Compact and Provably Secure Lattice-Based Signatures in Hardware

James Howe, Ciara Rafferty, Ayesha Khalid, Máire O'Neill.

May 2017

CENTRE FOR SECURE INFORMATION TECHNOLOGIES

Queen's University Belfast

@CSIT_QUB
@SAFECRYPTO

IEEE ISCAS 2017

Contributors

- Jointly collaborative work between:
  - James Howe, Ciara Rafferty, Ayesha Khalid, and Máire O'Neill,
    - Centre of Secure Information Technologies (CSIT), Queen's University Belfast, UK.

European Union H2020 SAFEcrypto Project: Advancing lattice-based cryptography in theory and practice (2015-2018).

@SAFEcrypto www.SAFEcrypto.eu

Presentation outline

- Motivation
- Introduction to (ideal) lattice-based cryptography
- Hardware design and optimisations
- Results and performance comparison
- Future research directions

What happens when quantum computers become a reality 10/15 years from now?

Commonly used public-key cryptographic algorithms
(based on integer factorisation and discrete log problem) such as:

RSA, DSA, Diffie-Hellman Key Exchange, ECC, ECDSA

will be vulnerable to Shor's algorithm and **will no longer be secure**.

What happens when quantum computers become a reality 10/15 years from now?

Commonly used public-key cryptographic algorithms
(based on integer factorisation and discrete log problem) such as:

RSA, DSA, Diffie-Hellman Key Exchange, ECC, ECDSA

will be vulnerable to Shor's algorithm and **will no longer be secure**.

IBM's quantum cloud computer goes commercial -*March 2017* nature

What happens when quantum computers become a reality 10/15 years from now?

Commonly used public-key cryptographic algorithms
(based on integer factorisation and discrete log problem) such as:

RSA, DSA, Diffie-Hellman Key Exchange, ECC, ECDSA

will be vulnerable to Shor's algorithm and **will no longer be secure**.

IBM's quantum cloud computer goes commercial -*March 2017* nature

Scientists are close to building a quantum computer that can beat a conventional one
–*December 2016* Science

What happens when quantum computers become a reality 10/15 years from now?

Commonly used public-key cryptographic algorithms
(based on integer factorisation and discrete log problem) such as:

RSA, DSA, Diffie-Hellman Key Exchange, ECC, ECDSA

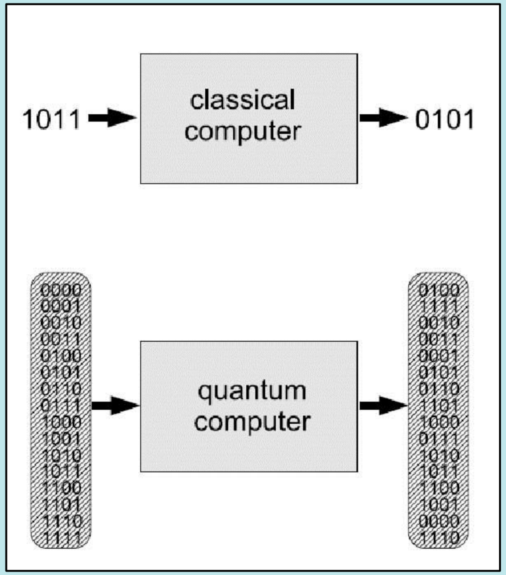
will be vulnerable to Shor's algorithm and **will no longer be secure**.

IBM's quantum cloud computer goes commercial -*March 2017* nature

Google moves closer to a universal quantum computer -*June 2016* nature

Scientists are close to building a quantum computer that can beat a conventional one –*December 2016* Science

- Why do we need post-quantum cryptography?
  - Quantum computers break ECC and RSA.
  - Classically hard computational problems are now trivial.
  - Governments and companies in preparation.



- Quantum computers exploit the power of quantum parallelism:
- Shor’s Algorithm (1994)
  - Used to quickly factorise large numbers (exponential speedup).
  - Significant implications for current cryptographic techniques.
- Grover’s Algorithm(1996)
  - Can be used to search an unsorted database faster than a conventional computer, effects security of AES, so AES-128 now 64-bit secure.

Motivation

- Quantum computers will cause complete insecurities in all currently implemented cryptography on the Internet, such as RSA, DSA, ECC, and ECDSA.
- Post-quantum cryptography currently protects against classical and quantum attacks.

- Quantum computers will cause complete insecurities in all currently implemented cryptography on the Internet, such as RSA, DSA, ECC, and ECDSA.
- Post-quantum cryptography currently protects against classical and quantum attacks.

Revealed: Google's plan for quantum computer supremacy

-*August 2016*

New Scientist

Motivation

- Quantum computers will cause complete insecurities in all currently implemented cryptography on the Internet, such as RSA, DSA, ECC, and ECDSA.
- Post-quantum cryptography currently protects against classical and quantum attacks.

Revealed: Google's plan for quantum computer supremacy
-*August 2016*
New Scientist

The quantum clock is ticking on encryption – and your data is under threat *-October 2016*
WIRED

- Quantum computers will cause complete insecurities in all currently implemented cryptography on the Internet, such as RSA, DSA, ECC, and ECDSA.
- Post-quantum cryptography currently protects against classical and quantum attacks.

Revealed: Google's plan for quantum computer supremacy -*August 2016*

New Scientist

Hacking, cryptography, and the countdown to quantum computing –*September 2016*

THE NEW YORKER

The quantum clock is ticking on encryption – and your data is under threat -*October 2016*

WIRED

- Quantum computers will cause complete insecurities in all currently implemented cryptography on the Internet, such as RSA, DSA, ECC, and ECDSA.
- Post-quantum cryptography currently protects against classical and quantum attacks.

Revealed: Google's plan for quantum computer supremacy -*August 2016*

New Scientist

Hacking, cryptography, and the countdown to quantum computing –*September 2016*

THE NEW YORKER

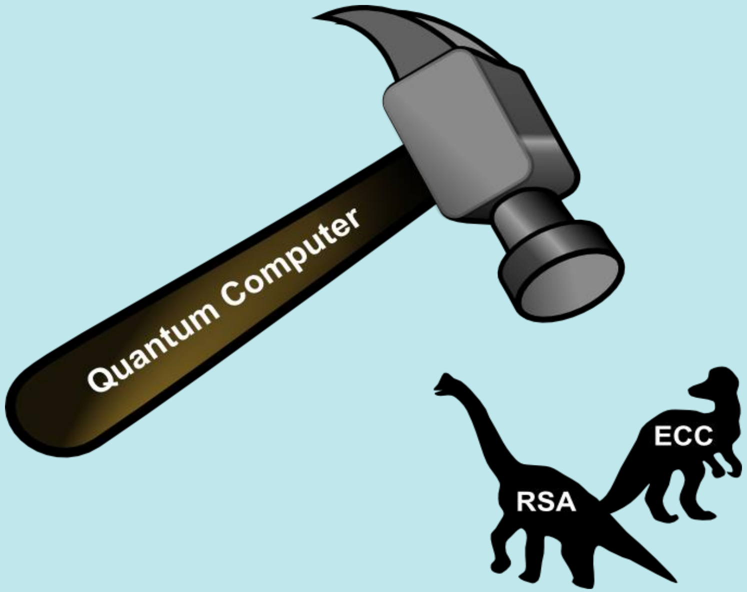
Google tests new crypto in Chrome to fend off quantum attacks –*July 2016*

WIRED

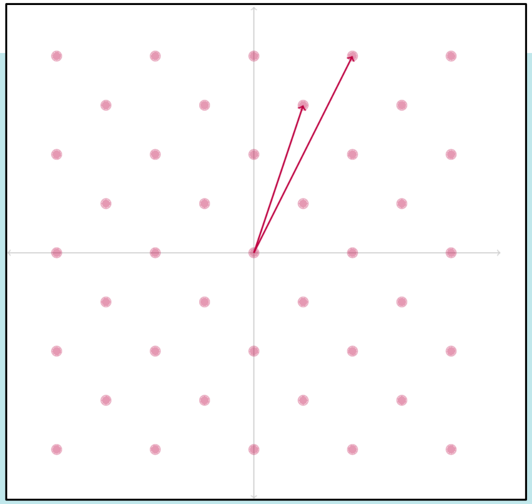
The quantum clock is ticking on encryption – and your data is under threat -*October 2016*

WIRED

- ETSI researching industrial requirements for quantum-safe real-world deployments.
  - ETSI Quantum-Safe Cryptography (QSC) Industry Specification Group (ISG).
- NIST plan to start post-quantum standardisation 30 Nov 2017
  - Similar to previous AES and SHA-3 standardisations.
- Why focus on lattice-based cryptography?
  - More versatile than code-based, MQ, and hash-based schemes.
  - Theoretical foundations are well-studied.
  - Uses in encryption, signatures, FHE, IBE, etc…

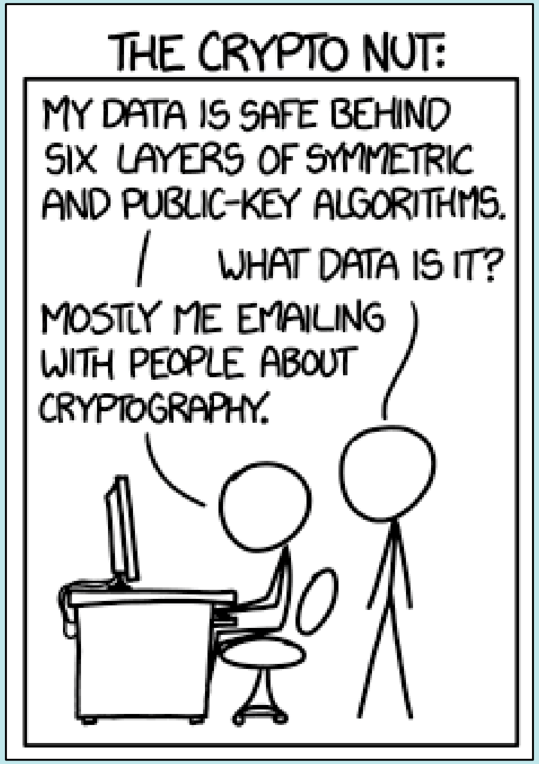


Lattice-Based Cryptography



- Lattice-based cryptography is important in its own right.
- Research in lattice-based cryptography is flourishing:
  - “New Hope” key exchange created.
  - “LPR” encryption outperforms RSA and ECC in s/w and h/w.
  - “BLISS” signatures outperform RSA and ECDSA in s/w and h/w.
- Lattice-based cryptography is already being considered:
  - VPN strongSwan supports signature and encryption within post-quantum mode.
  - New Hope awarded Internet Defense Prize Winner 2016.
  - Google experimenting with “New Hope” key exchange.
  - Horizon 2020 SAFECrypto Project.
    - Advancing lattice-based cryptography in theory and practice.

