleration
  - Have solutions to anticipate what we’ll need during following years
- Hash-Based Signatures
  - Performance/code size comparable to currently used algorithms
    - Usable in pre-quantum scenarios, too
  - HW hash significantly boosts performance
    - Custom drivers give extra speed-up
  - Ready solution → optimal choice for ultra low-power targets
- This work will have a significative impact on all targets currently equipped with cryptographic modules/software
  - Actively work with customers for a smooth transition to quantum resistant functionalities

References I

1. NIST Stateful Hash-Based Signatures web page. https://csrc.nist.gov/projects/stateful-hash-based-signatures
2. NIST SP 800-208. https://doi.org/10.6028/NIST.SP.800-208
3. RFC 8391. XMSS: eXtended Merkle Signature Scheme. https://datatracker.ietf.org/doc/html/rfc8391
4. RFC 8554. Leighton-Micali Hash-Based Signatures. https://datatracker.ietf.org/doc/html/rfc8554
5. Bos, J. W. ., Hülsing, A., Renes, J., & van Vredendaal, C. (2020). Rapidly Verifiable XMSS Signatures. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2021(1), 137–168. https://doi.org/10.46586/tches.v2021.i1.137-168
6. Fabio Campos, Tim Kohlstadt, Steffen Reith, Marc Stoettinger. LMS vs XMSS: Comparison of Stateful Hash-Based Signature Schemes on ARM Cortex-M4. IACR ePrint. https://eprint.iacr.org/2020/470
7. Ana Karina D. S. de Oliveira, Julio Lopez and Roberto Cabral. High Performance of Hash-based Signature Schemes. International Journal of Advanced Computer Science and Applications(IJACSA), 8(3), 2017. http://dx.doi.org/10.14569/IJACSA.2017.080358
8. Submission requirements and evaluation criteria for the post-quantum cryptography standardization process. US Department of Commerce, National Institute of Standards and Technology, 2016. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf

References II

9. Craig Gidney and Martin Ekerå. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. Quantum, 5:433, April 2021. doi:10.22331/q-2021-04-15-433.
10. Élie Gouzien and Nicolas Sangouard. Factoring 2048-bit RSA integers in 177 days with 13 436 qubits and a multimode memory. Phys. Rev. Lett., 127:140503, Sep 2021. doi:10.1103/PhysRevLett.127.140503.
11. Jakobsson et al. Fractal Merkle Tree Representation and Traversal, https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.65.6133&rep=rep1&type=pdf
12. NIST IR 8413, Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process, https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8413-upd1.pdf
13. Kyber, https://pq-crystals.org/kyber/data/kyber-specification-round3-20210804.pdf
14. Dilithium, https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf
15. Is AES-128 quantum safe?, https://crypto.stackexchange.com/questions/102671/is-aes-128-quantum-safe
16. G. Assael, P. Elbaz-Vincent and G. Reymond. Improving Single-Trace Attacks on the Number-Theoretic Transform for Cortex-M4. HOST 2023, San Jose, CA, USA.

life.augmented

Thanks for the attention!
Any question?

@ matteo.bocchi@st.com

life.augmented