Pre-Draft Public Comments on SP 800-197A

Comment period: June 6, 2025 – August 6, 2025

On June 6, 2025, NIST announced a proposal to develop three cryptographic accordions based on the HCTR2 technique and requested pre-draft public comments for an eventual specification in SP 800-197A. The comments that NIST received are collected below.

LIST OF COMMENTS

1. Comments from the National Cyber Security Centre (United Kingdom) July 24, 2025

NCSC agrees that the Hash-Encrypt-Hash paradigm is an effective approach for designing efficient accordions. HCTR2 appears a good choice for a birthday-bound accordion mode, whose claimed security properties meet the requirements of NIST IR 8552, and with good performance subject to these requirements. We look forward to discussing proposals for a beyond-birthday variant, and to participating in NIST-led community analysis of the security claims.

We have additional comments and suggestions regarding how HCTR2 could proceed towards a full specification:

Security:

- Some of HCTR2's performance advantages come at the cost of a proliferation of case checking in the security analysis, to bound the possibility that quantities collide. We have concerns that this brings risk that a mistake is made and not noticed. Complex security proofs can often contain mistakes, as for example in the case of OCB2, whose serious vulnerability went unnoticed for many years. We encourage community review of the proof before standardisation; for example, this seems a good candidate for formal verification with a tool such as Easy Crypt.
- To maximise opportunities for deployment, it would be desirable to have a security bound given in more specific variables, such as separating message length and tweak length if this yields a tighter bound.
- The specification permits input message and tweak lengths of almost the block size, but the security bound is only birthday in these. We recommend that in final standardisation, NIST should limit input lengths to ones which are secure.

AEAD via Encode-then-Encipher:

- When wrapping in Encode-then-Encipher for the purpose of AEAD, NIST IR 8552 depicts the padding at the end of the input. However, in the case of HCTR2, for performance reasons we suggest placing the padding in the M block, which already receives special treatment. This could be by placing the padding at the beginning, or by switching the order that M and N are read from the plaintext.
- In the case of variable tag lengths, we recommend cryptographically separating the different tag lengths. For example, all-zeroes padding could be modified by setting a single bit in the position adjacent to the payload. Without such separation, zero plaintext bits can be reallocated as tag bits and vice versa, while preserving the ciphertext.
- We note that a decision will need to be made as to how to handle the case that both the plaintext and the tag are short enough that they combine to less than one full block, the minimum message length for HCTR2. This is particularly likely to occur with 256-bit blocks and 128-bit tags. We note that protocol designers will need to accept that the minimum output length remains one full block, regardless.

NCSC thanks NIST for their effort to develop these new block cipher modes.

Charlotte S
Cryptography Research
National Cyber Security Centre

2. Comments from Bryan Queen (NSA Center for Cybersecurity Standards) August 4, 2025

Below are NSA's comments on NIST's proposal to employ variants of HCTR2 in its development of cryptographic accordions.

HCTR2 appears to be a reasonable choice on which to base a cryptographic accordion, per the design criteria outlined in NIST IR 8552. We have two comments.

Some AEAD modes (e.g., AES-GCM-SIV) lose security gracefully as the number of messages encrypted under a given key approaches the birthday bound; others do not. For HCTR2 this degradation of security is given in terms of the distinguishing advantage of an adversary attempting to distinguish it from an ideal accordion. When HCTR2 is used as an AEAD or for deterministic authenticated encryption, authentication fails with an unacceptable probability as the birthday bound is approached. There are collision events that leak the authentication key \bar{h} similarly to the GCM "forbidden attack" by Joux. It would be worthwhile to block such issues as it is often difficult to enforce strict data limits in practice.

The HCTR2 security proof implies loss of security when used with birthday-length inputs. As such, allowing the maximal tweak and message input sizes given in section 2.1 of the HCTR2 specification is not consistent with the proof. This inconsistency should be addressed in the eventual specification of accordions in NIST Special Publication (SP) 800-197A.

Please let us know if there are any questions. Thank you !

Regards,
Bryan

Bryan Queen, PMP, CMBB, CGPM
NSA Center for Cybersecurity Standards (CCSS)
Cybersecurity Collaboration Center

3. Comments from Christoph Dobraunig, Krystian Matusiewicz, and Alexander Tereschenko (Intel Labs and Intel Product Assurance and Security [IPAS] Crypto Team) August 5, 2025

Dear NIST team,

First, we want to thank NIST for pursuing the standardization of an accordion mode. We have several points we want to make.

1. We think that the set of planned accordions (Acc128, Acc256, BBBAcc) makes sense and covers the set of use cases the community discussed during the workshops. We would suggest specializing the names with the underlying block cipher to avoid any ambiguity in discussions and specifications, e.g., the first member of the set, Acc128 with AES, would be “Acc128-AES” or some such.
2. Choice of HCTR2. While we think that HCTR2 is a solid scheme, it comes with one main drawback – it needs three building blocks: a universal hash function, a stream cipher, and a block cipher. This leads to three downsides:
    1. Decryption needs the inverse of the block cipher. Hence, this inverse needs to be implemented, increasing the implementation complexity and inflating the code size for SW and power/performance/area envelope for HW.
    2. Since HCTR2 needs a block cipher, it can only be instantiated with blockciphers, leaving possible instantiation with NIST’s permutation-based standards out (Keccak-based, Ascon). Note that some of these NIST standards are already capable of providing a security level that a newly introduced 256-bit block cipher would provide.
    3. For the beyond-birthday bound instance, all three components must be instantiated such that they are beyond-birthday bound secure, again potentially increasing the proof and implementation complexity.
3. Focus on current software performance. We think that frequently-used cryptography will eventually be implemented in hardware. Hence, the focus on current software performance might lead to standardization of suboptimal constructions in hardware. We might make our lives easy now but pay the price for an easy path in the future. As another point, one of the major use cases for XTS-AES, potentially replaced by an accordion cipher mode, is memory encryption, which is usually done in hardware and requires high performance.
4. Huge message lengths required. We think the required message lengths of 2^64 and 2^96 are way too high. Especially since we are likely to deal here with at least two

passes over the message for encryption. Also, the message lengths might influence security negatively, and this might lead to designs accommodated for use cases that cannot be realistically achieved in practice. We suggest reformulating these limits as “total data processed under a single key” instead, which would be closer to practical use cases.

Christoph Dobraunig (*),

Krystian Matusiewicz (#),

Alexander Tereschenko (#, corresponding author)

(*) Intel Labs

(#) Intel Product Assurance and Security (IPAS) Crypto Team

4. Comments from the U.S. Department of Energy, August 5, 2025

Good evening,

Please find comments attached in response to the initial public draft, NIST SP 800-197A, Development of Cryptographic Accordions.

