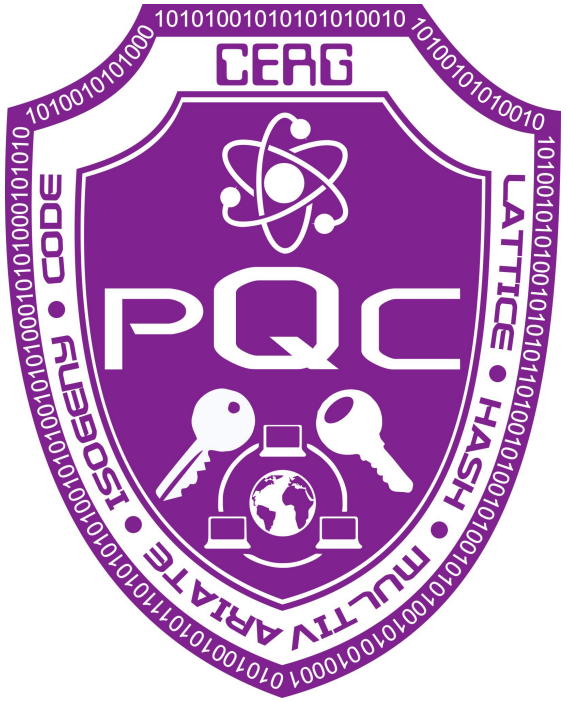
Post-Quantum Cryptography in Hardware and Embedded Systems: Toward Choosing the Most Efficient and Flexible New Public Key Cryptography Standards



Kris Gaj

George Mason University

Thank You!

Great thanks to

Dr. Krystian Matusiewicz & Dr. Piotr Sapiecha

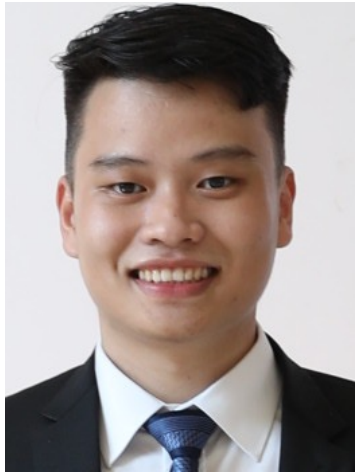
for the kind invitation
to give this talk!

CERG: Cryptographic Engineering Research Group

3 faculty members, 6 Ph.D. students
3 MS students, 9 affiliated scholars

CERG Group Members Supporting PQC

PhD Students



Viet

RTL Design of HW Accelerators for Lattice-based, Code-based, & Secret-key-based PQC

Kamyar

RTL Design of HW Accelerators for Lattice-based PQC

Side-Channel Analysis

RISC-V Accelerators

Luke

RTL Design of HW Accelerators for Lattice-based PQC

Duc

NEON-based SW implementations and HLS Design of HW Accelerators for Lattice-based PQC

Brian

NEON-based SW Implementations for Code-based PQC

CERG Affiliated Scholars Supporting PQC

Recent Graduates

Farnoud

SW/HW Codesign

RTL Accelerators

Experimental Setup for Timing Measurements

CAD Tools

Apple

Bakry

Experimental Setup for Side-Channel Analysis

Lightweight Architectures

PQSecure

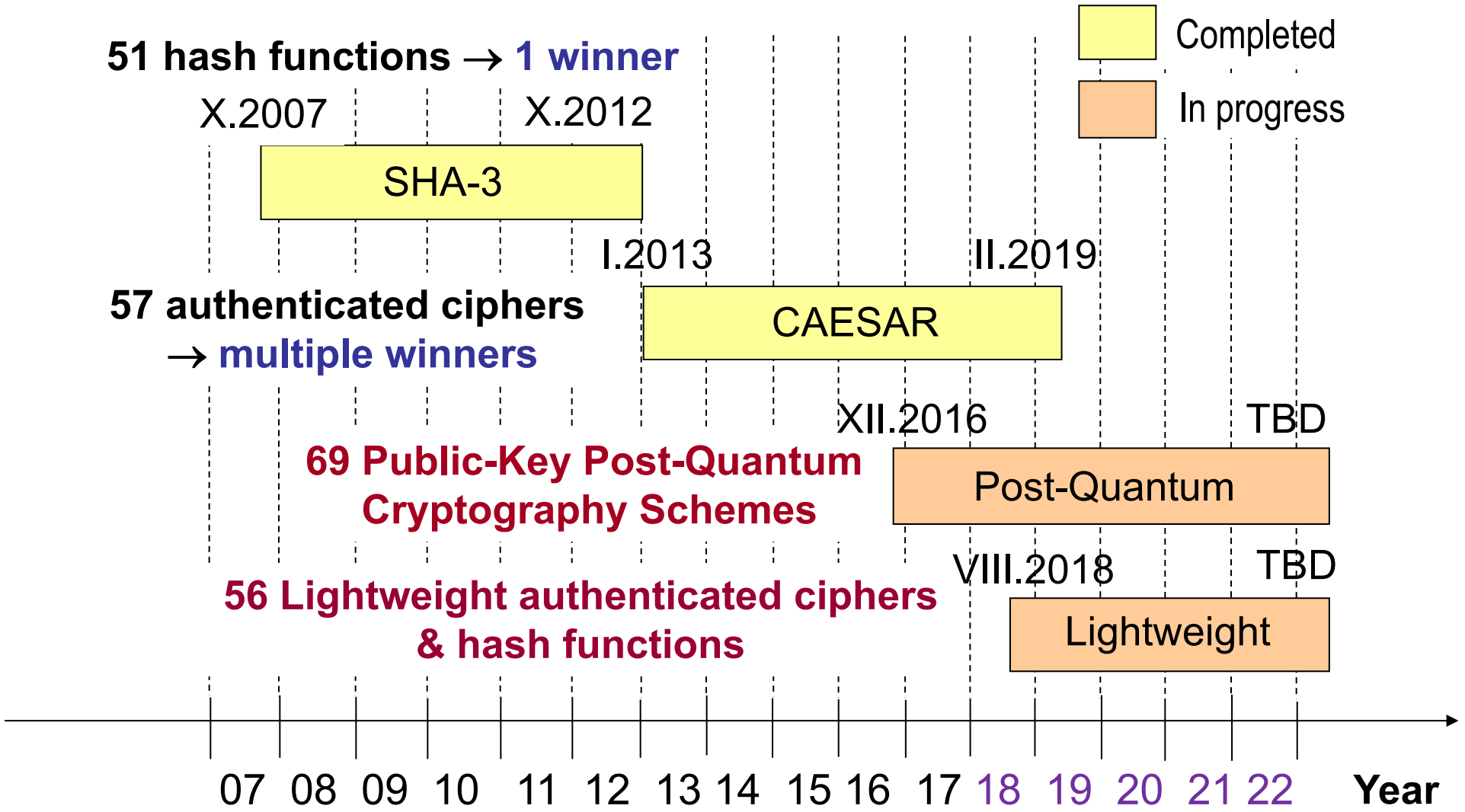
2019 Visitor

Michał

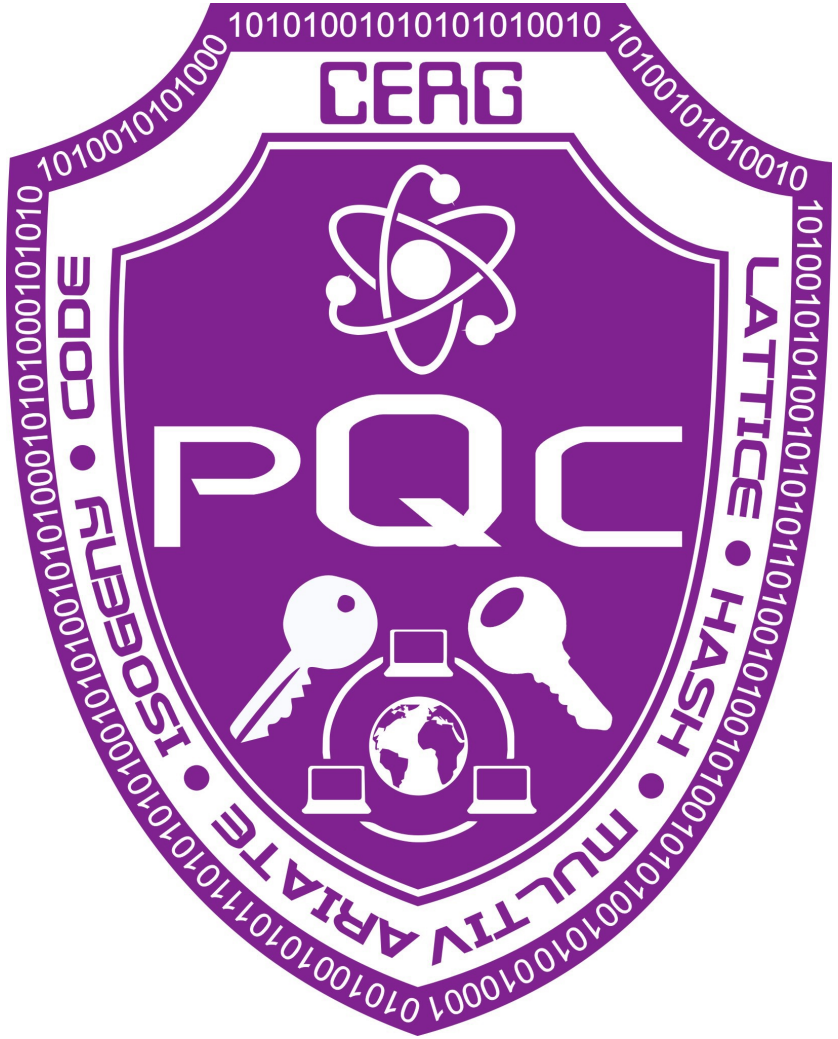
RTL Design of HW Accelerators for Lattice-based PQC & Lattice Sieving

Polish National Cyber Security Centre

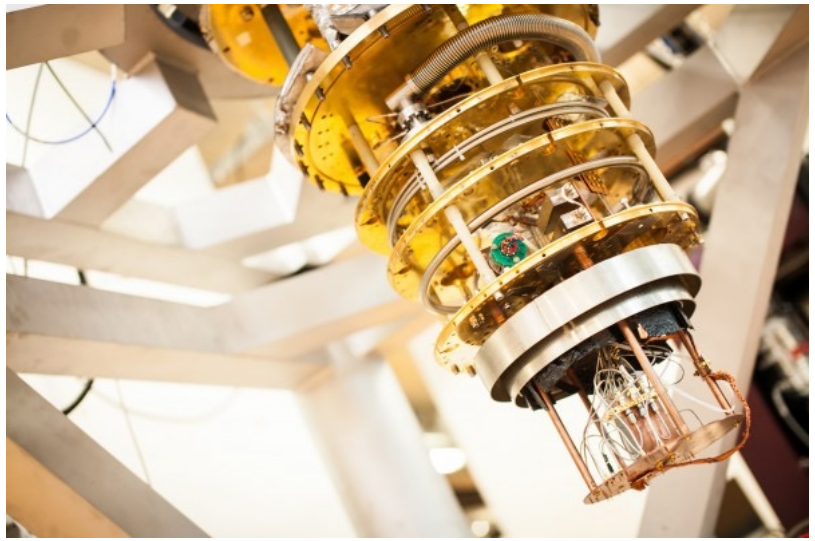
CERG Participation in Cryptographic Contests 2007-Present



Post-Quantum Cryptography in Hardware and Embedded Systems



Quantum Computers

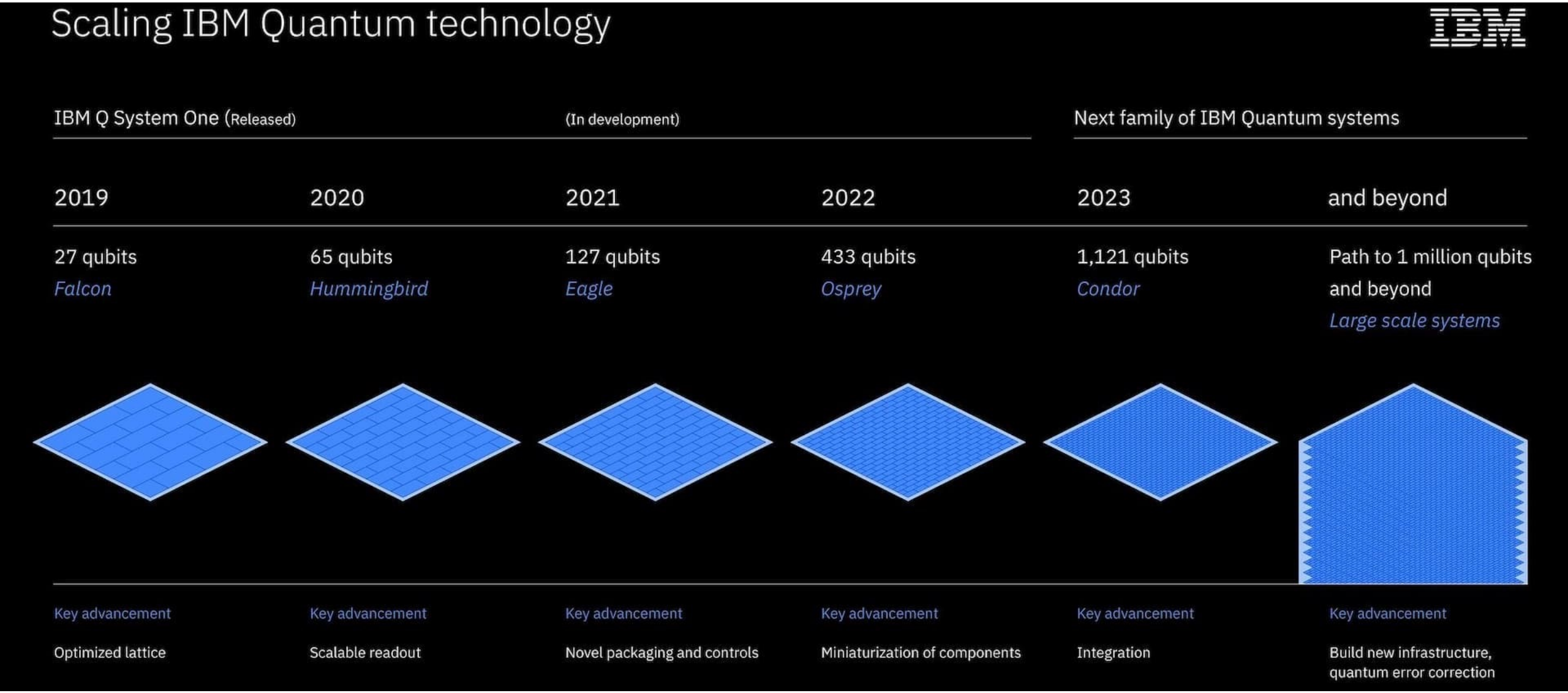


- Substantial investments by: Google, IBM, Intel, Microsoft, and governments of multiple countries

- Jan 2018: Intel's 49-qubit processor "Tangle Lake"
- Mar 2018: Google's 72-qubit processor "Bristlecone"
- 2020-2021: Three quantum computers developed at the University of Science and Technology of China reach quantum supremacy
- Nov 2021: IBM's 127-qubit quantum processor

Photos: https://www.technologyreview.com Source: https://en.wikipedia.org/wiki/Timeline_of_quantum_computing_and_communication

IBM Roadmap



Source: https://research.ibm.com/blog/ibm-quantum-roadmap

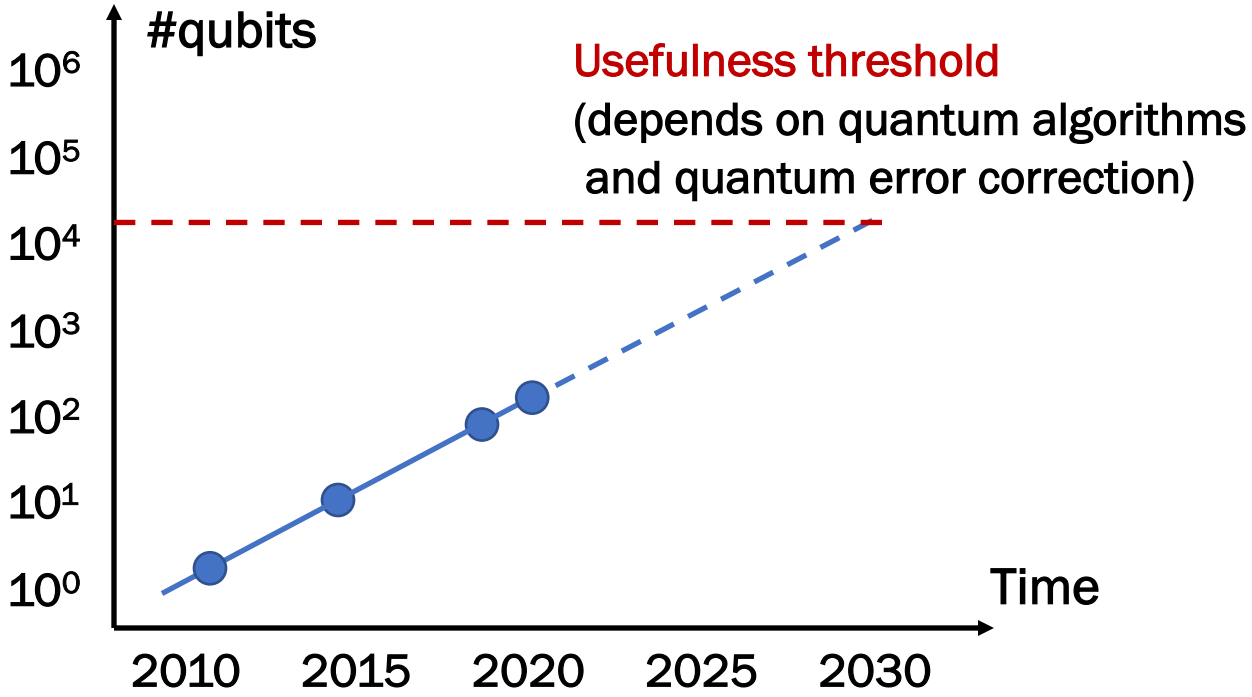
Effect on Public-Key Cryptography

1994: Shor's Algorithm, breaks major public key cryptosystems based on

Factoring:	RSA
Discrete logarithm problem (DLP):	DSA, Diffie-Hellman
Elliptic Curve DLP:	Elliptic Curve Cryptosystems

independently of the key size
assuming
a sufficiently powerful and reliable quantum computer available

How Real Is the Danger?



