First end-to-end PQC protected DPU-to-DPU communications

First end-to-end PQC protected DPU-to-DPU communications

A. Cano Aguilera,[1,✉] X. Arnal i Clemente,[1] D.C. Lawo,[1] I. Tafur Monroy,[1] and J.J. Vegas Olmos[2]

[1]*Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven, Netherlands*

[2]*Software Architecture, NVIDIA Corporation, Ofer Industrial Park Yokneam, Israel*

✉ E-mail: a.c.a.cano.aguilera@tue.nl

The appearance of quantum computing in the short foreseeable future and its capability to break conventional cryptographic algorithms forces to change the paradigm of secure real-time communications. Thus, government organizations, data centers, and enterprises among others are migrating their public key infrastructure towards using post-quantum cryptography (PQC) algorithms in order to mitigate the security threats posed by quantum computers. This letter presents the first quantum resilient secure end-to-end communication link based on PQC algorithms operating between two data-processing units DPU. Both data-processing units employ on-board ARM processors to perform the computationally expensive cryptographic building blocks—in that case CRYSTALS-Kyber as a key encapsulation mechanism and CRYSTALS-Dilithium for digital signature scheme in combination with advanced encryption standard with 256-bit key.

Introduction: Post-quantum cryptography (PQC) refers to cryptographic schemes that are designed to be secure against a cryptanalytic attacks by both quantum and classical computers. Since quantum computers and digital annealers [1] are becoming available, PQC needs to be implemented in real-time communication links.

There are currently two algorithms considered to implement PQC links by the National Institute of Standards and Technology (NIST) Post-Quantum Cryptography Standardization Process [2]: Dilithium [3], which is a digital signature scheme that is strongly secure under chosen message attacks based on the hardness of lattice problems over module lattices, and Kyber [4], which is a key encapsulation method (KEM) designed to be resistant to cryptanalytic attacks through the hardness of solving the learning-with-errors (LWE) [5] problem over module lattices. To the best of authors' knowledge, this letter presents for the first time a hybrid PQC link deployed on data processing units (DPU), where Dilithium is initially employed for digital signatures, and then Kyber is used to execute the exchange of keys. Finally, these PQC keys are used to encrypt and decrypt data through 256 bit advanced encryption standard (AES) [6] or Rijndael [7] ciphers.

Cryptography in general and PQC in particular are among the main time-consuming and complex tasks in worldwide communications. Executing cryptographic algorithms on traditional general-purpose processors can be time-consuming and inefficient. Therefore, ongoing research focuses on exploring avenues to enhance their optimization. Most prototype implementations use FPGAs as programmable fabric [8-10]. In addition, researchers have explored the integration of PQC on application-specific integrated circuits (ASIC) in various studies [11, 12]. However, in practical communication systems, the utilization of FPGAs or ASICs is infrequent due to concerns related to performance, cost, power consumption, and the additional burden imposed on the CPU when offloading tasks to these devices.

In this work, we use DPUs to create our communication link. . DPUs are designed to offload specific tasks from the server such as network processing, storage processing, or encryption. By offloading these tasks, DPUs can help reducing the load of server's CPU and thus improve the overall system performance and reduce power consumption. This is accomplished by means of using specialized hardware such as digital signal processors (DSPs), semiconductor intellectual property (IP) cores, or graphics processing units (GPUs).

This work demonstrates end-to-end PQC communications employing off-the-shelf high-capacity communication systems. A full software stack for Dilithium and Kyber is implemented on DPU units that then establish full PQC links. This letter is organized as follows: the main novelty of the letter is presented in the experimental setup, where we define the methodology of PQC calls in a real communication system as well as their integration in a system composed by DPUs, and then the obtained experimental results are shown; these results include performance improvement of CPU cycles as well as throughput when implementing the PQC algorithms through the utilization of hardware accelerations as well as throughput performance. Finally, a summary of the main novelty of the letter is given.