Respectfully,

Tasha Barnes

IM-1 Executive Secretariat, Communications Specialist

Contractor Supporting the Office of the Chief Information Officer

U.S. Department of Energy

[see DOE’s comments on the following pages]

Analysis of "PRE-DRAFT Call for Comments: NIST Launches Development of Cryptographic Accordions" (NIST SP 800-197A)

This document outlines NIST's new initiative to develop "cryptographic accordions," which are advanced block cipher modes designed to handle variable-length inputs and offer enhanced security principles.

1. Comprehensive Summary

1.1 Core Purpose and Technical Contribution

NIST's "Cryptographic Accordions" initiative aims to standardize new tweakable block cipher modes that can operate on variable-length inputs. The core contribution is to define "accordions" that serve simultaneously as modes of an underlying block cipher (like AES) and as tweakable block ciphers themselves capable of processing data of various lengths. This addresses limitations of NIST's current portfolio of block cipher modes, particularly concerning fixed-length inputs and security bounds. The goal is to provide more flexible and secure cryptographic primitives for applications such as authenticated encryption with associated data (AEAD), tweakable encryption for storage, and deterministic authenticated encryption.

1.2 Key Concepts and Proposed Designs

The draft introduces "cryptographic accordions" as tweakable, Variable-Input-Length Strong Pseudorandom Permutations (VIL-SPRPs) built from existing block ciphers. NIST proposes to develop three general-purpose variants:

- **Acc128:** Designed for typical usage (up to the birthday bound, expected limit 2^48 bits of total data under one key) with 128-bit block ciphers, primarily AES.

- **Acc256:** Intended for very high usage bounds, paried with a 256-bit block cipher. NIST is simultaneously considering the approval of Rijndael-256 (a variant of AES with 256-bit blocks) to serve this role.

- **BBBAcc (Beyond-Birthday-Bound Accordion):** Designed for extended usage beyond the birthday bound with AES, aiming for a much higher total data limit (at least 2^64 bits per key).

The proposal suggests the HCTR2 technique as the basis for these approved accordions, citing its maturity and existing deployment. Derived functions are also mentioned as input encodings that enable specific functions (ex., AEAED) from an accordion, promising enhanced security over current approved modes.

1.3 Problems Addressed and Significance

This initiative addresses several critical challenges in modern cryptography:

- **Fixed-Length Limitations:** Many existing block cipher modes operate on fixed-size blocks, requiring padding or complex constructions for variable-length data. Accordions offer a more natural and efficient solution for handling varied input lengths.

- **Security Bounds:** Current modes often have "birthday bound" limitations (ex., around 2^64 blocks for a 128-bit block cipher like AES), beyond which security may degrade. BBBAcc directly takes this by offering beyond-birthday-bound security, crucial for high-volume data processing.

- **Enhanced Functionality:** Accordions allow for derivation of functions like AEAD or tweakable encryption with stronger security guarantees, simplifying secure protocol design.

- **Future Proofing:** By exploring 256-bit block ciphers like Rijndael-256, NIST is preparing for potential future needs for larger block sizes and higher security. I would assume this is driven by advancements in classical and quantum computing.

1.4 Relevance of Other Standards and Related Concepts The document references previous NIST work that informed this development:

- **SP 800-38 series:** NIST's existing portfolio of block cipher modes, whose limitations are driving the need for accordions.

- **NIST Internal Report 8459:** Documents the specific limitations of the current mode portfolio.

- **NIST Workshops (2023 & 2024):** Public discussions that built consensus for accordion development.

- **NIST Interagency Report 8552:** Proposed technical requirements and formal goals for approved accordions.

- **Rijndael-256:** A proposed 256-bit block cipher variant of AES, intended to be standardized alongside Acc256. This is a significant development, as it directly addresses the need for wider blocks.

- **HCTR2 technique:** A specific Hash-Encrypt-Hash paradigm for designing efficient accordions, chosen for its maturity.

2. Ideas to Think About

1. **Deployment Strategy and Transition:** Given the current use of existing SP 800-38 modes, what is the plan and timeline for evaluating, testing, and potentially transitioning to these new accordion modes? Which applications are most critically impacted by the limitations of current modes (ex., high-volume data, variable-length data storage) and would benefit most from early adoption?

2. **Rijndael-256 Adoption:** How does the potential standardization of Rijndael-256 affect our cryptographic roadmap? Are there systems that could immediately leverage a 256-bit block cipher, and what are the implications for hardware acceleration and cryptographic module validations (ex., FIPS 140-3)?

3. **Security Assurance Requirements:** For which specific use cases (ex., long-term archival, high-throughput network encryption) are the beyond-birthday-bound properties of BBBAcc most critical? How do we quantify risk of exceeding birthday bounds with current modes versus the benefits of migrating to new ones?

4. **Vendor and Industry Readiness:** How will this new standardization effort impact our current cryptographic vendors and their product roadmaps? What is NIST's strategy for fostering broad industry adoption and ensuring interoperability across different implementations of these accordions?

5. **Integration with PQC Migration:** While note directly PQC algorithms, these new modes are foundational. How do these efforts align with and potentially accelerate or complicate our PQC migration strategy? Will PQC scheme rely on or integrate with these new wider-block or beyond birthday bound modes?

3. Implications & Considerations

1. **Strategic Shift Towards Wider Blocks:** The explicit development of Acc256 and the concurrent consideration of Rijndael-256 signal a significant strategic shift by NIST towards advocating for 256-bit block ciphers. This implies a future where wider blocks become standard, which could enhance security margins and efficiency for large-data applications. For governmental entities, this means evaluating the feasibility and benefits of upgrading systems to support 256-bit block ciphers, especially in the context of high-assurance and future-proof systems.

2. **Mitigating Quantum Search (Indirectly):** While AES (128-bit block) itself is a symmetric cipher, and Gover's algorithm could theoretically reduce its effective key length against quantum adversaries (ex., AES-256 would be need for ~128-bit quantum security), the move to sider 256-bit blocks (Rijndael-256) for Acc256 doesn't directly address key length quantum attacks but could offer other benefits. However, improved modes and higher security bounds reduce other attack surfaces, which is beneficial in a threat landscape that

includes quantum capabilities. For instance, achieving beyond birthday bound security for AES with BBBAcc is critical for long-term data protection even if the key is quantum secure.

3. **Complex Interdependencies:** The development of accordions introduces new dependencies and interactions with other cryptographic components. For instance, the use of tweakable encryption might affect how key management (SP 800-57, SP 800-133) or random number generation (SP 800-90A, or even the PIP RNG) are integrated. It is important to consider these systemic implications.

4. **Operational Efficiency and Performance:** These new modes promise enhanced security and flexibility. For the government client, this translates to potential operational efficiency gains by simplifying cryptographic implementations for variable-length data, reducing overhead from padding, and potentially improving throughput for high-volume applications like secure data storage or real-time communications.
5. **Long-Term Cryptographic Resilience:** By addressing limitations like birthday bounds and offering beyond birthday bound security, NIST is enhancing the long-term cryptographic resilience of systems. This proactive approach helps the governmental entities future-proof their infrastructure against evolving classical attacks and ensures that the underlying building blocks for future PQC applications are strong.

