HQC Post-Quantum Cryptography Decryption with Soft-Decision Reed-Solomon Decoder

Jiaxuan Cai and Xinmiao Zhang
Dept. of Electrical & Computer Engineering, The Ohio State University, Columbus, OH 43210 U.S.A.
{cai.1072, zhang.8952}@osu.edu

***Abstract*—Hamming Quasi-Cyclic (HQC) has been selected in the last round of the post-quantum cryptography standardization. The HQC decryption involves decoding a concatenated Reed-Muller (RM) code and Reed-Solomon (RS) code. The error-correcting capability of the decoding algorithms decides the codeword length, which in turn determines the key size and the ciphertext size, and affects the implementation complexity. Previous HQC instantiations have adopted hard-decision RS decoding, which is straightforward but offers limited error-correcting performance. This paper proposes soft-decision RS decoding for HQC in order to improve the correction capability and accordingly reduce the codeword length. Soft information is extracted from intermediate results of RM decoding to determine the reliability of RS decoder input symbols with small hardware overhead. The trade-off between error-correcting performance gain and implementation complexity is analyzed across various soft-decision decoding schemes, and erasure-only RS decoding is selected. Furthermore, low-complexity hardware architectures are developed for the soft information extraction and erasure-only RS decoder. For HQC-128, the proposed soft-decision decoding achieves a 13% reduction in codeword length compared to the current HQC specification with smaller estimated logic area and shorter latency for overall HQC decryption module.**

I. Introduction

In response to the challenges of quantum computing posing to conventional cryptosystems, the National Institute of Standards and Technology (NIST) initiated the post-quantum cryptography (PQC) standardization process [1]. In the most recent round, the code-based finalists included the McEliece cryptosystem [2]–[5], Bit Flipping Key Encapsulation (BIKE) [6]–[8], and Hamming Quasi-Cyclic (HQC) [9], and the HQC was selected for final standardization [10].

The HQC decryption process is composed of polynomial multiplication and subtraction, followed by the decoding of a concatenated Reed-Muller (RM) and Reed-Solomon (RS) code. To satisfy the security requirements of HQC-λ, the decoding failure rate (DFR) must be below $2^{-\lambda}$ [9]. The DFR is decided by the error-correcting performance of the decoding algorithms and the lengths of the codes.

The polynomial operations have been optimized in [11], [12]. The designs in [13]–[15] put together existing hardware architectures for the steps to implement the overall HQC system. The data flow was manually optimized to reduce the latency. High-level synthesis [16] and hardware-software co-design [17] were also explored in prior work. Algorithmic optimizations have been proposed for constrained processors [18], and a unified architecture for different security levels is designed in [19]. However, previous implementations use hard-decision RS decoders. Compared to hard-decision RS decoding, RS decoding algorithms that utilize soft inputs, which are reliability information of the received symbols, can achieve the target DFR with a shorter codeword length. This means that the sizes of the keys and ciphertexts can be reduced.

This paper proposes an efficient soft-decision RS decoding solution for the HQC system. Our proposed scheme effectively utilizes the soft reliability information generated during RM decoding to greatly improve the error-correcting performance and correspondingly reduce the codeword length without causing any hardware overhead. First, the type of soft information that can be made available and the error-correcting performance of the RS decoder are jointly evaluated. By analyzing the achievable error-correcting performance gain, the complexity of the soft-decision RS decoder, and complexity of soft information extraction of various schemes, it was proposed to utilize erasure-only RS decoder. The extracted soft reliability information is ranked to identify the least reliable symbol positions, which are marked as erasures, to improve the error-correcting capability. Then low-complexity hardware architectures are developed for the soft information extraction and erasure-only RS decoder. For HQC-128, the proposed design reduces the codeword length by 13% compared to the current specification while achieving smaller estimated area and shorter latency for the overall HQC decryption module.