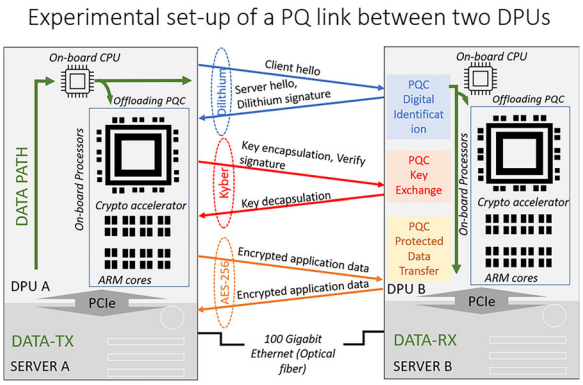

Fig. 1 *Experimental setup and handshake procedure to establish a quantum resilient link based on Dilithium and Kyber. In this setup, the transmitter (TX) initiates communication with the receiver (RX) by sending a ping. To establish authenticity, RX responds with a signature generated using Dilithium. Once TX confirms the identity of RX, they proceed with a key-exchange protocol to share an AES-256 key. As a result, the connection is established, enabling both RX and TX to transmit encrypted application data.*

Experiment: In this section, we present the experimental setup needed for achieving PQC-secure communications between DPUs. The setup is shown in Figure 1: it comprises two independent servers with their own central processing units (CPU) and two DPUs model MBF2H516A-CEEOT[1] that can process data at a 100 Gigabits per second (100G). The servers are interconnected with the DPUs through peripheral component interconnect express (PCIe) bridges. The DPUs are interconnected through optical pluggables using single-mode fiber. The DPUs include ARMv8 A72 cores for dedicated operations.

The software stack comprises the implementation of NIST selected algorithms Kyber and Dilithium. Kyber and Dilithium were selected based on Kyber's advancement to the 4th round of the NIST standardization process [2] as the only PQ KEM, and Dilithium's superior overall performance as it can be seen in Figure 2 compared to Falcon [13] and Sphincs+ [14], which are the other 4th round NIST-selected algorithms.

In an initial stage, Dilithium is employed to authenticate the identity of the user and validate its digital signature; then Kyber is employed for key exchange. Once this process is completed and the PQC link established, all data is encrypted with those PQC headers using AES-256.

The overall channel capacity is a direct relation between the processing capacity of the DPU and the capacity of the DPU interfaces. As DPU interfaces can operate at 100G regime, it becomes key to reduce the latency of the most complex PQC building blocks. In this demonstration, this is achieved by offloading heavy PQC functions from the main CPU of the DPU to dedicated on-board processors using ARM optimizations. We conducted our experiment using ARMv8 processors, incorporating the latest advancements in Dilithium [15] and Kyber [16] implementations. Our objective is to establish a benchmark with the results from this experiment, which will serve as a reference for future optimizations utilizing DPUs.

Results-CPU usage: The implementation of Kyber and Dilithium is processing intensive. Dilithium requires several distinct functions to be executed, some of them requiring 10^7 operation cycles. As the on-board

[1]https://docs.nvidia.com/networking/display/BlueField2DPUENUG/Specifications

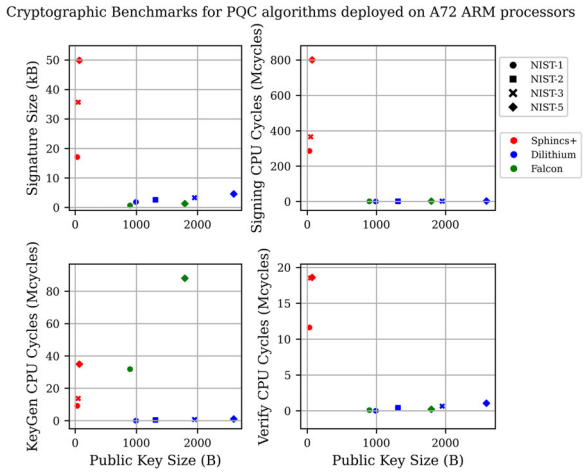

Fig. 2 *When deployed on DPUs, PQC signature algorithms were benchmarked according to their respective NIST security levels. Among them, Sphincs+ stands out as the only one based on Hash-signatures, but its public key size and signature size are excessively large, making it unsuitable for real-time communication implementations. On the other hand, Dilithium and Falcon, both based on lattices, offer distinct advantages. Dilithium outperforms in terms of key generation and signing algorithms, while Falcon excels in signature size and verification time.*