Comments for Considerations

1. Regarding the Choice of HCTR2 and Potential Alternatives:

While HCTR2 is acknowledged for its maturity and existing deployment, it might be worth considering strong, formally verified alternative designs that may offer:

- **Even greater performance characteristics** for extremely high-throughput government systems, particularly in software implementations, while maintaining security guarantees.

- **Simpler security proofs or easier integration paths** with existing hardware cryptographic accelerators where possible.

- **Explicit analysis of quantum resilience beyond key size;** for example, if any of the underlying permutations used within accordion constructions could be susceptible to quantum speedups beyond generic search algorithms.

We recommend a transparent comparison of HCTR2 against other leading “Hash-Encrypt-Hash" paradigm candidates, specifically highlighting their respective advantages/disadvantages concerning implementation complexity, provable security under various assumptions, and performance across diverse hardware platforms relevant to government deployments.

2. Emphasis on Rijndael-256 and Wider Block Adoption:

We strongly support the concurrent consideration and standardization of Rijndael-256 to serve as the underlying block cipher for ACC256. The strategic move towards 256-bit block ciphers is critical for the following reasons:

- **Enhanced Security Margin:** Wider blocks inherently offer a larger state space, which can contribute to greater long-term cryptographic resilience, especially when considering advancements in classical cryptoanalysis techniques.

- **Efficiency for Large Data:** For large-scale data storage, high volume network traffic, and big data analytics within government systems, operating on wider blocks can significantly reduce the number of block cipher calls, leading to improved throughput and efficiency.

- **Future Proofing:** This aligns with a forward-looking cryptographic roadmap, anticipating future needs that may exceed the practical limits of 128-bit block ciphers, even those with beyond birthday bound properties.

We recommend NIST prioritize the finalization of Rijndael-256 in parallel with Acc256 development to enable a timely adoption.

3. Clarification on Integration with Post-Quantum Cryptography (PQC) Efforts:

While cryptographic accordions are not PQC algorithms themselves, they form fundamental building blocks. We request NIST to provide explicit guidance or an anticipated roadmap on:

- **How PQC schemes will interface with these new modes:** Will PQC key establishment or signature schemes rely on the properties of ACC128, Acc256, or BBBAcc for their underlying symmetric operations (ex., encryption of encapsulations, hashing within signatures)?

- **The impact of wider blocks on PQC efficiency:** Could the use of Rijndael-256 with Acc256 potentially simplify or optimize certain PQC constructions that require larger intermediate states or high-bandwidth symmetric operations?

- **Guidance for hybrid PQC transitions:** As government agencies transition to hybrid PQC deployments, clear recommendations on how to integrate these new modes alongside existing or new PQC primitives will be useful for seamless and secure migration.

4. Implementation Guidance and Side-Channel Considerations:

We recommend that the final publication include:

- **More detailed implementation guidance** for Acc128, Acc256, and BBBAcc, especially regarding efficient software and hardware implementations, to facilitate rapid adoption by developers and vendors.

- **Explicit consideration of side-channel resistance:** Given the critical nature of government applications, cryptographic implementations must be resilient to side-channel attacks. We encourage NIST to include recommendations or a framework for analyzing side-channel resistance of implementations of cryptographic accordions, particularly concerning the tweak processing and internal state manipulations.

5. Comments from Byeonghak Lee (Samsung SDS) August 6, 2025

Hello, this is Byeonghak Lee from Samsung SDS.

First, we would like to thank you for the call for accordion ciphers. We have a question about the AEAD use case.

Some recent AEAD schemes [1, 2] enjoy BBB security against nonce-respecting adversaries while still being secure up to the birthday bound, even when nonces are repeated. If an accordion cipher can achieve BBB security subject to the tweak repetition limitation, then an AEAD based on this accordion cipher will achieve BBB security against adversaries with limited nonce reuse for both encryption and decryption queries. For example, Dutta and Nandi proposed a TBC-based accordion cipher with this property called THCTR [3]. Another example can be created using block ciphers, as shown below.

```
Encipher(T, M_L || M_R):
  MM = M_L XOR H(M_R)
  CC = CLRW2(T, MM)
  C_R = M_R XOR HteC(T || (MM XOR CC))
  C_L = CC XOR H(C_R)
```

where CLRW2 is a BC-based TBC construction with BBB security [4], HteC is a BC-based variable-output-length PRF with BBB security [5], H is almost xor-universal hash function, and key inputs are omitted.

We would like to ask NIST's opinion on this property.

[1] AES-GCM-SIV: Specification and Analysis, https://eprint.iacr.org/2017/168
[2] Toward a Fully Secure Authenticated Encryption Scheme From a Pseudorandom Permutation, https://eprint.iacr.org/2021/1168
[3] Tweakable HCTR: A BBB Secure Tweakable Enciphering Scheme, https://eprint.iacr.org/2019/1324
[4] Tight Security of Cascaded LRW2, https://eprint.iacr.org/2019/1495
[5] Making GCM Great Again: Toward Full Security and Longer Nonces, https://eprint.iacr.org/2025/577

Best regards,
Byeonghak

Byeonghak Lee, PhD
Security Algorithm Lab / Samsung SDS

6. Comments from Yusuke Naito (Mitsubishi Electric Corporation) August 6, 2025

Dear NIST,

I would like to support standardizing accordions designed based on the HCTR2 technique because HCTR2 [1] has a wide adoption, e.g., the Linux kernel [2] and file encryption in Android [3]. Hence, I support standardizing HCTR2 as a birthday-bound secure accordion Acc128. Meanwhile 64-bit security may not be sufficient for distributed systems as reported in [4]. Hence, I support standardizing a beyond-birthday-bound secure accordion BBBAcc. For BBBAcc, our CHCTR family is suitable. Our CHCTR family archives 2n/3-bit VIL-STPRP security in the multi-user setting when using an n-bit block cipher. The CHCTR family includes two accordions, CHCTR2 and CHCTR2-1k. CHCTR2 just iterates HCTR2 twice with two independent keys. CHCTR2-1k is a single-key variant of CHCTR2. The specifications are given in the attached file. I will publish the proofs at ePrint soon.

[1] Crowley, P., Huckleberry, N., Biggers, E.: Length-preserving encryption with HCTR2. IACR Cryptol. ePrint Arch. (2021), https://eprint.iacr.org/2021/1441.

[2] The Linux Kernel: hctr2.c. https://github.com/torvalds/linux/blob/master/crypto/hctr2.c, accessed: 2025-05-15.

[3] Android Open Source Project: File-based encryption. https://source.android.com/docs/security/features/encryption/file-based (2023).