II. Background Information

The HQC-public key encryption (PKE) scheme is composed of three components: key generation, encryption, and decryption. It operates over binary field $GF(2)$ in the polynomial quotient ring $\mathcal{R} = GF(2)[X]/(X^n - 1)$. Each polynomial in the ring can be also represented by a vector consisting of its coefficients. In key generation, $\mathbf{x}$ and $\mathbf{y}$ of fixed weight w, along with $\mathbf{h}$, are sampled from $\mathcal{R}$. Then the secret key and public key are generated as $\mathbf{sk} = (\mathbf{x}, \mathbf{y})$ and $\mathbf{pk} = (\mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{hy})$, respectively. The encryption randomly samples $\mathbf{r_1}$ and $\mathbf{r_2}$ of weight w_r and error polynomial $\mathbf{e}$ with weight w_e from $\mathcal{R}$. Then the public key is used to encrypt a message $\mathbf{m}$ into a ciphertext $\mathbf{c} = (\mathbf{u} = \mathbf{r}_1 + \mathbf{hr}_2, \mathbf{v} = \mathbf{mG} + \mathbf{sr}_2 + \mathbf{e})$, where $\mathbf{mG}$ denotes the codeword generated by encoding $\mathbf{m}$ using concatenated RM and RS codes. In decryption, the polynomial $\mathbf{v} - \mathbf{uy}$ is processed by a concatenated RM and RS decoder to

This material is based upon work supported by the National Science Foundation under Award No. 2052641.

TABLE I
HQC PARAMETER SETS [9]

Sec. level (bits)	RM code	RS code	n	w	$w_r = w_e$
128	$[384, 8]$	$[46, 16]$	17669	66	75
192	$[640, 8]$	$[56, 24]$	35851	100	114
256	$[640, 8]$	$[90, 32]$	57637	131	149

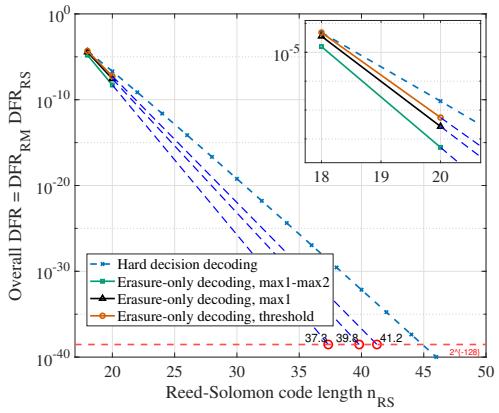

Fig. 1. DFRs of the concatenated RM $[384, 8]$ and RS $[N_{RS}, 16]$ code for HQC-128.

retrieve the original message $\mathbf{m}$. The version of HQC selected for final standardization is Key Encapsulation Mechanism (KEM), which is constructed from the PKE components to securely establish a shared secret key. However, the same concatenated RM and RS codes are utilized.

The HQC decryption carries out RM decoding followed by RS decoding. Denote the length and dimension of an RM (RS) code by $[n, k]_{RM(RS)}$. To avoid algebraic attacks, the size of the ring, n, is the smallest primitive prime greater than $n_{RM}n_{RS}$ [9]. Table I lists the parameters to achieve 128, 192, and 256 bits of security. RM codes are defined over $GF(2)$. The lowest $n_{RM} \cdot n_{RS}$ bits of $\mathbf{v} - \mathbf{uy}$ are taken as the input to the concatenated decoding. They are grouped into n_{RS} vectors of n_{RM} bits, and each vector is individually processed by the RM decoder. The output of each RM decoding instance has k_{RM} bits and is interpreted as a single symbol over $GF(2^{k_{RM}})$. The RS code is constructed over $GF(2^{k_{RM}})$, and each symbol of RM decoder output becomes one input symbol for RS decoding. Representing the RS decoding output symbol in binary format gives the message $\mathbf{m}$.

The decoding of RM codes can be carried out by the fast Hadamard transform (FHT) algorithm [20], whose outputs include $2^{k_{RM}-1}$ integers. The binary representation of the index of the integer with the largest magnitude along with its sign are used to form the k_{RM}-bit decoding output. Hard-decision RS decoding consists of syndrome computation, key equation solver (KES), and Chien search steps. The syndrome computation and Chien search are straightforward. The KES can be efficiently implemented by the enhanced parallel inversionless Berlekamp-Massey (ePIBM) algorithm [21].

The DFR of the concatenated code is the product of the DFRs of the RM and RS codes. To achieve λ bits of security, the number of message bits for RS codes needs to be λ. Besides, the DFR is required to be under $2^{-\lambda}$ to avoid various attacks [9]. For the same code dimension and same decoding algorithm, the DFR decreases as the codeword length increases. Assuming the parameters of the RM code remain fixed, to reduce the key length, the minimum n_{RS} to use for a certain decoding algorithm can be determined from a plot of the DFR vs. n_{RS} curve. Such a curve for $\lambda = 128$ and an RS code of dimension $k_{RS} = 16$ over $GF(2^8)$ with hard-decision decoding is depicted in Fig. 1 from theoretical analysis [9]. It can be observed that n_{RS} needs to be at least 45 to reach a DFR lower than $2^{-\lambda}$. Since $k_{RS} = 16$, n_{RS} must be an even number and $n_{RS} = 46$ is chosen in the HQC proposal [9].