"There is a 1 in 5 chance that some fundamental public-key crypto will be broken by quantum by 2029."
Dr. Michele Mosca
Deputy Director of the Institute for Quantum Computing, University of Waterloo
2020

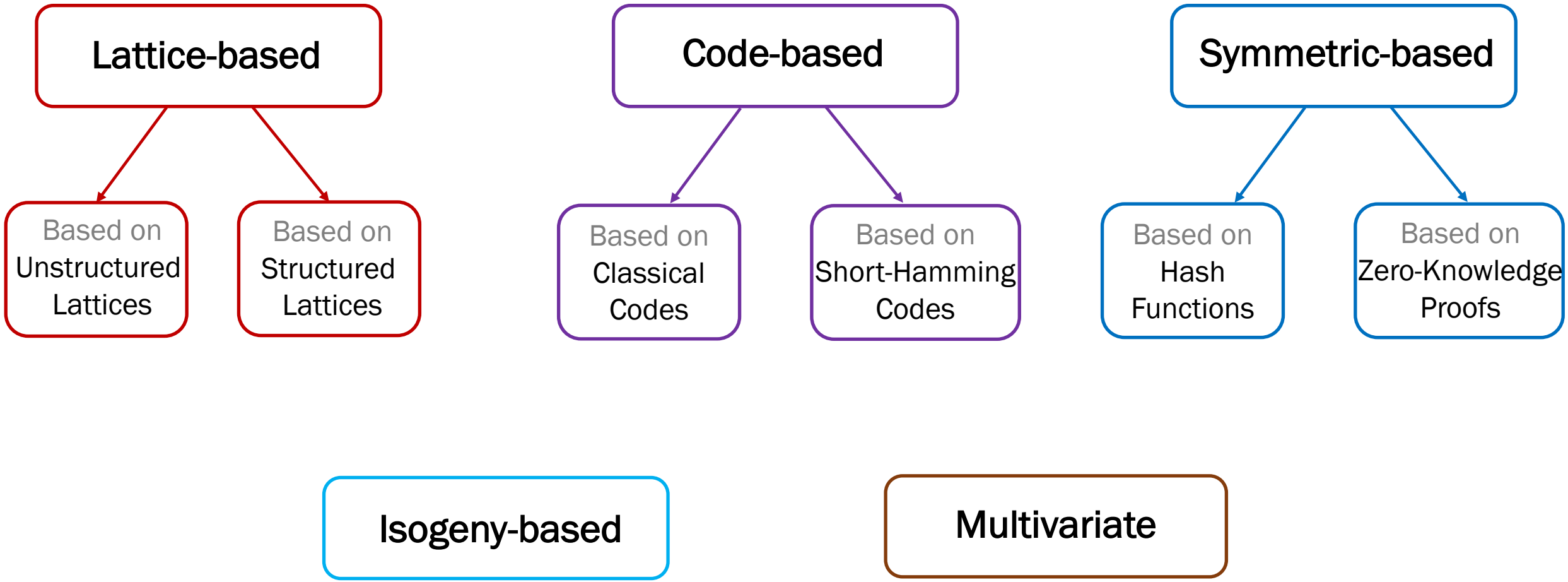
Source: Vandersypen, PQCrypto 2017; Lily Chen, seminar, 2020

Post-Quantum Cryptography (PQC)

- Public-key cryptographic algorithms for which there are no known attacks using quantum computers
  - Capable of being implemented using any traditional methods, including software and hardware
  - Running efficiently on any modern computing platforms: PCs, tablets, smartphones, servers with FPGA accelerators, etc.
- Based entirely on traditional semiconductor VLSI technology!

The biggest revolution in cryptography, since the invention of public-key cryptography in 1970s!!!

PQC Families and Subfamilies



Two Major Types of Schemes & Corresponding Families

Post-Quantum
Public Key Exchange

Post-Quantum
Digital Signatures

Lattice-based

Code-based

Multivariate

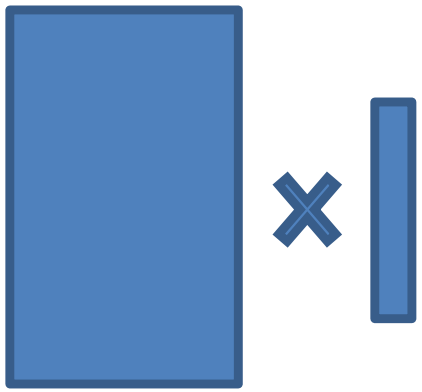
Isogeny-based

Symmetric-based

Lattice-Based Schemes

Based on
Unstructured Lattices
(a.k.a. random lattices)

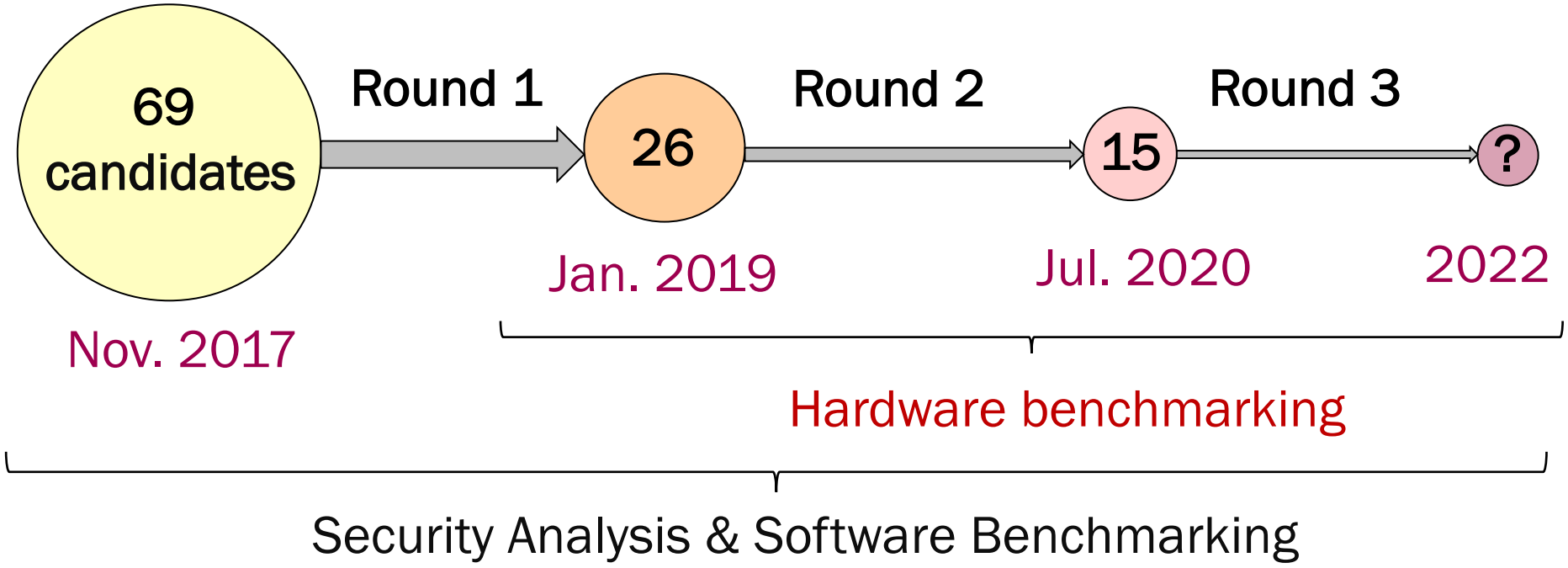
- Keys have the form of large matrices
- Major operation:
  matrix-by-vector multiplication
- Large public keys
- Low performance
- Low risk of attacks



Based on
Structured Lattices
(a.k.a. ideal lattices)

- Keys have the form of polynomials
- Major operation:
  polynomial multiplication
- Moderate public keys
- High performance
- Moderate risk of attacks

NIST PQC Standardization Process



Five Security Levels

Level	Security Description
1	At least as hard to break as AES-128 using exhaustive key search
2	At least as hard to break as SHA-256 using collision search
3	At least as hard to break as AES-192 using exhaustive key search
4	At least as hard to break as SHA-384 using collision search
5	At least as hard to break as AES-256 using exhaustive key search

Round 3 Candidates

FINALISTS

Encryption/KEM

- Lattice-based
  - CRYSTALS-KYBER
  - NTRU
  - SABER
- Code-based
  - Classic McEliece

Digital Signature

- Lattice-based
  - CRYSTALS-DILITHIUM
  - FALCON
- Multivariate
  - Rainbow

ALTERNATE

Encryption/KEM

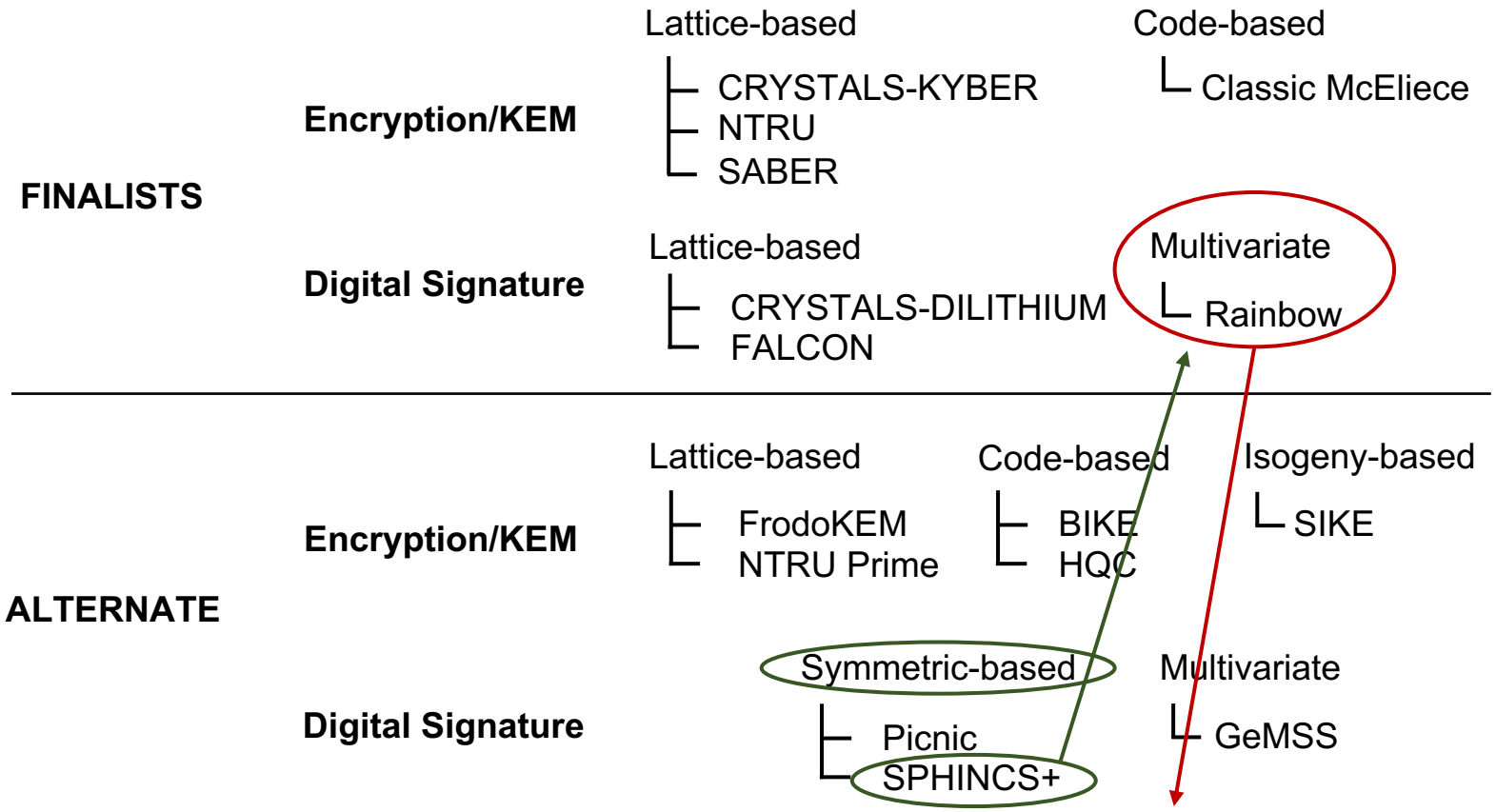
- Lattice-based
  - FrodoKEM
  - NTRU Prime
- Code-based
  - BIKE
  - HQC
- Isogeny-based
  - SIKE

Digital Signature

- Symmetric-based
  - Picnic
  - SPHINCS+
- Multivariate
  - GeMSS

Recent Developments

Round 3 Candidates



Breaking Rainbow Takes a Weekend on a Laptop
by Ward Beullens, https://eprint.iacr.org/2022/214, received 21 Feb 2022

Favorites for first-generation standards

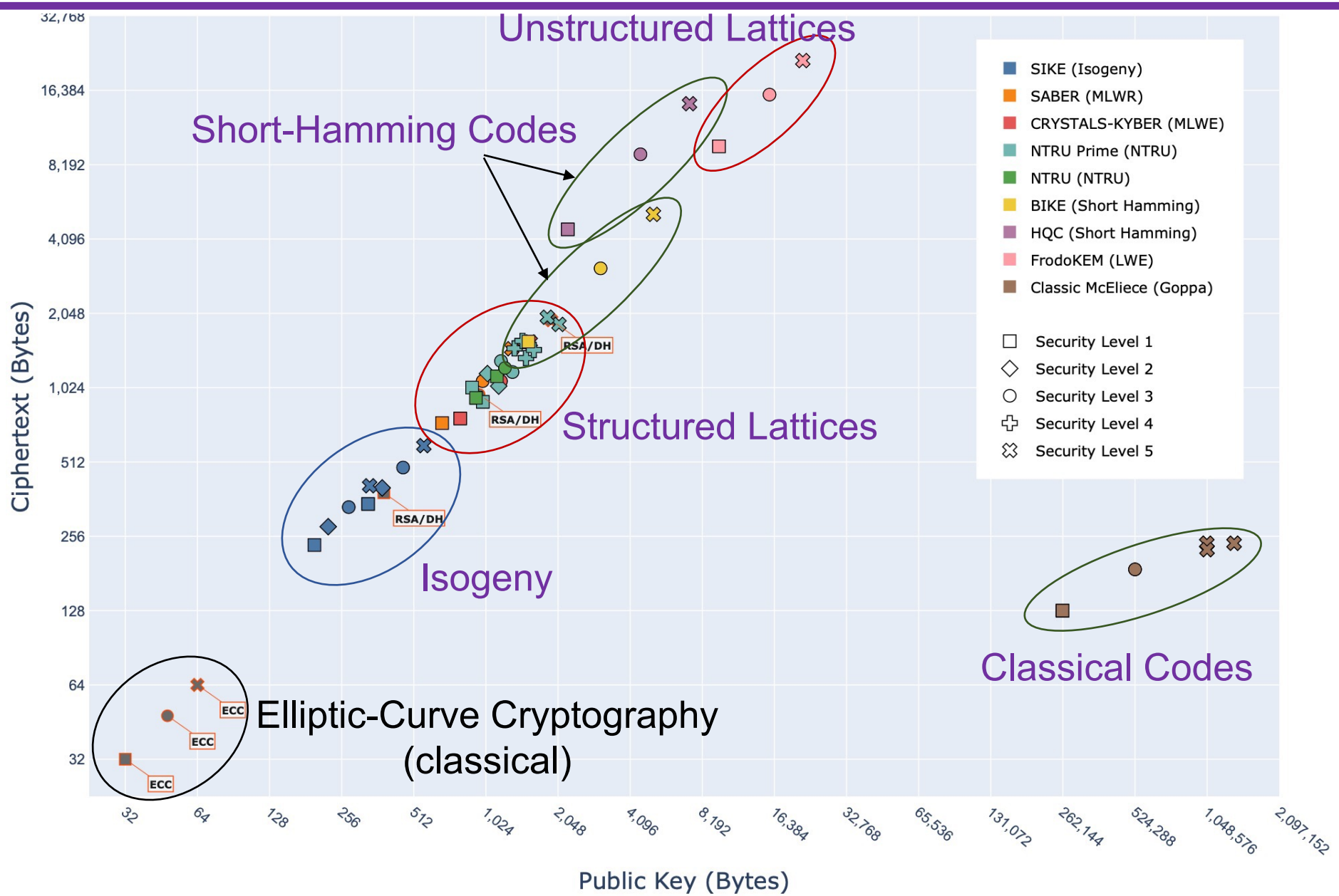
Key Exchange (Key Encapsulation Mechanism – KEM)

Based on structured lattices	CRYSTALS-KYBER SABER NTRU
Based on classical codes	Classic McEliece

