2024 21st International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE) | 979-8-3503-7754-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/CCE62852.2024.10770880

Hardware Architecture for the SHA-3 Family in CRYSTALS-KYBER: Post-Quantum Cryptography

Adriana Pérez-Navarro
Departamento de Computación
Cinvestav
Mexico City, Mexico
adriana.perez@cinvestav.mx

Brisbane Ovilla-Martinez
Departamento de Computación
Cinvestav
Mexico City, Mexico
0000-0002-2861-863

***Abstract*—With the immediate advances in quantum computing, standard public key cryptosystems are expected to become obsolete. For this reason, Post-Quantum Cryptography (PQC) emerges as a new area of research for developing cryptographic systems that are resistant to conventional and quantum attacks. It could thus replace traditional public-key cryptographic solutions. This work describes the design hardware implementation of one of the most critical components in the NIST-PQC standard, CRYSTALS-KYBER. The symmetric primitives used in CRYSTALS-KYBER: SHA3-256, SHA3-512, SHAKE-128, and SHAKE-256, all based on the Keccak permutation, are integrated as a parameterizable hardware module. The proposed design focuses on embedded systems with a constraint on resources and power consumption and is implemented on a Xilinx Field Programmable Gate Arrays Artix-7. The efficiency and performance of the proposed architecture are compared in terms of area, frequency, and clock cycles with the state of the art.**

***Index Terms*—Post-Quantum Cryptography, CRYSTALS-KYBER, Hardware Design, FPGA, SHA-3.**

I. INTRODUCTION

Currently, traditional public key cryptosystems, such as Rivest–Shamir–Adleman (RSA) and Elliptic Curve Cryptography (ECC), are expected to become obsolete. For this reason, Post-Quantum Cryptography represents the standards that establish secure mechanisms for key exchange and digital signature schemes in our digital security infrastructure. In these algorithms, security depends on the large integer factorization problem and the elliptic discrete logarithm problem in a finite field, respectively.

Nowadays, the standard algorithms for Public Key Cryptography (PCK) are resistant to classical attacks such as brute force, traditional mathematical attacks, or side-channel attacks since they cannot solve these problems in a reasonable amount of time due to the sheer number of possible solutions that need to be tested. However, due to advances in the field of quantum computers, PKC is likely to be vulnerable to polynomial-time realizations of Shor's [2] algorithm on a quantum computer, in which the basic idea is the period search process using the quantum Fourier transform, which takes a function $f(x)$ and calculates its period [1]. This implies that it is necessary to define and design alternative cryptosystems to classical public key cryptography that maintain security against traditional computer attacks and, at the same time, guarantee security against quantum computing attacks.

The National Institute of Standards and Technology (NIST), in December 2016, initiated a process to solicit, evaluate, and standardize new public key algorithms resistant to quantum attacks [3] . In particular, the Key-Encapsulation Mechanism (KEM) and digital signature schemes are part of its efforts to promote security in public key cryptography and to establish cryptographic standards that can withstand attacks from both technologies, conventional computers and quantum computers.

A KEM allows the secure transmission of a shared secret via a public key algorithm, which can then be expanded to generate symmetric keys, which is more efficient for transmitting long messages to a PKC scheme [4].

In the third round of the Post-Quantum Cryptography (PQC) standardization process, the NIST has selected the CRYSTALS-KYBER [5] algorithm as the best combination of performance and key sizes, allowing two parties to exchange the key easily, as well as its speed of operation. It is based on the Module Learning With Errors (M-LWE) [6] hardness.

The SHA-3 family plays a crucial role in the CRYSTALS-KYBER cryptographic scheme, as it is a key component in several stages of the algorithm. In particular, the CRYSTALS-KYBER hash module accounts for a large portion of the total resource consumption; it includes the SHA3-256 and SHA3-512 hash functions, as well as the SHA-3 family's extensible output functions SHAKE128 and SHAKE256 based on the Keccak-f[1600] permutation but with different pricing and different message suffixes added for domain separation.

The Field Programmable Gate Arrays (FPGA) technology provides a favorable hardware platform for prototyping the algorithms and looking for accelerating PQC algorithms. The design of hardware architectures for computationally intensive CRYSTALS-KYBER modules, such as the extensive use of hashing Keccak, the Number Theoretic Transform (NTT), and Inverse NTT (INTT), has generated significant interest in FPGA, particularly for embedded systems with a constraint on resources and power consumption.