III. SOFT-DECISION RS DECODING FOR HQC

This section proposes to utilize soft-decision decoding for HQC decryption to lower the DFR and accordingly reduce the codeword length and key size. Various schemes trading off error-correcting performance and complexity have been investigated. It is discovered that erasure-only RS decoding can effectively utilize the soft reliability information with low hardware complexity.

A. Reliability Information Extraction from RM Decoding

In RM decoding, the FHT generates a vector of $2^{k_{RM}-1}$ integers. The one with the largest magnitude is considered to be the most correlated to the correct message bits. Its index in binary representation together with the sign bit are utilized to generate the k_{RM}-bit output. The magnitudes of the integers vary. They actually give us reliability information about the RM decoding outputs and hence RS decoding inputs.

Various settings have been exploited to utilize the magnitudes of the integers to generate reliability information. The maximum magnitude of the $2^{k_{RM}}$ integers, denoted by max_1, itself can be considered as reliability information. Alternatively, if max_1 is substantially larger than max_2, which is the magnitude of the second largest integer, the symbol corresponding to max_1 is also more likely to be correct. Both of these approaches have been investigated in our work and it was found that using $max_1 - max_2$ leads to better estimation on the RS input reliability and hence lower DFR. However, fully pipelinable architecture with a parallel comparable tree for finding max_1 is needed to reduce the latency. Finding max_2 within the same latency causes large area overhead. Therefore, max_1 itself is chosen to generate soft inputs to RS decoder in our design to reduce the hardware complexity.

B. Erasure-Only RS decoding

Chase decoding and erasure decoding are two types of soft-decision RS decoding schemes with low hardware complexity. However, the Chase decoding would need two candidates symbols for each unreliable input symbol. Finding the second candidate would require the max_2 value, and is not further investigated in this work. In this design, the focus is given to erasure-only RS decoding.

Two methods for assigning erasures were investigated in this work. The first one selects the $n_{RS} - k_{RS}$ RM outputs with the smallest max_1 to be the erasures. The simulation results are shown in Fig. 1, and the curve is extrapolated using the method

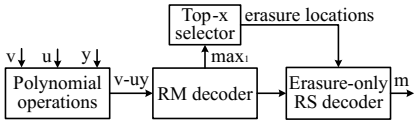

Fig. 2. Top-level block diagram of HQC decryption with the proposed erasure-only RS decoder.

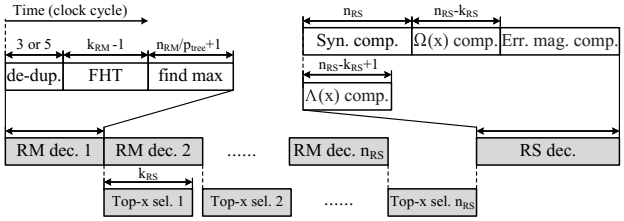

Fig. 3. Computation schedule of the proposed concatenated RM and RS decoder.

in [6] to lower DFR region. It can be observed that using $n_{RS} = 40$ can meet the requirement of DFR$< 2^{-128}$. For RS $[40, 16]$ codes, 40-16>16. Although the 16 RM outputs with the largest max_1 can be selected and assigned as non-erasures instead, the selector needs extra hardware to implement.

The second method tries to minimize the hardware complexity of erasure assignment by using a threshold. If the max_1 generated from the RM decoding is lower than the threshold, then the corresponding symbol is assigned as an erasure. Such a scheme only requires simple comparators. Simulations have been carried out to exhaustively search for the optimal threshold, and the DFR using the optimal threshold is also shown in Fig. 1. It can be observed that there is a significant gap between the DFRs of the first and second erasure assignment methods. An (n_{RS}, k_{RS}) RS code can correct up to $n_{RS} - k_{RS}$ erasures. When the number of erasures found from threshold comparison is more than $n_{RS} - k_{RS}$, extra erasures are discarded. When there are less than $n_{RS} - k_{RS}$ erasures, random symbols are used to fill the number. These two cases of imprecise erasure assignment cause the performance degradation. To mitigate the degradation, more thresholds can be utilized. However, our simulations show that the improvement is marginal.