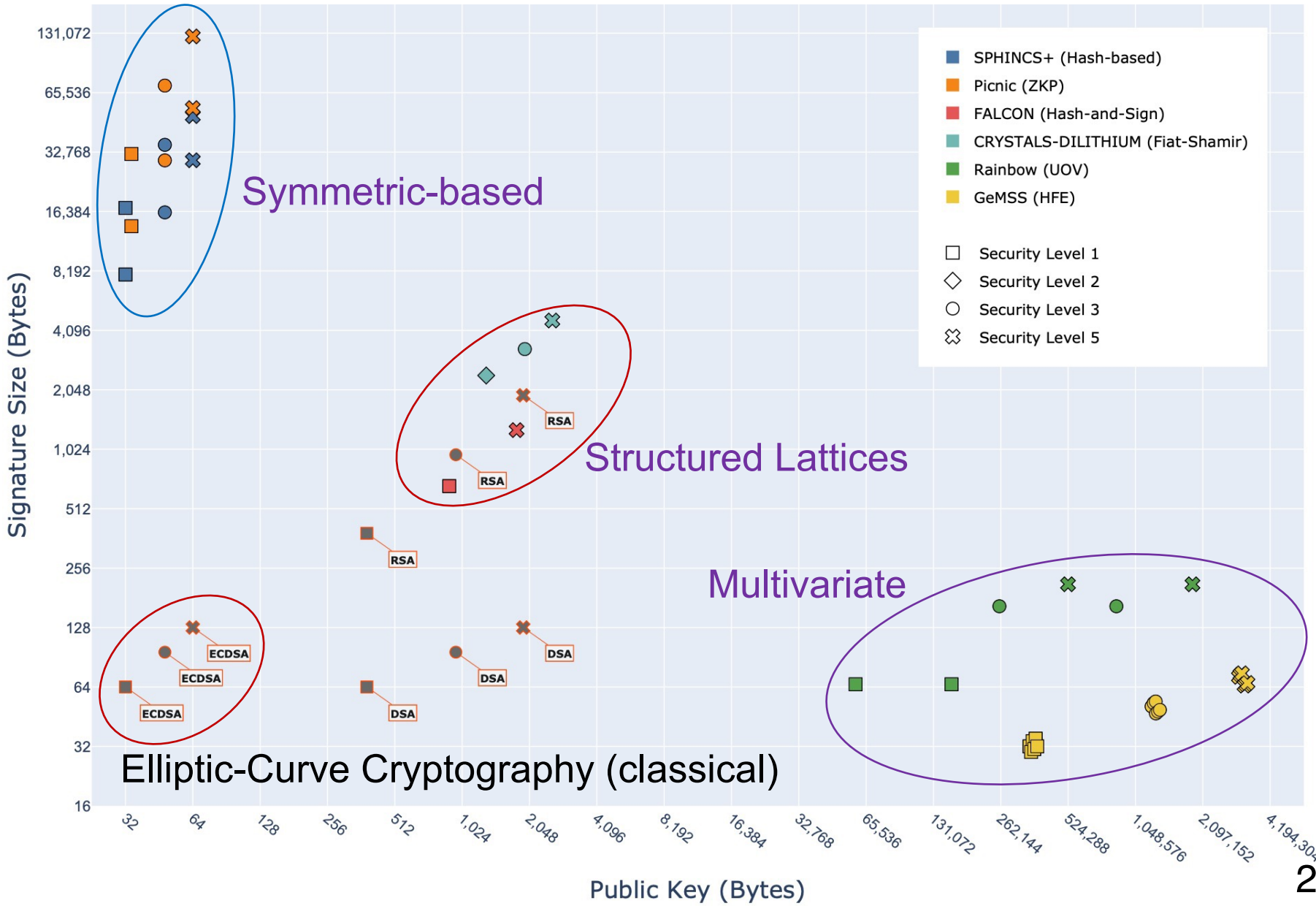
Digital Signatures

Based on structured lattices	CRYSTALS-DILITHIUM FALCON
Symmetric-based (hash-based)	SPHINCS+

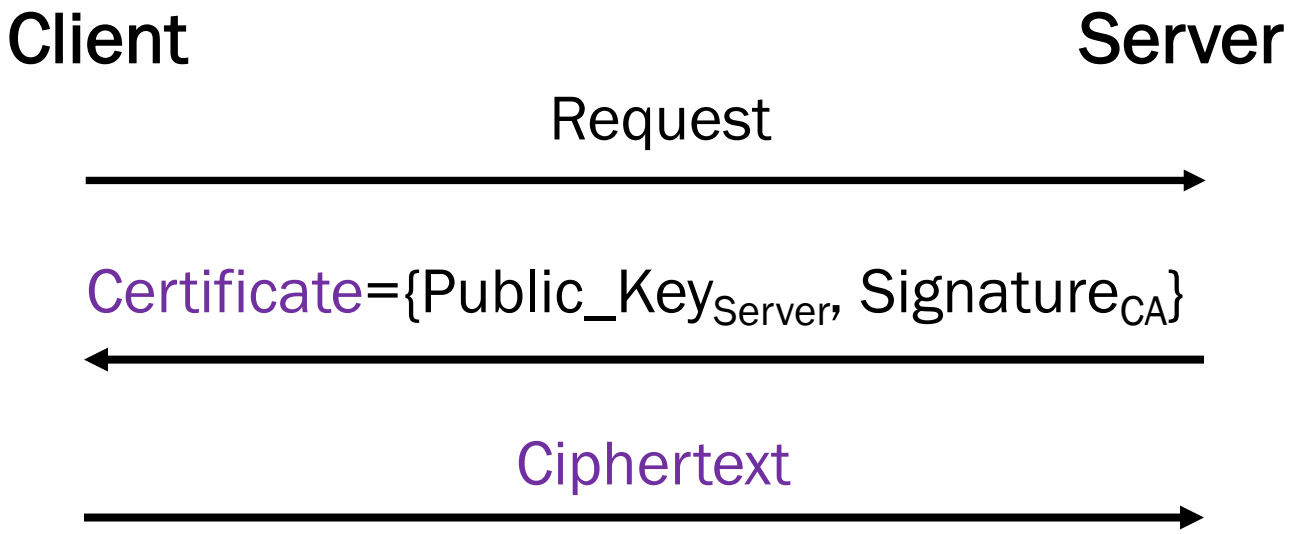
Round 3 PQC Key Exchange + Classical PKE



Round 3 + Classical Digital Signature Schemes



Certificate Size Ratio

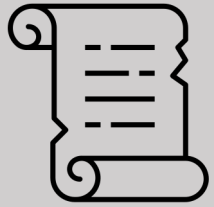


$$\text{Certificate Size Ratio} = \frac{\text{Classic McEliece + SPINCS+}}{\text{SABER + CRYSTALS DILITHIUM}} > 100$$

Evaluation Criteria

Size of Keys, Ciphertext, and Signatures

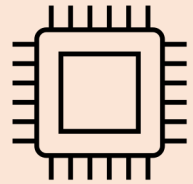
Security



Patent Issues

Software Efficiency

Hardware Efficiency



Simplicity

Flexibility

CERG Major Contributions

High-Speed Hardware Implementations of KEMs:

- NTRU (first)
- CRYSTALS-Kyber
- Saber

High-Speed Hardware Implementations of Digital Signatures:

- CRYSTALS-Dilithium
- Falcon (verification only) (first)

Lightweight Hardware Implementations of KEMs Resistant Against Side-Channel Attacks

- Saber (first)

NEON-Based Software Implementations

- NTRU
- CRYSTALS-Kyber
- Saber

Hardware Benchmarking Methodology

Design Approach

Rounds 1 & 2

Software/Hardware Co-Design (SW/HW)

- SW: C, assembly
- HW RTL: VHDL, Verilog, Chisel
- HW HLS: C, C++, System C

- Short development time

- Communication overhead
- Strong dependence on a partitioning scheme
- Inconclusive results

High-Level Synthesis (HLS)

- HW: C, C++, System C

- Short development time

- Lower performance in terms of speed and/or area (for PQC, some reports showing 2-4 orders of magnitude difference)

Round 3

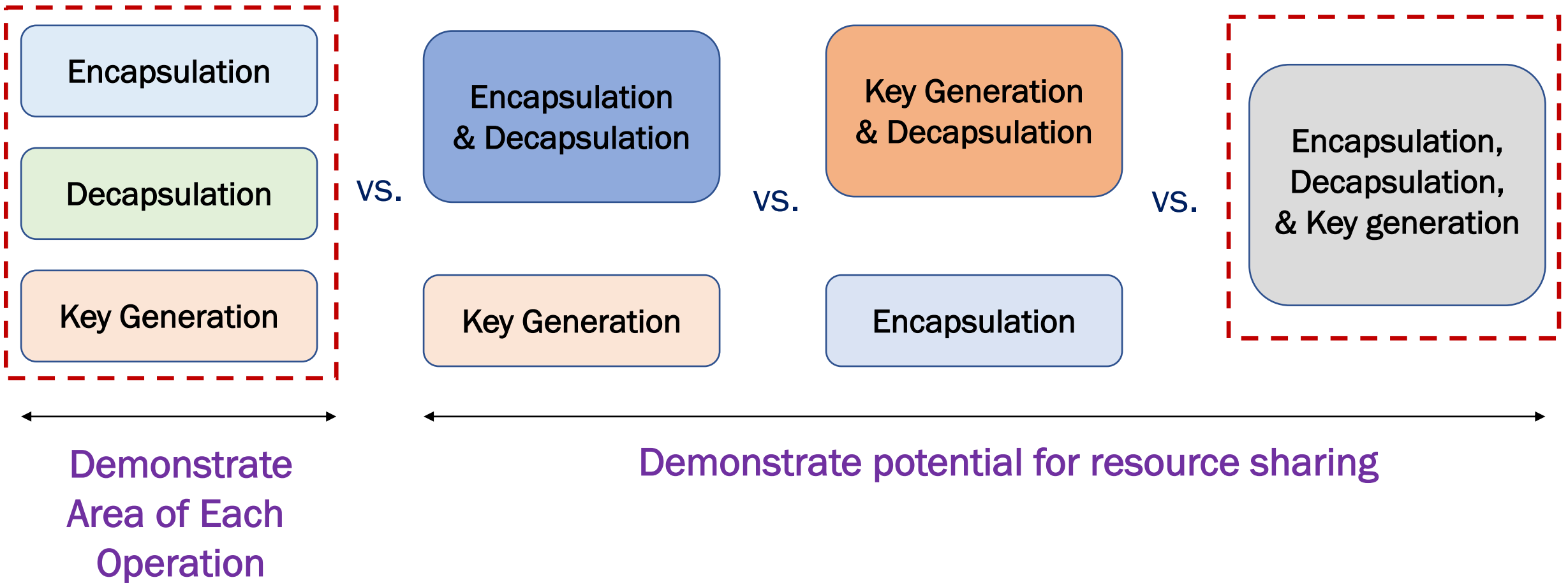
Register-Transfer Level (RTL)

- HW: VHDL, Verilog, Chisel

- Industry standard
- Highest-performance
- Best trade-offs between speed vs. area

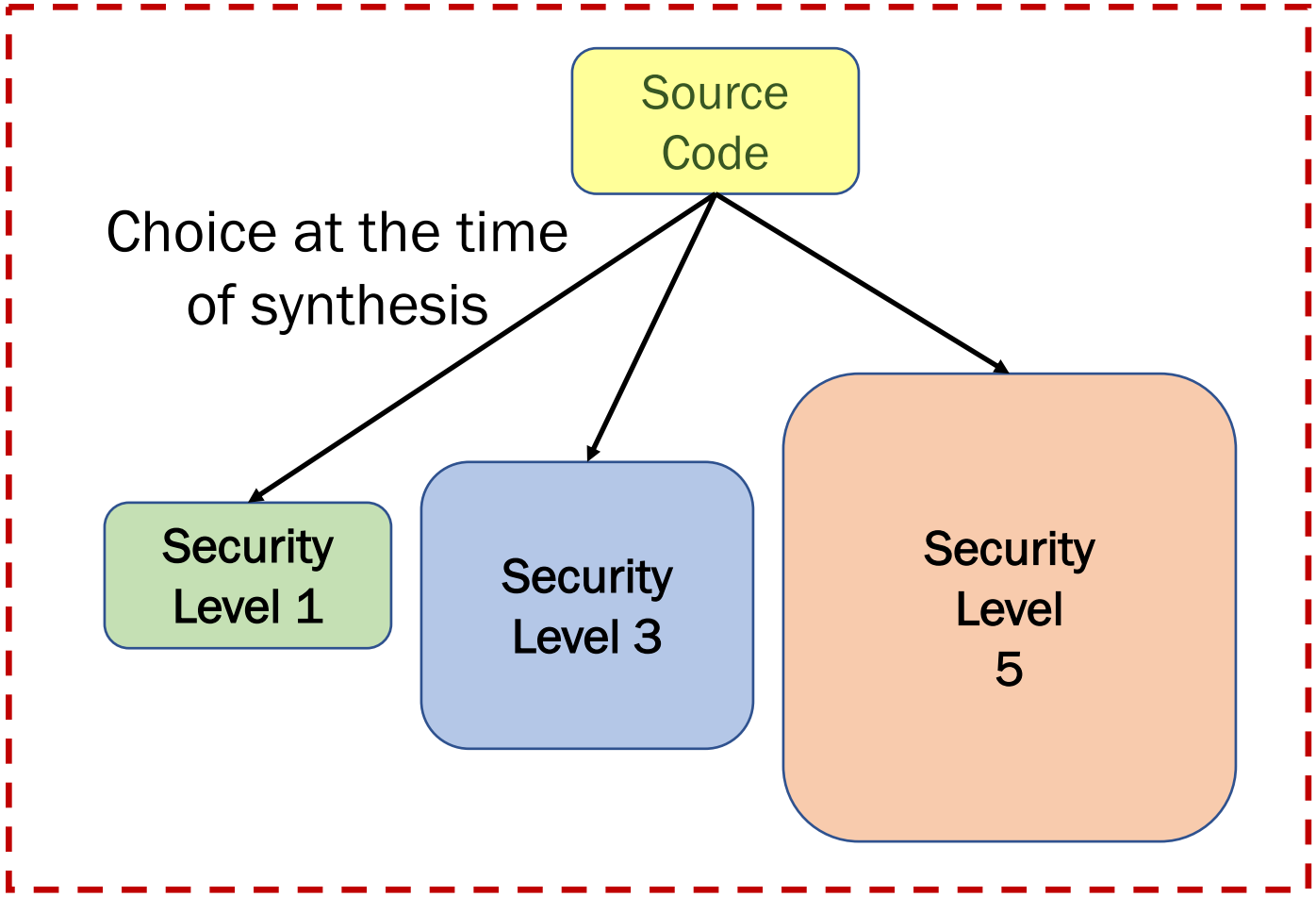
- Long development time

Operations Supported by Each Core



Each core can operate with its own maximum clock frequency

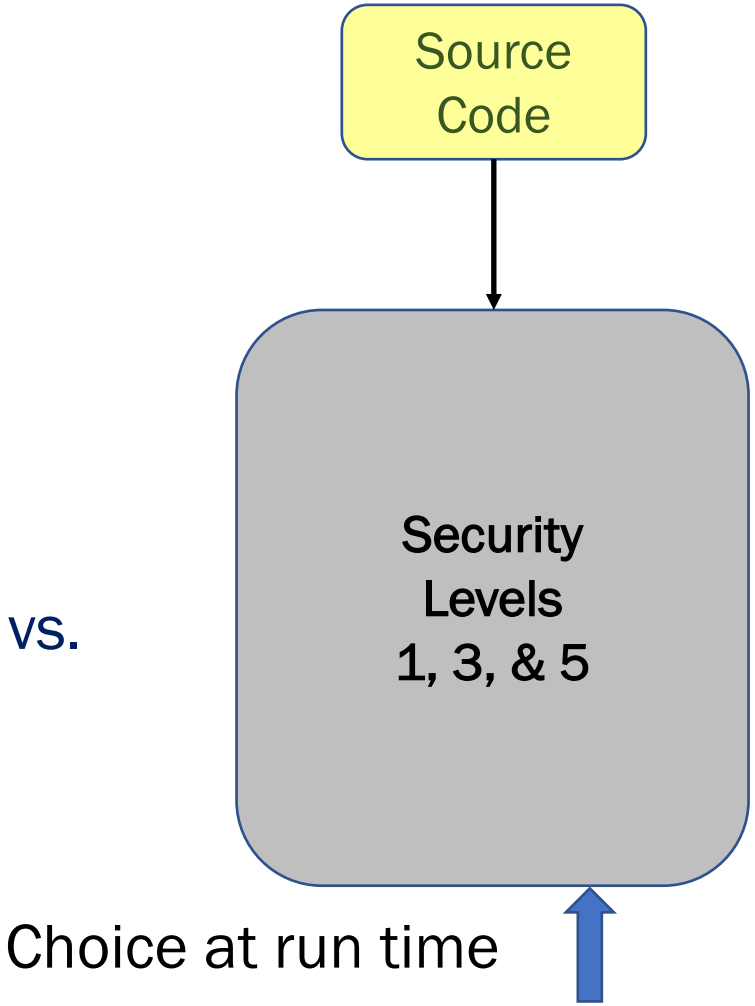
Security Levels Supported by Each Core



Multiple result sets with minimal effort

3 areas, 3 clock frequencies

vs.



1 area, 1 clock frequency

Design Space Exploration

High-speed

Primary Metrics:

- Latency
- #Operations_per_s

Secondary Metrics:

- Area
- Power

Balanced

Primary Metrics:

- Latency · Area
- #Operations_per_s / Area

Lightweight

Primary Metrics:

- Area
- Power

Secondary Metrics:

- Latency
- #Operations_per_s

FPGA Platforms & Tools

Platforms:

Artix-7:	XC7A200T-3,	28 nm technology
134,600 LUTs	365 BRAMs	740 DSPs
Zynq UltraScale+:	ZU7EV-3,	16 nm technology
230,400 LUTs	312 BRAMs	1,728 DSPs

Tools:

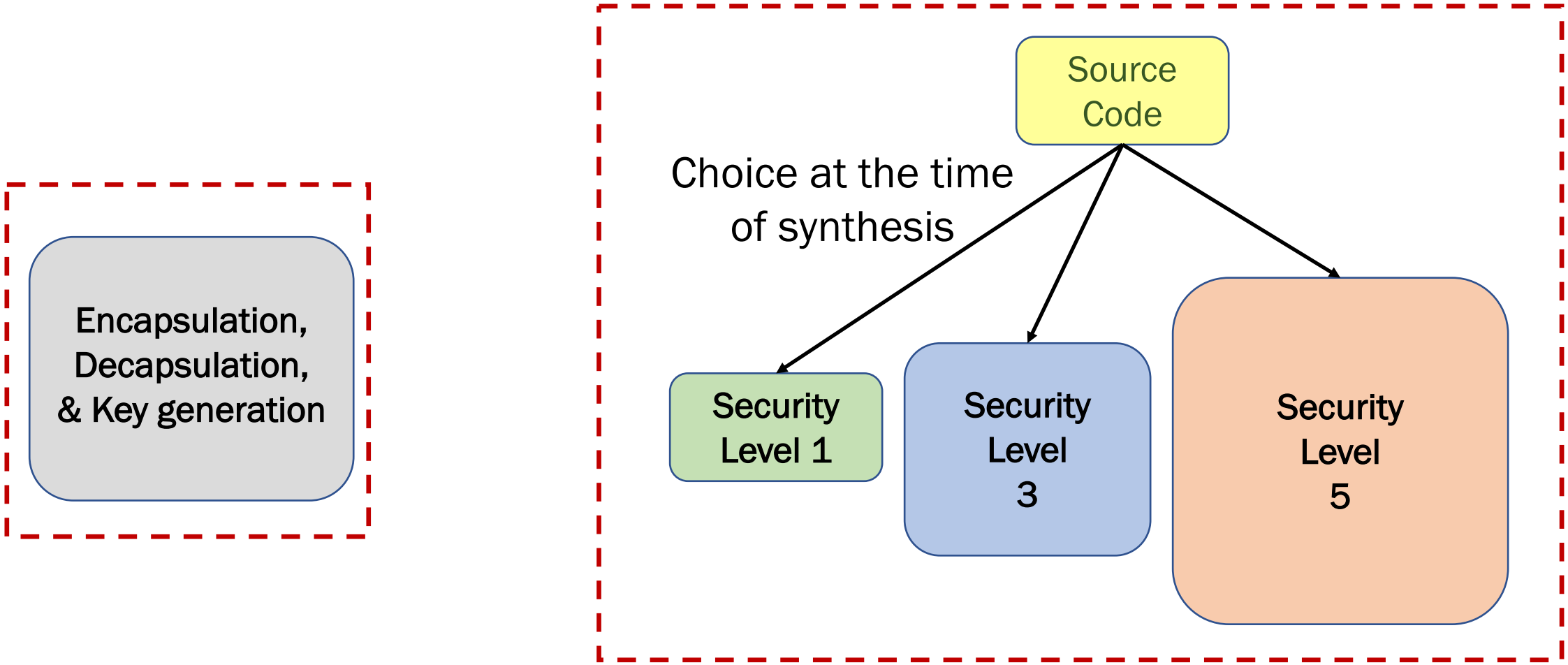
Vivado WebPack 2020.1 (free)