- Digital signatures are very important.
  - Authenticates message source.
  - Validates that data sent is unaltered/trusted.
  - Identifies person, legally, like written signature.
- Used everyday within bank transfers, smart cards, SSL etc…
  - Currently uses RSA or ECDSA.
  - These will be obsolete with quantum computers.
- Hardware-based signatures are becoming more prominent.
  - Will become prominent within IoT & the cloud.
  - Required for V2X communications.



Ring-TESLA vs BLISS

- For lattice-based digital signatures, current state-of-the-art is BLISS.
- Recently announced was Ring-TESLA, an efficient lattice-based signature scheme.
- Shown to compete with state-of-the-art in software.

BLISS vs.	Ring-TESLA
Patented NTRU assumptions.	**Strong** security assumptions (Ring-LWE).
No worst-case to average-case hardness.	**Includes** worst-case to average-case hardness.
Costly discrete Gaussian sampling.	**No** on-device discrete Gaussian sampling.
Large polynomial multiplier used.	**Evaluate** generic low-area poly. multiplier.
Parameters not chosen via security reduction.	**Simpler** parameter selection, tightly secure.

Discrete Gaussian Samplers

- Attacks found in the BLISS algorithm.
- Cache attack (software) targets the discrete Gaussian sampler component.
- Discrete Gaussian samplers are known to be a side-channel target in software and hardware.
- Not known yet how to fix this issue in software.
- Ring-TESLA uses discrete Gaussian samplers independent of secret computations.

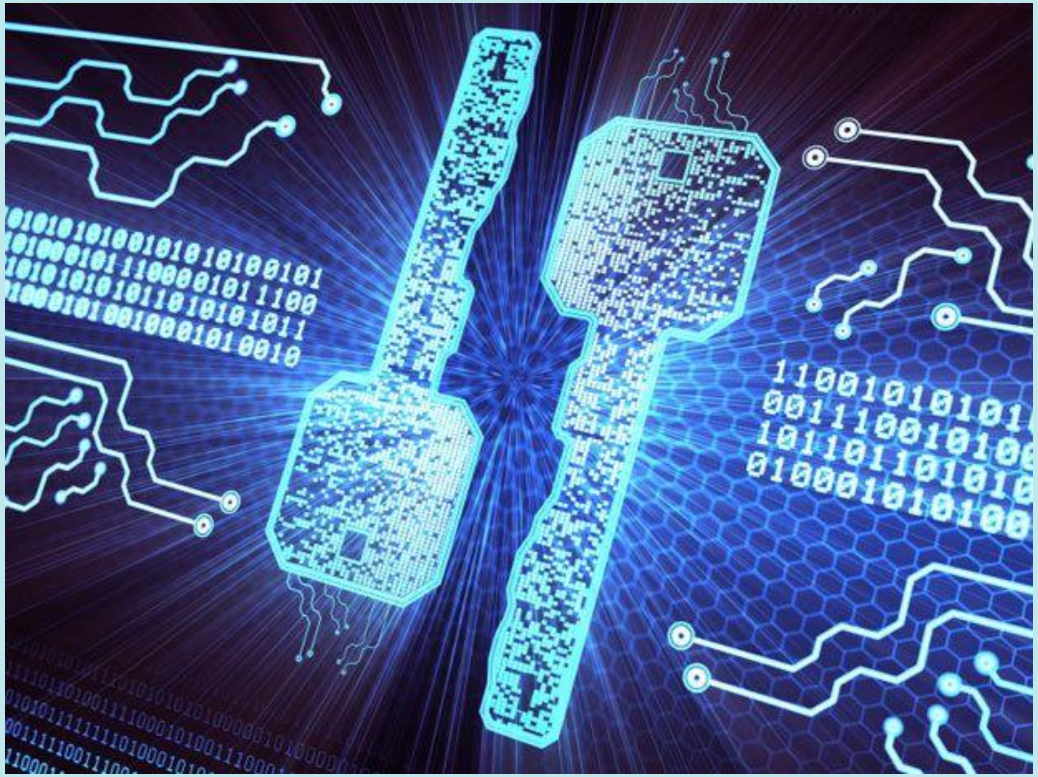
Flush, Gauss, and Reload – A Cache Attack on the BLISS Lattice-Based Signature Scheme

Leon Groot Bruinderink[1], Andreas Hülsing[1], Tanja Lange[1], and Yuval Yarom[2]

[1] Department of Mathematics and Computer Science
Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, NL
l.groot.bruinderink@tue.nl, andreas@huelsing.net, tanja@hyperelliptic.org

[2] The University of Adelaide and NICTA
yval@cs.adelaide.edu.au

- Generic hardware architecture.
- Evaluate low-area hardware design.
- Offers better security, slower throughput.
- Spartan-6 FPGA targeted for comparisons.



Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):
Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1 \boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$
Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1\boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; a_1, a_2, s, e_1, e_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1\boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2\boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{sc}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2\boldsymbol{c} \bmod q$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1 \boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1 \boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2 \boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{s}\boldsymbol{c}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1 \boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2 \boldsymbol{c} \bmod q$

Verify($\mu; \boldsymbol{z}, \boldsymbol{c}; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{t}_1, \boldsymbol{t}_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1 \boldsymbol{z} + \boldsymbol{t}_1 \boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2 \boldsymbol{z} + \boldsymbol{t}_2 \boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1 \boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; a_1, a_2, s, e_1, e_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$
- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1 \boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2 \boldsymbol{y} \bmod q$

Compute the hash function:
- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:
- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{s}\boldsymbol{c}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1 \boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2 \boldsymbol{c} \bmod q$

Verify($\mu; z, c; a_1, a_2, t_1, t_2$):

Compute hash inputs:
- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1 \boldsymbol{z} + \boldsymbol{t}_1 \boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2 \boldsymbol{z} + \boldsymbol{t}_2 \boldsymbol{c} \bmod q$

Compute the hash function:
- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:
- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1\boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; a_1, a_2, s, e_1, e_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1\boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2\boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{sc}$ $\leftarrow$signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2\boldsymbol{c} \bmod q$

Verify($\mu; z, c; a_1, a_2, t_1, t_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1\boldsymbol{z} + \boldsymbol{t}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2\boldsymbol{z} + \boldsymbol{t}_2\boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1\boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; a_1, a_2, s, e_1, e_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1\boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2\boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{sc}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2\boldsymbol{c} \bmod q$

Verify($\mu; z, c; a_1, a_2, t_1, t_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1\boldsymbol{z} + \boldsymbol{t}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2\boldsymbol{z} + \boldsymbol{t}_2\boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1\boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1\boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2\boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{s}\boldsymbol{c}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2\boldsymbol{c} \bmod q$

Verify($\mu; \boldsymbol{z}, \boldsymbol{c}; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{t}_1, \boldsymbol{t}_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1\boldsymbol{z} + \boldsymbol{t}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2\boldsymbol{z} + \boldsymbol{t}_2\boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1 \boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; a_1, a_2, s, e_1, e_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1 \boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2 \boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{s}\boldsymbol{c}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1 \boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2 \boldsymbol{c} \bmod q$

Verify($\mu; z, c; a_1, a_2, t_1, t_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1 \boldsymbol{z} + \boldsymbol{t}_1 \boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2 \boldsymbol{z} + \boldsymbol{t}_2 \boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1 \boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; a_1, a_2, s, e_1, e_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1 \boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2 \boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{s}\boldsymbol{c}$ $\leftarrow$signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1 \boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2 \boldsymbol{c} \bmod q$

Verify($\mu; z, c; a_1, a_2, t_1, t_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1 \boldsymbol{z} + \boldsymbol{t}_1 \boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2 \boldsymbol{z} + \boldsymbol{t}_2 \boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1\boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; a_1, a_2, s, e_1, e_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1\boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2\boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{sc}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2\boldsymbol{c} \bmod q$

Verify($\mu; z, c; a_1, a_2, t_1, t_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1\boldsymbol{z} + \boldsymbol{t}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2\boldsymbol{z} + \boldsymbol{t}_2\boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1\boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1\boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2\boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{sc}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2\boldsymbol{c} \bmod q$

Verify($\mu; \boldsymbol{z}, \boldsymbol{c}; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{t}_1, \boldsymbol{t}_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1\boldsymbol{z} + \boldsymbol{t}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2\boldsymbol{z} + \boldsymbol{t}_2\boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1\boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1\boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2\boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boxed{\boldsymbol{sc}}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boxed{\boldsymbol{e}_1\boldsymbol{c}} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boxed{\boldsymbol{e}_2\boldsymbol{c}} \bmod q$

Verify($\mu; \boldsymbol{z}, \boldsymbol{c}; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{t}_1, \boldsymbol{t}_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1\boldsymbol{z} + \boldsymbol{t}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2\boldsymbol{z} + \boldsymbol{t}_2\boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1\boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; a_1, a_2, s, e_1, e_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1\boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2\boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boxed{\boldsymbol{sc}}$ ←signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boxed{\boldsymbol{e}_1\boldsymbol{c}} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boxed{\boldsymbol{e}_2\boldsymbol{c}} \bmod q$

Verify($\mu; z, c; a_1, a_2, t_1, t_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1\boldsymbol{z} + \boxed{\boldsymbol{t}_1\boldsymbol{c}} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2\boldsymbol{z} + \boxed{\boldsymbol{t}_2\boldsymbol{c}} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

- If $\boldsymbol{c} = \boldsymbol{c}'$

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Ring-TESLA signature scheme[1] (Akleylek *et al.* '16)

KeyGen(a_1, a_2):

Discrete Gaussian polynomials: $\mathbf{s}, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow D_\sigma^n$, $\mathbf{t}_1 \equiv \boldsymbol{a}_1\boldsymbol{s} + \boldsymbol{e}_1 \bmod q$, $\mathbf{t}_2 \equiv \boldsymbol{a}_2 + \boldsymbol{e}_2 \bmod q$

Secret-Key: $(\boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ // Public-Key: $(\boldsymbol{t}_1, \boldsymbol{t}_2)$.