[4] Panos Kampanakis, Matt Campagna, Eric Crocket, and Adam Petcher. Practical Challenges with AES-GCM and the need for a new mode and wide-block cipher. The Third NIST Workshop on Block Cipher Modes of Operation 2023.

Best regards,
Yusuke Naito
Mitsubishi Electric Corporation

[see the referenced attachment on the following pages]

CHCTR Family for BBBAcc

Yusuke Naito

Mitsubishi Electric Corporation

1 Introduction

We suggest a CHCTR family as BBBAcc. The CHCTR family includes two accordions with the cascaded $\mathsf{HCTR2}$ structures [1]. CHCTR is an abbreviation of cascaded HCTR. Regarding security, our accordions instantiated with an n-bit block cipher achieve $2n/3$-bit online security in the sense of multi-user variable-input-length tweakable strong-pseudorandom permutation ($\mathsf{mu\text{-}VIL\text{-}STPRP}$) in the standard model by assuming that the underlying block cipher is multi-user strong-pseudorandom permutation ($\mathsf{mu\text{-}SPRP}$) secure and the underlying keyed hash function is almost XOR universal and regular.

The first scheme of the CHCTR family is $\mathsf{CHCTR2}$, which just iterates $\mathsf{HCTR2}$ twice with two independent keys. Hence, $\mathsf{CHCTR2}$ can be implemented by using an $\mathsf{HCTR2}$ implementation as a black box.

The second scheme of the CHCTR family is $\mathsf{CHCTR2\text{-}1k}$, which is a single-key variant of $\mathsf{CHCTR2}$.

In the following sections, we give specifications of $\mathsf{CHCTR2}$ and $\mathsf{CHCTR2\text{-}1k}$. The proofs will be published at ePrint soon.

2 Notations

Let ε be the empty string. Let $\{0,1\}^i$ be the set of all i-bit strings and $\{0,1\}^0 \coloneqq \{\varepsilon\}$. Let $\mathsf{msb}_i(X)$ be the most significant i bits of a bit string X. Let $\mathtt{bin}_n(i)$ be an n-bit representation of an integer $i \geq 0$. For integers $0 \leq i \leq j$, let $[i,j] \coloneqq \{i, i+1, \ldots, j\}$ and $[j] \coloneqq [1,j]$. If $j < i$ then $[i,j] \coloneqq \emptyset$. Let $|D|_n \coloneqq \lceil |D|/n \rceil$ be an n-bit block length of D. For a value or a set X, $Y \leftarrow X$ means that X is assigned to Y. $(D_1, D_2) \xleftarrow{n,|D|-n} D$ means that a bit string D is partitioned into the n-bit string D_1 and the $(|D|-n)$-bit string D_2 such that $D = D_1 \| D_2$, and $(D_1, \ldots, D_\ell) \xleftarrow{n} D$ means that D is partitioned into the n-bit strings $D_1, \ldots, D_\ell$ such that $|D_i| = n$ for $i = 1, \ldots, \ell - 1$, $|D_\ell| \leq n$, and $D = D_1 \| \cdots \| D_\ell$.

3 Primitives

We fix the key and block sizes of the underlying block ciphers to k and n bits. Let t_{max} and m_{max} be the maximum n-bit block lengths of tweaks and plaintexts, respectively. Let $\mathcal{T} \coloneqq \{T \in \{0,1\}^* \mid |T| \leq t_{\mathsf{max}} n$ and $\mathcal{M} \coloneqq \{M \in \{0,1\}^* \mid n \leq |M| \leq m_{\mathsf{max}} n\}$ be the sets of tweaks and plaintexts, respectively.

3.1 Block Cipher (BC)

Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be an encryption of a BC and $E^{-1} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be its decryption. For a key $K \in \{0,1\}^k$, the encryption is denoted by $E_K(\cdot) = E(K,\cdot)$. The decryption is denoted by $E_K^{-1}(\cdot) = E^{-1}(K,\cdot)$. Let $E_K^{\pm} \coloneqq (E_K, E_K^{-1})$.

3.2 Keyed Hash Function

$\mathsf{HCTR2}$ and the CHCTR family use a family of keyed hash functions where the key space is $\{0,1\}^n$. Let $\mathsf{H} : \{0,1\}^n \times \mathcal{T} \times \mathcal{M} \to \{0,1\}^n$ be a family of keyed hash functions. H with a key $H \in \{0,1\}^n$ is denoted by H_H.

3.3 HCTR2

$\mathsf{HCTR2}$ [1] is a birthday-bound-secure accordion based on a BC E_K and a keyed hash function H_H. Let $K \in \{0,1\}^k$ be a BC key. The hash key is defined by using E_K: $H = E_K(\mathtt{bin}_n(0))$. $\mathsf{HCTR2}$ has three rounds. In the first round, H_H is performed. In the second round, E_K is performed once to generate IV of the encryption scheme $\mathsf{XCTR}[E_K]$, and then $\mathsf{XCTR}[E_K]$ is performed. In the third round, H_H is performed.

$\mathsf{HCTR2}$ consists of encryption and decryption algorithms. These algorithms and the underlying modes $\mathsf{HCTR2core}$ and XCTR are defined in Algorithm 1 and depicted in Fig. 1. $\mathsf{HCTR2core}$ is a core algorithm of $\mathsf{HCTR2}$ that can be used in common by the encryption and decryption algorithms. P_1 and P_2 are n-bit permutations used in the left and right parts, respectively. In $\mathsf{HCTR2}$, $P_1 = E_K$ in the encryption, $P_1 = E_K^{-1}$ in the decryption, and $P_2 = E_K$. $\mathsf{HCTR2}$ is the encryption algorithm that takes a plaintext $M \in \mathcal{M}$ and a tweak $T \in \mathcal{T}$, and returns the ciphertext $C \in \mathcal{M}$. $\mathsf{HCTR2}^{-1}$ is the decryption one that takes a ciphertext $C \in \mathcal{M}$ and a tweak $T \in \mathcal{T}$, and returns the plaintext $M \in \mathcal{M}$.

4 CHCTR Family

We present a CHCTR family of $2n/3$-bit secure accordions. The CHCTR family consists of two accordions $\mathsf{CHCTR2}$ and $\mathsf{CHCTR2\text{-}1k}$. Our concrete proposal uses AES [4] for the BC E and $\mathsf{POLYVAL}$ [2, 3] for the family of the keyed hash functions H.