Although the selector requires extra complexity, it contributes to a small portion of the overall decryptor area. Considering this, the first method of erasure assignment is adopted in our design, since it leads to substantial RS codeword length reduction. Our design is not using max_2 for hardware complexity purpose. To justify the claims made in the previous subsectin, the DFR in the case that the $n_{RS} - k_{RS}$ RM outputs with the smallest $max_1 - max_2$ are assigned erasures is also shown in Fig. 1. It can be observed that using $max_1 - max_2$ would further reduce the required RS codeword length to 38.

IV. Hardware Architecture and Comparisons

This section first develops an efficient HQC decryption architecture with the proposed soft-decision RS decoder. Then the hardware complexity is analyzed and compared with that of the best prior design.

Fig. 2 presents the top-level block diagram of the HQC decryption. The input ciphertexts $\mathbf{v}$ and $\mathbf{u}$ are first processed by the polynomial operation module to compute $\mathbf{v} - \mathbf{uy}$. Since all involved polynomials are binary, the multiplication can be implemented by using the shift-and-add architectures proposed in [14], [15]. For each of the w nonzero entries in $\mathbf{y}$, the dense $\mathbf{u}$ is shifted by the index of that entry, and the shifted terms are sequentially accumulated to produce the final product. Then the lowest $n_{RM} \cdot n_{RS}$ coefficients of $\mathbf{v} - \mathbf{uy}$ are partitioned into n_{RS} segments of n_{RM} bits. Each segment serves as an input to the RM decoder. Fig. 3 shows the computation schedule of the concatenated RM and RS decoders. In the RM decoder, depending on the security level, the n_{RM} bits are divided into three or five groups, which are added up bit-wise. This is referred to as the de-duplication process. Then the $n_{RM}/3$(or 5) $= 2^{k_{RM}-1}$-entry vector is sent to the FHT, which can be implemented by using the standard iterative butterfly architecture [22]. The output of the FHT is a vector consisting of $2^{k_{RM}-1}$ integers. A binary tree can be used to find the max_1 value and its index, from which the output of RM decoding is generated. The parallelism of the tree, p_{tree}, can be chosen to trade off latency and complexity.

The RM decoding for the n_{RS} segments is carried out one after another. The max_1 value from each RM decoding is sent to the top-x selector as it is generated. The top-x selector takes one input at a time and iteratively finds the k_{RS} largest max_1 values. Different from existing top-x sorters [23], our design does not need the k_{RS} entries to be sorted with respect to their relative magnitudes. Hence, the implementation architecture can be substantially simplified. Our proposed top-x selector keeps the k_{RS} largest max_1 values along with their indices in the corresponding RM decoding output vector among all the inputs it has received at any time. Inspired by the algorithm [24], our design also tracks the smallest entry among the k_{RS} max_1 values. When a new input arrives, it is compared against this minimum value. If the input is greater, then it replaces the minimum entry. Then comparisons are carried out among the updated set of k_{RS} entries to find the minimum value. Since RM decoding takes many clock cycles, a simple comparator in a feedback loop can be utilized to complete the comparison in $k_{RS} - 1$ clock cycles. After all RM decoding outputs have been processed, the registers in our top-x selector hold the k_{RS} largest max_1 values and their indices.

The symbols received by the RS decoder other than those with the k_{RS} largest max_1 values are set as erasures. Erasure-only RS decoding consists of three main components: syndrome computation, erasure locator and evaluator polynomials computation, and erasure magnitude computation. Assume that the vector of n_{RS} symbols received by the RS decoder is $[r_0, r_1, \ldots, r_{n_{RS}-1}]$. They are considered as the coefficients of a polynomial $r(X) = r_0 + r_1 X + \cdots + r_{n_{RS}-1} X^{n_{RS}-1}$. The symbols set to erasures are replaced by zero. Then the $n_{RS} - k_{RS}$ syndromes are computed as $S_i = \sum_{j=0}^{n-1} r_j \alpha^{ij}, (0 \le i < n_{RS} - k_{RS})$, where α is a primitive element of $GF(2^{k_{RM}})$. The architecture for computing S_i is shown in Fig. 4(a). A total of $n_{RS} - k_{RS}$ such units are utilized in parallel to compute all syndromes in n_{RS} clock cycles.