In PQC, the use of LUTs typically most limiting ⇒ Area represented by #LUTs

All results reported after placing & routing

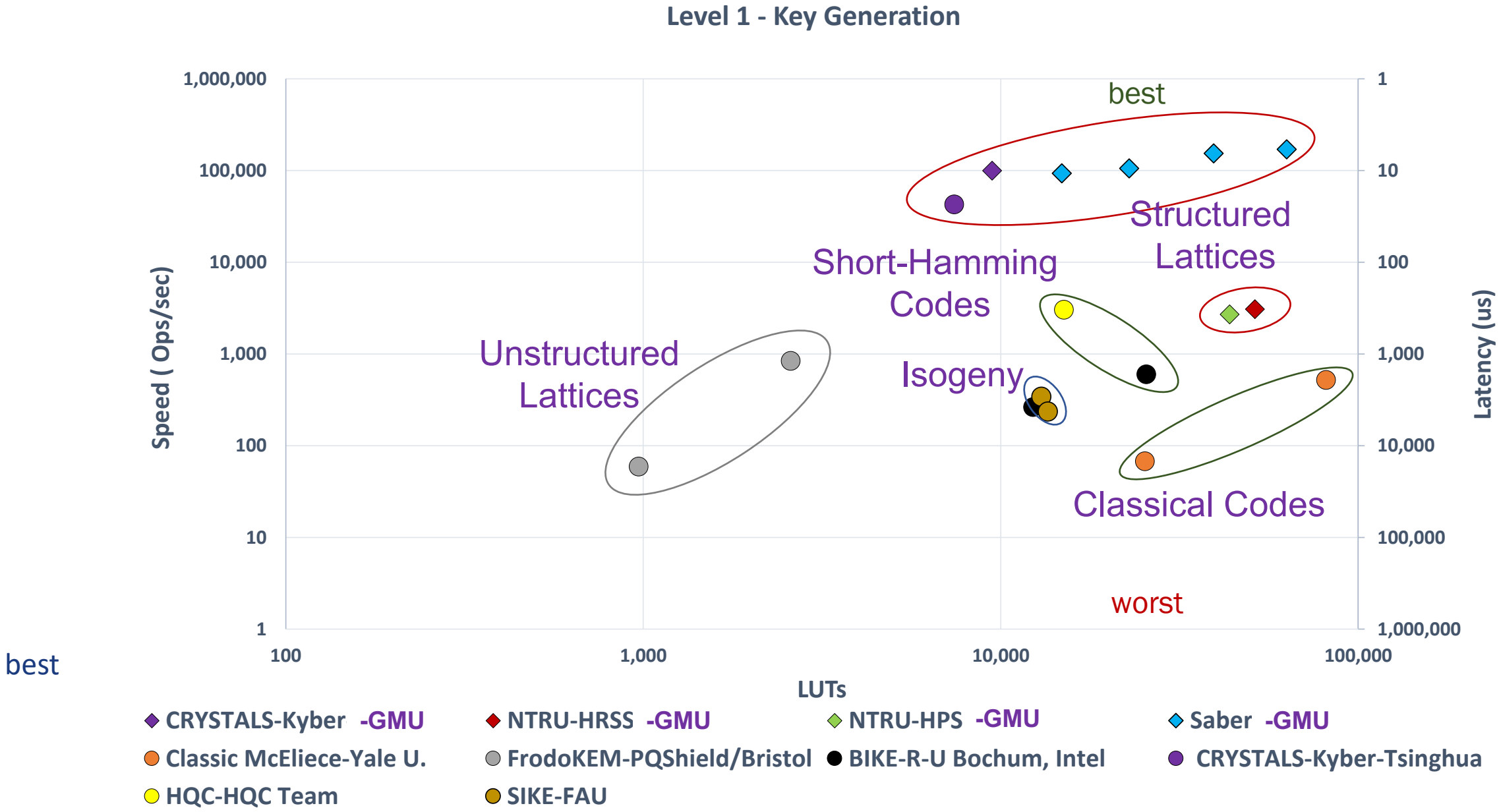
Results for KEMs in Hardware

Assumptions

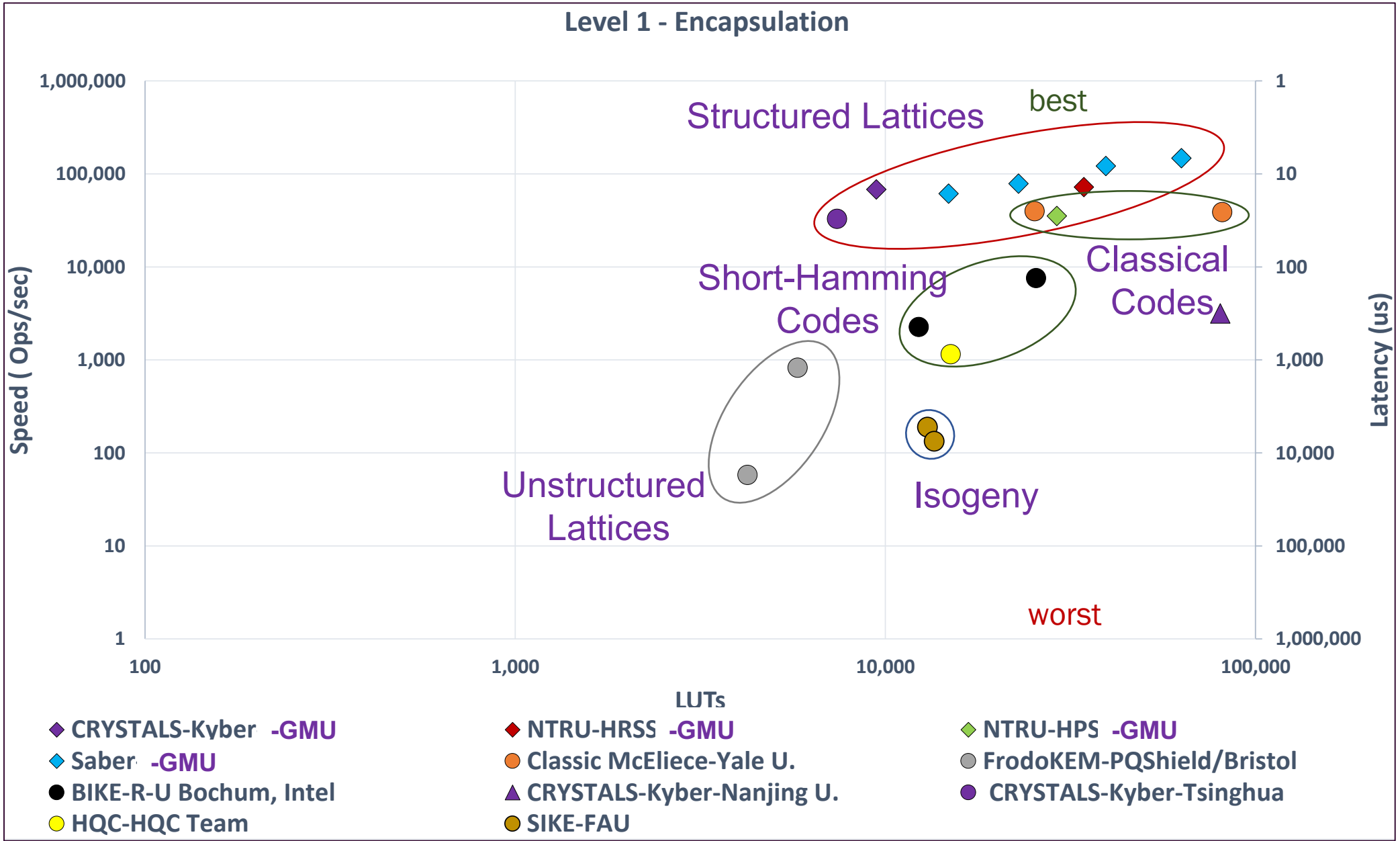


- 3 operations and 1 security level supported by each core
- 3 cores per algorithm