Sign($\mu; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{s}, \boldsymbol{e}_1, \boldsymbol{e}_2$):

Uniform polynomial: $\boldsymbol{y} \leftarrow \mathbb{Z}_q[\boldsymbol{x}]/(\boldsymbol{x}^n + 1)$

- $\boldsymbol{v}_1 \equiv \boldsymbol{a}_1\boldsymbol{y} \bmod q, \boldsymbol{v}_2 \equiv \boldsymbol{a}_2\boldsymbol{y} \bmod q$

Compute the hash function:

- $\boldsymbol{c} = H(\boldsymbol{v}_1 || \boldsymbol{v}_2, \mu)$

Compute signature/rejections:

- $\boldsymbol{z} \equiv \boldsymbol{y} + \boldsymbol{sc}$ $\leftarrow$signature
- $\boldsymbol{w}_1 \equiv \boldsymbol{v}_1 + \boldsymbol{e}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2 \equiv \boldsymbol{v}_2 + \boldsymbol{e}_2\boldsymbol{c} \bmod q$

Verify($\mu; \boldsymbol{z}, \boldsymbol{c}; \boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{t}_1, \boldsymbol{t}_2$):

Compute hash inputs:

- $\boldsymbol{w}_1' \equiv \boldsymbol{a}_1\boldsymbol{z} + \boldsymbol{t}_1\boldsymbol{c} \bmod q$
- $\boldsymbol{w}_2' \equiv \boldsymbol{a}_2\boldsymbol{z} + \boldsymbol{t}_2\boldsymbol{c} \bmod q$

Compute the hash function:

- $\boldsymbol{c}' = H(\boldsymbol{w}_1' || \boldsymbol{w}_2', \mu)$

Accept/reject signature:

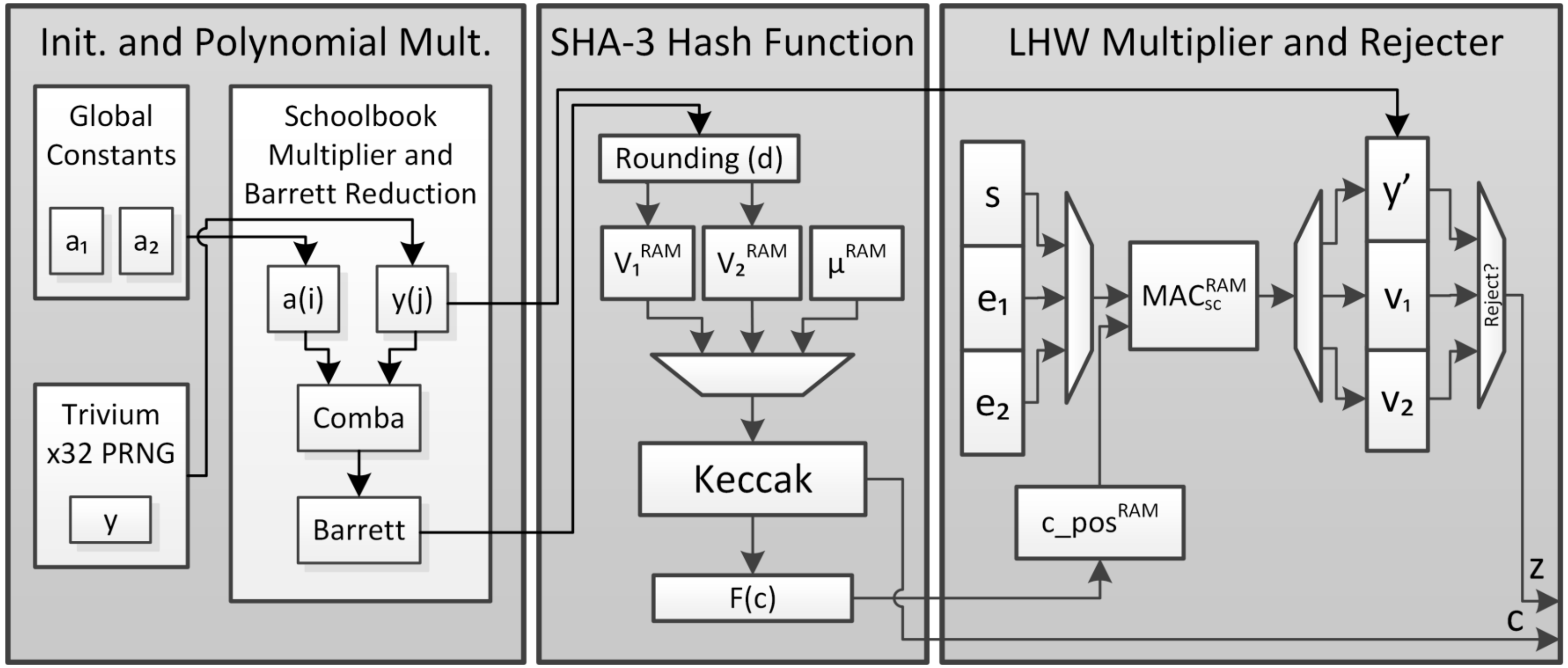
- If $\boldsymbol{c} = \boldsymbol{c}'$

128-bit security parameters:
$n = 512$,
$q = 51750913$,
$\sigma = 52$.

Signature is 11.9 kb,
public-key is 26 kb,
and secret-key is 13.7 kb.

1) Sedat Akleylek, Nina Bindel, Johannes A. Buchmann, Juliane Krämer, and Giorgia Azzurra Marson. An efficient lattice-based signature scheme with provably secure instantiation. In AFRICACRYPT, pages 44–60, 2016.

Architecture of Ring-TESLA Sign



High-level architecture of the Ring-LWE signature scheme, Ring-TESLA.

- The first hardware design of a Ring-LWE signature scheme.
- First low-area signature scheme in lattice-based cryptography.
- Generic hardware designs for sign and verify.
- Numerous parallel multipliers used for a variety of results.

Architecture of the Ring-TESLA Signature Scheme

Algorithm 1 Ring-TESLA Sign

procedure $\textsc{Sign}(\mu, \mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$

$\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\mathbf{c} = F(c)$

$\mathbf{z} \leftarrow \mathbf{y} + \mathbf{s}\mathbf{c}$

$\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d-L}$

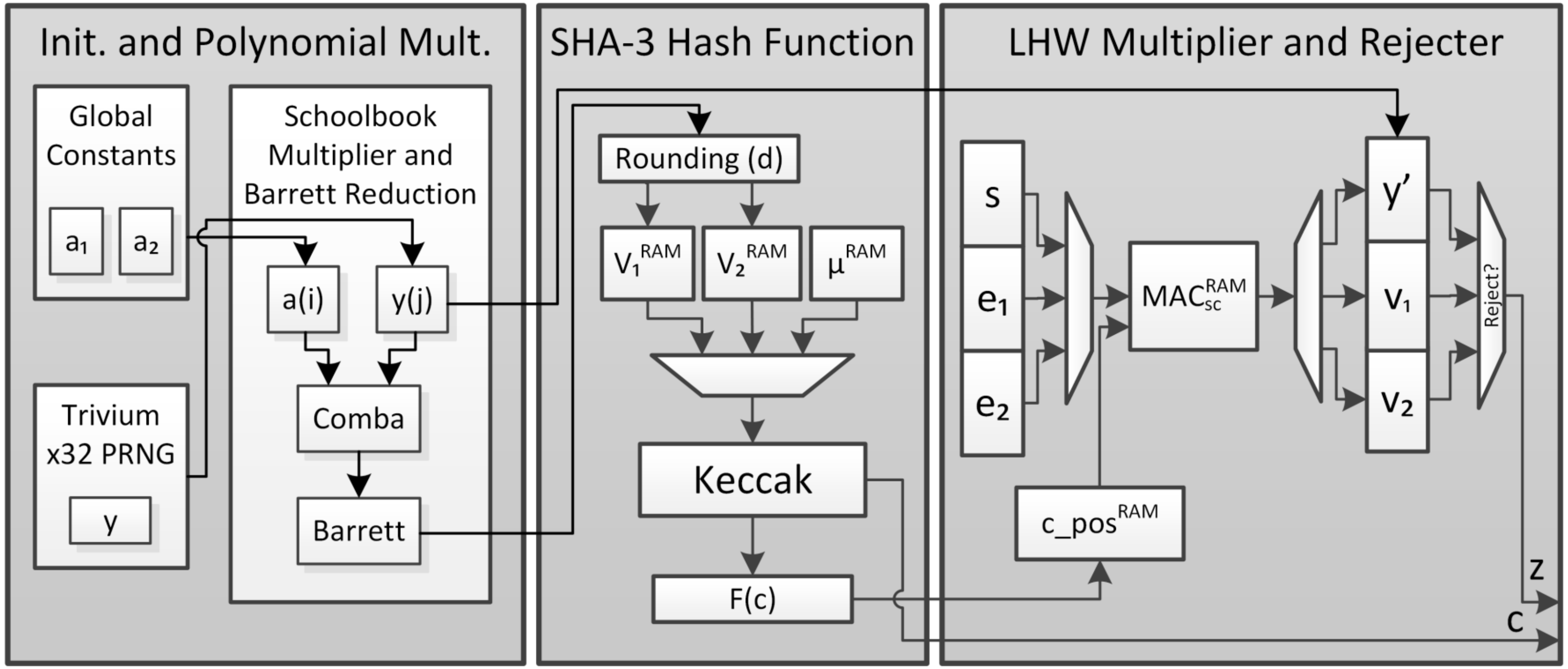
or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