Instead of using the ePIBM architecture [21] to compute

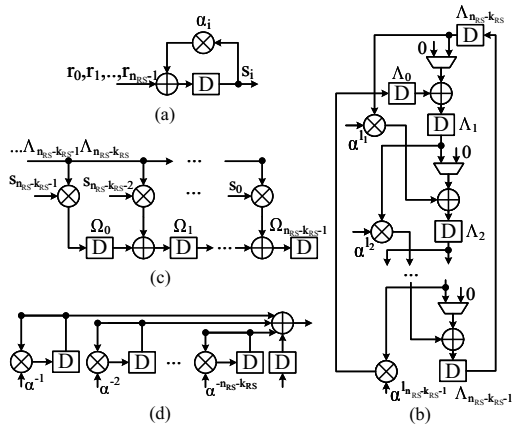

Fig. 4. Architectures of the erasure-only RS decoder. (a) Syndrome computation; (b) $\Lambda(X)$ computation; (c) $\Omega(X)$ computation; (d) Chien search.

error locator and evaluator polynomials for hard-decision error-correcting RS decoding as in prior works [14], [15], our proposed design adopts direct polynomial multiplication to compute the erasure locator and evaluator polynomials to reduce the complexity. Denote the erasure locations by $l_0, l_1, \cdots, l_{n_{RS}-k_{RS}-1}$. The erasure locator polynomial is defined as $\Lambda(X) = \prod_{i=0}^{n_{RS}-k_{RS}-1}(X - \alpha^{l_i})$. This polynomial can be computed using the parallel architecture presented in Fig. 4(b) simultaneously with the syndrome computation. The registers Λ_0 and Λ_1 are initialized to α^{l_0} and 1, respectively, for the first term $(X - \alpha^{l_0})$. In each clock cycle, one more $(X - \alpha^{l_i}), i > 0$ is multiplied, and the coefficients of $\Lambda(X)$ are located at the registers after $n_{RS} - k_{RS} + 1$ clock cycles. The syndromes can be considered as the coefficients of a syndrome polynomial $S(X)$. The erasure evaluator polynomial $\Omega(X)$ is defined as $\Omega(X) = S(X)\Lambda(X) \bmod X^{n_{RS}-k_{RS}}$. This operation is realized by a polynomial multiplication architecture similar to that of an FIR filter as shown in Fig. 4(c), where the coefficients of $\Lambda(X)$ are supplied serially and the coefficients of $S(X)$ act as filter coefficients. The coefficients of $\Omega(X)$ are located in the registers after the last coefficient of $\Lambda(X)$ is processed.

The magnitudes of the erased symbols are computed using Forney's formula $e_{i_l} = \Omega(\alpha^{-i_l})/\Lambda'(\alpha^{-i_l})$, which can be reformulated as $e_{i_l} = \alpha^{-i_l}\Omega(\alpha^{-i_l})/\Lambda_{odd}(\alpha^{-i_l})$ [25], where $\Lambda_{odd}(X)$ is a polynomial consisting of the odd coefficients of $\Lambda(X)$. $\Omega(\alpha^{-i_l})$ and $\Lambda_{odd}(\alpha^{-i_l})$ can be computed using the Chien search architecture. Since n_{RS} is small and the latency of the RS decoding contributes to a small portion of the overall HQC decryption latency, a serial Chien search architecture as depicted in Fig. 4(d) is employed for each of $\Omega(X)$ and $\Lambda_{odd}(X)$ in our design to reduce the area requirement.

Table II lists the hardware requirement and latency of each component in the proposed erasure-only RS decoder with the reduced codeword length. For the operations in $GF(2^8)$, the areas for an adder, a general multiplier, and an inverter are approximately the area requirement of 8, 100, and 148 XOR gates, respectively. The areas of 1 bit of memory, multiplexer, and register can be estimated as the areas of 1, 1, and 3 XOR gates, respectively [25]. The area of a constant multiplier depends on the constant operand, and can be estimated as the area of 15 XOR gates on average in our design [26].