4.1 CHCTR2

$\mathsf{CHCTR2}$ just iterates $\mathsf{HCTR2}$ twice, and thus can be implemented by using an $\mathsf{HCTR2}$ implementation as a black box. It uses two independent keys K_1 and K_2, which are respectively used for the first and second $\mathsf{HCTR2}$ calls. The encryption $\mathsf{CHCTR2}$ and the decryption $\mathsf{CHCTR2}^{-1}$ are defined in Algorithm 2. The encryption is depicted in Figure 2. The encryption takes a plaintext $M \in \mathcal{M}$

Algorithm 1 HCTR2

Encryption $\mathsf{HCTR2}[E_K, \mathsf{H}](T, M)$

$H \leftarrow E_K(\mathtt{bin}_n(0))$; $L \leftarrow E_K(\mathtt{bin}_n(1))$
$C \leftarrow \mathsf{HCTR2core}[E_K, E_K, \mathsf{H}](H, L, T, M)$
return C

Decryption $\mathsf{HCTR2}^{-1}[E_K^{\pm}, \mathsf{H}](T, C)$

$H \leftarrow E_K(\mathtt{bin}_n(0))$; $L \leftarrow E_K(\mathtt{bin}_n(1))$
$M \leftarrow \mathsf{HCTR2core}[E_K^{-1}, E_K, \mathsf{H}](H, L, T, C)$
return M

Core Procedure $\mathsf{HCTR2core}[P_1, P_2, \mathsf{H}](H, L, T, D)$

$(D_0, D_*) \xleftarrow{n, |D|-n} D$
$X_0 \leftarrow D_0 \oplus \mathsf{H}_H(T, D_*)$
$Y_0 \leftarrow P_1(X_0)$; $IV \leftarrow X_0 \oplus Y_0 \oplus L$; $D'_* \leftarrow \mathsf{XCTR}[P_2](IV, D_*)$
$D'_0 \leftarrow Y_0 \oplus \mathsf{H}_H(T, D'_*)$
$D' \leftarrow D'_0 \| D'_*$
return D'

Subroutine $\mathsf{XCTR}[P_2](IV, D_*)$

for $i = 1, \ldots, |D_*|_n$ **do** $Y_i \leftarrow P_2(IV \oplus \mathtt{bin}_n(i))$ **end for**
$Y \leftarrow \mathsf{msb}_{|D_*|}(Y_1 \| \cdots \| Y_{|D_*|_n})$; $D'_* \leftarrow D_* \oplus Y$
return D'_*

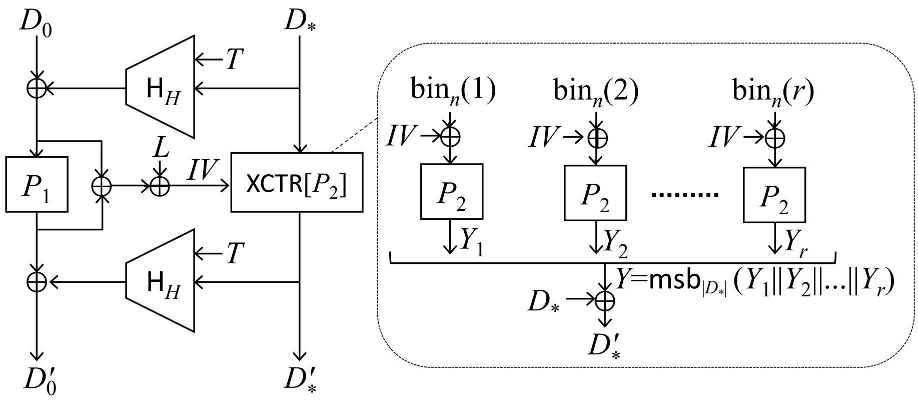

Fig. 1. $\mathsf{HCTR2core}$ and XCTR. The hash key is defined as $H = E_K(\mathtt{bin}_n(0))$ and the mask value is defined as $L = E_K(\mathtt{bin}_n(1))$. In the encryption of $\mathsf{HCTR2}$, $P_1 = P_2 = E_K$. In the decryption of $\mathsf{HCTR2}$, $P_1 = E_K^{-1}$ and $P_2 = E_K$.

and a tweak $T \in \mathcal{T}$, and returns the ciphertext $C \in \mathcal{M}$. The decryption takes a ciphertext $C \in \mathcal{M}$ and a tweak $T \in \mathcal{T}$, and returns the plaintext $M \in \mathcal{M}$.

With a polynomial hash function with a homomorphic property such as POLYVAL [2, 3], i.e., $\mathsf{H}_{H_1 \oplus H_2}(T, R) = \mathsf{H}_{H_1}(T, R) \oplus \mathsf{H}_{H_2}(T, R)$, the two hash

Algorithm 2 $\mathsf{CHCTR2}$

Encryption $\mathsf{CHCTR2}[E_{K_1}, E_{K_2}, \mathsf{H}](T, M)$

1: $D \leftarrow \mathsf{HCTR2}[E_{K_1}, \mathsf{H}](T, M)$
2: $C \leftarrow \mathsf{HCTR2}[E_{K_2}, \mathsf{H}]\,(T, D)$
3: **return** C

Decryption $\mathsf{CHCTR2}^{-1}[E^{\pm}_{K_1}, E^{\pm}_{K_2}, \mathsf{H}](T, C)$

1: $D \leftarrow \mathsf{HCTR2}^{-1}[E^{\pm}_{K_2}, \mathsf{H}](T, C)$
2: $M \leftarrow \mathsf{HCTR2}^{-1}[E^{\pm}_{K_1}, \mathsf{H}]\,(T, D)$
3: **return** M

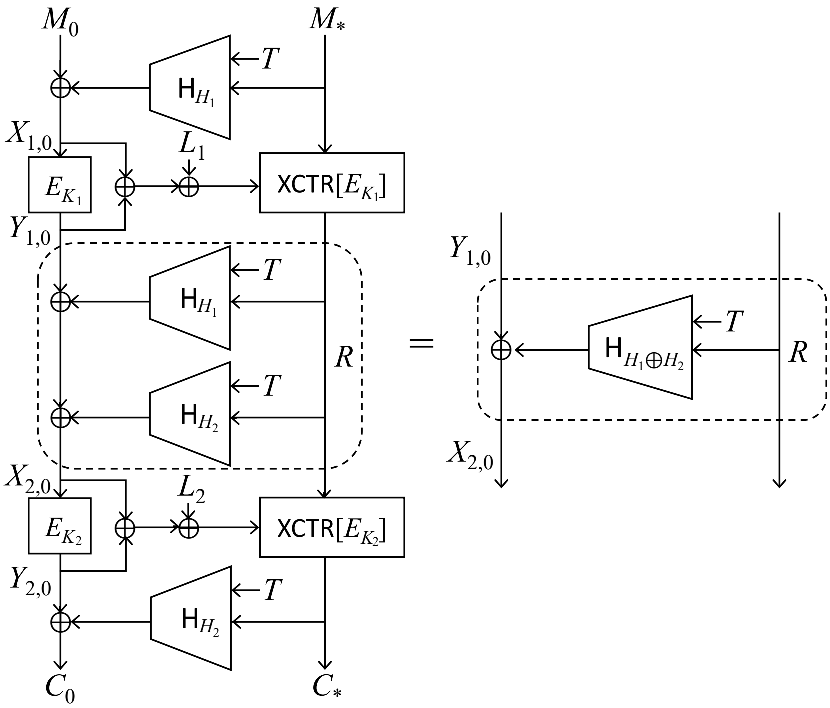

Fig. 2. Encryption of $\mathsf{CHCTR2}$. The hash keys are defined as $H_1 = E_{K_1}(\mathtt{bin}_n(0))$ and $H_2 = E_{K_2}(\mathtt{bin}_n(0))$. The mask values are defined as $L_1 = E_{K_1}(\mathtt{bin}_n(1))$ and $L_2 = E_{K_2}(\mathtt{bin}_n(1))$.

evaluations in the middle layer of $\mathsf{CHCTR2}$ can be merged with the single hash evaluation with the key $H_1 \oplus H_2$. Hence, $\mathsf{CHCTR2}$ can be performed with two BC calls, three hash calls, and two XCTR calls.