The motivation of this work is to contribute to the security of communications against quantum attacks on embedded devices. A key part of the CRYSTALS-KYBER is the use of the SHA-3 family of hash functions, which plays a crucial

979-8-3503-7754-5/24/$31.00

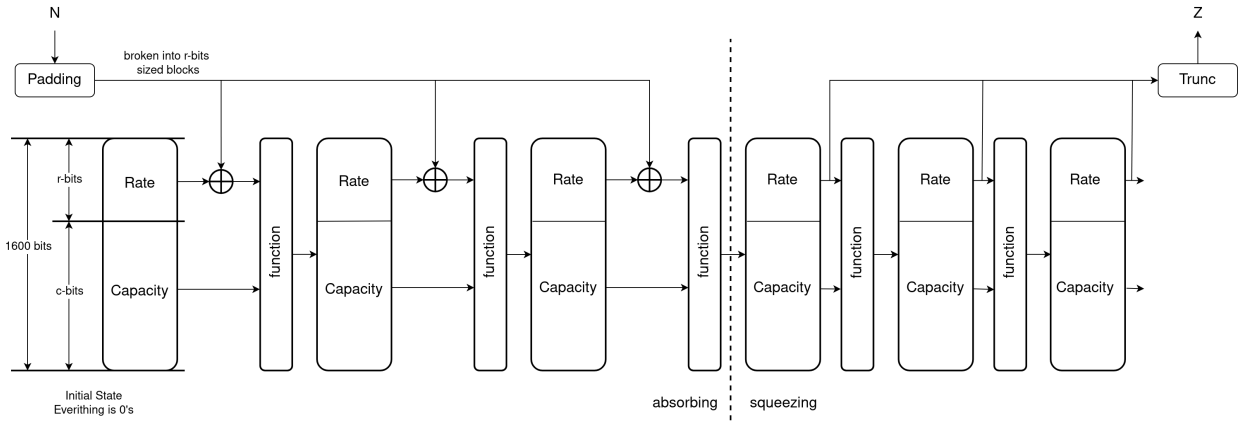

Fig. 1: Sponge Function Keccak-f[1600].

role in its operation. Implementing SHA-3 in hardware can significantly improve the speed and energy efficiency of the cryptographic system.

The main contribution of this work is the design and implementation of a hardware module dedicated to CRYSTALS-KYBER-768 and SHA-3 primitives. For this purpose, we implemented a VHDL architecture that integrates the SHA-3 primitives used in CRYSTALS-KYBER, SHA3-256, SHA3-512, SHAKE-128, and SHAKE-256 as a single parameterizable module.

The rest of this paper is organized as follows: in Section II, the CRYSTALS-KYBER algorithms will be presented as well the SHA-3 primitives (Section II-A). Section III reports the hardware implementation design process. Section IV will present the results obtained and a comparison with other implementations; finally, section V will show the conclusions obtained during the implementation of the module and future work.

II. PRELIMINARIES

CRYSTALS-KYBER is a Lattice-Based cryptography (LBC) algorithm used as KEM. Compared to other post-quantum KEM schemes, CRYSTALS-KYBER offers significant advantages in terms of efficiency, while guaranteeing resistance against known quantum and classical attacks [7]. Designed based on the assumptions of Module-LWE, CRYSTALS-KYBER's security depends on the challenge of solving the learning with errors problem within module networks. Being a KEM, it is composed of three main algorithms: key generation, encapsulation, and decapsulation.

TABLE I: KYBER PARAMETERS.

	n	k	q	$\eta 1$	$\eta 2$	(d_u, d_v)	δ
Kyber512	256	2	3329	3	2	(10,4)	2^{-139}
Kyber768	256	3	3329	2	2	(10,4)	2^{-164}
Kyber1024	256	4	3329	2	2	(10,5)	2^{-174}

Specifications and parameters: CRYSTALS-KYBER offers three different security levels, each with its configuration parameters. These variants (Table: I), KYBER512, KYBER768 and KYBER1024, provide security levels comparable to AES-128, AES-192 and AES-256, respectively [8]. The derived parameter δ is the probability of the decapsulation of a valid KYBER.CCAKEM ciphertext will fail.

