Innovative FPGA Implementation of High-Level Synthesis in Post-Quantum Cryptography: Hardware Acceleration for Falcon Digital Signature

FPGA實現高階合成於後量子密碼學：Falcon數位簽章之硬體加速整合快速傅立葉變換與數論轉換

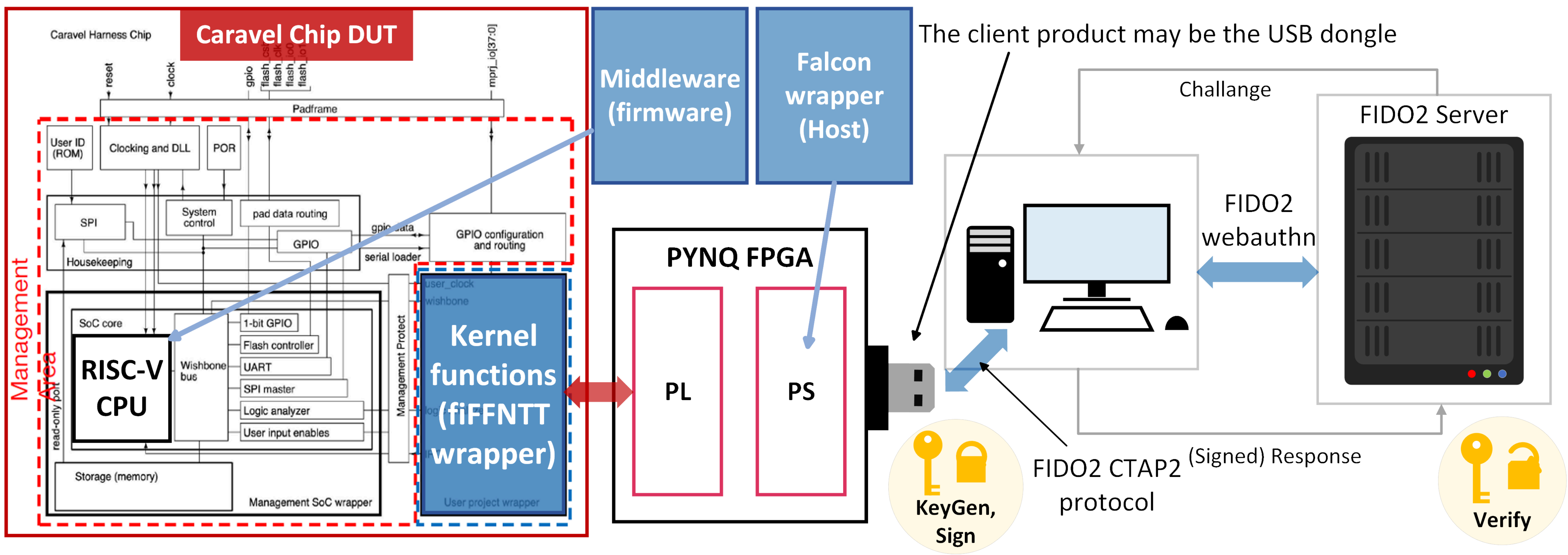
指導教授: 賴瑾 組員: 陳冠晰, 劉祐瑋, 陳昇達, 王彥智, 陳柏翰 組別: A400

Abstract

Falcon is an innovative lattice-based digital signature scheme that represents a significant leap forward in the field of post-quantum cryptography. Our research targets to overcome this challenge of lengthy execution time in software. We speed up the Falcon computation by accelerating the critical kernels, such as Fast Fourier Transform (FFT), inverse FFT (iFFT), Number Theoretic Transform (NTT), and inverse NTT (iNTT), enhancing both computational efficiency and security. We employ High-Level Synthesis (HLS) to speed up the hardware development process and restructure the data path to optimize performance.