Algorithm 3 CHCTR2-1k

Encryption $\mathsf{CHCTR2\text{-}1k}[E_K, \mathsf{H}](T, M)$

$H_1 \leftarrow E_K(\mathtt{bin}_n(0))$; $H_2 \leftarrow E_K(\mathtt{bin}_n(2))$
$D \leftarrow \mathsf{HCTR2core}[E_K, E_K, \mathsf{H}](H_1, 0^n, T, M)$
$C \leftarrow \mathsf{HCTR2core}[E_K, E_K, \mathsf{H}]\,(H_2, 0^n, T, D)$
return C

Decryption $\mathsf{CHCTR2\text{-}1k}^{-1}[E_K^{\pm}, \mathsf{H}](T, C)$

$H_1 \leftarrow E_K(\mathtt{bin}_n(0))$; $H_2 \leftarrow E_K(\mathtt{bin}_n(2))$
$D \leftarrow \mathsf{HCTR2core}[E_K^{-1}, E_K, \mathsf{H}](H_2, 0^n, T, C)$
$M \leftarrow \mathsf{HCTR2core}[E_K^{-1}, E_K, \mathsf{H}]\,(H_1, 0^n, T, D)$
return M

4.2 CHCTR2-1k

CHCTR2-1k is a single-key variant of CHCTR2 and can be implemented by using an HCTR2core implementation as a black box. Note that, in CHCTR2-1k, the XORs with L_1 and L_2 are absent, and HCTR2core can be called as a black box with the zero mask $L = 0^n$. Let K be the BC's key. The hash keys are defined as $H_1 = E_K(\mathtt{bin}_n(0))$ and $H_2 = E_K(\mathtt{bin}_n(2))$. The encryption CHCTR2-1k and the decryption $\mathsf{CHCTR2\text{-}1k}^{-1}$ are defined in Algorithm 3. The encryption is depicted in Figure 3. The encryption takes a plaintext $M \in \mathcal{M}$ and a tweak $T \in \mathcal{T}$, and returns the ciphertext $C \in \mathcal{M}$. The decryption takes a ciphertext $C \in \mathcal{M}$ and a tweak $T \in \mathcal{T}$, and returns the plaintext $M \in \mathcal{M}$.

With a polynomial hash function with the homomorphic property, the two hash evaluations in the middle layer of CHCTR2-1k can be merged with the single hash evaluation with the key $H_1 \oplus H_2$. Hence, CHCTR2-1k can be performed with two BC calls, three hash calls, and two XCTR calls.

References

[1] Crowley, P., Huckleberry, N., Biggers, E.: Length-preserving encryption with HCTR2. IACR Cryptol. ePrint Arch. (2021), `https://eprint.iacr.org/2021/1441`

[2] Gueron, S., Langley, A., Lindell, Y.: AES-GCM-SIV: specification and analysis. IACR Cryptol. ePrint Arch. p. 168 (2017)

[3] Gueron, S., Langley, A., Lindell, Y.: AES-GCM-SIV: Nonce misuse-resistant authenticated encryption. RFC **8452**, 1–42 (2019)

[4] NIST: Advanced encryption standard (AES). FIPS 197 (2023), published November 26, 2001; Updated May 9, 2023, `https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf`

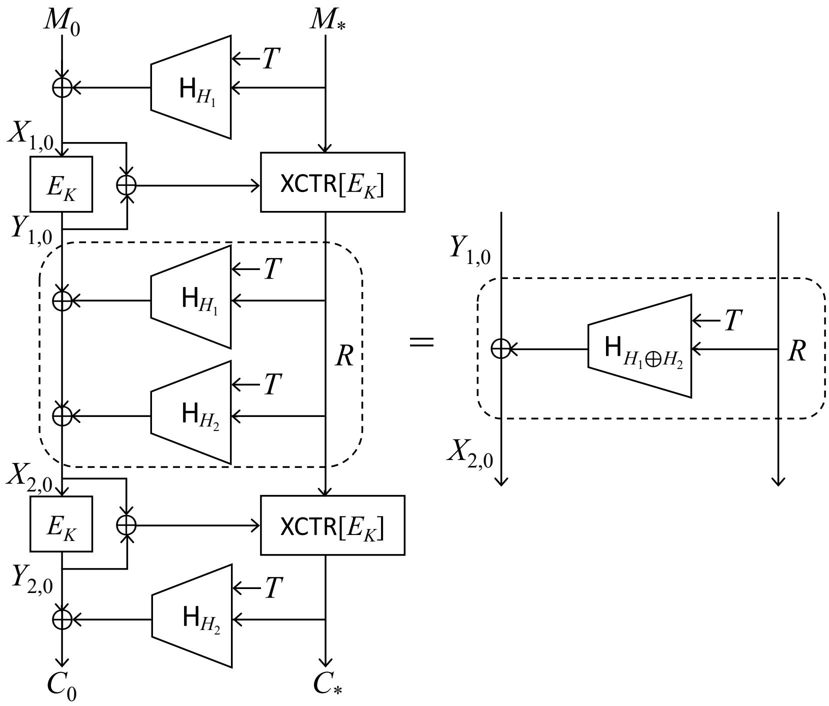

Fig. 3. Encryption of CHCTR2-1k. The hash keys are defined as $H_1 = E_K(\mathtt{bin}_n(0))$ and $H_2 = E_K(\mathtt{bin}_n(2))$.

7. Comments from John Preuß Mattsson (Ericsson) August 6, 2025

Dear NIST,

Thanks for your continuous efforts to produce well-written open-access security documents.

Please find attached Ericsson’s comments on the NIST’s proposal for accordion development.