TABLE II
HARDWARE REQUIREMENT AND LATENCY OF THE PROPOSED ERASURE-ONLY (40, 16) RS DECODER OVER $GF(2^8)$

	General mult.	Add.	Inv.	Constant mult.	Mux (8-bit)	Reg. (8-bit)	Latency (# of clks)
Syn. comp.	0	24	0	24	0	24	40
$\Lambda(X)$ comp.	23	23	0	0	23	25	25
$\Omega(X)$ comp.	24	23	0	0	0	24	24
Mag. comp.	2	0	1	37	0	39	40
Total	49	70	1	61	23	112	104

TABLE III
AREA AND LATENCY COMPARISON FOR THE HQC-128 DECRYPTION

	Proposed		Prior works [14], [15]	
Codeword len. (bits) (normalized)	15361 (0.87)		17669 (1)	
	Area (# of XORs)	Latency (# of clks)	Area (# of XORs)	Latency (# of clks)
Poly. operations	34332	8125	39006	9313
RM decoder	20362	760	19702	874
Top-x selector	748	+ 16	/	/
RS decoder	9683	104	11816	122
Total (norm.)	65125 (0.92)	9005 (0.87)	70524 (1)	10309 (1)

The area and latency of the overall HQC-128 decryption utilizing the proposed erasure-only RS decoder and top-x selector are analyzed and listed in Table III. Due to the improved error-correcting capability of the proposed soft-decision decoder, n_{RS} is reduced from 46 to 40. Hence, the codeword length of the proposed design is reduced to 15361, which is the smallest primitive prime larger than $n_{RS}n_{RM} = 15360$. The area and latency of each module in the prior HQC decryptors with the original RS $[46, 16]$ code using the hard-decision ePIBM decoder [14], [15] are also listed in the table. Since the critical path delays are similar, the latency is compared by the number of clock cycles. The polynomial operations process 128 bits at a time. The FHT takes $k_{RM} - 1 = 7$ clock cycles. Using $p_{tree} = 16$ and taking into account the de-duplication, each RM decoding is completed in $3 + 7 + (128/16 + 1) = 19$ clock cycles. The proposed design has smaller area and shorter latency than the prior works in polynomial operations, RM decoder, and RS decoder because of the smaller n_{RS}. The top-x selector needs k_{RS} clock cycles to incorporate one entry to the set of entries it keeps. All these updates overlap with RM decoding except the last one. Hence, it adds $k_{RS} = 16$ clock cycles to the latency. Overall, our proposed design reduces the codeword length, area requirement and latency by 13%, 8%, and 13%, respectively.

V. CONCLUSIONS

This work presents an HQC decryption module with reduced key length through utilizing soft-decision RS decoder. A top-x selector is adopted to generate the soft information from RM decoding results and a low-complexity erasure decoder is proposed to efficiently utilize the soft information. Efficient hardware architectures for soft information extraction and RS decoding are developed. The proposed design achieves significant key length reduction with lower hardware complexity.

REFERENCES

[1] NIST. *Post-Quantum Cryptography*. NIST. Accessed: Sep. 2024. [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography

[2] R. Misoczki, J.-P. Tillich, N. Sendrier, and P. S. L. M. Barreto, "MDPC-McEliece: new McEliece variants from moderate density parity-check codes," *Proc. IEEE Intl. Symp. on Info. Theory*, pp. 2069-2073, Oct. 2013.

[3] J. Cai and X. Zhang, "Low-complexity parallel Min-sum medium-density parity-check decoder for McEliece cryptosystem," *IEEE Trans. on Circuits and Syst.-I*, vol. 70, no.12, pp. 5328-5338, Oct. 2023.

[4] J. Cai and X. Zhang, "Highly efficient parallel row-layered Min-Sum MDPC decoder for McEliece cryptosystem," *arXiv preprint arXiv:2407.12695*, Jul. 2024.

[5] J. Cai and X. Zhang, "Low-latency parallel row-layered Min-sum MDPC decoder for McEliece cryptosystem," *Proc. IEEE Workshop on Signal Processing Syst.*, pp. 147-152, Nov. 2024.

[6] N. Aragon et al. *BIKE: Bit Flipping Key Encapsulation, Round 4 Submission*. NIST. Accessed: Sep. 2024. [Online]. Available: https://bikesuite.org

[7] J. Cai and X. Zhang, "Efficient layered new bit-flipping QC-MDPC decoder for BIKE post-quantum cryptography," *Proc. IEEE Intl. Symp. on Circuits and Syst.*, pp. 1-5, May 2025.

[8] J. Cai and X. Zhang, "Low-latency parallel row-layered Min-sum decoders with scheduling optimization for MDPC code-based post-quantum cryptography ," *Springer Journ. of Signal Processing Syst.*, vol. 97, no.1, pp. 257–268, Sep. 2025.