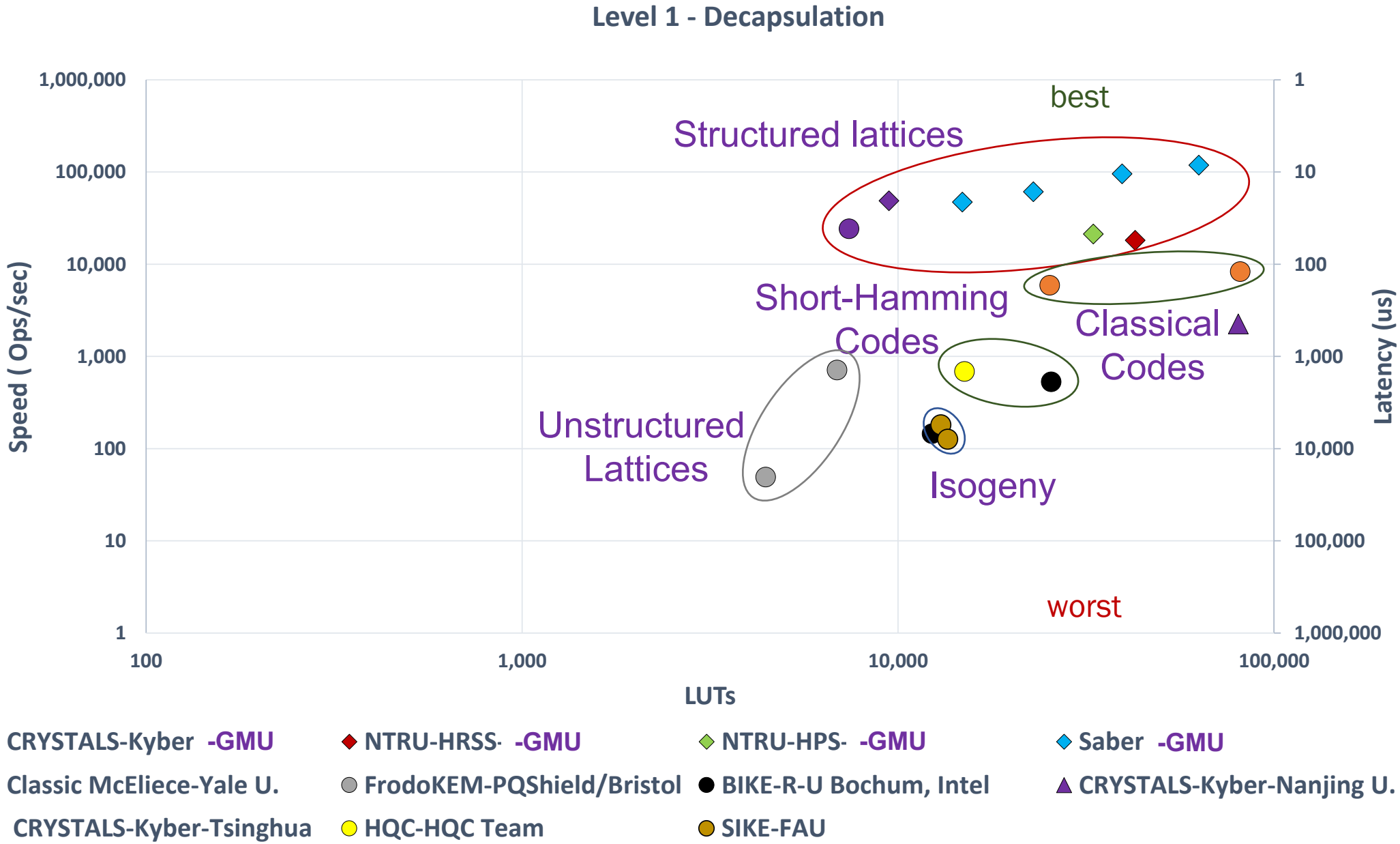
Level 1: Key Generation on Artix-7



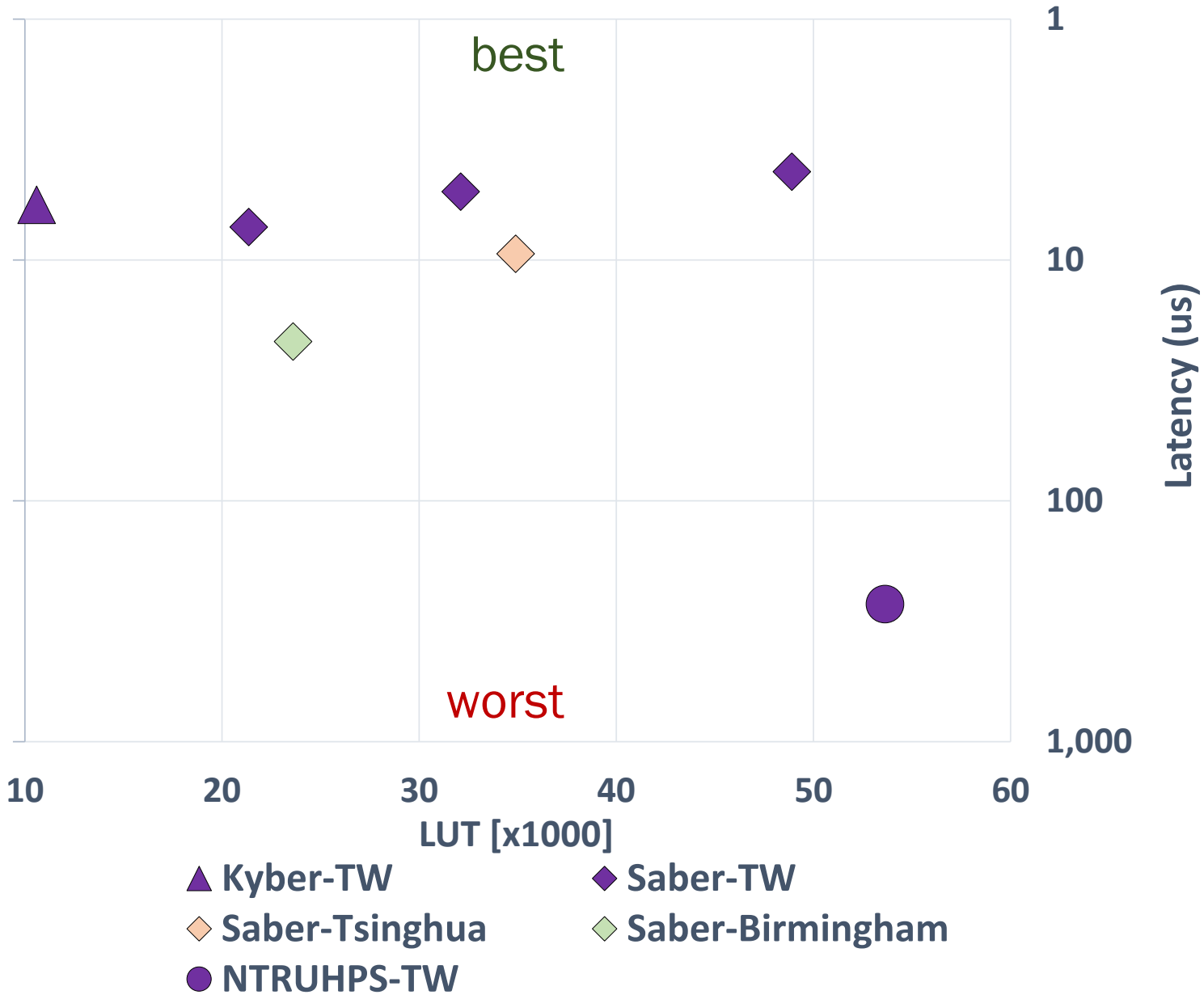
Level 1: Encapsulation on Artix-7



Level 1: Decapsulation on Artix-7

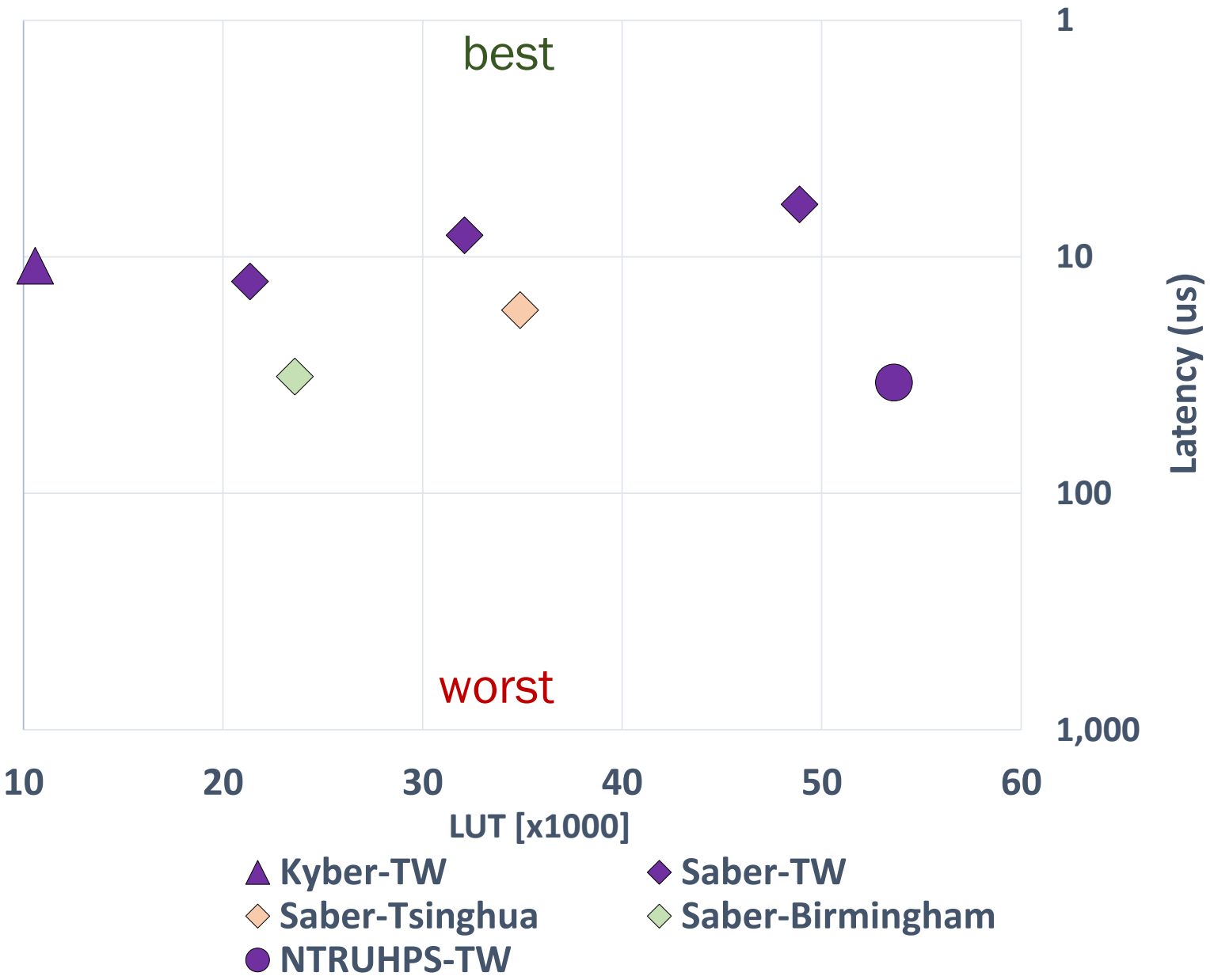


Level 3: Key Generation on Zynq UltraScale+

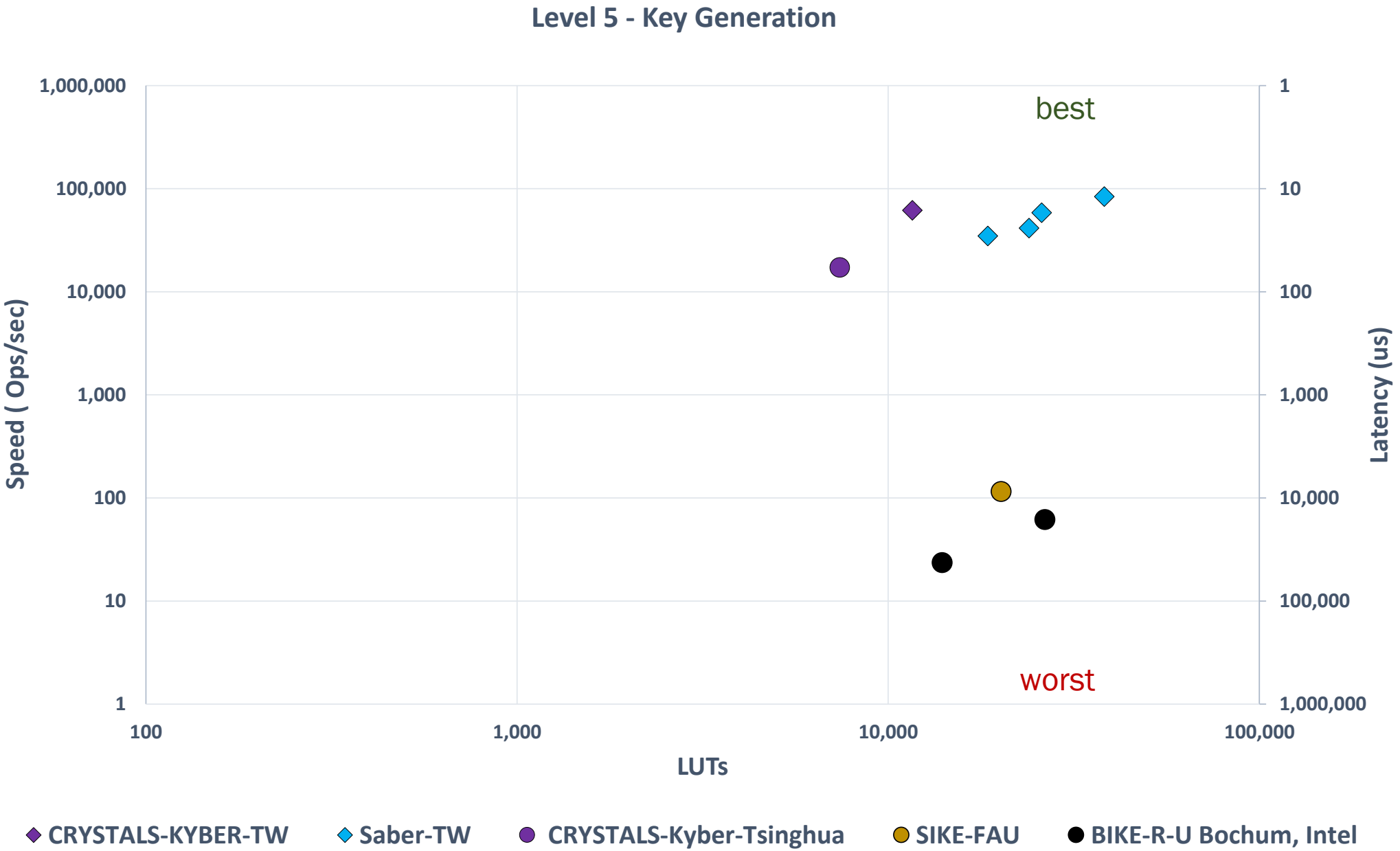


Level 3: Encapsulation on Zynq UltraScale+

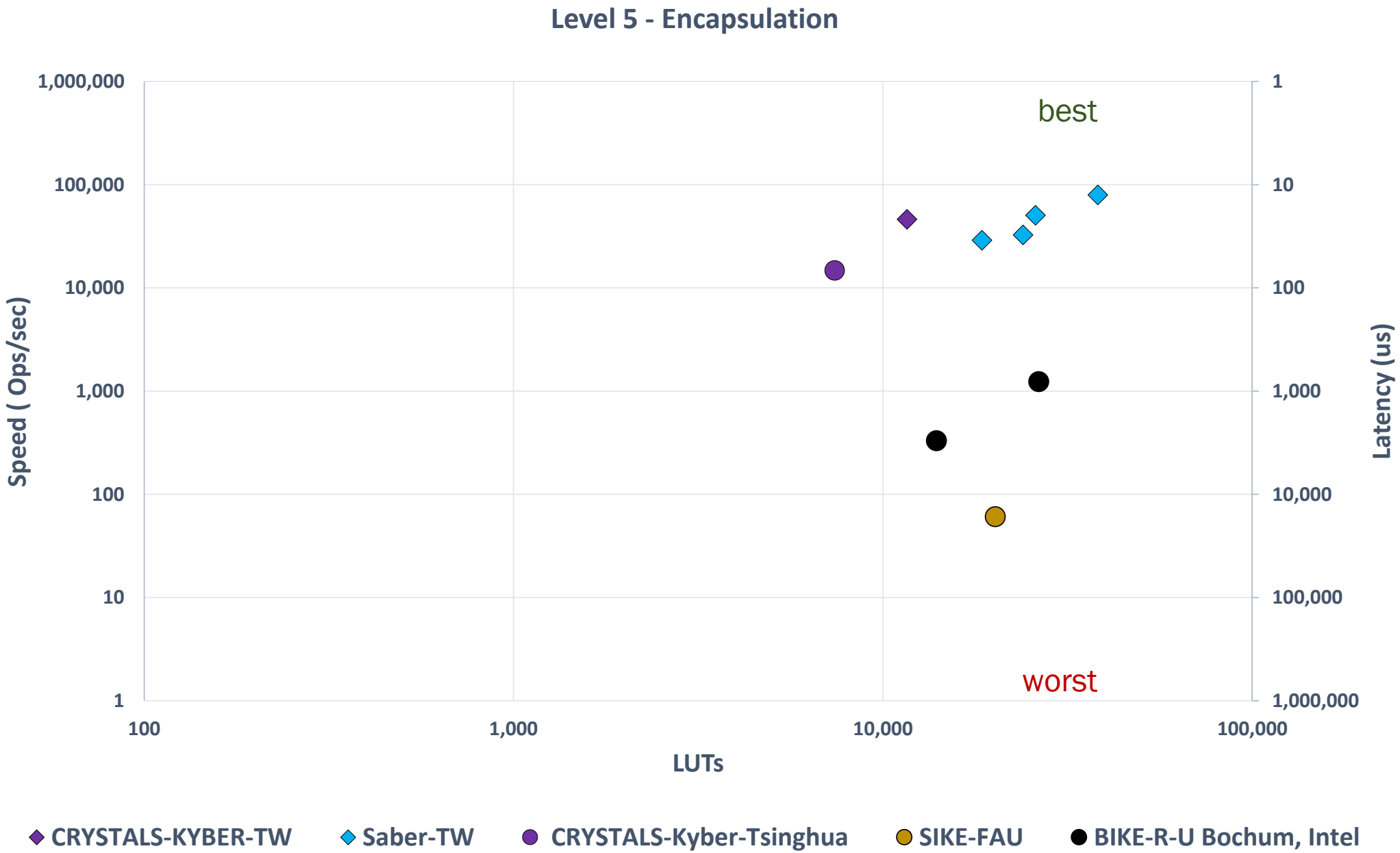
Level 3: Decapsulation on Zynq UltraScale+



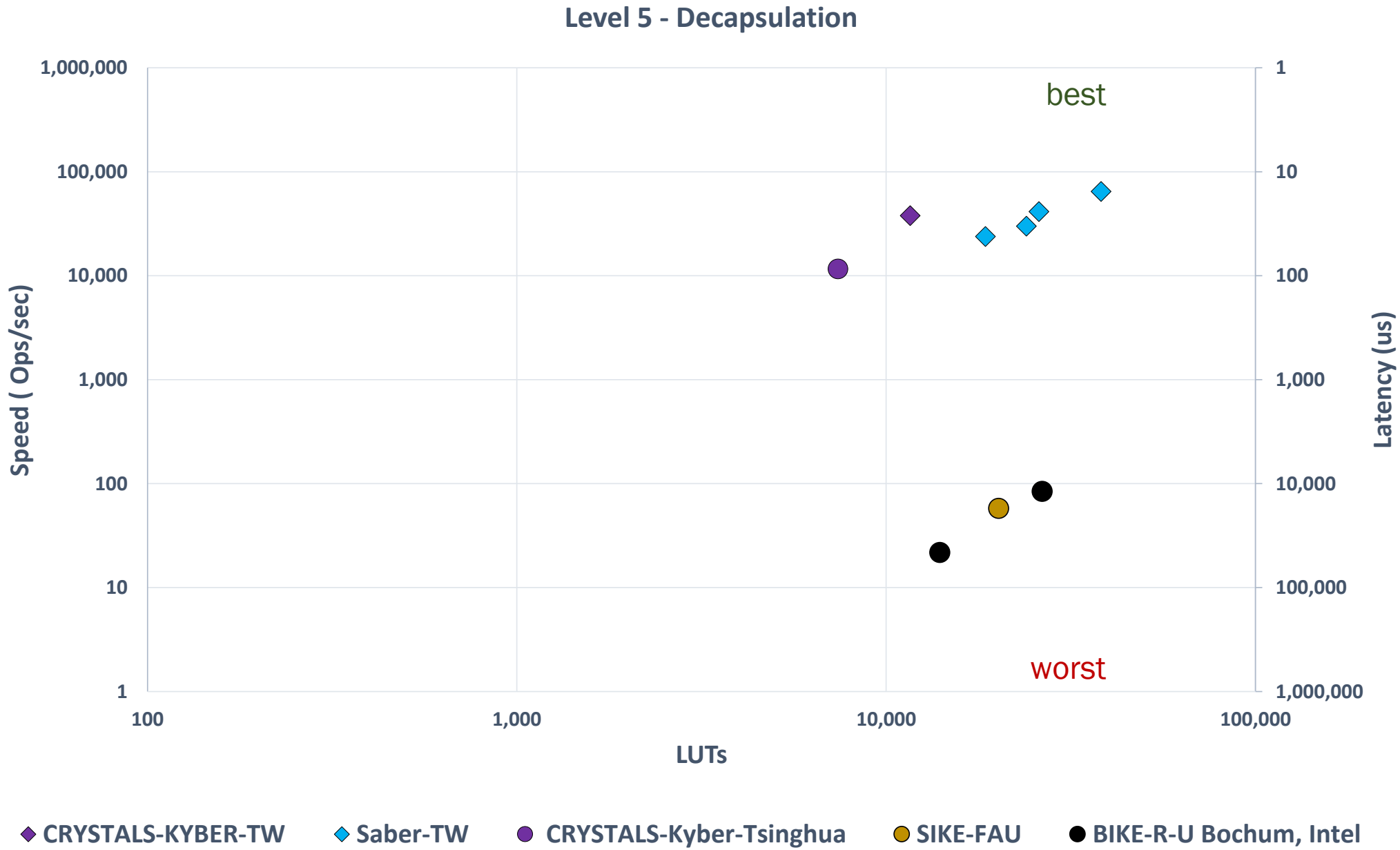
Level 5: Key Generation on Artix-7



Level 5: Encapsulation on Artix-7



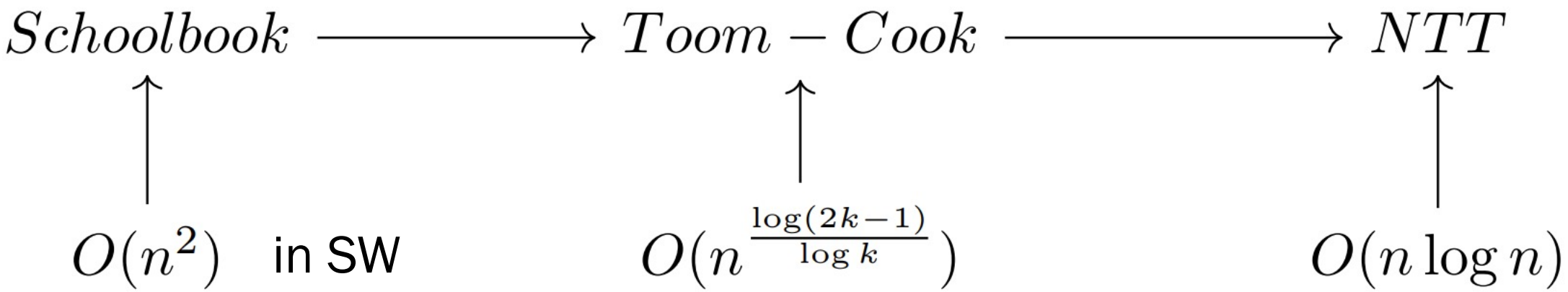
Level 5: Decapsulation on Artix-7



Design Choices

Most Commonly-Used Algorithms for Polynomial Multiplication

Number Theoretic Transform



$O(n)$ in HW

Typically:

k=2: Karatsuba : $O(n^{1.58})$

k=3: Toom-3 : $O(n^{1.46})$

k=4: Toom-4 : $O(n^{1.40})$

Choice of a Polynomial Multiplier

	Small Coefficient Range	Number of coefficients	NTT friendly ring	One Operand in NTT domain	"Small" × "Large" Polynomial Multiplication in KeyGen/Encaps/Decaps	"Large" × "Large" Polynomial Multiplication in KeyGen/Encaps/Decaps
ntruhrss701	[-1..1]	701	N	N	5/1/3	8* /–/1
ntruhps2048677		677			5/1/3	8* /–/1
ntruhps4096821		821			5/1/3	8* /–/1
Kyber512	[-3..3], [-2..2]	256	Y	Y	4/6/8	–/–/–
Kyber768	[-2..2]				9/12/15	–/–/–
Kyber1024	[-2..2]				16/20/24	–/–/–
LightSaber-KEM	[-5..5]	256	N	N	4/6/8	–/–/–
Saber-KEM	[-4..4]				9/12/15	–/–/–
FireSaber-KEM	[-3..3]				16/20/24	–/–/–

* Part of polynomial inversion

Choice of a Polynomial Multiplier

CRYSTALS-Kyber

"Small" x "Large"

k x NTT-based

k= 2, 3, 4
for Security Levels 1, 3, 5

+ Karatsuba
during pointwise multiplication

NTRU

"Large" x "Large"

Toom-Cook

Toom-3 + Karatsuba

Based on 15·d DSP units

d= 2, 3

"Small" x "Large"

Schoolbook

when one polynomial ternary, i.e., w/ coefficients {-1, 0, 1}

Saber

"Small" x "Large"