Restart

end if

return $(\mathbf{z}, c)$

end procedure



Architecture of the Ring-TESLA Signature Scheme

Algorithm 1 Ring-TESLA Sign

procedure $\textsc{Sign}(\mu, \mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$

$\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\mathbf{c} = F(c)$

$\mathbf{z} \leftarrow \mathbf{y} + \mathbf{sc}$

$\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d - L}$

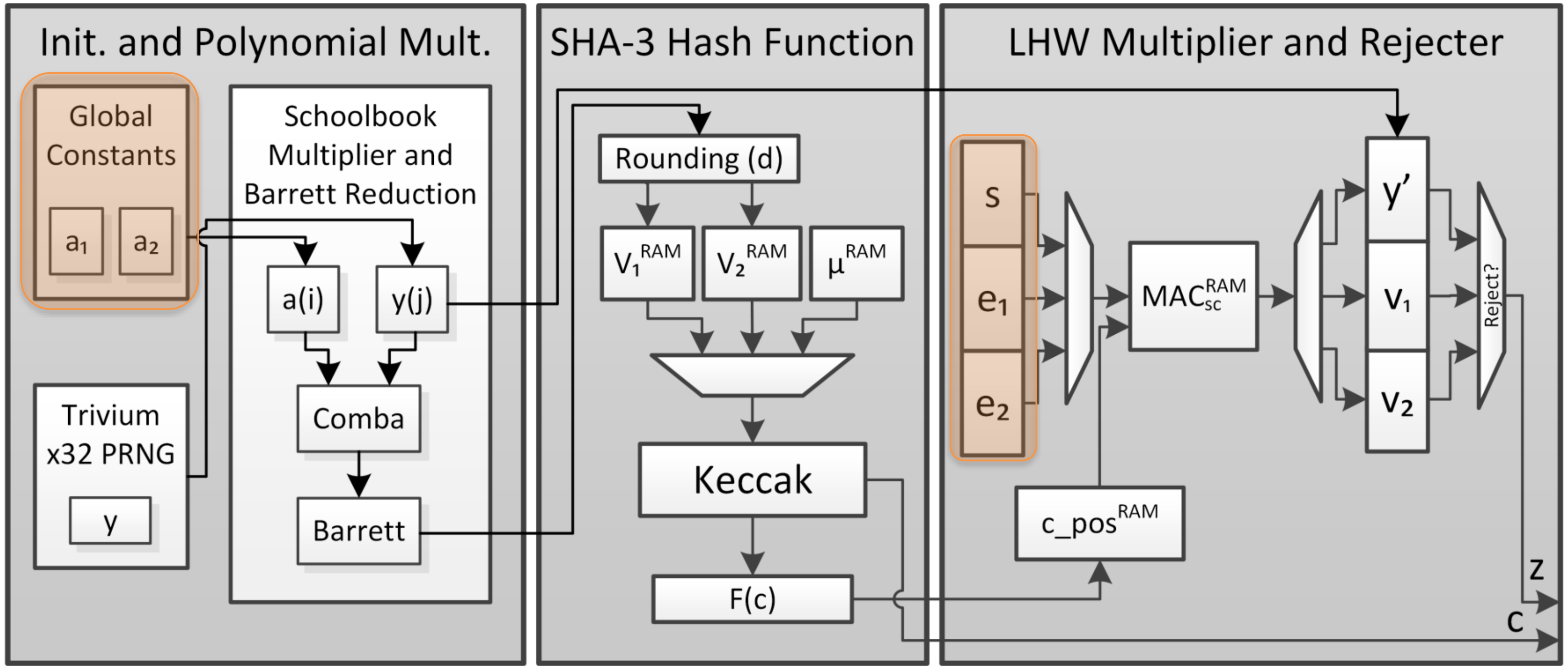
or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

Restart

end if

return $(\mathbf{z}, c)$

end procedure



Architecture of the Ring-TESLA Signature Scheme

Algorithm 1 Ring-TESLA Sign

procedure $\textsc{Sign}(\mu, \mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$

$\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\mathbf{c} = F(c)$

$\mathbf{z} \leftarrow \mathbf{y} + \mathbf{s}\mathbf{c}$

$\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

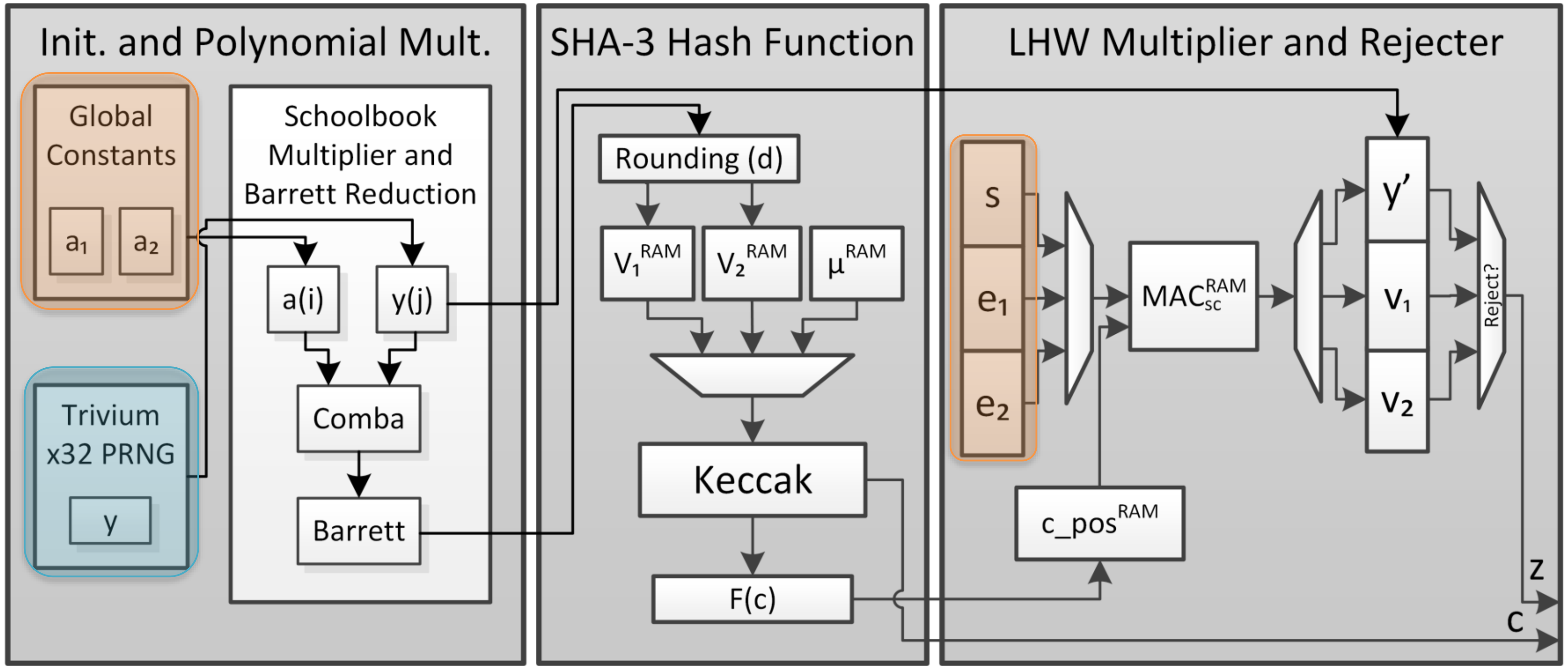
if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d - L}$ or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

Restart

end if

return $(\mathbf{z}, c)$

end procedure



Architecture of the Ring-TESLA Signature Scheme

Algorithm 1 Ring-TESLA Sign

---

procedure $\textsc{Sign}(\mu, \mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$

$\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\mathbf{c} = F(c)$

$\mathbf{z} \leftarrow \mathbf{y} + \mathbf{s}\mathbf{c}$

$\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

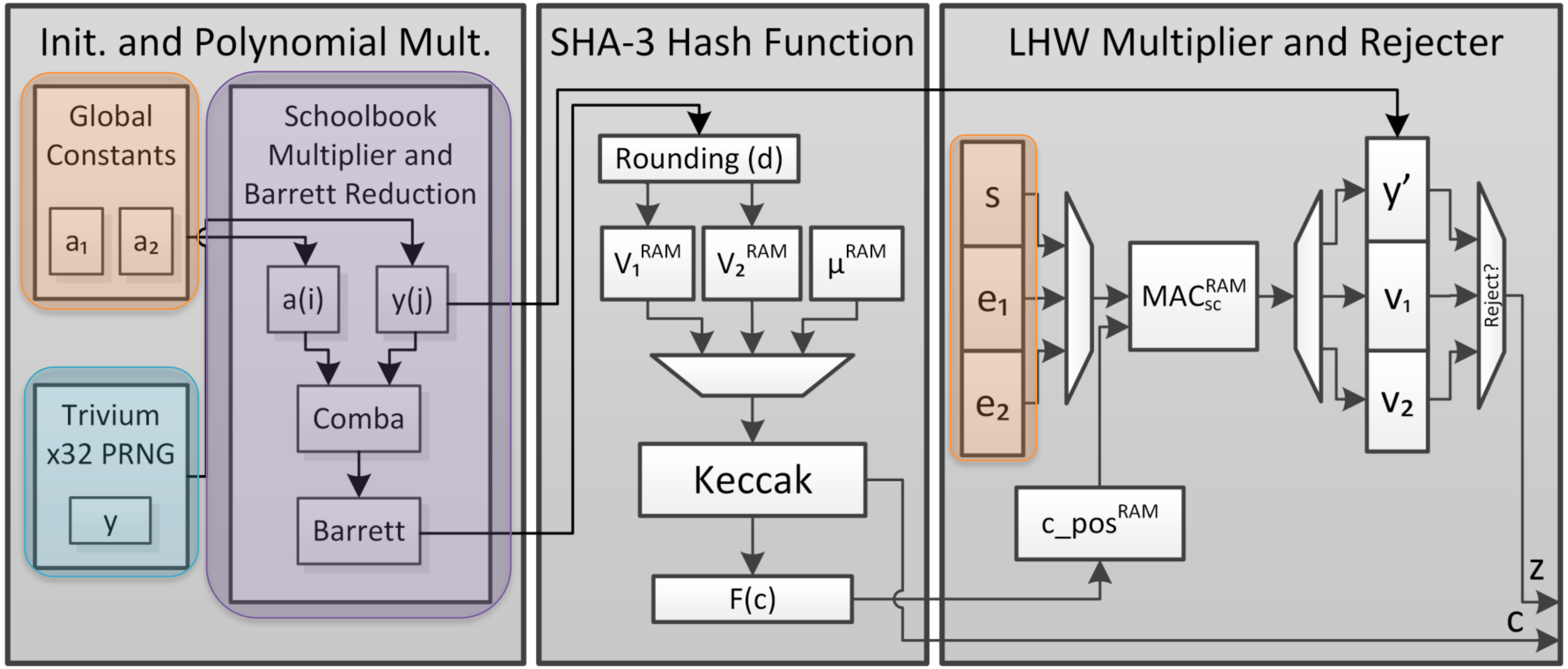
if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d-L}$ or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