[9] P. Gaborit et al. *Hamming Quasi-Cyclic (HQC)*. NIST. Accessed: Sep. 2025. [Online]. Available: https://pqc-hqc.org/

[10] G. Alagic et al. "Status report on the fourth round of the NIST post-quantum cryptography standardization process," NIST, NIST IR 8545, Mar. 2025.

[11] Y. Tu, P. He, C. K. Koc, and J. Xie, "LEAP: lightweight and efficient accelerator for sparse polynomial multiplication of HQC," *IEEE Trans. on Very Large Scale Integr. Syst.*, vol.31, no.6, pp. 892–896, Mar. 2023.

[12] P. He, Y. Tu, and J. Xie, "LOCS: low-latency and constant-timing implementation of fixed-weight sampler for HQC," *Proc. IEEE Intl. Symp. on Circuits and Syst.*, pp. 1-5, May 2023.

[13] S. Deshpande, M. Nawan, K. Nawaz, J. Szefer, and C. Xu, "Fast and efficient hardware implementation of HQC," *IACR Cryptol. ePrint Arch.*, vol. 2022, no. 1, pp. 1183-1212, Sep. 2023.

[14] C. Li, S. Song, J. Tian, Z. Wang, and C. K. Koc, "An efficient hardware design for fast implementation of HQC," *Proc. IEEE Intl. Syst.-on-Chip Conf.*, pp. 1-6, Sep. 2023.

[15] F. Antognazza, A. Barenghi, and G. Pelosi, "A high efficiency hardware design for the post-quantum KEM HQC," *Proc. IEEE Intl. Symp. on Hardware Oriented Sec. and Trust*, pp. 1-11, May 2024.

[16] C. Aguilar-Melchor et al. "Towards automating cryptographic hardware implementations: a case study of HQC," *Proc. Intl. Workshop on Code-Based Crypto.*, pp. 62-76, May 2022.

[17] M. Schoffel, J. Feldmann, and N. Wehn, "Code-based cryptography in IoT: a HW/SW co-design of HQC," *Proc. IEEE World Forum on Inet. of Things*, pp. 1-7, Jun. 2023.

[18] R. Aissaoui , J.-C. Deneuville, C. Guerber, and A. Pirovano, "A performant quantum-resistant KEM for constrained hardware: optimized HQC," *Proc. Intl. Conf. on Sec. and Crypto.*, pp. 668-673. Sep. 2024.

[19] F. Antognazza, A. Barenghi, and G. Pelosi, "An efficient and unified RTL accelerator design for HQC-128, HQC-192, and HQC-256," *IEEE Trans. on Computers*, vol. 74, no. 7, pp. 2306-2320, Jul. 2025.

[20] Y. Be'ery and J. Snyders, "Optimal soft decision block decoders based on fast Hadamard transform," *IEEE Trans. on Inf. Theory*, vol. 32, no. 3, pp. 355–364, May 1986.

[21] Y. Wu, "New scalable decoder architectures for Reed Solomon codes," *IEEE Trans. on Comms.*, vol. 63, no. 8, pp. 2741–2761, Jun. 2015.

[22] M. Hashemipour-Nazari, K. Goossens, and A. BalatsoukasStimming, "Hardware implementation of iterative projection-aggregation decoding of Reed-Muller codes," *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Processing*, pp. 8293–8297, Jun. 2021.

[23] W. Chen, W. Li, and F. Yu, "A hybrid pipelined architecture for high performance tok-x sorting on FPGA," *IEEE Trans. on Circuits and Syst.-II*, vol. 67, no. 8, pp. 1449-1453, Aug. 2020.

[24] A. Shanbhag, H. Pirk, and S. Madden, "Efficient top-k query processing on massively parallel hardware," *Proc. Intl. Conf. on Mgmt. of Data*, pp. 1557–1570, May 2018.

[25] J. Zhu and X. Zhang, "High-speed re-encoder design for algebraic soft-decision Reed–Solomon decoding," *Proc. IEEE Intl. Symp. on Circuits and Syst.*, pp. 465–468, May 2010.

[26] X. Zhang and Y. Zheng, "Systematically re-encoded algebraic soft-decision Reed–Solomon decoder," *IEEE Trans. on Circuits and Syst.-II*, vol. 59, no. 6, pp. 376-380, May 2012.