Framework for Evaluating Software/Hardware Implementations of Post-Quantum Public-Key Algorithms Using Zynq SoC

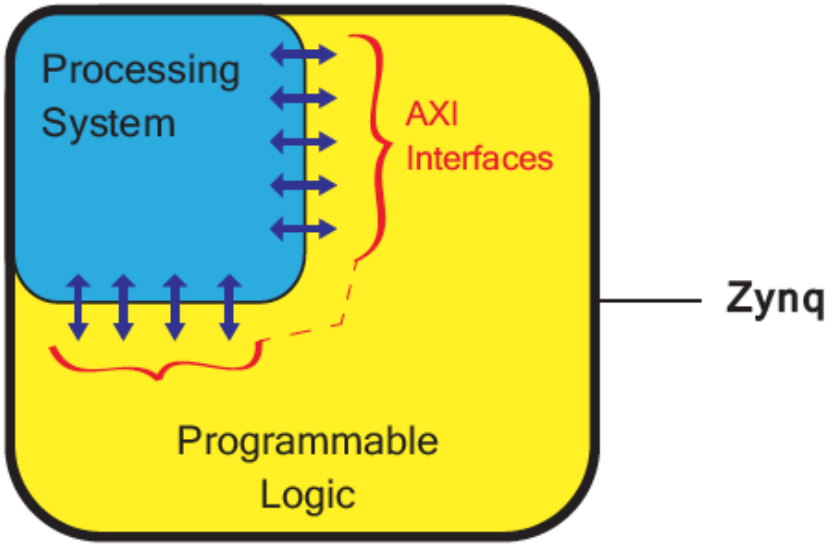
Brian Loop, Ahmed Ferozpuri, and Kris Gaj
George Mason University
USA

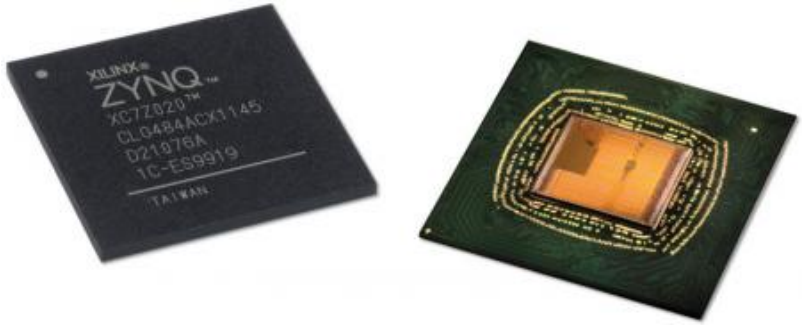
http:/cryptography.gmu.edu
https://cryptography.gmu.edu/athena

Motivation

- Multiple families of post-quantum cryptosystems with different performance in software & hardware at the same security level
- Early benchmarking can help focusing the attention of cryptographers & cryptanalysts on the most promising algorithms and parameter sets
- Hardware & embedded software will play major role in providing security for Internet of Things
- Complex PQC schemes provide interesting opportunities for software/hardware codesign