Schoolbook

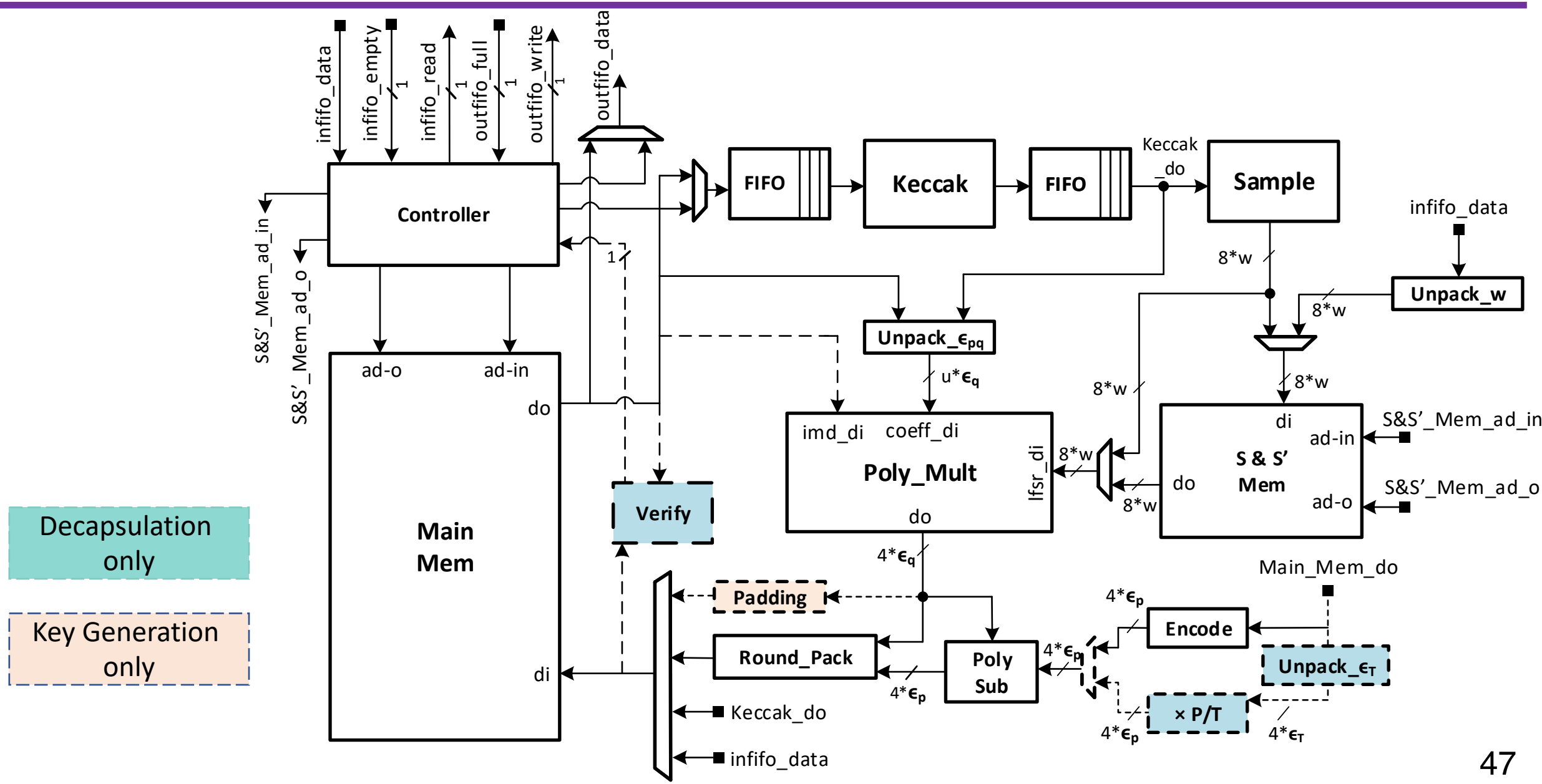
u – unrolling factor
(#coefficients of B multiplied by A)

u= 1, 2, 4

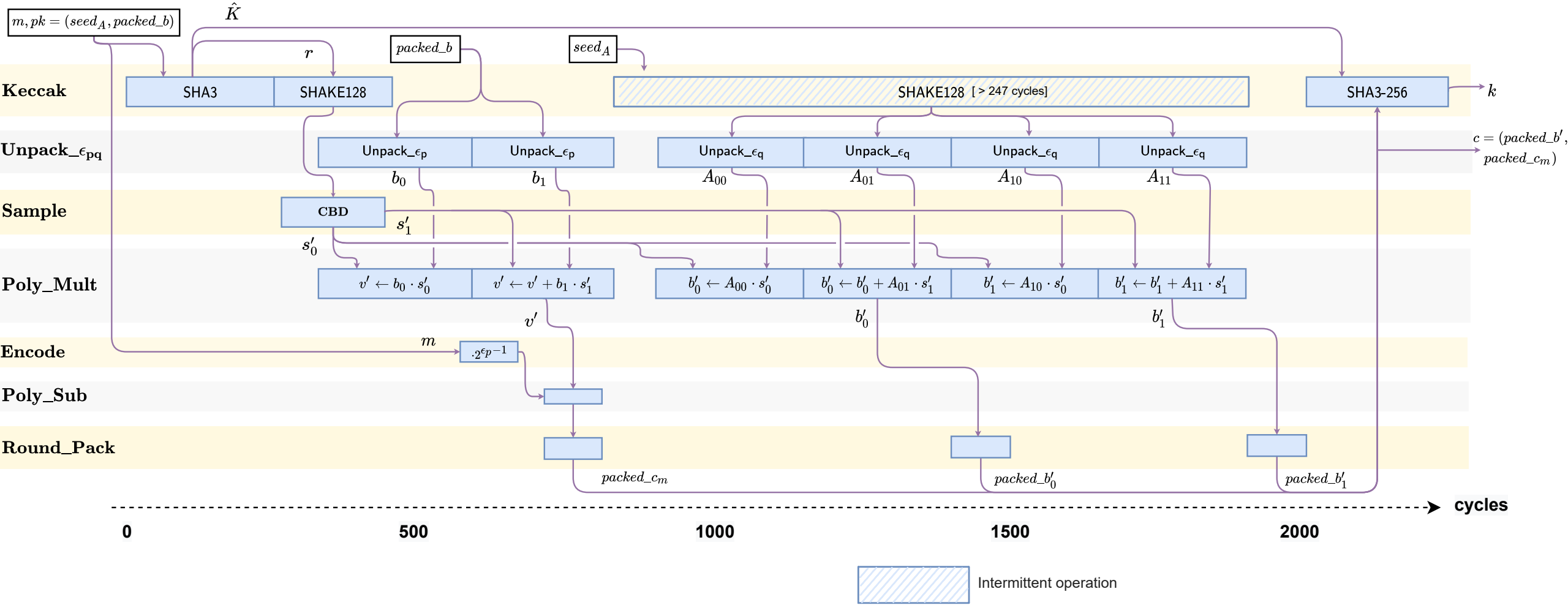
or

NTT-based

Example of a Block Diagram: Saber

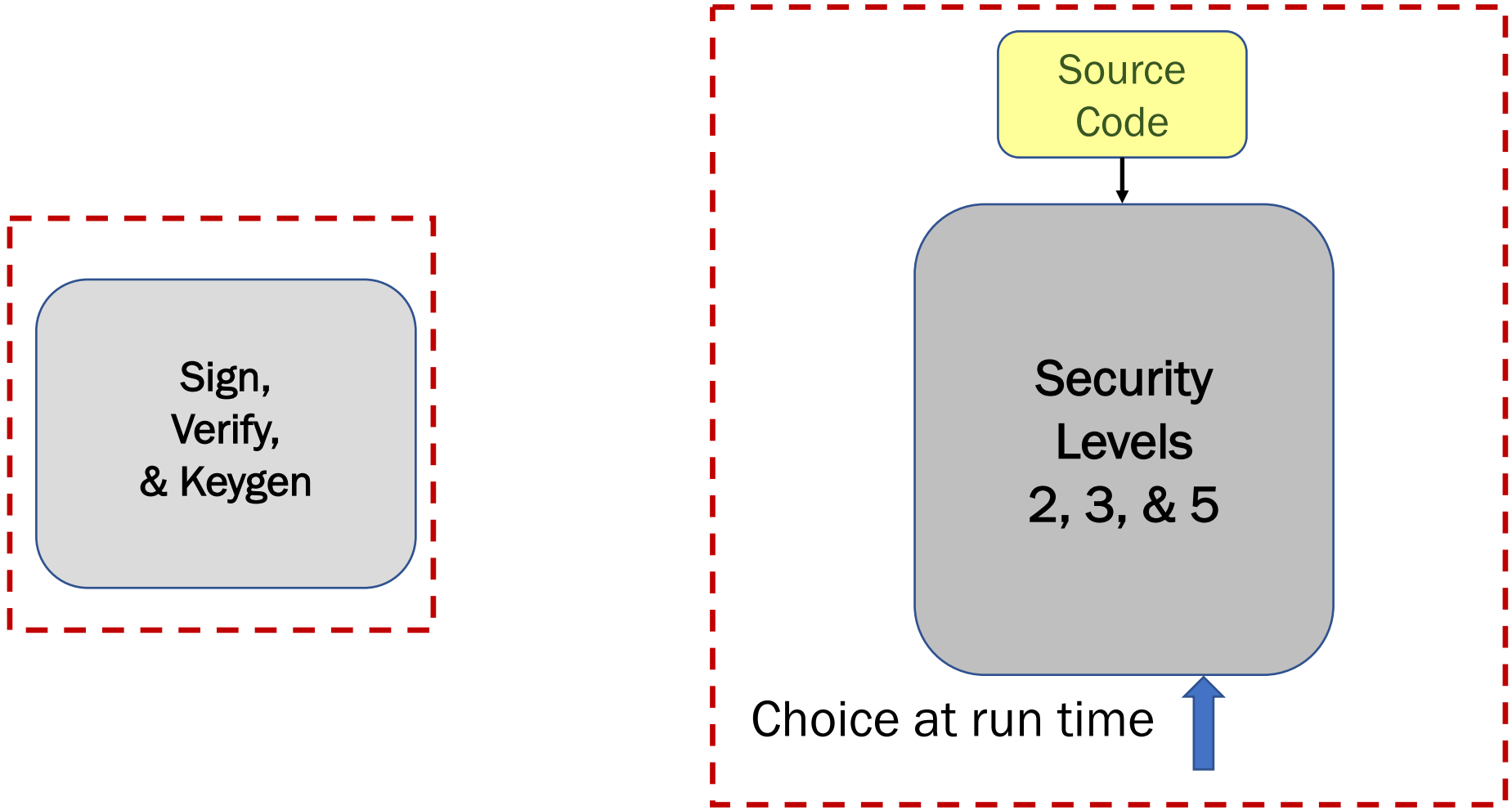


Example of Scheduling Diagram: Saber Encapsulation



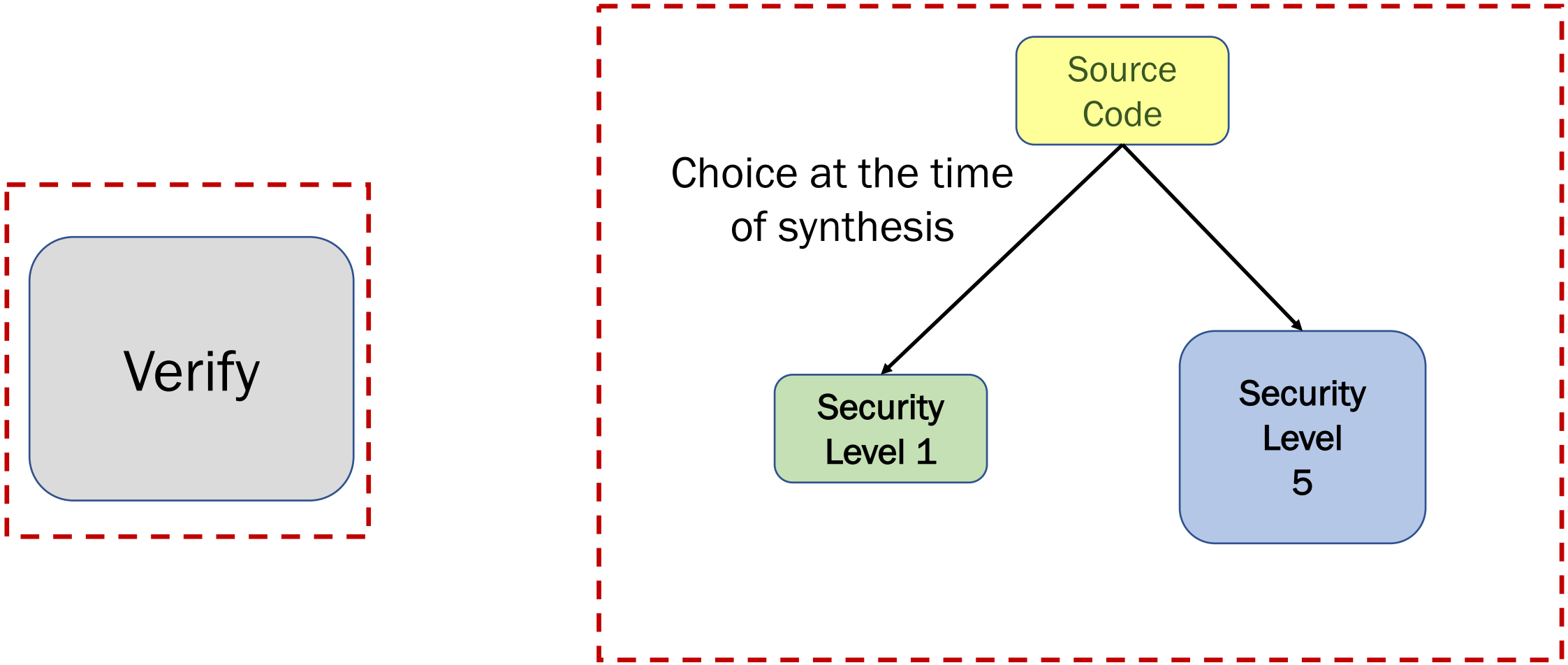
Results for Digital Signatures in Hardware

Assumptions for CRYSTALS-Dilithium



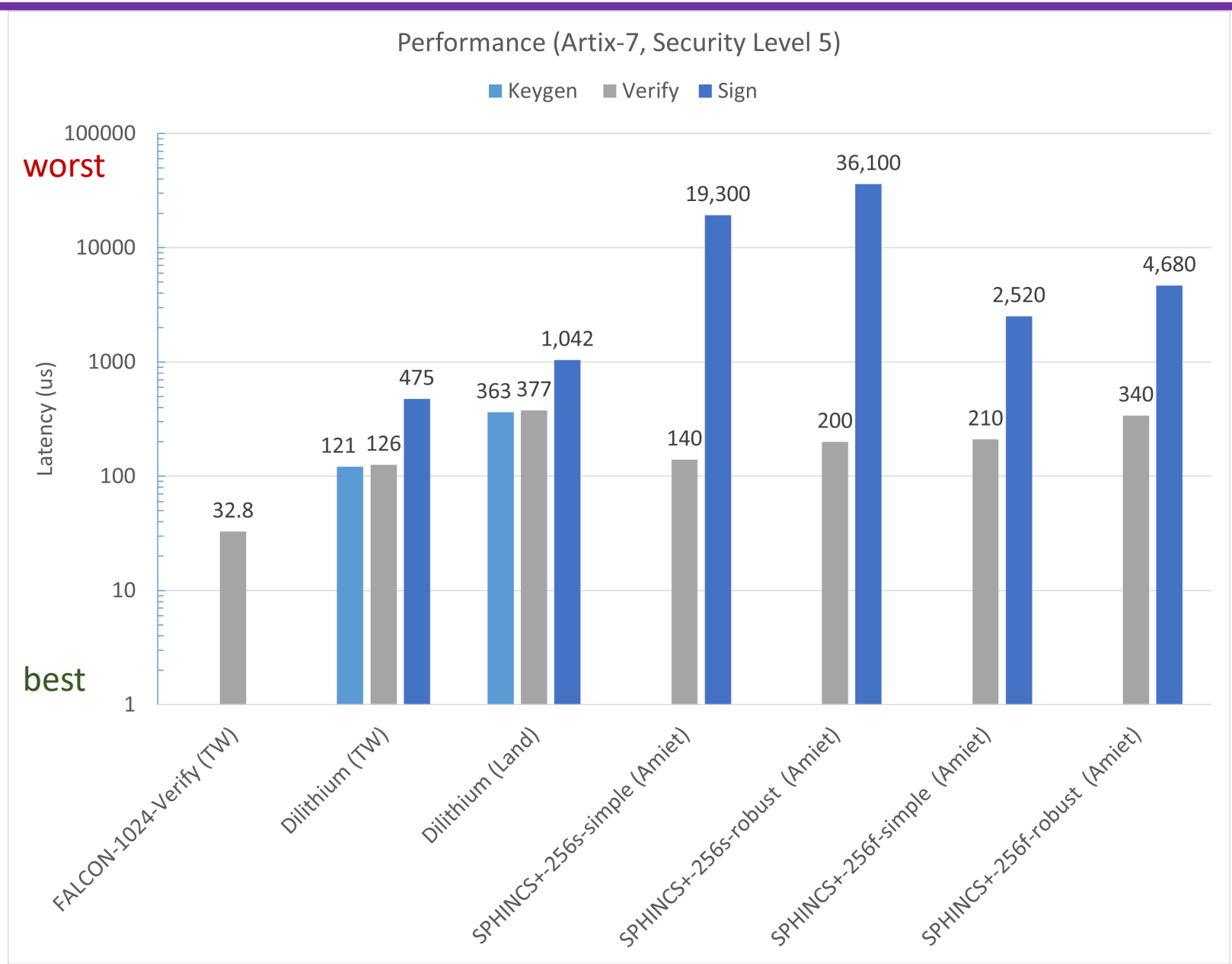
- 3 operations and 3 security levels supported by each core
- 1 core per algorithm

Assumptions for Falcon



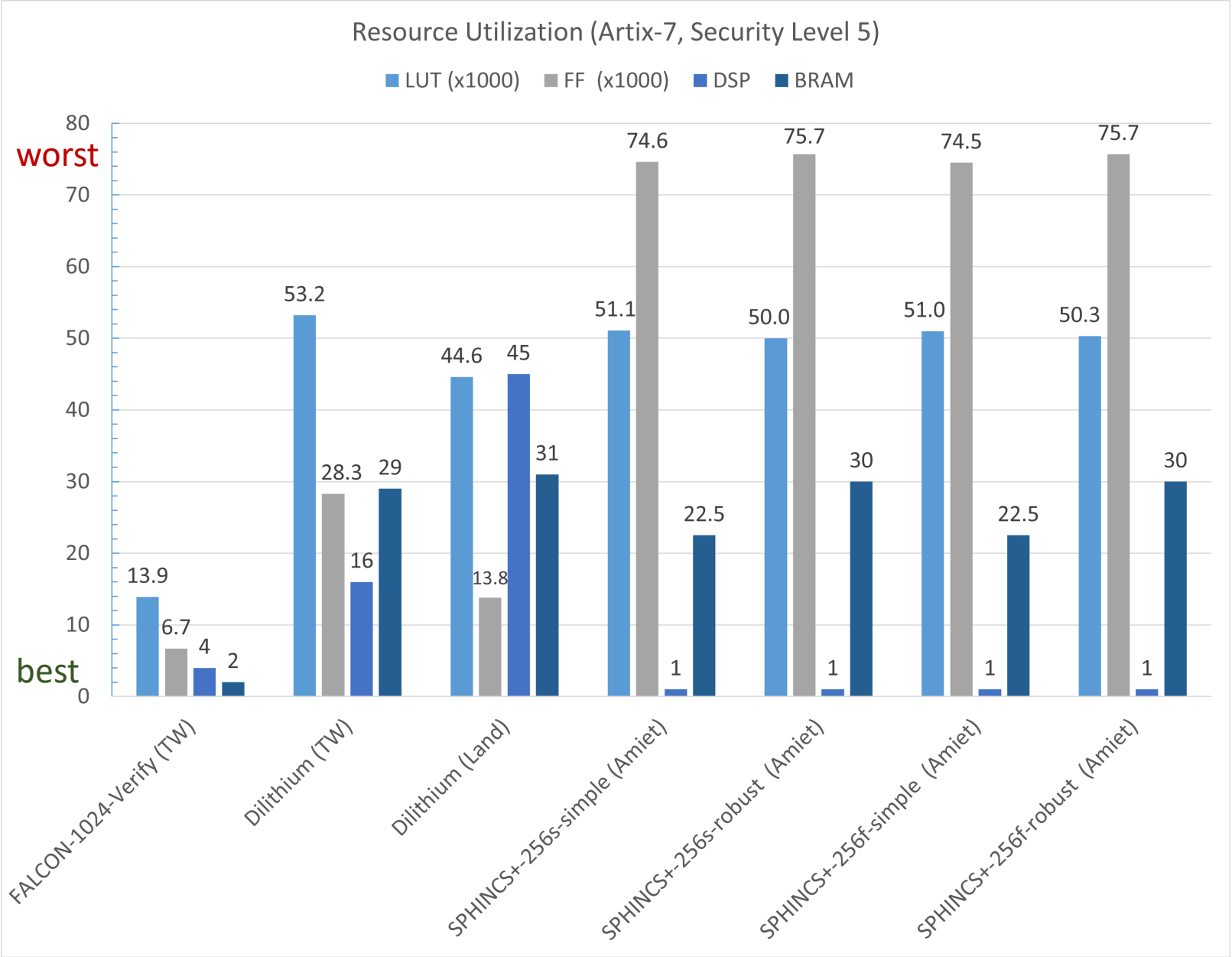
- 1 operation and 1 security level supported by each core
- 2 cores per algorithm