CRYSTALS-KYBER algorithm operates within a polynomial ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$, where the parameters are fixed: q is the modulus (with $q = 3329$) and n is the degree of the polynomial (with $n = 256$) where k = 2, 3 or 4, representing the three different post-quantum security levels. The prime modulus is fixed for the three CRYSTALS-KYBER security levels.

CRYSTALS-KYBER employs an extendable output function (**XOF**), two hash functions (**H** and (**G**)), a pseudo-random function (**PRF**), and a key-derivation function (**KDF**). All these primitives are instantiated with functions from the Federal Information Processing Standard (FIPS-202 standard), as outlined in Table. II. In an internal key generation, a hash function is applied to the wrapping key, which is stored in the dewrapping key. During encapsulation, a hash function combines the wrapping key with a random value and derives the shared key. In dewrapping, a hash function is applied again to recompute the shared key and verify the ciphertext. If verification fails, the implicit rejection mechanism employs another hash function to modify the shared key using the ciphertext and a random value stored in the secret key.

The second critical component for speed within CRYSTALS-KYBER is the symmetric primitives [3]. SHA3 has the reputation of not being the fastest hash function in software. For this reason, a dedicated SHA3 CRYSTALS-KYBER primitive SHA3 primitive hardware has been implemented, which incorporates a control unit that monitors all modes of operation.

A. Study of SHA-3

Secure Hash Algorithm-3 (SHA-3) is a family of functions based on an instance of the Keccak algorithm. The SHA-3 family comprises four cryptographic hash functions and two extensible output functions. These six functions share the

TABLE II: KYBER PRIMITIVES STANDARD'S PARAMETERS.

SHA3 primitives	Kyber primitives	Padding	Size in bits		
			r	c	output length
SHA3-256	**H**	$M \parallel 0x06 \parallel 0x00 \ldots \parallel 0x80$	1088	512	256
SHA3-512	**G**		576	1024	512
SHAKE128	**XOF**	$M \parallel 0x06 \parallel 0x00 \ldots \parallel 0x80$	1344	256	unlimited
SHAKE256	**PRF, KDF**		1088	512	unlimited

structure of the sponge construct; functions with this structure are called sponge functions.

The four SHA-3 functions are considered modes of operation of the KECCAK-p[1600,24] permutation. The state of the KECCAK-p[b, nr] permutation consists of b bits. The specifications of this standard contain two other quantities related to b: b/25 and $\log_2(b/25)$, denoted by ω and ℓ, respectively.

The Keccak-f [1600] permutation is constructed using three main components: an underlying function for fixed-length strings, denoted as f; a parameter called the rate, represented by r; and a padding rule referred to as pad (Fig. 1). These components work together to perform the necessary permutation in the Keccak hash algorithm, where f transforms the strings, r determines the bits processed per round, and pad ensures the correct format of the messages.

Figure 1 shows the function's two phases.

1) Absorption phase: The message is decomposed into blocks of r bits generating $M = M_1||...||M_s$. If the last block is incomplete, M_s is filled with the corresponding rule. The bit rate of the initialized state is XORed with the first part of the input. The new bit rate, together with the capacity of the initialized state matrix, will form a new state used in the f permutation.
2) Squeeze phase: The output of the hash is generated from the internal state of the sponge. If the desired hash length is less than or equal to the rate length r, then this output value is taken directly from the rate of the internal state. When the desired hash length exceeds r, the Keccak permutation function is applied to the internal state until enough output is produced to reach the desired hash length.

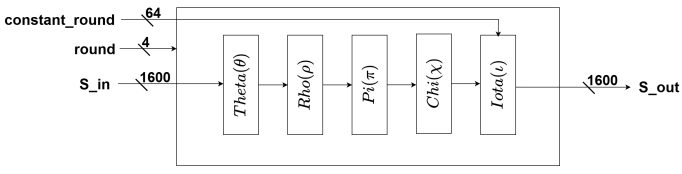

Fig. 2: A round of KECCAK-p[b, nr].

For SHA-3 functions, a two-bit or four-bit suffix is added to the M message to produce the input string N to KECCAK[c], and additional bits are added as part of the multi-rate padding rule (See Table. II).

A round of KECCAK-p[b, nr] is composed of five separate steps: theta ($theta$), rho (rho), pi (pi), chi (chi) and iota ($iota$). The process of each iteration of the algorithm (Fig. 2) takes a 1600-bit state consisting of a 5x5 matrix of 64-bit words representing the current state of the system as input and produces a new updated state matrix as output. Except the mapping function $iota$ has a second input variable, an integer called the round index.