Best Regards,

John Preuß Mattsson

[see Ericsson’s comments on the following pages]

ERICSSON

Date: August 6, 2025

Ericsson AB
Group Function Technology
SE-164 80 Stockholm
SWEDEN

Comments on Accordion Development

Dear NIST,

Thank you for your continued efforts to produce well-written, user-friendly, and open-access security documents. Well-designed accordions with derived functions [1] could provide significantly improved properties compared to many of the cipher modes currently approved by NIST. Beyond very strong cryptographic properties, we believe the derived functions should prioritize usability and usable security. Interfaces and guidelines should be designed to minimize the demands on users and implementers, as well as the adverse consequences of human mistakes [2]. Ideally, usage limits should be high enough that they need not be considered in practice. The practical adoption of the derived functions will heavily depend on performance and other properties. We think the accordion project should design for the next 50 years, not for the most limiting existing AES APIs. The standardization effort should analyze and standardize several accordions with a common set of derived functions.

Please find below Ericsson's detailed comments on NIST proposal [3] to develop three general-purpose accordions:

- *"A cryptographic accordion is a tweakable, variable-input-length strong pseudorandom permutation (VIL-SPRP) constructed from an underlying block cipher"*

  We do not think NIST should define accordions as block cipher modes, even if the three initially proposed accordions are all based on block ciphers. Looking ahead, we hope to see accordions, both in the literature and standardized by NIST, CFRG, based on other primitives such as Keccak, Ascon, AEGIS, or SNOW 5G, which reuse the NIST-standardized derived functions for accordions. Furthermore, support for variable-length tweaks is essential for certain derived functions and should be included in the definition. We propose the following formulation:

  "A cryptographic accordion is a variable-input-length strong pseudorandom permutation (VIL-SPRP) with support for variable-length tweaks, for example constructed from an underlying block cipher"

- We think Acc128, Acc256, and BBBAcc, based on HCTR2 [4], provide a strong starting point if one limits oneself to block-cipher modes. However, it's difficult to assess whether all three are necessary, or whether they are sufficient, until their security, complexity, limits, and performance characteristics are better understood.

- The security of Acc128, Acc256, and BBBAcc are all based AES/Rijndael. A necessity for cryptographic agility is to have a cryptographic primitive to switch to. With the deprecation of Triple DES, NIST does not have a standardized alternative to be used for encryption in the event that AES/Rijndael is broken. As stated in [5], we think NIST should also standardize an accordion based of Keccak [6–7] to enable cryptographic agility.

- We appreciate that NIST is proposing concrete solutions by suggesting HCTR2 as a foundation. However, many high-level questions about the overall goals remain unresolved. Are accordions intended to replace AES-GCM, or are they meant to serve as high-security options for use cases where performance is less critical? If the goal is to replace AES-GCM, then very high performance would be required, likely achievable only through constructions based on the AES round function, such as AEGIS [8], or modern stream ciphers using SIMD instructions, like SNOW 5G [9]. Similarly, NIST has recently proposed standardizing functions for deriving key-nonce pairs for GCM [10]. It remains unclear whether AEAD derived functions for accordions are intended to be used with similar key-nonce derivation mechanisms or to operate independently. If the final total data limits are indeed 2^{48} and 2^{64} bits, the use of a separate key-nonce derivation function will likely be necessary. We recommend that NIST provide updated guidance on the overall goals.

- NIST's requirements for cryptographic accordions [1] states that an accordion should support Beyond-Birthday-Bound (BBB) security to the highest extent reasonable. We note that Acc128, and Acc256 do not support BBB at all, which contradicts these requirements. As we previously stated in [5], we believe the term "beyond-birthday-bound security" should not be used as a requirement. Similarly [1] states that an accordion should support Key-Dependent-Input (KDI) security to the highest extent reasonable, while also stating that KDI security is generally unachievable and unlikely to be relevant for the intended use cases. We recommend that NIST provide updated guidance on the requirements for accordions. Academic research into the multi-key security of HCTR2 would be highly valuable.

- We like the XCTR mode used in HCTR2 [4]. Using a little-endian block counter and XORing it with the value S appears to be an optimal construction for implementing counter mode on modern CPUs.

- *"Acc128 would support an underlying block cipher with 128-bit blocks (i.e., AES). Due to the birthday bound, NIST expects to limit the total data processed under a single key to 2^{48} bits."*

  This is a stricter formulation than IR 8552 [1] that has the requirement of **at least** 2^{48} bits. We would expect this limit to be ≈ 2^{59} blocks = 2^{66} bits. As shown in Section 4 of [11], the entropy loss of a pseudorandom permutation (PRP) in counter mode, which provides an upper bound on the amount of information an attacker can theoretically recover, is $\approx \sigma^2 \,/\, 2^b \,/\, \ln 4$ where σ is the number of encrypted blocks. This result also apply to the XCTR mode used in HCTR2. We agree with ANSSI [12] and EUCC [13] that $\sigma \lesssim\ 2^{59}$ blocks is a suitable limit for non-BBB block cipher modes. This appears to be a well-thought-out and balanced requirement, effectively limiting the amount of information an attacker could recover to no more than $\approx 2^{-10.47} \approx 0.0007$ bits of entropy. NIST has given no motivation for the 2^{48} bits = 2^{41} blocks limit. Unless NIST provides some convincing arguments, we think the limits for Acc128, if standardized, should be 2^{59} blocks.

We also think NIST should follow the recommendation from the Accordion workshop and use byte alignment instead of bit alignment. This considerably reduces complexity in implementation and testing.

- *"BBBAcc would support extended usage beyond the birthday bound with AES. NIST expects the analogous limit on the total data processed under a single key to be at least 2^{64} bits."*

  2^{64} bits = 2^{57} blocks seems low for a mode that claims to offer beyond-birthday-bound security. Can a cipher with a 2^{57}-block limit, below the commonly accepted birthday-bound limit of 2^{59} blocks [11–13], even be called beyond-birthday-bound? 2^{57} blocks falls short of NIST's own definition of BBB security in Section 2.2.2 of [1], which states that the advantage should be negligibly small, even when 2^{64} or more blocks of input are processed. NIST has given no motivation for the 2^{48} and 2^{64}-bit limits. We believe NIST should clarify its rationale and share its reasoning behind the proposed usage limits. While many current NIST specifications for encryption and PRFs should have stricter limits, the suggested limits for Acc128 and BBBAcc seem too strict.

  Furthermore, it would be beneficial for NIST to clarify how it intends to modify HCTR2 in the design of BBBAcc. Is NIST planning to use the approach in [14] or something else? Beyond-Birthday-Bound (BBB) designs typically suffer from lower performance, more complex security proofs, and greater implementation challenges compared to birthday-bound designs. BBBAcc likely needs to use a 256-bit hash function $H_{\bar{h}}$ and a 256-bit block cipher E_k internally, which is complex and overlaps with the much simpler Acc256 design. As stated in [15], *"one must balance the risk of flaws in an increasingly complex implementation with the risk of a cryptanalytic breakthrough. Because more security products fail due to implementation or configuration errors than failures in their underlying cryptographic algorithms, spending limited resources to add cryptographic complexity can at times weaken security rather than improve it."*

- We note that 2^{48} and 2^{64} bits are less than the $\approx 2^{71}$ bits = 2^{64} blocks limit currently allowed in the AES-GCM specification SP 800-38D. One stated motivation for accordions is to process more data than is possible with AES-GCM. With much more restrictive limits than AES-GCM, the value of standardizing Acc128 and BBBAcc is questionable.