Restart

end if

return $(\mathbf{z}, c)$

end procedure



Architecture of the Ring-TESLA Signature Scheme

Algorithm 1 Ring-TESLA Sign

procedure SIGN(μ, $\mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2$)

$\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\mathbf{c} = F(c)$

$\mathbf{z} \leftarrow \mathbf{y} + \mathbf{s}\mathbf{c}$

$\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

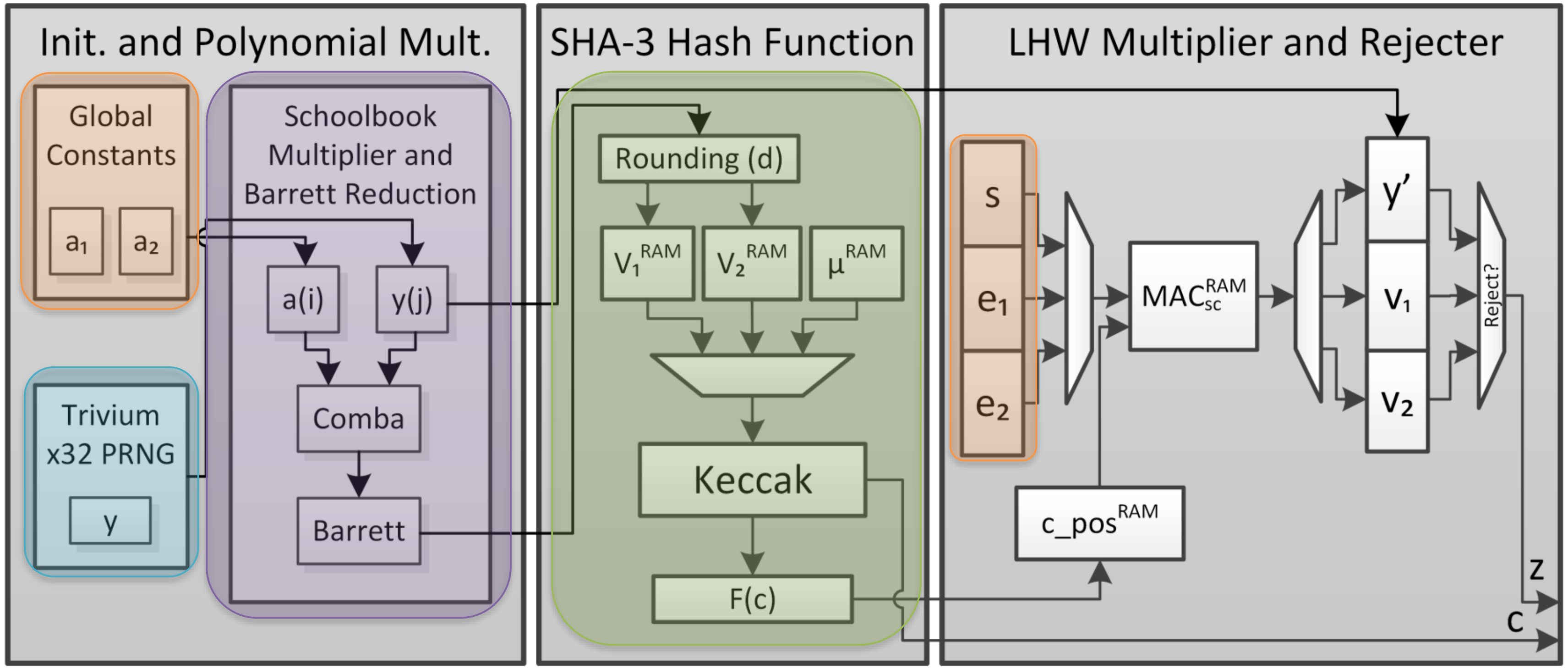
if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d-L}$ or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

Restart

end if

return $(\mathbf{z}, c)$

end procedure



Architecture of the Ring-TESLA Signature Scheme

Algorithm 1 Ring-TESLA Sign

procedure SIGN(μ, $\mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2$)

    $\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

    $\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

    $\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

    $c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

    $\mathbf{c} = F(c)$

    $\mathbf{z} \leftarrow \mathbf{y} + \mathbf{sc}$

    $\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

    $\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

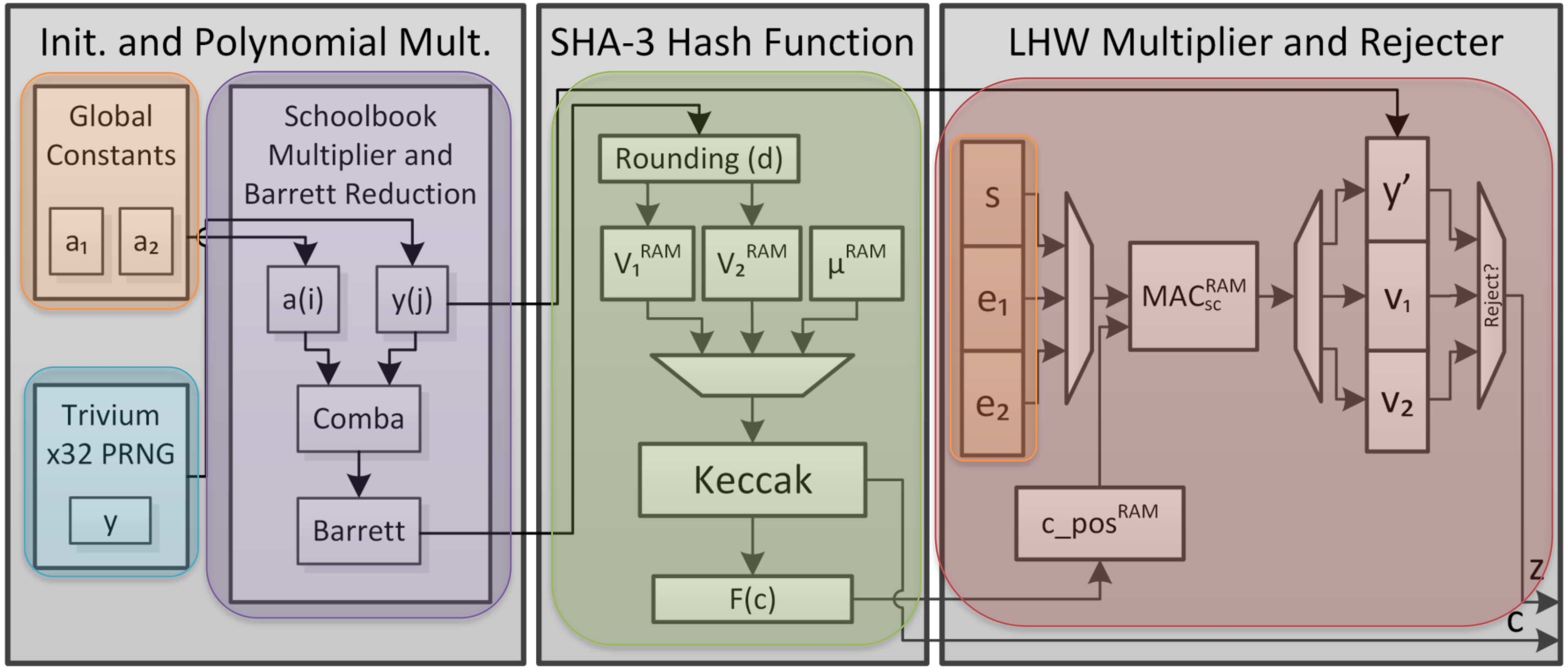
    if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d - L}$ or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

        Restart

    end if

    return $(\mathbf{z}, c)$

end procedure



FSM of the Ring-TESLA Hardware Architecture

Algorithm 1 Ring-TESLA Sign

procedure $\textsc{Sign}(\mu, \mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$