Level 5: All Operations on Artix-7: Latency



TW– This Work

Level 5: All Operations on Artix-7: Resource Utilization

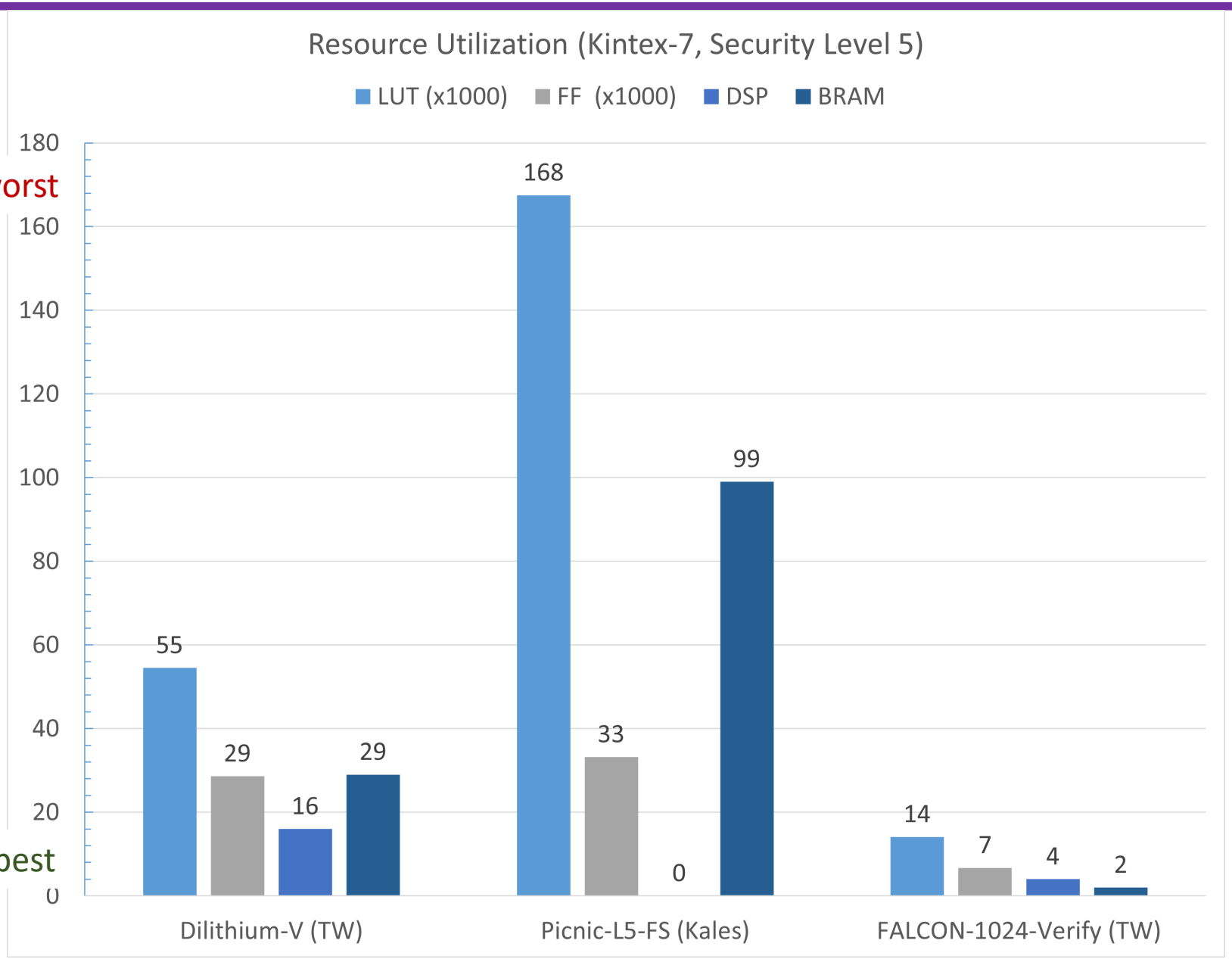


TW– This Work

Level 5: All Operations on Kintex-7: Latency

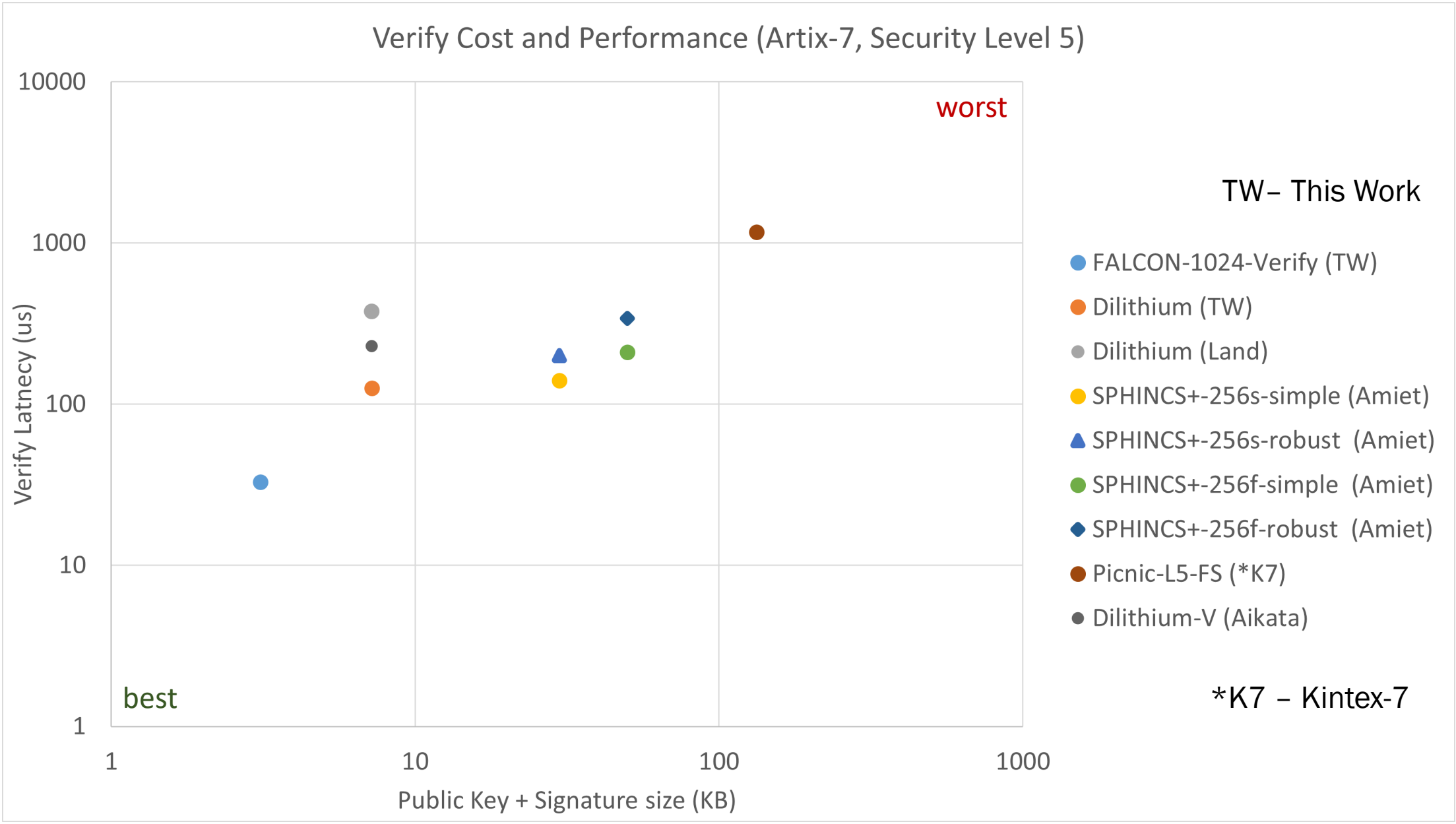
TW– This Work

Level 5: All Operations on Kintex-7: Resource Utilization



TW– This Work

Level 5: Signature Verification: Artix-7: Latency vs. Certificate Size



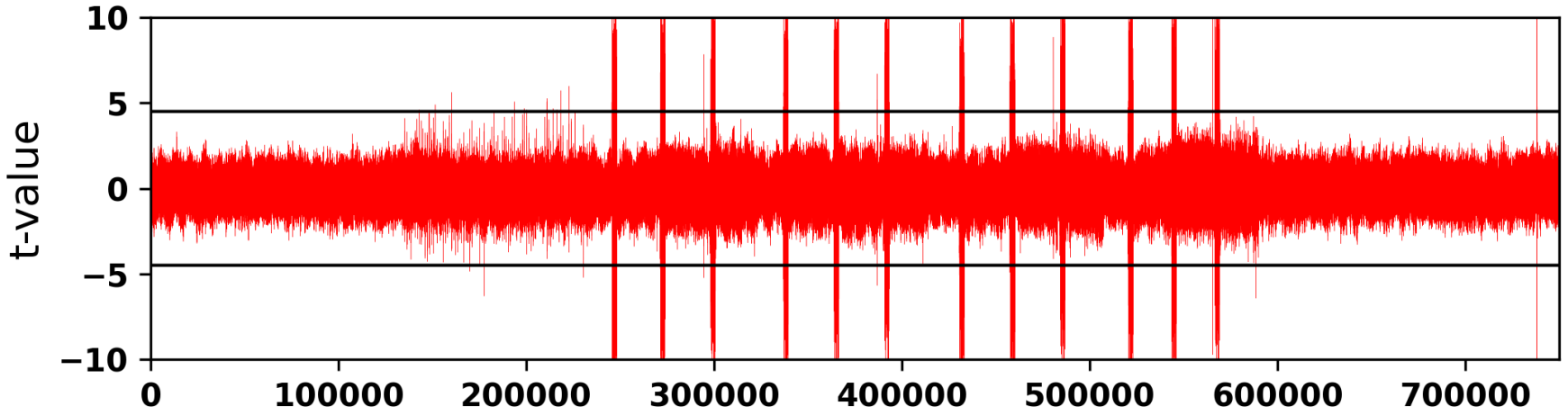
Results for the Lightweight Implementation Resistant Against SCA

Approach

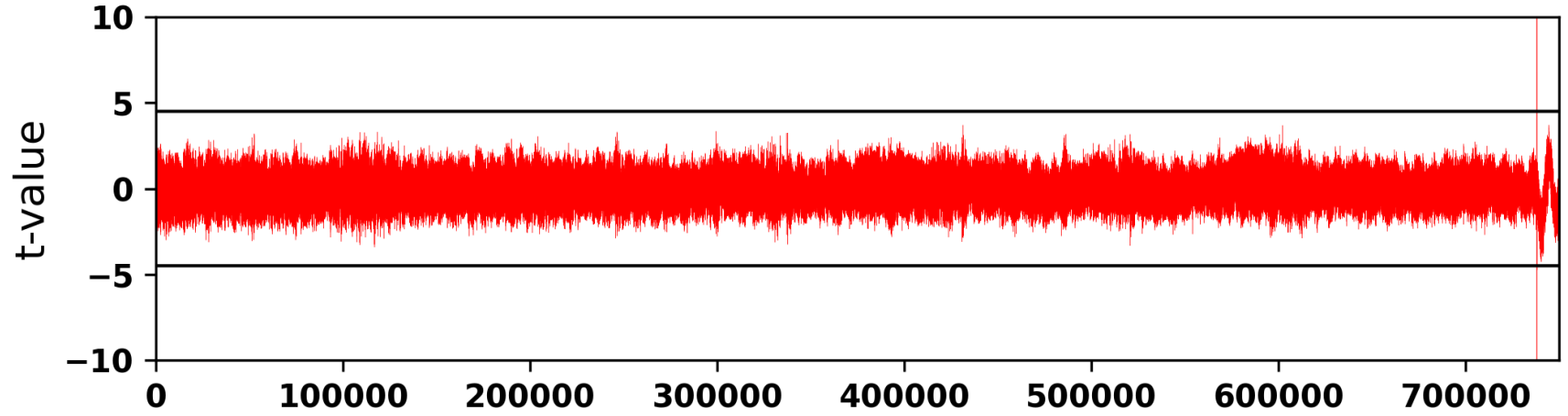
- Lightweight unprotected implementation of Saber
- Protected implementation based on arithmetic and Boolean masking
  - $X = X0 \text{ xor } X1 \longrightarrow$ Boolean
  - $X = X0 + X1 \bmod q \longrightarrow$ Arithmetic
- Arithmetic shares for polynomial arithmetic, Boolean in SHA-3
- Partially based on the protected software implementation by Beirendonck et al., 2020

Experimental Verification Using Test-Vector Leakage Assessment

Unprotected



Protected



Overhead of the GMU Protected Implementation of Saber

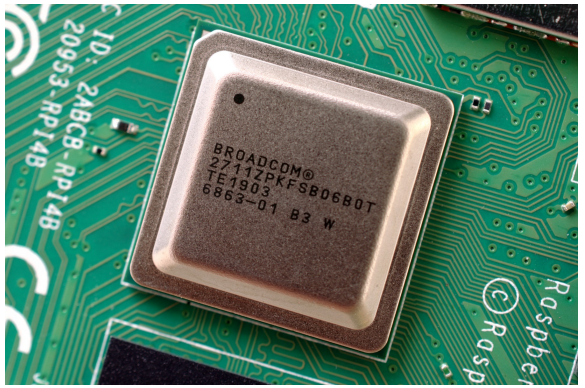
- Clock cycles for decapsulation:
  52,758 → 72,005 [x 1.36]
- #LUTs:
  6,713 → 19,299 [x 2.87]
- #DSPs:
  32 → 64 [x 2.00]

NEON-Based Software Implementations

NEON

- NEON is an alternative name for ASIMD - Advanced Single Instruction Multiple Data extension to the ARM Instruction Set Architecture, mandatory since ARMv7-A.
- NEON provides 32x**128-bit** vector registers. Compared with Single Instruction Single Data (SISD), NEON can have ideal speed-up in the range 2..16 (for 64..8-bit operands).

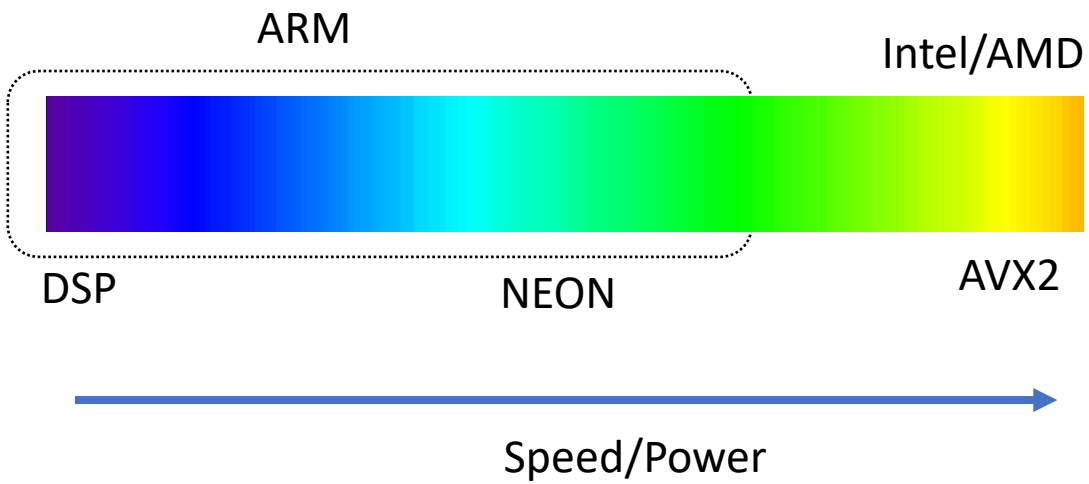
Firestorm core of Apple M1:
part of new MacBook Air, MacBook Pro,
Mac Mini, iMac, and iPad Pro



Cortex-A72 of Broadcom SoC, BCM2711:
part of the Raspberry Pi 4
single-board computer