- Given an approved 256-bit block cipher, Acc256 is the easiest to analyze with its excellent security, low complexity, and good limits. The only downside is incompatibility with some existing AES implementations and non-optimal performance on other AES implementations. A new universal hash function using a polynomial over GF(2^{256}), or a finite field of similar size, would have to be defined, but this should be relatively straightforward. As Acc256 primarily targets processors that support carryless multiplication, a binary field is likely faster and simpler than a prime field [4]. If development of an accordion is higher priority than the standardization of a wide block cipher, we think an accordion based on Keccak, preferable with twelve rounds, is the most straightforward solution [6–7].

Regarding Rijndael-256, we have trust in Rijndael-256 as currently specified [16]. ETSI SAGE and 3GPP have recently standardized the use of Rijndael-256 for authentication and key generation in MILENAGE-256 [17]. We do not consider related-key attacks to be of practical concern. However we agree with comments that the AES/Rijndael key schedule is non-optimal and could be improved [18]. A redesigned key schedule could offer stronger security properties and potentially even improved performance compared to the current key expansion. In contrast to the NSA [16], we believe a revised key schedule should be non-linear to ensure better diffusion.

Figure 1 illustrates an example key schedule that takes a 128-bit key and generates 256 bytes of pseudo random values that are not very strong as a keystream but strong enough to serve as round keys. The implementation can likely be optimized further. Something similar could be used as a modified Rijndael-256 key schedule.

- The discussion on accordions, Rijndael-256, and PQC algorithms clearly demonstrate that hardware and software APIs which do not expose subfunctions, such as the AES round function and KECCAK-$p[b, n_r]$, greatly limit innovation and can significantly reduce the performance (or security) of future standards. We recommend that NIST strongly encourage implementations to support the AES round function and KECCAK-$p[b, n_r]$. These subfunctions should also be testable for conformance under the Cryptographic Algorithm Validation Program (CAVP).

John Preuß Mattsson,
Expert Cryptographic Algorithms and Security Protocols, Ericsson
On behalf of the Ericsson Cryptography Team

```
void KeyExpansion128(uint8_t* expKey, uint8_t K[16],
uint64_t Tweak, uint8_t n, uint8_t t, uint32_t domain)
{
  __m512i x0 = Tweak||n||t||domain||0;
  __m512i x1 = Tweak||n||t||domain||1;
  __m512i x2 = Tweak||n||t||domain||2;
  __m512i x3 = Tweak||n||t||domain||3;
  __m512i key = load(K);

  x0 = _mm512_xor_si512(x0, key);
  x1 = _mm512_xor_si512(x1, key);
  x2 = _mm512_xor_si512(x2, key);
  x3 = _mm512_xor_si512(x3, key);

  for (int i = 1; i < 8; i++)
  {
    x0 = _mm512_aesenc_epi128(x0, key);
    x1 = _mm512_aesenc_epi128(x1, key);
    x2 = _mm512_aesenc_epi128(x2, key);
    x3 = _mm512_aesenc_epi128(x3, key);
  }

  _mm512_storeu_si512(expKey + 64 * 0, x0);
  _mm512_storeu_si512(expKey + 64 * 1, x1);
  _mm512_storeu_si512(expKey + 64 * 2, x2);
  _mm512_storeu_si512(expKey + 64 * 3, x3);
}
```

Figure 1. Example key schedule that takes a 128-bit key and generates 256 bytes of pseudo random values that are not very strong as a keystream but strong enough to serve as round keys. The implementation can likely be optimized further. Something similar could be used as a modified Rijndael-256 key schedule.

References

[1] Requirements for Cryptographic Accordions
https://nvlpubs.nist.gov/nistpubs/ir/2025/NIST.IR.8552.pdf

[2] NIST Cryptographic Standards and Guidelines Development Process
https://nvlpubs.nist.gov/nistpubs/ir/2016/nist.ir.7977.pdf

[3] PRE-DRAFT Call for Comments: NIST Launches Development of Cryptographic Accordions
https://csrc.nist.gov/pubs/sp/800/197/a/iprd

[4] Length-preserving encryption with HCTR2
https://eprint.iacr.org/2021/1441.pdf

[5] Comments on NIST's Requirements for an Accordion Cipher
https://csrc.nist.gov/csrc/media/Events/2024/accordion-cipher-mode-workshop-2024/documents/papers/comments-on-NIST-reqs-accordion-cipher.pdf

[6] SHAKE modes of operation
https://csrc.nist.gov/csrc/media/Presentations/2023/shake-modes-of-operation/images-media/sess-3-daemen-bcm-workshop-2023.pdf

[7] Deck-Based Wide Block Cipher Modes
https://csrc.nist.gov/csrc/media/Presentations/2023/deck-based-wide-block-cipher-modes/images-media/sess-3-mennink-bcm-workshop-2023.pdf

[8] The AEGIS Family of Authenticated Encryption Algorithms
https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-aegis-aead

[9] SNOW-Vi: an extreme performance variant of SNOW-V for lower grade CPUs
https://eprint.iacr.org/2021/236.pdf

[10] Pre-Draft Call for Comments: GCM and GMAC Block Cipher Modes of Operation
https://csrc.nist.gov/pubs/sp/800/38/d/r1/iprd

[11] Collision-Based Attacks on Block Cipher Modes - Exploiting Collisions and Their Absence
https://eprint.iacr.org/2024/1111.pdf

[12] Guide des Mécanismes cryptoraphiques
https://cyber.gouv.fr/sites/default/files/2021/03/anssi-guide-mecanismes_crypto-2.04.pdf

[13] Agreed Cryptographic Mechanisms
https://certification.enisa.europa.eu/document/download/a845662b-aee0-484e-9191-890c4cfa7aaa_en?filename=ECCG%20Agreed%20Cryptographic%20Mechanisms%20version%202.pdf

[14] A BBB Secure Accordion Mode from HCTR
https://csrc.nist.gov/csrc/media/Presentations/2024/a-bbb-secure-accordion-mode-from-hctr/images-media/sess-8-lee-acm-workshop-2024.pdf

[15] The Commercial National Security Algorithm Suite 2.0 and Quantum Computing FAQ
https://media.defense.gov/2022/Sep/07/2003071836/-1/-1/0/CSI_CNSA_2.0_FAQ_.PDF

[16] The Rijndael Block Cipher
https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf

[17] Specification of the MILENAGE-256 algorithm set
https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=4243

[18] Pre-Draft Public Comments on SP 800-197
https://csrc.nist.gov/files/pubs/sp/800/197/iprd/docs/sp800-197-pre-draft-public-comments.pdf