III. DESIGN HARDWARE IMPLEMENTATION

This section presents the architecture and key features of the implemented SHA3 module. It is a hardware accelerator designed for implementing hash and XOF functions compliant with the SHA3 algorithm, specifically tailored for the CRYSTALS-KYBER application.

The development of a server-level SHA-3 module for CRYSTALS- KYBER poses the challenge of optimizing the establishment of cryptographic keys. Since CRYSTALS- KYBER is a post-quantum scheme, its implementation on servers with many simultaneous requests can generate a high load. The challenge is to ensure an efficient key establishment process, minimizing latency and resource usage without compromising security against quantum threats.

This design is thought within a SoC (System on Chip) architecture that combines a PS (Processing System) and a PL (Programmable Logic). This is to integrate the complete CRYSTALS-KYBER cryptosystem in the future, thus taking advantage of both worlds. Specifically, registers handle the inputs and outputs, allowing us to establish efficient communication between the PS and the PL. The registers act as data buffers that facilitate the transfer of information without the need for complex or costly communication mechanisms in terms of resources.

A. Keccak-p core

In order to design the Keccak-p core used in CRYSTALS-KYBER, the module must supports the generation of the four primitives of the SHA-3 family, SHA3-256, SHA3-512, SHAKE128 and SHAKE256, where each one has a different configuration as is the described in Table II. These primitives share the same permutation function KECCAK-p[b, nr] described in subsection II-A. A control unit named CtrlPermutation controls the round function and the number of rounds.

Keccak-permutation is the core for the primitives to be implemented. The surrounding modules control the inputs,

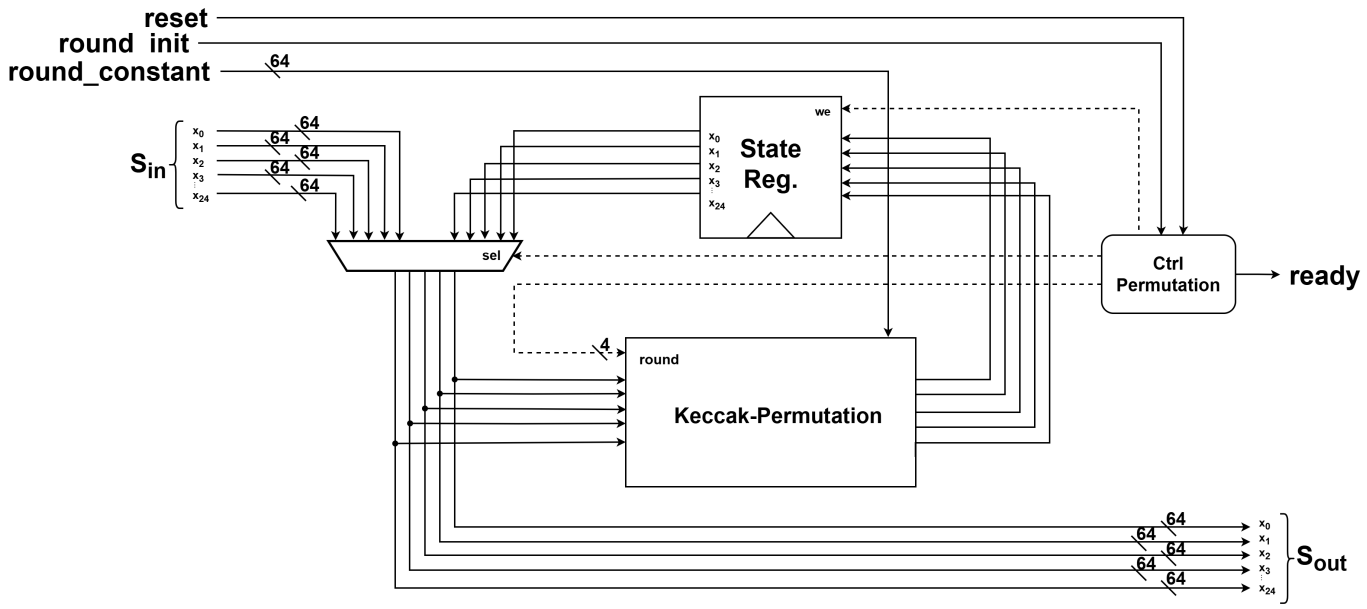

Fig. 3: Keccak-p core.

outputs, starts, and rounds performed by Keccak-permutation. In the first round, the initial state is all zeros, while the mux controls that for the following iterations, the input state is the previous output state reg of the Keccak permutation.