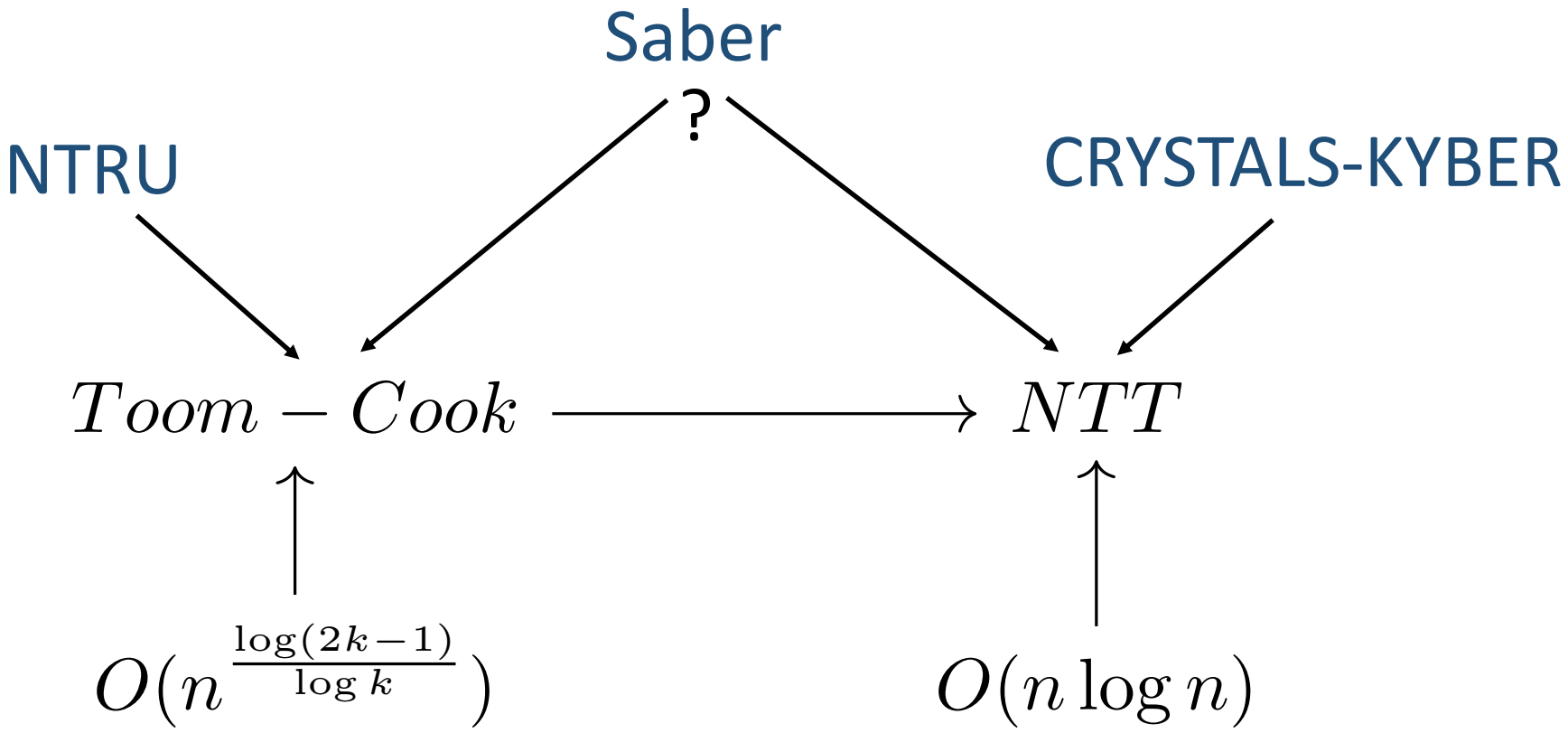
NEON Project Goals

- Most software implementations of PQC candidates on:
  - Intel/AMD (w/ AVX2 extension)
  - ARM Cortex-M4 (w/ DSP extension)



- We developed constant-time, optimized ARMv8 implementations of 3 KEM finalists:
  - CRYSTALS-Kyber
  - NTRU
  - Saber

Optimal Choice of Algorithms



Based on the analysis of algorithms, their parameters, and AVX2 implementations for the 3 lattice-based KEMs finalists

NEON Benchmarking Methodology

Apple M1 System on Chip	Firestorm core, 3.2 GHz[1], MacBook Air
Broadcom BCM2711 System on Chip	**Cortex-A72** core, 1.5 GHz, Raspberry Pi 4
Operating System	MacOS 11.4, Arch Linux (March 25, 2021)
Compiler	clang 12.0 (MacBook Air), clang 11.1 (Raspberry Pi 4)
Compiler Options	-O3 -mtune=native -fomit-frame-pointer
Cycles count on Cortex-A72	PAPI[2]
Cycles count on Apple M1	Modified[3] from Dougall Johnson's work[4]
Iterations	10,000,000 on Apple M1 to force CPU to run on high-performance "FireStorm" core; 1,000,000 otherwise

[1] https://www.anandtech.com/show/16252/mac-mini-apple-m1-tested

[2] D. Terpstra, H. Jagode, H. You, and J. Dongarra, "Collecting Performance Data with PAPI-C," in Tools for High Performance Computing, 2009

[3] https://github.com/GMUCERG/PQC_NEON/blob/main/neon/kyber/m1cycles.c

[4] https://github.com/dougallj

NTT vs. Toom-Cook for Saber

All values in cycles

Apple M1 3.2 GHz	Encap			Decap		
	Toom	NTT	Toom/NTT	Toom	NTT	Toom/NTT
lightsaber	37,187	35,949	103%	35,318	34,142	103%
saber	59,838	55,892	107%	57,955	54,117	107%
firesaber	87,899	82,776	106%	86,724	81,983	106%

Cortex-A72 1.5 GHz	Encap			Decap		
	Toom	NTT	Toom/NTT	Toom	NTT	Toom/NTT
lightsaber	130,097	116,105	112%	131,187	115,859	113%
saber	213,574	183,230	116%	215,364	183,208	117%
firesaber	321,637	265,626	121%	329,566	270,989	121%

On Apple M1, NTT better by 3-7%

On Cortex-A72, NTT better by 12-21%

Ranking for NEON Implementations

Rank	neon Cortex-A72 E	kc	↑	D	kc	↑
1	ntru-hrss701	93.6	1.00	kyber512	94.1	1.00
2	kyber512	95.3	1.02	lightsaber	131.2	1.39
3	lightsaber	130.1	1.39	ntru-hps677	205.8	2.19
4	ntru-hps677	181.7	1.94	ntru-hrss701	262.9	2.79
1	kyber768	151.0	1.00	kyber768	149.8	1.00
2	saber	213.6	1.41	saber	215.4	1.44
3	ntru-hps821	232.6	1.54	ntru-hps821	274.5	1.83
1	kyber1024	223.8	1.00	kyber1024	220.7	1.00
2	firesaber	321.6	1.44	firesaber	329.6	1.49

Rank	neon Apple M1 E	kc	↑	D	kc	↑
1	ntru-hrss701	22.7	1.00	kyber512	29.4	1.00
2	kyber512	32.5	1.43	lightsaber	35.3	1.20
3	lightsaber	37.2	1.63	ntru-hps677	54.5	1.85
4	ntru-hps677	60.1	2.64	ntru-hrss701	60.7	2.06
1	kyber768	49.2	1.00	kyber768	45.7	1.00
2	saber	59.9	1.22	saber	58.0	1.27
3	ntru-hps821	75.7	1.54	ntru-hps821	69.0	1.51
1	kyber1024	71.6	1.00	kyber1024	67.1	1.00
2	firesaber	87.9	1.23	firesaber	86.7	1.29

Decapsulation ranking of NEON implementations at Levels 1, 3 and 5

Encapsulation ranking of NEON implementations at Level 3 and 5:

1. CRYSTALS-Kyber
2. Saber [1.27-1.49 slower]
3. NTRU (Levels 1 & 3 only) [1.51-1.83 slower]

Consistent between Cortex-A72 and Apple M1.

Exception: Encapsulation at Level 1

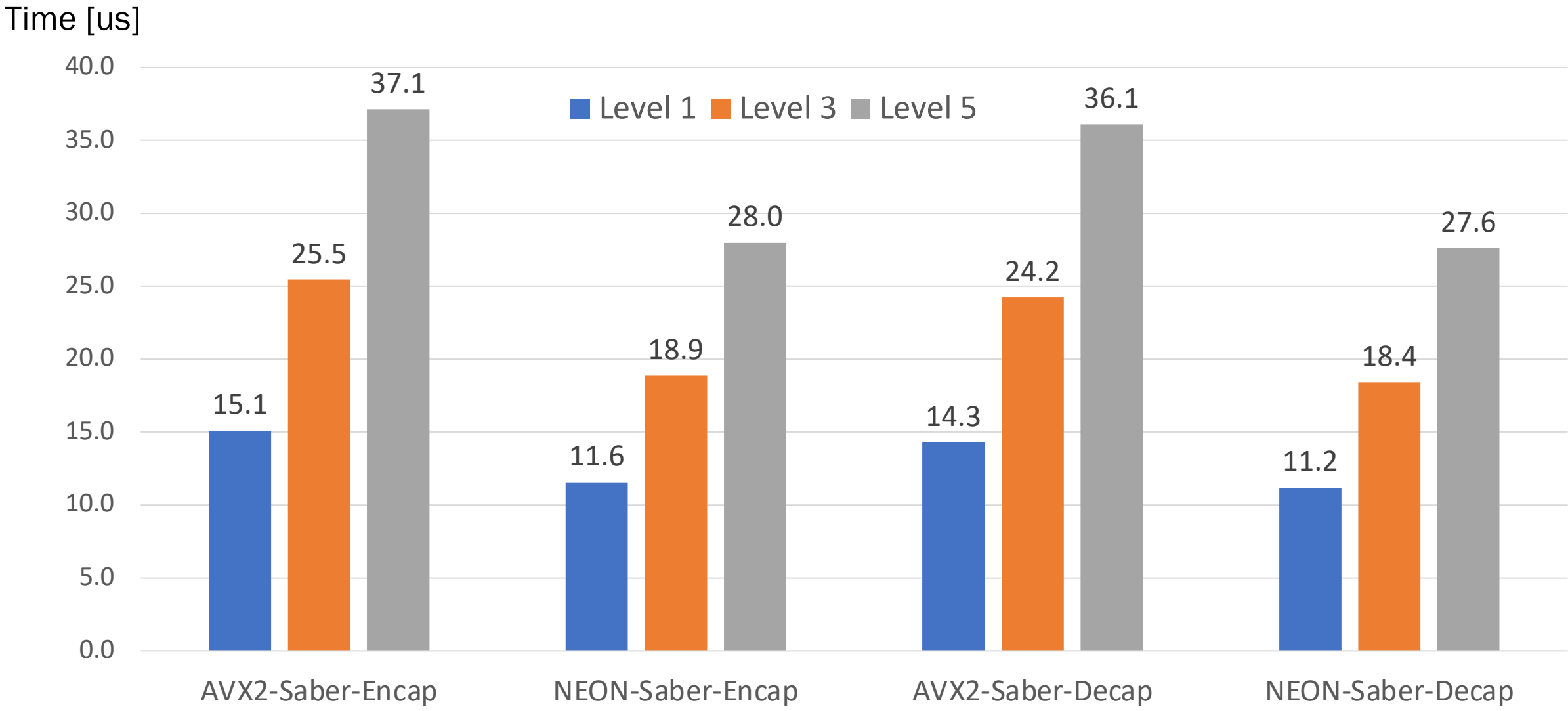
1. NTRU
2. CRYSTALS-Kyber [1.02-1.43 slower]
3. Saber [1.39-1.63 slower]

Apple M1 w/NEON @ 3.2 GHz vs. Intel Core i7-8750H w/AVX2 4.1 GHz

Apple M1 / Core i7-8750H	ref (*kc*)		neon (*kc*)		AVX2 (*kc*)		ref/neon		AVX2/neon	
	E	**D**	**E**	**D**	**E**	**D**	**E**	**D**	**E**	**D**
NTRU-HPS677	183.1	430.4	60.1	54.6	47.6	32.5	3.05	7.89	0.79	0.60
NTRU-HRSS701	152.4	439.9	22.8	60.8	28.8	33.9	6.68	7.24	1.26	0.56
LIGHTSABER	50.9	54.9	37.2	35.3	35.1	32.3	1.37	1.55	0.94	0.91
KYBER512	75.7	89.5	32.6	29.4	23.2	17.5	2.33	3.04	0.71	0.59
NTRU-HPS821	245.3	586.5	75.7	69.0	56.1	40.7	3.24	8.49	0.74	0.59
SABER	90.4	96.2	59.9	58.0	54.3	53.8	1.51	1.66	0.91	0.93
KYBER768	119.8	137.8	49.2	45.7	33.9	26.0	2.43	3.02	0.69	0.57
FIRESABER	140.9	150.8	87.9	86.7	78.9	78.1	1.60	1.74	0.90	0.90
KYBER1024	175.4	198.4	71.6	67.1	45.2	35.5	2.45	2.96	0.63	0.53

Intel Core i7 using 6-10% fewer clock cycles

Apple M1 w/NEON @ 3.2 GHz vs. Intel Core i7-8750H w/AVX2 4.1 GHz Frequency Scaling Effect



Time measured with the ns accuracy using clock_gettime() on a MacBook Air and a PC laptop

Conclusions

Conclusions

- High-speed hardware for KEMs:
  - CRYSTALS-Kyber and Saber comparable; Saber more flexible
  - NTRU and Classic McEliece significantly slower for key generation and somewhat slower for decapsulation and encapsulation
  - SIKE, BIKE, HQC, and FrodoKEM orders of magnitude slower
- High-speed hardware for Digital Signatures:
  - CRYSTALS-Dilithium efficient and easy to implement
  - FALCON Verify operation the fastest, but KeyGen and Sign prohibitively complicated
  - SPHINCS+ and Picnic outperformed by CRYSTALS-Dilithium
- Lightweight hardware for KEMs w/ SCA countermeasures:
  - Saber relatively easy to protect against side-channel attacks
- NEON-based software implementations
  - CRYSTALS-Kyber slightly faster than Saber; NTRU noticeably behind in most cases

Gazing the PQC Crystal Ball

NIST possible choices

	Conservative (security-based)	Cautious	Efficient & Flexible
Key Encapsulation Mechanism:	Classic McEliece	NTRU	SABER *or* CRYSTALS-Kyber
Digital Signature:	SPHINCS+	None	CRYSTALS-Dilithium *or* FALCON

Q&A

Thank You!

Questions?  Comments?

CERG: http://cryptography.gmu.edu

ATHENa: http://cryptography.gmu.edu/athena

Choose: PQC

Backup

Related Developments

NSA's Cybersecurity Perspective on PQC, Jul 2020

- Strong preference for Lattice-Based Cryptography
  - "fairly well-studied"
  - "secure when well-parameterized"
  - "among the most efficient"
- Planned adoption for National Security Systems (NSS)

Concerns about the viability of the majority of lattice-based schemes, 2021

- Patent issues
- New S-unit attack by Dan Bernstein, et al.

2022-2024: Draft of First-Generation Standards, Round 4, On Ramp for non-lattice Digital Signature

SIAM Conference on Applied Algebraic Geometry, Aug. 2021

Plenary Talk

S-unit attacks

Daniel J. Bernstein

University of Illinois at Chicago; Ruhr University Bochum

Includes new joint work with
Kirsten Eisenträger, Tanja Lange, Karl Rubin,
Alice Silverberg, and Christine van Vredendaal.

Builds upon vast previous literature;
see upcoming paper for credits.

Unproven Conjuncture

Conjectured scalability: $\exp(n^{1/2+o(1)})$

Simple algorithm variant, skipping many speedups:

Take traditional $\log y \in n^{1/2+o(1)}$.
Take $S = \infty \cup \{P : \#(R/P) \leq y\}$.
Precompute $\left\{S\text{-unit } u \in R\text{: } \sum_i u_i^2 \leq n^{1/2+o(1)}\right\}$.

Compute S-generator g of I.
Replace g with gu/v having log vector closest to I;
repeat until stable $\Rightarrow$ small S-generator of I.
Multiply by $P_c P_{-c}$ gens $\Rightarrow$ short element of I.
Repeat $y^{O(1)}$ times, avoiding cycles; take shortest.

Heuristics $\Rightarrow \eta \leq n^{1/2+o(1)}$, time $\exp(n^{1/2+o(1)})$.
"Vector within ϵ of shortest in subexponential time."

Dan Bernstein's Classification

Two different optimization goals

If goal is to minimize enc + dec time, best option is
Quotient NTRU: original 1998 Hoffstein–Pipher–Silverman NTRU.
Keygen: $G = e/a$. Enc: $B = Gb + d$. Dec: ...

If goal is to minimize keygen + enc + dec time, best option is
Product NTRU: 2010 Lyubashevsky–Peikert–Regev (LPR).
Keygen: $A = aG + e$. Enc: $B = Gb + d$, $C = M + Ab + c$. Dec: ...

NTRU's `ntruhrss` and `ntruhps` options:	Quotient NTRU.
NTRU Prime's `sntrup` option:	Quotient NTRU.
NTRU Prime's `ntrulpr` option:	Product NTRU.
SABER:	Product NTRU.
Kyber:	Product NTRU.

Dan Bernstein's Patent Analysis

Original NTRU was patented. Patent expired in 2017.

U.S. patent 9094189 until 2032 threatens Product NTRU (LPR). Was filed before LPR was published. Kept quiet for many years. Litigation against this patent was filed in 2017 and gave up in 2021.

U.S. patent 9246675 until 2033 threatens Product NTRU with compressed ciphertexts. Was filed before 2014 Peikert paper claimed LPR ciphertext compression as an "innovation". Apparently stopped Google's first post-quantum experiment, 2016.

Ongoing arguments: "Non-applicability ... to Kyber and Saber"; but "doctrine of equivalents"; NIST's secret patent analysis; ...

Dan Bernstein's Risk Analysis

Highly unstable attack picture! What do we do?

For each KEM family: Use biggest keys you can afford.
Can also choose a KEM family to eliminate *some* attack avenues:

submission	NTRU		NTRU Prime		SABER	Kyber	Frodo
KEM family	`ntruhrss`	`ntruhps`	`sntrup`	`ntrulpr`	`saber`	`kyber`	`frodo`
lattices	risk	risk	risk	risk	risk	risk	risk
derandomization				risk	risk	risk	risk
decryption failures					risk	risk	risk
structured lattices	risk	risk	risk	risk	risk	risk	
cyclotomics	risk	risk			risk	risk	
reducibility	risk	risk			risk	risk	
quotients	risk	risk	risk				
extra samples				risk	risk	risk	risk
non-QROM FO	risk	risk	risk	risk	risk	risk	risk
non-QROM 2				risk	risk	risk	risk

Dan Bernstein's Risk Analysis

submission	NTRU		NTRU Prime		SABER	Kyber	Frodo
KEM family	ntruhrss	ntruhps	sntrup	ntrulpr	saber	kyber	frodo
Known attack avenues not ruled out by theorems							
lattices	risk	risk	risk	risk	risk	risk	risk
derandomization				risk	risk	risk	risk
decryption failures					165	174	138
structured lattices	risk	risk	risk	risk	risk	risk	
cyclotomics	risk	risk			risk	risk	
reducibility	risk	risk			risk	risk	
quotients	risk	risk	risk				
extra samples				risk	risk	risk	risk
non-QROM FO	risk	risk	risk	risk	risk	risk	risk
non-QROM 2				risk	risk	risk	risk
Known patent threats							
patent 9094189				risk	risk	risk	
patent 9246675				risk	risk	risk	

D.J. Bernstein, https://ntruprime.cr.yp.to/warnings.html