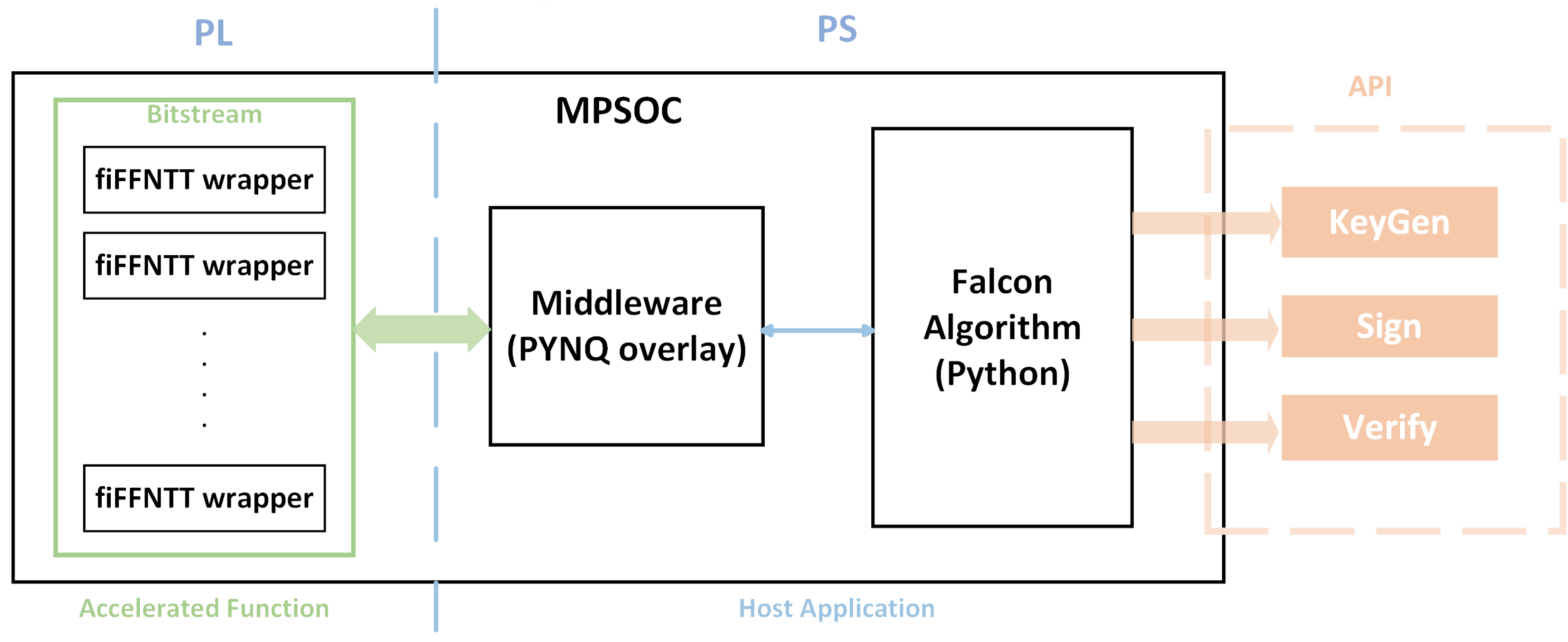
A particularly innovative aspect of our work is the hardware/software co-design approach. This method enables the Falcon algorithm to be executed more efficiently on hardware platforms. The hardware/software co-design is optimized by moving the kernel management from software to firmware running in an embedded RISC-V CPU in the FPGA. This implementation allows the full Falcon process to be executed on the board, significantly reducing execution times compared to its software-only counterpart.