Once the Keccak-p core is built, the mechanisms controlling the inputs, outputs, and state storage must be defined.

Fig. 3 illustrates the main blocks of the architecture for the Keccak-p core from the general diagram. It has a 1600-bit register (State Reg), a permutation control unit (CtrlPermutation) with three states (reset, wait, round), and a multiplexer that controls the input to the permutation.

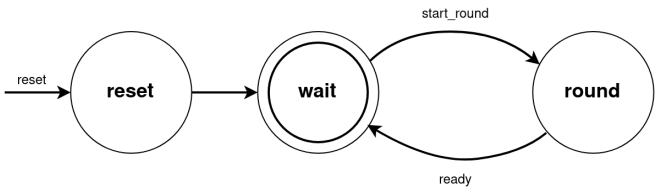

Fig. 4: Permutation control for Keccak-p core.

These modules control the rounds to be given, the input and output data, and the reset together with the control signals.

- Keccak-Permutation: as shown in Fig 2, takes a matrix representing the current state of the system as input and produces a new updated state matrix, a round number of four bits, and a constant round of 64 bits. Depending on the number of rounds, these are initialized using a different constant.
- Ctrl Permutation (Fig.4): the main function of this machine is to control the flow of data to the permutation of the input state or the state stored in the registers, as well as the number of rounds. When the rounds have been completed, the `valid` signal is sent to indicate that the output contains valid data, and `ready` high to indicate that a new series of rounds can be started. It also switches to the `wait` state to wait for a new `start_round` signal.

B. SHA3 Module

It can operate with single-block and multi-block messages as required by an instance of the CRYSTALS-KYBER-768 SHA3-256 primitive. A multi-block message is an entry whose length exceeds the maximum bit rate specific primitive can process.

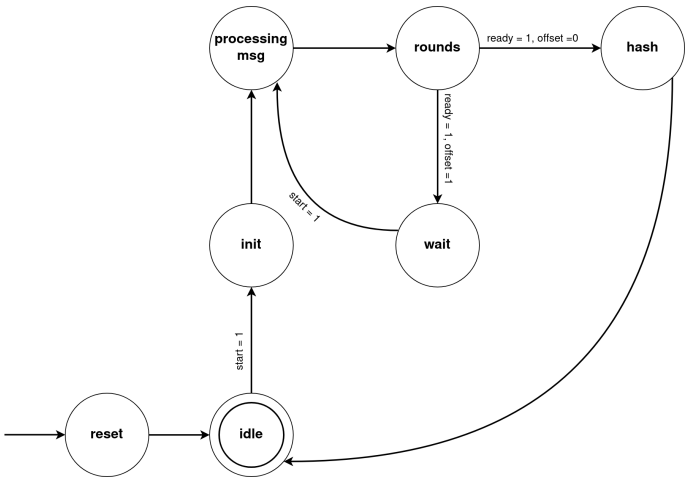

Fig. 5: Ctrl permutation SHA3 Module.

Figure 6 shows the design of our SHA-3 module covering the four primitives used by CRYSTALS-KYBER.

- Pad: The unit necessary to properly prepare the input message implements the padding rules according to the

