Blending Scheduling Barriers: A Hybrid Approach for FPGA-based Post-Quantum Cryptography

***Abstract*—FPGAs are promising platforms for Post-Quantum Cryptography (PQC) hardware through electronic design automation tools such as High-Level Synthesis (HLS). Previous endeavors have demonstrated that incorporating pragmas and making adjustments to the code can enhance the Quality of Results (QoR) [1] [2]. However, manually modifying the original code presents two notable drawbacks: (1) the performance can be suboptimal, and (2) it is highly prone to interpretation that may compromise functionality. In this work, we leverage innovative HLS techniques based on dynamic scheduling, that typically do not require any code alterations to maximize performance to the greatest extent possible, using principles that are similar or equivalent or as done for out-of-order (OoO) CPUs. In particular, we highlight the benefits of combining static and dynamic HLS and how it opens new challenges for further research in this field.**

I. Introduction

HLS enables the development of algorithms coded in C/C++, followed by the automatic generation of the corresponding hardware description. However, the QoR very seldom leads toward performance-optimized solutions because of the complexity of the software code and limitations in extracting memory parallelism and dependencies at compile time. The intuitive way to workaround the slowdown due to the false dependencies is using novel HLS techniques that are able to detect dependencies at runtime. However, dynamically scheduled pipelines intrinsically bring resource overhead, requiring a thrifty use only when needed. In this work, we explore methodologies that use a combination of both static and dynamic scheduling techniques. This hybrid approach can strike a balance between resources and adaptability in HLS.

II. Hybrid Scheduling: Methodology

Scheduling Partitioning. Anticipating all possible pipelines that suffer from suboptimal results is not a trivial task. To discover that, we analyze the scheduling results after static HLS. If a loop schedule shows an iteration latency as the initiation interval (potentially affected by false or true loop-carried dependencies), we mark it as a module requiring dynamic scheduling.

Split and and Recombine. We split the part to be synthesized into the two different tool flows, adapting the interfaces and recombining them after the design has been singularly generated with the given technique. The final design will be a circuit that combines the two different scheduling typologies.

III. Preliminary Results

We applied our methodology to Kyber PQC, one of the standard candidates announced by NIST. The two modules requiring dynamic scheduling are *ntt* and *invntt*. As expected, they brought on an important overhead in resource utilization.

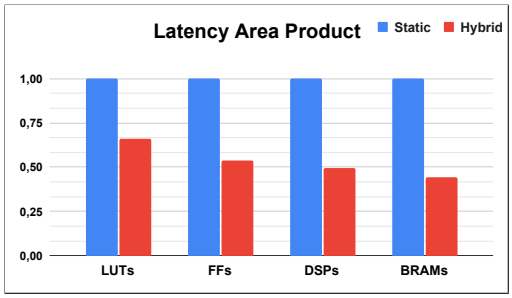

Fig. 1. Latency Area Product normalized for Kyber **crypto_kem_enc**. Both versions use the reference code. The hybrid solution is more efficient as the clock cycle reduction is more than the area overhead.

Nevertheless, we find that on the scale of the entire module, the area introduced is a lot less significant after recombination with the static design. A full dynamic design, in contrast, could be up to 100× larger with no more speedup than the one demonstrated in this work (dynamic unneeded everywhere).

TABLE I
Hybrid design uses +48% LUTs, +21% FFs, and +10% DSPs, with a speedup of 2.2× in clock cycles. The clock frequency is 100MHz. The target is Kintex-7 AMD FPGA.

Version	Latency (cc)	Speedup (×)	LUTs	FFs	DSPs	BRAMs
Static	170064	-	55292	38938	117	59
Hybrid	75679	2.25	82167	47118	130	59

IV. Conclusion and Future Work

Combining two different HLS scheduling techniques shows very encouraging results. As reported in Table I the design runs twice as fast while introducing reasonable overhead in resource utilization, hence a more efficient design considering the latency area product as shown in Fig. 1. Future work includes the analysis and exploration of dynamic scheduling pipelines and their effects on cryptographic security and vulnerabilities to side-channel attacks as well as energy efficiency trade-offs. Also, we plan to apply our method to the other PQC schemes to normalize the hybrid approach. Considerable effort still lies ahead, but we envision that mixing scheduling techniques will lead to high-quality electronic designs while designing FPGA-based PQC using HLS.

References

[1] R. Kastner, J. Matai, and S. Neuendorffer, "Parallel Programming for FPGAs," *ArXiv e-prints*, May 2018.

[2] A. Guerrieri, G. D. S. Marques, F. Regazzoni, and A. Upegui, "Optimizing post-quantum cryptography codes for high-level synthesis," in *2022 Euromicro Conference on digital systems Design (DSD22)*, Gran Canaria, Spain, 2022, pp. 361–67.