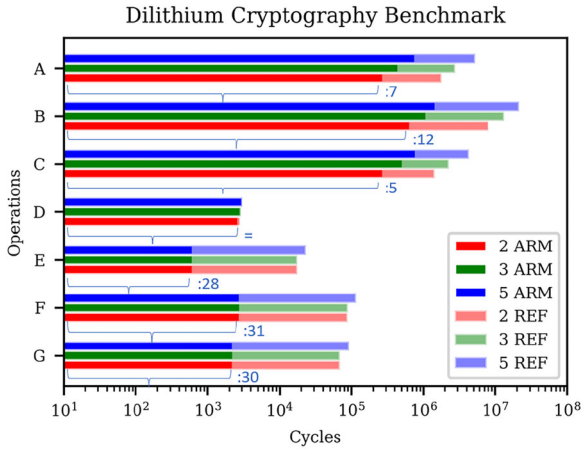

Fig. 3 *Performance of Dilithium main subroutines. According to the reference code [3]: A=verify; B=sign: c=keypair; D=poly_challenge; E=poly_pointwise_montgomery; F=poly_invntt_tomont; and G=poly_ntt. The figure provides a comparison between the reference code (REF) and an implementation with ARM optimizations. Results show that there is a reduction in CPU cycles by up to a factor of 31.*

ARM processors were clocked at 2.75 GHz, we can already predict subsecond performance but above the millisecond regime. Similarly, Kyber needs to call multiple separate key functions, with some reaching 5×10^5 operation cycles.

PQC schemes, including PQ signatures and PQ key-exchanges, consist of KeyGeneration, encapsulation/signing, and decapsulation/signature verification, utilizing sub-routines such as number theoretic transform and inverse number theoretic transform for polynomial multiplication, Montgomery reductions for polynomial arithmetic, and hash functions like SHA2 and SHA3 for sampling [3, 4], with parallelizable functions like NTT, INTT, and Montgomery reductions providing significant speed improvements of up to 57 times using ARM instructions on DPUs, while non-parallelizable functions like SHA3 hash functions have received less attention in literature and thus, are not explored in this letter.

In order to bring down the PQC algorithms to the millisecond regime, the most intensive processing functions have been offloaded to a dedicated ARM processor for cryptographic hardware accelerations. The results bench-mark reference codes and the proposed stack with hardware accelerations are shown in Figures 3 and 4 for Dilithium and Kyber, respectively.

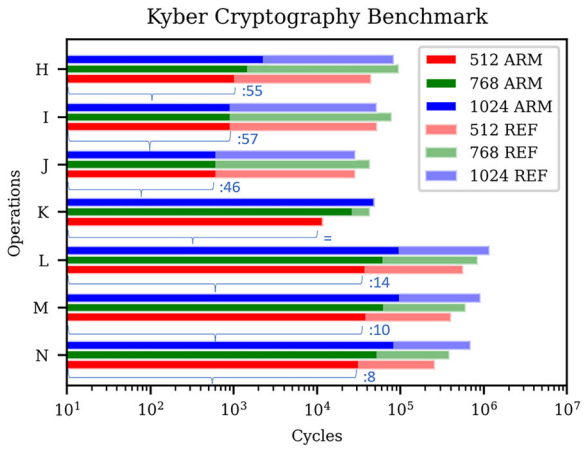

Fig. 4 *An analysis is conducted on the CPU performance of Kyber's main functions, utilizing function names obtained from the Kyber reference code [4]. The functions examined include: H=mont_red; I=invntt; J=ntt; K=gen_a; L=crypto_kem_dec; M=crypto_kem_enc; and N=crypto_kem_keypair To evaluate their performance, both the reference (REF) and ARM optimized implementation codes are tested on a DPU, and the resulting data is presented. The findings reveal a significant reduction of 57 times in CPU cycle, as clearly depicted in the graph.*

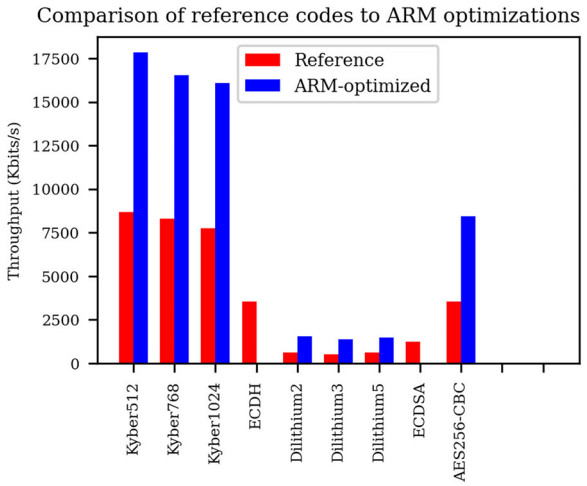

Fig. 5 *Throughput achieved for the diverse algorithms used in our PQC link. Results show that PQC algorithms like Dilithium and RSA can outperform their classical counterparts ECDH and ECDSA.*

The results for Dilithium shown Figure 3 highlight improvement in CPU cycles factor ranging from 5 to 31 in key functions. Similarly, Kyber results are shown in Figure 4 and yield improvement rates ranging from 8 up to 57 times.