$\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\mathbf{c} = F(c)$

$\mathbf{z} \leftarrow \mathbf{y} + \mathbf{s}\mathbf{c}$

$\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d-L}$

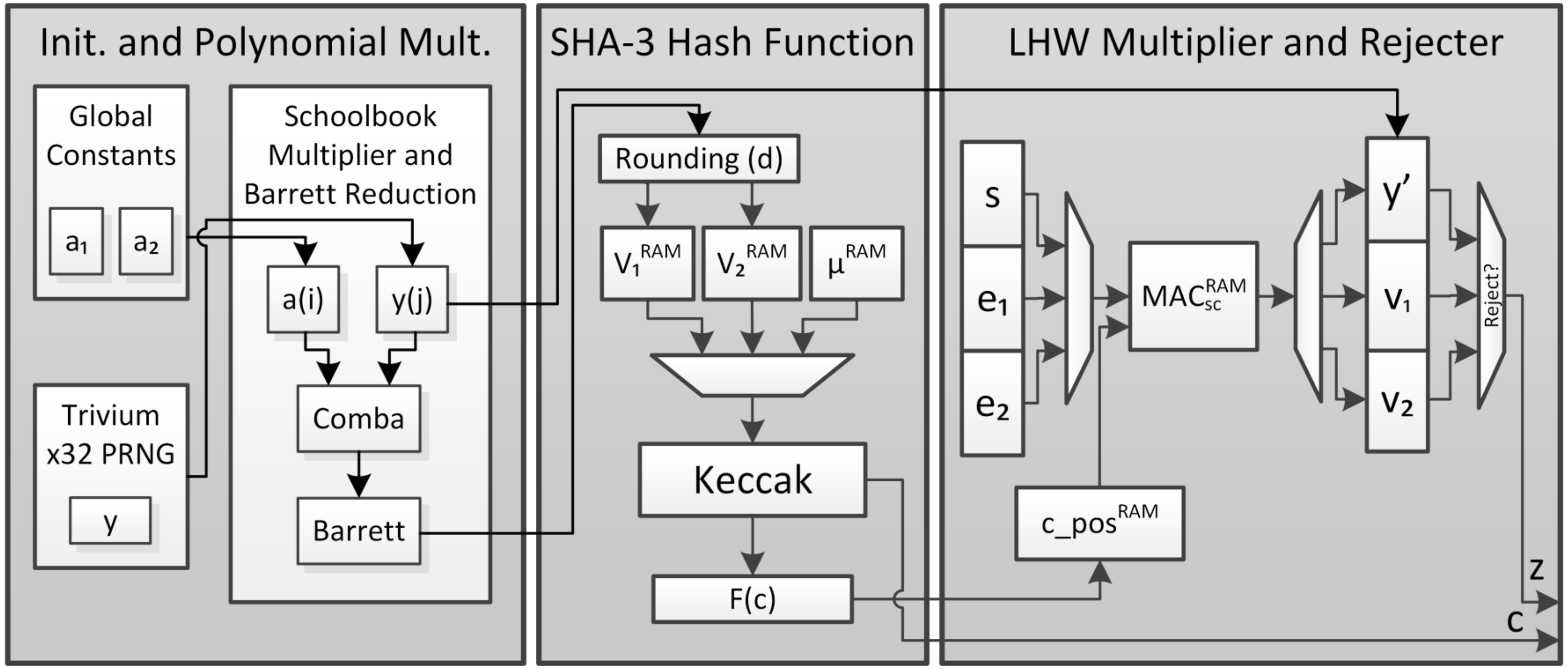
or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

Restart

end if

return $(\mathbf{z}, c)$

end procedure



FSM of the Ring-TESLA Hardware Architecture

Algorithm 1 Ring-TESLA Sign

procedure $\textsc{Sign}(\mu, \mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$

$\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\mathbf{c} = F(c)$

$\mathbf{z} \leftarrow \mathbf{y} + \mathbf{s}\mathbf{c}$

$\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d - L}$

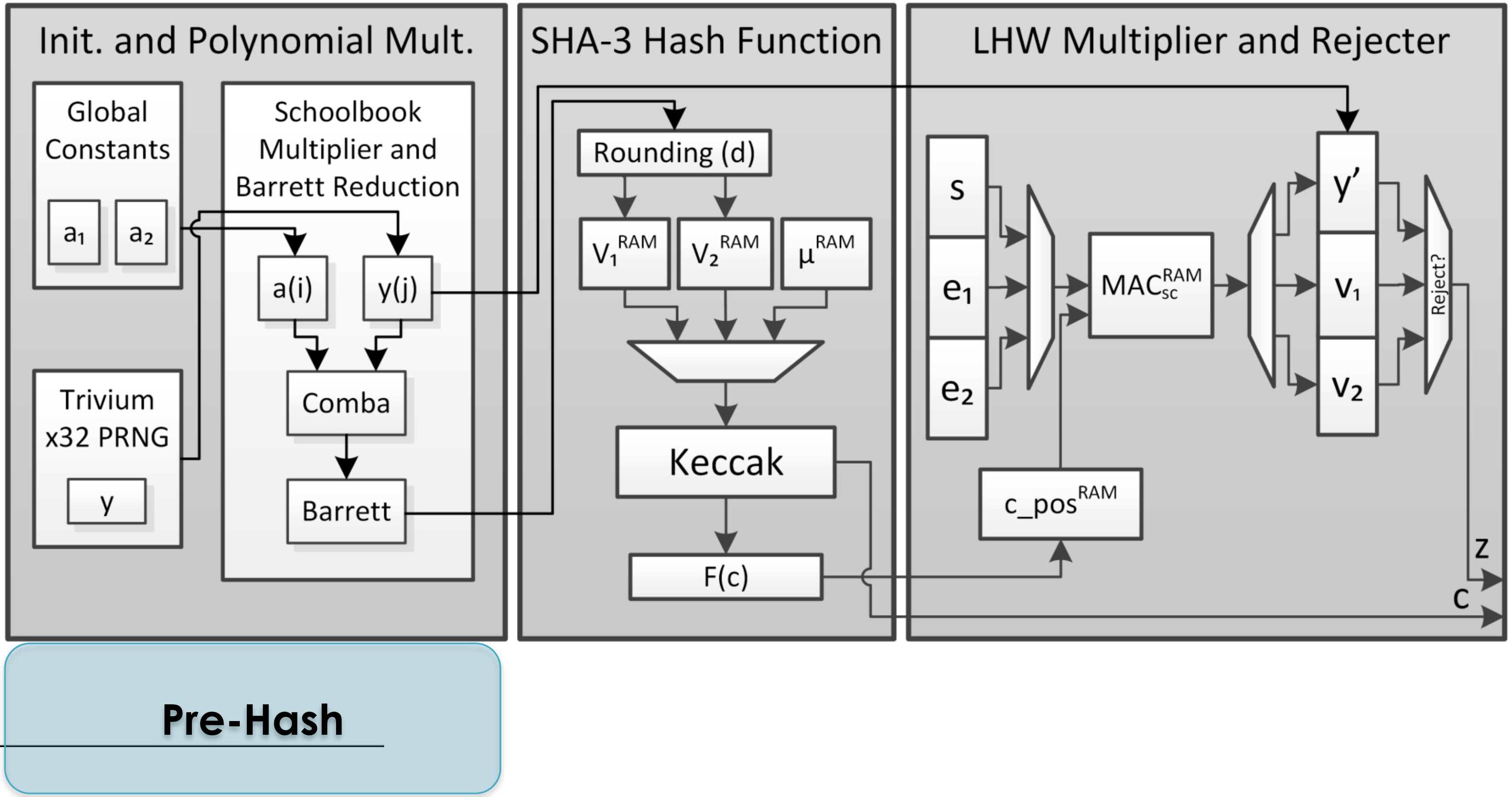
or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

Restart

end if

return $(\mathbf{z}, c)$

end procedure



FSM of the Ring-TESLA Hardware Architecture

Algorithm 1 Ring-TESLA Sign

procedure SIGN(μ, $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{s}$, $\mathbf{e}_1$, $\mathbf{e}_2$)

$\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\mathbf{c} = F(c)$

$\mathbf{z} \leftarrow \mathbf{y} + \mathbf{sc}$

$\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d-L}$

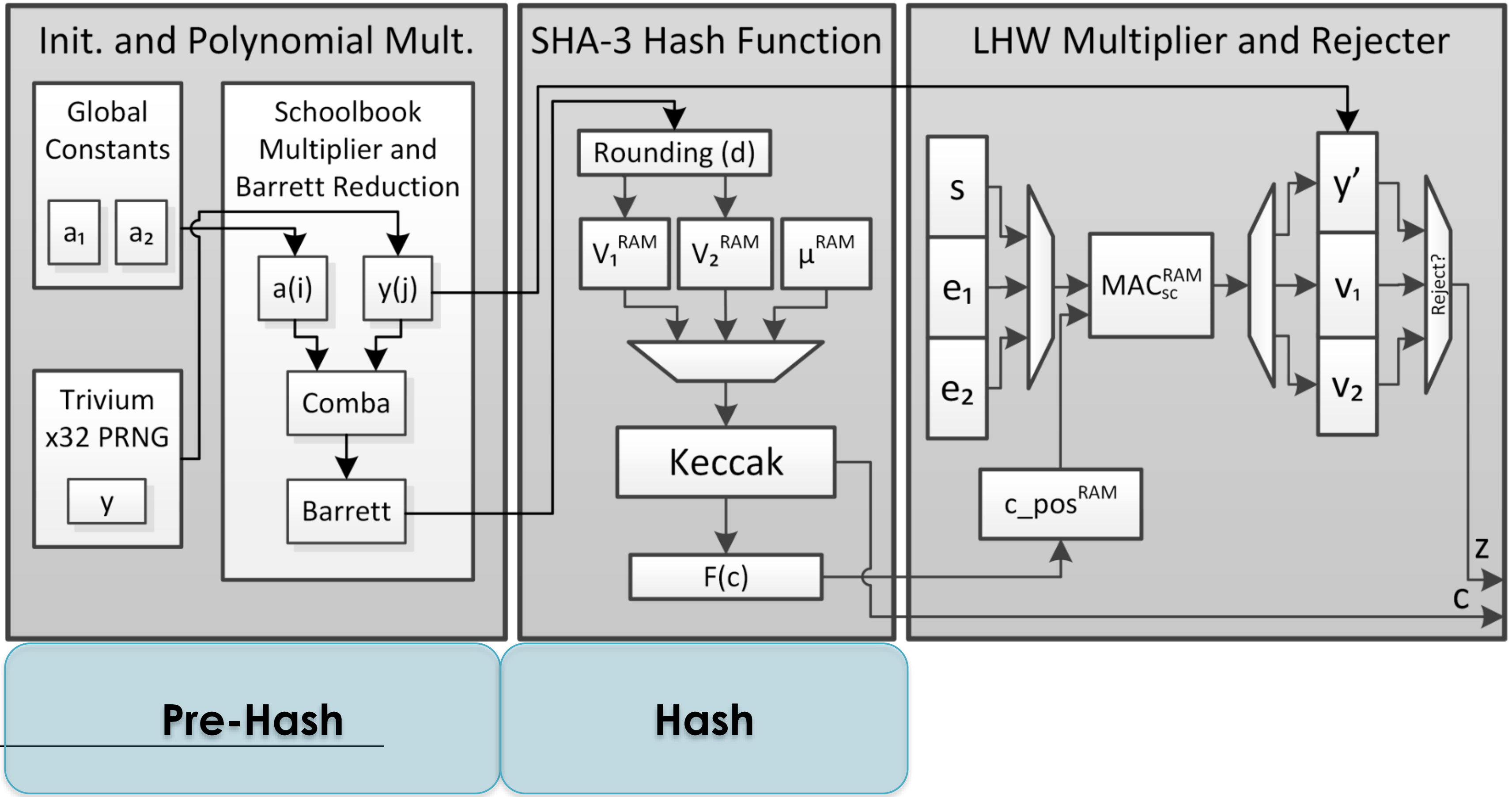
or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