Our Platform - Xilinx Zynq System on Chip

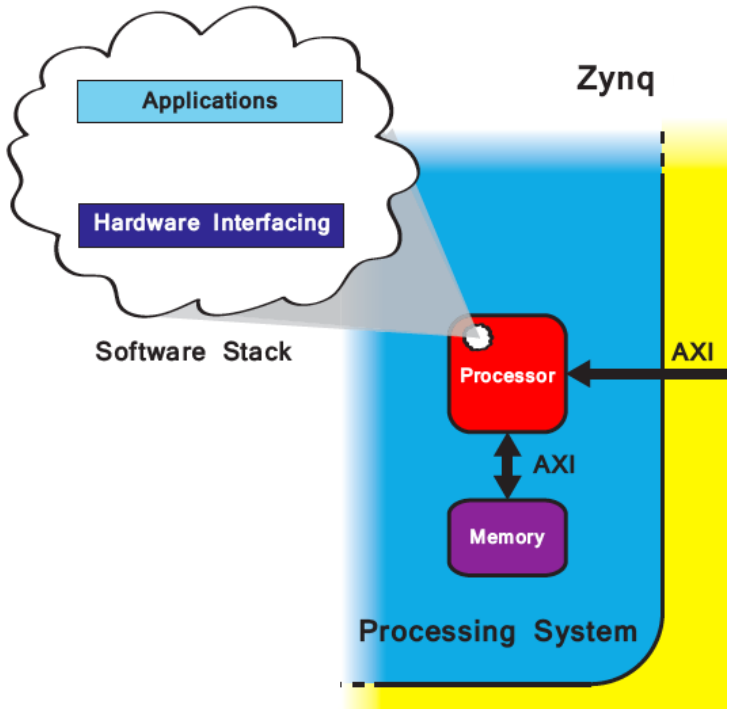




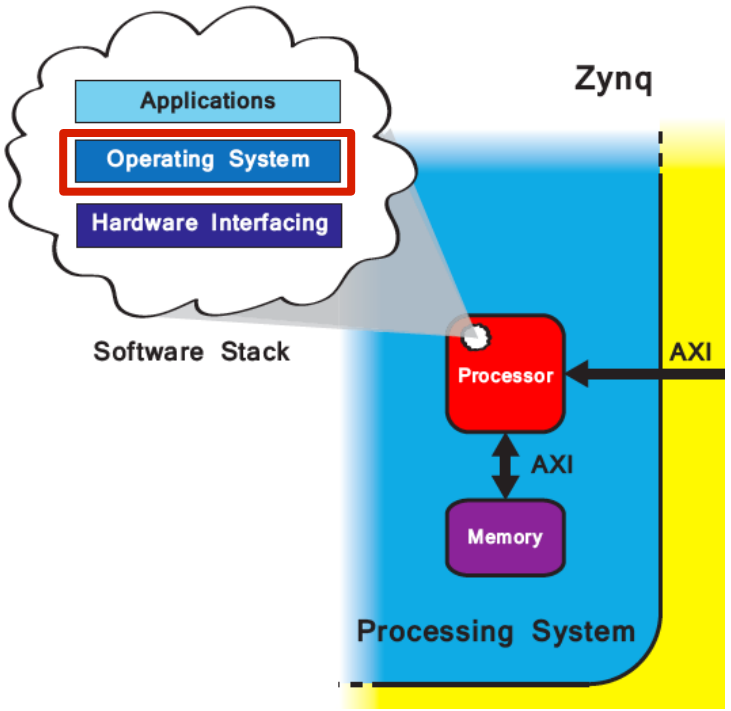
- Processing System (PS) – Dual-core ARM microprocessor system
- Programmable Logic (PL) – Series 7 FPGA logic
- Advanced eXtensible Interface 4 (AXI4) – 4th generation of ARM high-performance interface

Two Primary Modes of Operation

Bare Metal



Linux



Two Primary Modes of Operation

Bare Metal

- Full control over software execution
- Very small overhead
- Limited functionality
- Suitable for straightforward and repetitive tasks

Linux

- Provides separation between software and hardware
- Multiple software libraries, such as OpenSSL
- User friendly
- Open-source platform
- Constantly updated
- Extensively tested
- Wide range of applications, from supercomputers to the Internet of Things devices
- Possible overhead