Both Dilithium and Kyber clearly benefit in terms of cycle utilization reduction in functions that are highly parallelisable, e.g. the (NTT), which in essence conducts a series of concurrent polynomial multiplications [3, 4].

Results-network performance: Once Dilithium and Kyber algorithms are integrated into a handshake pipeline to establish a PQC channel between the two DPUs, all data is then encapsulated through AES-256 (key length of 256 bits). Figure 5 shows the throughput performance for different PQC implementations, for the reference model and for ARM-processor optimized PQC algorithms. This plot highlights two relevant pieces of knowledge: Dilithium reaches much lower aggregated throughput than Kyber (as anticipated by Figures 3 and 4, as Dilithium requires more building blocks which consume more cycles), and ARM-optimizations double the throughput across the board.

The implemented algorithms for Kyber include its 512, 768, and 1024 variations, which aim at security roughly equivalent to AES-128/192/256, respectively. The throughput performance indicates that

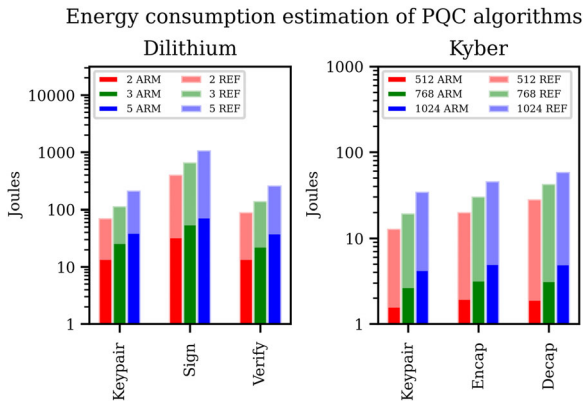

Fig. 6 *Estimation of the energy consumption of 1000 instantiations of Kyber and Dilithium in our PQC link. Our comparison includes the three different versions of Dilithium and Kyber, considering their security level as well as their corresponding ARM optimized and reference (REF) implementations.*

once the optimization is chosen, a small 8% penalty is taken for Kyber to operate as its highest robustness level.

Figure 5 also shows that Kyber key exchange outperforms its classical analogous elliptic curve Diffie–Hellman (ECDH) and elliptic curve digital signature (ECDSA).

Results-energy consumption: When real communication systems utilize accelerations and ARM instructions, the overall energy consumption decreases in a roughly linear manner with the CPU cycles required for its operations. An estimate of the energy consumption during the implementation is presented below in Figure 6, assuming the PQC link is functioning between two DPUs equipped with ARMv8 processors that operate at a minimum clock frequency of 600 MHz and a maximum frequency of 1500 MHz.

Conclusion: This letter presented the first end-to-end PQC link operating between two server systems through dedicated DPUs. DPUs execute all the networking functions required to establish a link with Dilithium and Kyber in combination with classic AES-256 encryption. The high-processing requirements of PQC algorithms are met by offloading their functions to dedicated on-board ARM processors, which provide hardware accelerations, therefore yielding reduction factors in cycles reaching from 5 up to 57 for specific functions.

Finally, throughput performance results are provided for a link operating with both the reference designs for PQC algorithms and the ARM-optimized algorithms, yielding more than 2.5 increasing factor of the capacity per link when using the latter as it can be seen in Figure 5. Multiple implementations of PQC have been studied in the past.

PQC enables security in a future with quantum computers; this letter has shown that PQC can be utilized in real-life scenarios incurring no sustancial penalties in terms of latency on the authentication and key exchange segments of the communication.

Author contributions: The authors provide a transparent account of their individual contributions to this research paper on the author section of this document.

Acknowledgments: This work was partly funded by the QUARC project by the European Union Horizon Europe research and innovation program within the framework of Marie Skłodowska-Curie Actions with grant number 101073355.

Conflict of interest statement: The authors declare that they have no financial or personal relationships that could potentially influence the research, analysis, or interpretation of the data presented in this paper. This includes any financial affiliations, employment, consultancies, stock ownership, or honoraria from organizations that may have a direct or indirect interest in the subject matter discussed.