Restart

end if

return $(\mathbf{z}, c)$

end procedure



FSM of the Ring-TESLA Hardware Architecture

Algorithm 1 Ring-TESLA Sign

procedure $\textsc{Sign}(\mu, \mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$

$\quad \mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\quad \mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\quad \mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$\quad c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\quad \mathbf{c} = F(c)$

$\quad \mathbf{z} \leftarrow \mathbf{y} + \mathbf{s}\mathbf{c}$

$\quad \mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\quad \mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

$\quad$ **if** $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d - L}$

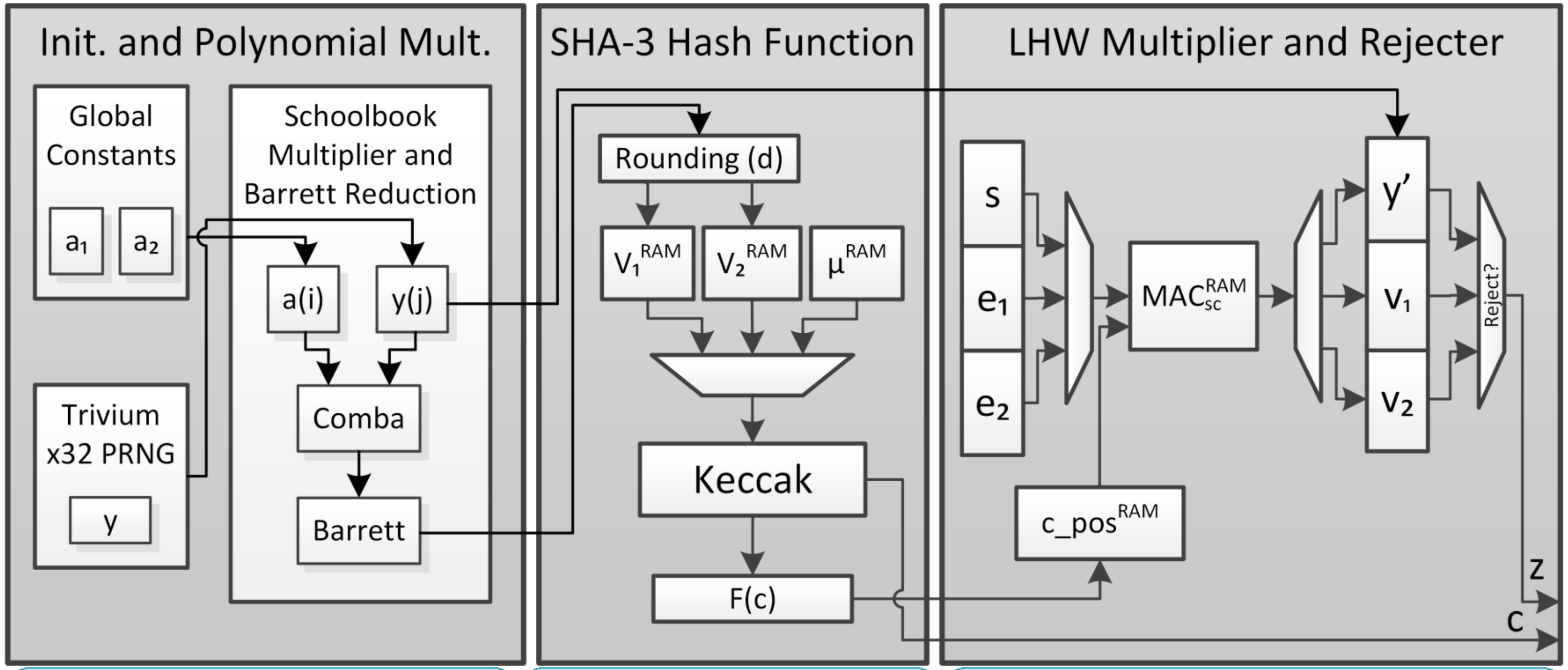
$\qquad$ or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

$\qquad$ Restart

$\quad$ **end if**

$\quad$ **return** $(\mathbf{z}, c)$

end procedure



Pre-Hash

Hash

Post-Hash

FSM of Ring-TESLA Architecture

Algorithm 1 Ring-TESLA Sign

procedure $\textsc{Sign}(\mu, \mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$

$\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$

$\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$

$\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$

$c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$

$\mathbf{c} = F(c)$

$\mathbf{z} \leftarrow \mathbf{y} + \mathbf{s}\mathbf{c}$

$\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$

$\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$

if $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d-L}$

or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**

Restart

end if

- Pipeline created for pre-hash computations.
- After pre-hash polynomial multiplication;
  - $\boldsymbol{y}$ is copied to another register for $\boldsymbol{z}$.
  - $\boldsymbol{y}$ is generated for next signature in parallel.
- Hash, LHW calculations of $\boldsymbol{z}$, $\boldsymbol{w_1}$, and $\boldsymbol{w_2}$, and rejections then outside the critical path.
- Sign/Verify critical path thus pre-hash phase.

FSM of Ring-TESLA Architecture

Algorithm 1 Ring-TESLA Sign

procedure $\textsc{Sign}(\mu, \mathbf{a}_1, \mathbf{a}_2, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$

- $\mathbf{y} \xleftarrow{\$} \mathcal{R}_{q,[B]}$
- $\mathbf{v}_1 \equiv \mathbf{a}_1\mathbf{y} \mod q$
- $\mathbf{v}_2 \equiv \mathbf{a}_2\mathbf{y} \mod q$
- $c = H(\lfloor \mathbf{v}_1 \rceil_{d,q}, \lfloor \mathbf{v}_2 \rceil_{d,q}, \mu)$
- $\mathbf{c} = F(c)$
- $\mathbf{z} \leftarrow \mathbf{y} + \mathbf{sc}$
- $\mathbf{w}_1 \equiv \mathbf{v}_1 - \mathbf{e}_1\mathbf{c} \mod q$
- $\mathbf{w}_2 \equiv \mathbf{v}_2 - \mathbf{e}_2\mathbf{c} \mod q$
- **if** $[\mathbf{w}_1]_{2^d}, [\mathbf{w}_2]_{2^d} \notin \mathcal{R}_{2^d-L}$ or $\mathbf{z} \notin \mathcal{R}_{B-U}$ **then**
  - Restart
- **end if**

- Pipeline created for pre-hash computations.
- After pre-hash polynomial multiplication;
  - $\boldsymbol{y}$ is copied to another register for $\boldsymbol{z}$.
  - $\boldsymbol{y}$ is generated for next signature in parallel.
- Hash, LHW calculations of $\boldsymbol{z}$, $\boldsymbol{w_1}$, and $\boldsymbol{w_2}$, and rejections then outside the critical path.
- Sign/Verify critical path thus pre-hash phase.

Signature ♯1 | Poly. Mult. ⇒ | Hash ⇒ | LHW

Signature ♯2 | Poly. Mult. ⇒ | Hash ⇒ | LHW

⋮ ⋱

Signature ♯n | Poly. Mult. ⇒ | Hash ⇒ | LHW

- Ring-TESLA, ideal lattice-based signatures on a Spartan 6 – LX25 FPGA.
- Significantly smaller than other lattice-based signature designs, suffers in throughput.
- Significantly smaller and faster in comparison to RSA and ECDSA.

Operation, Configuration	Security	Device	LUT/FF/SLICE	BRAM/DSP	MHz	Cycles	Ops/sec
Ring-TESLA (Sign, SB-I)	128-bits	S6 LX25	4447/3345/1257	3/6	190	1835540	104
Ring-TESLA (Sign, SB-II)	128-bits	S6 LX25	4828/3790/1513	4/8	196	917771	214
Ring-TESLA (Sign, SB-IV)	128-bits	S6 LX25	5071/3851/1503	4/12	187	458891	408
Ring-TESLA-(Sign, SB-VIII)	128-bits	S6 LX25	6848/5457/2254	4/20	180	229446	785
Ring-TESLA (Verify, SB-I)	128-bits	S6 LX25	3714/3023/1172	3/6	188	1835540	102
Ring-TESLA (Verify, SB-II)	128-bits	S6 LX25	3917/3253/1238	3/8	194	917771	212
Ring-TESLA (Verify, SB-IV)	128-bits	S6 LX25	4793/3939/1551	3/12	186	458891	406
Ring-TESLA (Verify, SB-VIII)	128-bits	S6 LX25	6473/5582/2103	3/20	178	229446	776
GLP (Sign, Schoolbook x3)	80-bits	S6 LX16	7465/8993/2273	30/28	162	-	931
GLP (Verify, Schoolbook x3)	80-bits	S6 LX16	6225/6663/2263	15/8	158	-	998
BLISS (Sign, NTT)	128-bits	S6 LX25	7193/6420/2291	6/5	139	15864	8761
BLISS (Verify NTT)	128-bits	S6 LX25	5065/4312/1687	4/3	166	16346	17101
RSA (Sign)	103-bits	V5 LX30	3237 slices	7/17	200	-	89
ECDSA (Sign)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	139	-	-
ECDSA (Verify)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	110	-	

Ring-TESLA Hardware Results

- Ring-TESLA, ideal lattice-based signatures on a Spartan 6 – LX25 FPGA.
- Significantly smaller than other lattice-based signature designs, suffers in throughput.
- Significantly smaller and faster in comparison to RSA and ECDSA.