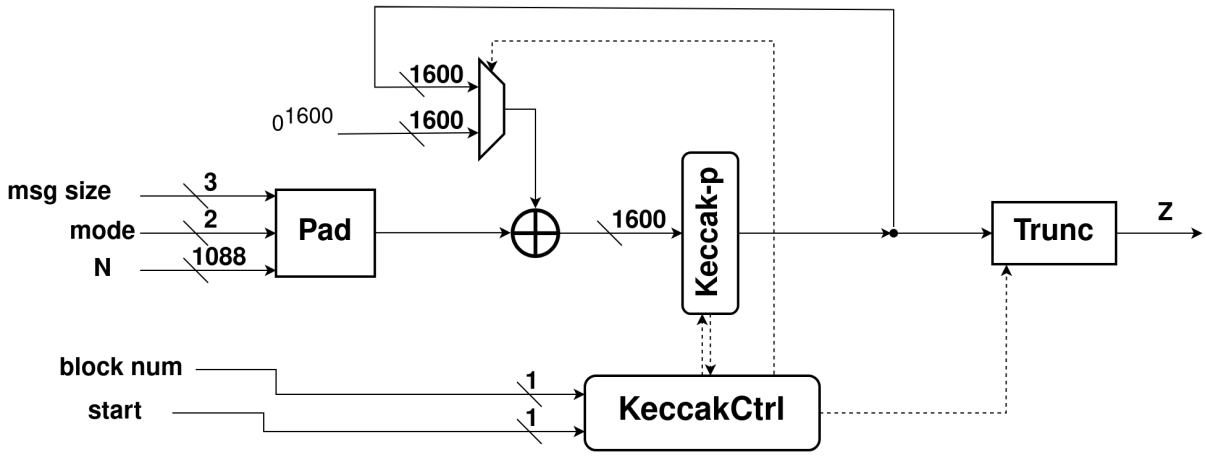

Fig. 6: SHA-3 Module.

rules defined in Section II-A, based on the msg size and mode parameters.

- Trunc: Controls the output bits corresponding to each specific primitive. The maximum output value is 1024 for the SHAKE256-PRF primitive.
- Xor: According to the primitive being processed it is responsible for building the appropriate 1600-bit state. Regardless of the required primitive, the 1344 evaluation from the Pad will be processed, since it is the maximum r size for the SHAKE128 primitive. The multiplexer controls the data that will xor when there is more than one message block.
- Ctrl Permutation: This controls that one message block is read at a time, and if the offset = '1', then read another message block (Fig. 5). This state machine directly controls the machine that controls the permutation.

The SHA3 primitives used in CRYSTALS-KYBER share all the steps of the Keccak Permutation. They differ only in how they treat the input data and in the output of the data. Depending on the cryptographic primitive, either SHA3-256, SHA3-512, SHAKE128, or SHAKE256 (mode) and the message size (msg size), the padding and absorption of the corresponding data will be performed. While the signal (block num) indicates whether more than one message block is required. For this, a study was conducted on the input and output data of the cryptographic primitives in CRYSTALS-KYBER-768.

IV. RESULTS

In this section, we present our VHDL language implementation results for a Xilinx Artix-7 FPGA and compare them to those found in the literature.

The Zynq-7000 FPGA and the Artix-7 FPGA share similarities in terms of architecture and general resources, as they both belong to the Xilinx 7-series device family. This means that both contain programmable logic elements such as LUTs (Look-Up Tables), Flip-Flops, Digital Signal Processing (DSP) slices, and Block RAM (BRAM). The main difference between the Artix-7 and the Zynq-7000 lies in integrating an ARM processor in the Zynq-7000, which allows both software processing (in the ARM processor) and hardware acceleration (in the FPGA).

Table III comprehensively compares the proposed SHA3 design with related works. Resource utilization (e.g., LUTs, FFs, Slices, and Cycles) within the context of CRYSTALS-KYBER can serve to assess computational performance. All the compared works use the Artix-7 FPGA, while our work uses the Zynq-7000. The choice of the Zynq-7000 offers us advantages in terms of hardware and software integration because of its SoC (System on Chip) architecture.

Regarding area utilization, there was an improvement of 11% compared to the other implementations' mean value, using more slices than the designs of [13] and [11] but less than that of [10].

TABLE III: COMPARISON OF SHA3 MODULE WITH RELATED WORKS ON FPGA PLATFORM

Design	Method	Device	Area			Max freq.	Cycles
			#LUTs	#FFs	#Slices	[MHz]	[CCs]
Guo et al. [13]	HW	Artix-7	4614	1771	1407	159	-
Bisheh-N et al. [12]	HW	Artix-7	4405	1629	1825	115	24
Banerjee et al. [11]	HW/SW	Artix-7	5784	1605	1716	25	24
Dolmeta et al. [10]	HW	Artix-7	9651	8697	3413	250	39
This work	**HW**	**Artix-7 / Zynq-7000**	**5079**	**2638**	**2212**	**145**	**29**

In terms of maximum frequency, our work reaches 145 MHz, which is competitive. It is higher than that of [12]. and [11] but lower than that of [13] and [10]. The high frequency of [10] is associated with their higher resource consumption.

When it comes to clock cycles (CCs), our implementation requires 29 cycles, which is a moderate number compared to other works. It is more than the 24 cycles required by [12] and [11], but less than the 39 cycles from [10].