System View

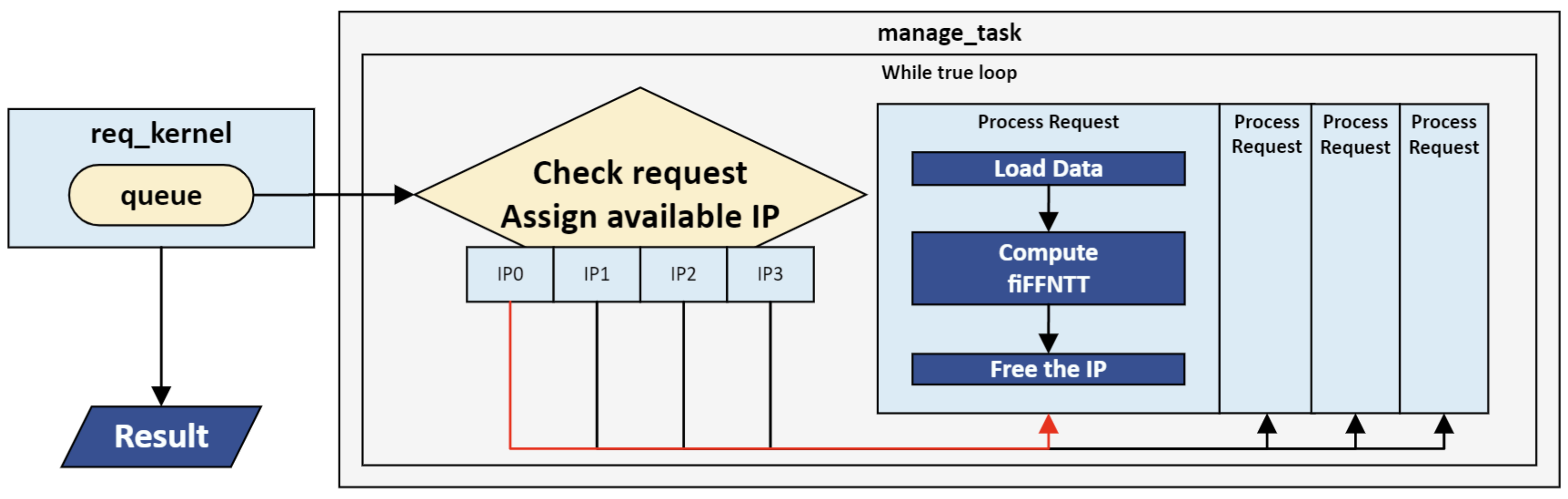


The kernels are integrated into the Caravel SoC (which will be taped-out in the project's next phase). The Caravel chip DUT communicates with FPGA through a customized interface. The Falcon authentication function is realized in an FPGA through a USB dongle, which communicates with the PC via FIDO2 CTAP2.

HW/SW Codesign



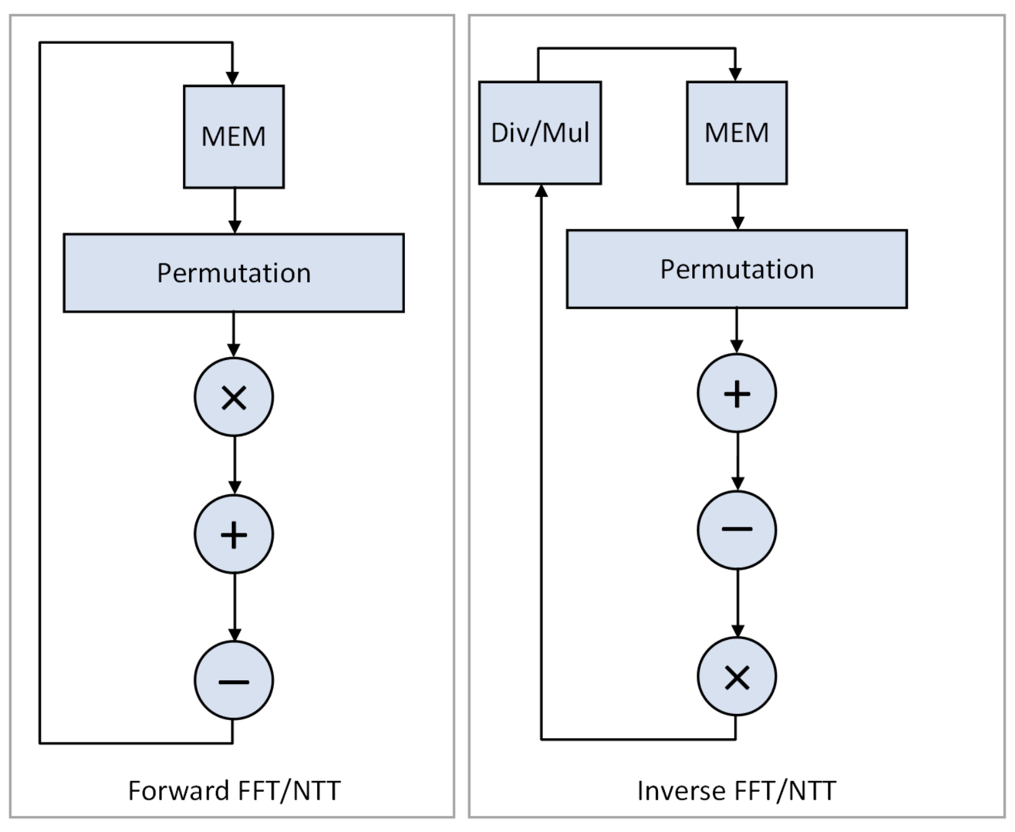
We integrated multiple kernels into the FPGA to allow parallel processing. This integration led to the development of a middleware located at the SoC side of the FPGA, capable of preloading and handling multiple requests from the host program (Falcon algorithm) and returning values as required. To reduce the high communication overhead between hardware and software (indicated by the green double arrow), the middleware will be written into firmware and loaded onto the RISC-V CPU of the Caravel SoC.