Operation, Configuration	Security	Device	LUT/FF/SLICE	BRAM/DSP	MHz	Cycles	Ops/sec
Ring-TESLA (Sign, SB-I)	128-bits	S6 LX25	4447/3345/1257	3/6	190	1835540	104
Ring-TESLA (Sign, SB-II)	128-bits	S6 LX25	4828/3790/1513	4/8	196	917771	214
Ring-TESLA (Sign, SB-IV)	128-bits	S6 LX25	5071/3851/1503	4/12	187	458891	408
Ring-TESLA-(Sign, SB-VIII)	128-bits	S6 LX25	6848/5457/2254	4/20	180	229446	785
Ring-TESLA (Verify, SB-I)	128-bits	S6 LX25	3714/3023/1172	3/6	188	1835540	102
Ring-TESLA (Verify, SB-II)	128-bits	S6 LX25	3917/3253/1238	3/8	194	917771	212
Ring-TESLA (Verify, SB-IV)	128-bits	S6 LX25	4793/3939/1551	3/12	186	458891	406
Ring-TESLA (Verify, SB-VIII)	128-bits	S6 LX25	6473/5582/2103	3/20	178	229446	776
GLP (Sign, Schoolbook x3)	80-bits	S6 LX16	7465/8993/2273	30/28	162	-	931
GLP (Verify, Schoolbook x3)	80-bits	S6 LX16	6225/6663/2263	15/8	158	-	998
BLISS (Sign, NTT)	128-bits	S6 LX25	7193/6420/2291	6/5	139	15864	8761
BLISS (Verify NTT)	128-bits	S6 LX25	5065/4312/1687	4/3	166	16346	17101
RSA (Sign)	103-bits	V5 LX30	3237 slices	7/17	200	-	89
ECDSA (Sign)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	139	-	-
ECDSA (Verify)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	110	-	

- Ring-TESLA, ideal lattice-based signatures on a Spartan 6 – LX25 FPGA.
- Significantly smaller than other lattice-based signature designs, suffers in throughput.
- Significantly smaller and faster in comparison to RSA and ECDSA.

Operation, Configuration	Security	Device	LUT/FF/SLICE	BRAM/DSP	MHz	Cycles	Ops/sec
Ring-TESLA (Sign, SB-I)	128-bits	S6 LX25	4447/3345/1257	3/6	190	1835540	104
Ring-TESLA (Sign, SB-II)	128-bits	S6 LX25	4828/3790/1513	4/8	196	917771	214
Ring-TESLA (Sign, SB-IV)	128-bits	S6 LX25	5071/3851/1503	4/12	187	458891	408
Ring-TESLA-(Sign, SB-VIII)	128-bits	S6 LX25	6848/5457/2254	4/20	180	229446	785
Ring-TESLA (Verify, SB-I)	128-bits	S6 LX25	3714/3023/1172	3/6	188	1835540	102
Ring-TESLA (Verify, SB-II)	128-bits	S6 LX25	3917/3253/1238	3/8	194	917771	212
Ring-TESLA (Verify, SB-IV)	128-bits	S6 LX25	4793/3939/1551	3/12	186	458891	406
Ring-TESLA (Verify, SB-VIII)	128-bits	S6 LX25	6473/5582/2103	3/20	178	229446	776
GLP (Sign, Schoolbook x3)	80-bits	S6 LX16	7465/8993/2273	30/28	162	-	931
GLP (Verify, Schoolbook x3)	80-bits	S6 LX16	6225/6663/2263	15/8	158	-	998
BLISS (Sign, NTT)	128-bits	S6 LX25	7193/6420/2291	6/5	139	15864	8761
BLISS (Verify NTT)	128-bits	S6 LX25	5065/4312/1687	4/3	166	16346	17101
RSA (Sign)	103-bits	V5 LX30	3237 slices	7/17	200	-	89
ECDSA (Sign)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	139	-	-
ECDSA (Verify)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	110	-	

- Ring-TESLA, ideal lattice-based signatures on a Spartan 6 – LX25 FPGA.
- Significantly smaller than other lattice-based signature designs, suffers in throughput.
- Significantly smaller and faster in comparison to RSA and ECDSA.

Operation, Configuration	Security	Device	LUT/FF/SLICE	BRAM/DSP	MHz	Cycles	Ops/sec
Ring-TESLA (Sign, SB-I)	128-bits	S6 LX25	4447/3345/1257	3/6	190	1835540	104
Ring-TESLA (Sign, SB-II)	128-bits	S6 LX25	4828/3790/1513	4/8	196	917771	214
Ring-TESLA (Sign, SB-IV)	128-bits	S6 LX25	5071/3851/1503	4/12	187	458891	408
Ring-TESLA-(Sign, SB-VIII)	128-bits	S6 LX25	6848/5457/2254	4/20	180	229446	785
Ring-TESLA (Verify, SB-I)	128-bits	S6 LX25	3714/3023/1172	3/6	188	1835540	102
Ring-TESLA (Verify, SB-II)	128-bits	S6 LX25	3917/3253/1238	3/8	194	917771	212
Ring-TESLA (Verify, SB-IV)	128-bits	S6 LX25	4793/3939/1551	3/12	186	458891	406
Ring-TESLA (Verify, SB-VIII)	128-bits	S6 LX25	6473/5582/2103	3/20	178	229446	776
GLP (Sign, Schoolbook x3)	80-bits	S6 LX16	7465/8993/2273	30/28	162	-	931
GLP (Verify, Schoolbook x3)	80-bits	S6 LX16	6225/6663/2263	15/8	158	-	998
BLISS (Sign, NTT)	128-bits	S6 LX25	7193/6420/2291	6/5	139	15864	8761
BLISS (Verify NTT)	128-bits	S6 LX25	5065/4312/1687	4/3	166	16346	17101
RSA (Sign)	103-bits	V5 LX30	3237 slices	7/17	200	-	89
ECDSA (Sign)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	139	-	-
ECDSA (Verify)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	110	-	

- Ring-TESLA, ideal lattice-based signatures on a Spartan 6 – LX25 FPGA.
- Significantly smaller than other lattice-based signature designs, suffers in throughput.
- Significantly smaller and faster in comparison to RSA and ECDSA.

Operation, Configuration	Security	Device	LUT/FF/SLICE	BRAM/DSP	MHz	Cycles	Ops/sec
Ring-TESLA (Sign, SB-I)	128-bits	S6 LX25	4447/3345/1257	3/6	190	1835540	104
Ring-TESLA (Sign, SB-II)	128-bits	S6 LX25	4828/3790/1513	4/8	196	917771	214
Ring-TESLA (Sign, SB-IV)	128-bits	S6 LX25	5071/3851/1503	4/12	187	458891	408
Ring-TESLA-(Sign, SB-VIII)	128-bits	S6 LX25	6848/5457/2254	4/20	180	229446	785
Ring-TESLA (Verify, SB-I)	128-bits	S6 LX25	3714/3023/1172	3/6	188	1835540	102
Ring-TESLA (Verify, SB-II)	128-bits	S6 LX25	3917/3253/1238	3/8	194	917771	212
Ring-TESLA (Verify, SB-IV)	128-bits	S6 LX25	4793/3939/1551	3/12	186	458891	406
Ring-TESLA (Verify, SB-VIII)	128-bits	S6 LX25	6473/5582/2103	3/20	178	229446	776
GLP (Sign, Schoolbook x3)	80-bits	S6 LX16	7465/8993/2273	30/28	162	-	931
GLP (Verify, Schoolbook x3)	80-bits	S6 LX16	6225/6663/2263	15/8	158	-	998
BLISS (Sign, NTT)	128-bits	S6 LX25	7193/6420/2291	6/5	139	15864	8761
BLISS (Verify NTT)	128-bits	S6 LX25	5065/4312/1687	4/3	166	16346	17101
RSA (Sign)	103-bits	V5 LX30	3237 slices	7/17	200	-	89
ECDSA (Sign)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	139	-	-
ECDSA (Verify)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	110	-	

- Ring-TESLA, ideal lattice-based signatures on a Spartan 6 – LX25 FPGA.
- Significantly smaller than other lattice-based signature designs, suffers in throughput.
- Significantly smaller and faster in comparison to RSA and ECDSA.

Operation, Configuration	Security	Device	LUT/FF/SLICE	BRAM/DSP	MHz	Cycles	Ops/sec
Ring-TESLA (Sign, SB-I)	128-bits	S6 LX25	4447/3345/1257	3/6	190	1835540	104
Ring-TESLA (Sign, SB-II)	128-bits	S6 LX25	4828/3790/1513	4/8	196	917771	214
Ring-TESLA (Sign, SB-IV)	128-bits	S6 LX25	5071/3851/1503	4/12	187	458891	408
Ring-TESLA-(Sign, SB-VIII)	128-bits	S6 LX25	6848/5457/2254	4/20	180	229446	785
Ring-TESLA (Verify, SB-I)	128-bits	S6 LX25	3714/3023/1172	3/6	188	1835540	102
Ring-TESLA (Verify, SB-II)	128-bits	S6 LX25	3917/3253/1238	3/8	194	917771	212
Ring-TESLA (Verify, SB-IV)	128-bits	S6 LX25	4793/3939/1551	3/12	186	458891	406
Ring-TESLA (Verify, SB-VIII)	128-bits	S6 LX25	6473/5582/2103	3/20	178	229446	776
GLP (Sign, Schoolbook x3)	80-bits	S6 LX16	7465/8993/2273	30/28	162	-	931
GLP (Verify, Schoolbook x3)	80-bits	S6 LX16	6225/6663/2263	15/8	158	-	998
BLISS (Sign, NTT)	128-bits	S6 LX25	7193/6420/2291	6/5	139	15864	8761
BLISS (Verify NTT)	128-bits	S6 LX25	5065/4312/1687	4/3	166	16346	17101
RSA (Sign)	103-bits	V5 LX30	3237 slices	7/17	200	-	89
ECDSA (Sign)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	139	-	-
ECDSA (Verify)	128-bits	V5 LX110	32299 LUT/FF pairs	10/37	110	-	

Conclusions and future work

- The first hardware design of a Ring-LWE signature scheme.
- First low-area signature scheme in lattice-based cryptography.
- Generic hardware designs for sign and verify, important for parameters changes.
- Numerous parallel multipliers used for a variety of results.
- Consider hardware-friendly parameters in the future.
- Consider high-throughput, large polynomial multiplier in the future.

Thank you!

Any questions?