V. Conclusions and Future Work

This paper presents a balanced hardware architecture for CRYSTALS-Kyber cryptographic primitives in terms of resources and performance. The design of a dedicated architecture that can perform all SHA-3 primitives used in the algorithm can significantly reduce the hardware resource occupation, obtaining a component with higher performance. Our architecture has been implemented for CRYSTALS-Kyber-768 cryptographic primitives. For this we make an implementation in FPGA with VHDL achieving a balance in the state of the art. Our design can be integrated into hardware implementations, or hardware/software co-designs.

We believe that this work contributes to the efficient and safe deployment of CRYSTALS-Kyber KEM in resource-constrained environments and embedded systems, consuming only 10.55% LUT, 3.62% FF and 0.5% DSP of a low-cost FPGA, AMD Artix-7. Thus, sufficient area is preserved for full CRYSTALS-Kyber implementation. Future work concerns the integration of the proposed SHA3 module into a complete CRYSTALS-Kyber hardware/Software architecture, thus enabling the establishment of low-latency secure post-quantum keys in embedded devices.

References

[1] CH Ugwuishiwu, UE Orji, CI Ugwu, and CN Asogwa. *An overview of quantum cryptography and Shor's algorithm*. Int. J. Adv. Trends Comput. Sci. Eng, vol. 9, no. 5, 2020.

[2] Thomas Monz, Daniel Nigg, Esteban A. Martinez, Matthias F. Brandl, Philipp Schindler, Richard Rines, Shannon X. Wang, Isaac L. Chuang, and Rainer Blatt. *Realization of a scalable Shor algorithm*. Science, vol. 351, no. 6277, pp. 1068–1070, 2016. American Association for the Advancement of Science.

[3] National Institute of Standards and Technology (NIST). *Post-quantum cryptography*, 2021. https://csrc.nist.gov/projects/post-quantum-cryptography.

[4] Victor Shoup. *A proposal for an ISO standard for public key encryption*. Cryptology ePrint Archive, 2001.

[5] Chad Boutin. *NIST announces first four quantum-resistant cryptographic algorithms*. National Institute of Standards and Technology, 2022.

[6] He Li, Yongming Tang, Zhiqiang Que, and Jiliang Zhang. *FPGA Accelerated Post-Quantum Cryptography*. *IEEE Transactions on Nanotechnology*, 21:685–691, 2022. DOI: 10.1109/TNANO.2022.3217802.

[7] Gorjan Alagic, Daniel Apon, David Cooper, Quynh Dang, Thinh Dang, John Kelsey, Jacob Lichtinger, Yi-Kai Liu, Carl Miller, and others. *Status report on the third round of the NIST post-quantum cryptography standardization process*. US Department of Commerce, National Institute of Standards and Technology, 2022.

[8] SAOUDI, Mohamed, et al. Low latency FPGA implementation of NTT for Kyber. Microprocessors and Microsystems, 2024, vol. 107, p. 105059

[9] National Institute of Standards and Technology. SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions (FIPS PUB 202). National Institute of Standards and Technology, 2015. http://dx.doi.org/10.6028/NIST.FIPS.202.

[10] Alessandra Dolmeta, Maurizio Martina, and Guido Masera. *Hardware architecture for CRYSTALS-Kyber post-quantum cryptographic SHA-3 primitives*. In *2023 18th Conference on Ph.D Research in Microelectronics and Electronics (PRIME)*, pages 209-212, 2023. IEEE. https://doi.org/10.1109/PRIME58259.2023.10161780.

[11] Utsav Banerjee, Tenzin S. Ukyab, and Anantha P. Chandrakasan. *Sapphire: A configurable crypto-processor for post-quantum lattice-based protocols*. arXiv preprint arXiv:1910.07557, 2019.

[12] Mojtaba Bisheh-Niasar, Reza Azarderakhsh, and Mehran Mozaffari-Kermani. *Instruction-set accelerated implementation of CRYSTALS-Kyber*. IEEE Transactions on Circuits and Systems I: Regular Papers, 68(11):4648–4659, 2021. IEEE.

[13] Wenbo Guo, Shuguo Li, and Liang Kong. *An Efficient Implementation of Kyber*. IEEE Transactions on Circuits and Systems II: Express Briefs, 69(3):1562-1566, 2022. DOI: 10.1109/TCSII.2021.3103184.