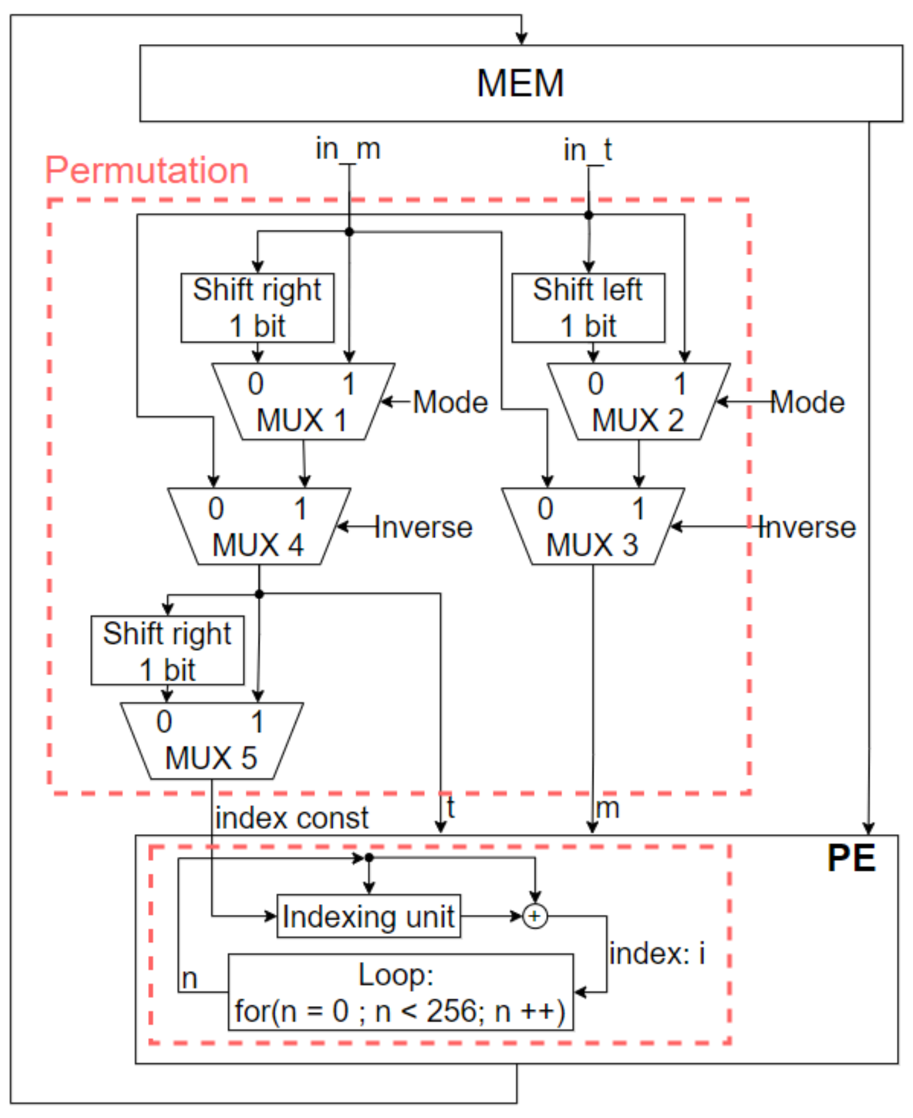


Although the way the middleware manages requests has greatly accelerated the entire Falcon flow, running the middleware on the PS side as software would still result in a hardware/software communication overhead issue. Further optimization is possible, so we plan to transform the middleware into firmware and place it on the hardware side to reduce communication overhead.

Hardware Acceleration: Kernel Function

We combined FFT, iFFT, NTT, and iNTT into one hardware IP to optimize hardware resources. Our optimization started by implementing a one-port memory with a custom permutation algorithm. We used a processing element (PE) with an in-place memory buffer, a double shifter to reduce FPGA DSP usage, sharing memory with different datatypes for FFT/NTT, and decomposing the complex and Montgomery multiplication with shared multiplier in the PE to limit hardware resource usage.





Performance

Kernel execution time

Function	FFT (ms)	iFFT (ms)	NTT (ms)	iNTT (ms)
Python	28.3617	29.9833	32.8956	34.3557
Original HLS	1.7429	2.2315	3.3797	4.3449
Final optimization	0.1372	0.1631	0.1951	0.1773

Falcon execution time

Version	KeyGen (ms) (compute pk)	Sign (ms)	Verify (ms)
Original software	100.4	2053.3	139.9
HW/SW co-design with middleware	18.9	747.5	60.7

Reference

Fouque, Pierre-Alain, et al. "Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU Specification." v1.2, 1 Oct. 2020, https://falcon-sign.info/

Michael Schmid, Dorian Amiet, Jan Wendler, Paul Zbinden, and Tao Wei, "Falcon Takes Off - A Hardware Implementation of the Falcon Signature Scheme," 2023