Data availability statement: The data that support the findings of this study are available upon request. Due to confidentiality and privacy restrictions, the raw data cannot be publicly shared. However, interested researchers may contact Eindhoven University of Technology at a.c.a.cano.aguilera@tue.nl to request access to the data for the purpose of replication or further analysis. Any data provided will be de-identified and comply with relevant data protection regulations.

Received: *14 June 2023* Accepted: *18 July 2023*
doi: 10.1049/ell2.12901

References

1 Parra-Rodriguez, A., et al.: Digital-analog quantum computation. *Phys. Rev. A* **101**, 022305 (2020). doi:https://link.aps.org/doi/10.1103/PhysRevA.101.022305
2 Alagic, G., et al. *Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process*. National Institute of Standards and Technology, Gaithersburg, MD (2022)
3 Ducas, L., et al.: Crystals-Dilithium: a lattice-based digital signature scheme. *IACR Trans. Cryptographic Hardware and Embedded Syst.* **2018**(1), 238–268 (2018). doi:https://doi.org/10.13154/tches.v2018.i1.238-268
4 Bos, J., et al.: Crystals - kyber: A CCA-secure module-lattice-based KEM. In: *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 353–367. IEEE, Piscataway, NJ (2018)
5 Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. *J. ACM* **60**(6), 1–35 (2013). doi:https://doi.org/10.1145/2535925
6 Dworkin, M., et al.: *Federal Information Processing Standards (NIST FIPS) Advanced Encryption Standard (aesAES)*. National Institute of Standards and Technology, Gaithersburg, MD (2001)
7 Daemen, J., Rijmen, V.: The block cipher rijndael. In: Quisquater, J.J., Schneier, B., (eds.) *Smart Card Research and Applications*, pp. 277–284. Springer Berlin, Heidelberg (2000)
8 Gunasekaran, M., Rahul, K., Yachareni, S.: Virtex 7 FPGA implementation of 256 bit key AES algorithm with key schedule and sub bytes block optimization. In: *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, pp. 1–6. IEEE, Piscataway, NJ (2021)
9 Xing, Y., Li, S.: A compact hardware implementation of CCA-secure key exchange mechanism CRYSTALS-Kyber on FPGA. *IACR Trans. Cryptogr. Hardware Embedded Syst.* **2021**(2), 328–356 (2021). DOI:https://doi.org/10.46586/tches.v2021.i2.328-356
10 Dang, V.B., Mohajerani, K., Gaj, K.: High-speed hardware architectures and FPGA benchmarking of CRYSTALS-Kyber, NTRU, and Saber. *IEEE Trans. Comput.* **72**(2), 306–320 (2023). DOI:https://doi.org/10.1109/TC.2022.3222954
11 Bisheh-Niasar, M., Azarderakhsh, R., Mozaffari-Kermani, M.: Instruction-set accelerated implementation of CRYSTALS-Kyber. *IEEE Trans. Circuits Syst. I: Regul. Pap.* **68**(11), 4648–4659 (2021). doi:https://doi.org/10.1109/TCSI.2021.3106639
12 Imran, M., Aikata, A., Roy, S. S., Pagliarini, S. (2023). High-speed Design of Post Quantum Cryptography with Optimized Hashing and Multiplication. IEEE Transactions on Circuits and Systems II: Express Briefs, 1–1. doi:https://doi.org/10.1109/tcsii.2023.3273821
13 Fouque, P.A., et al.: Falcon: Fast-fourier lattice-based compact signatures over NTRU. *Post-Quantum Cryptogr. Stand.* **36**(5), 1–75 (2018)
14 Bernstein, D.J., et al.: The SPHINCS+ signature framework. In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. CCS '19*, pp. 2129–2146. Association for Computing Machinery, New York, NY (2019)
15 Becker, H., et al.: Neon NTT: Faster Dilithium, Kyber, and Saber on Cortex-A72 and Apple M1. *IACR Trans. Cryptogr. Hardware Embedded Syst.* **2022**(1), 221–244 (2021). doi:https://doi.org/10.46586/tches.v2022.i1.221-244
16 Sanal, P., et al.: Kyber on ARM64: compact implementations of Kyber on 64-bit ARM Cortex-A processors. In: Garcia-Alfaro, J., Li, S., Poovendran, R., Debar, H., Yung, M., (eds.) *Security and Privacy in Communication Networks*, pp. 424–440. Springer, Cham (2021)