Currently Investigated PQC Schemes

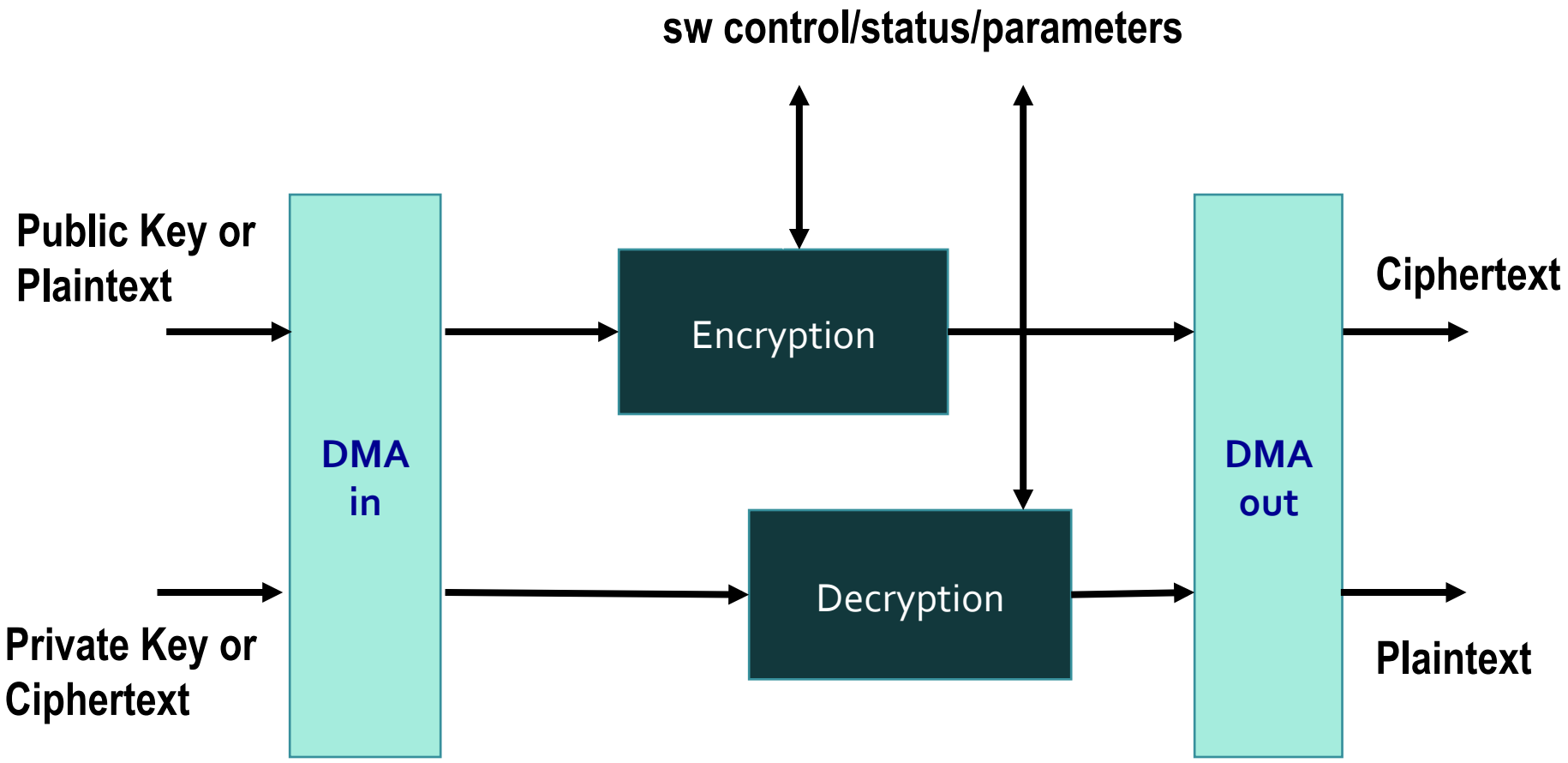
1. Encryption Scheme – Code-based QC-MDPC

- Based on the McEliece PKC
- Key sizes reduced by using the Quasi-Cyclic Moderate Density Parity-Check (QC-MDPC) codes
- Security level: 80 bits (due to area limitations)
- Parameters: n_0=2, N=9600, R=4800, W=90, T=84
- Speed-optimized implementation based on the work by Stefan Heyse, Ingo von Maurich, and Tim Güneysu

2. Signature Scheme – Lattice-based BLISS

- Bimodal Lattice Signature Scheme (BLISS)
- Security level: 192 bits (BLISS-IV)
- Parameters: n=512, q=12,289, δ_1=0.45, δ_2=0.06, α=0.55, etc.
- Security-optimized implementation based on the work by Thomas Pöppelmann, Léo Ducas, and Tim Güneysu

Hardware Accelerators



- Developed in VHDL
- Based on the open-source codes from Ruhr-University Bochum, Germany
- Standard external DMA (Direct Memory Access) cores

Software Drivers

Bare Metal

- Generated automatically by Xilinx tools

Linux

- Based on the open source driver cryptodev
- Allows writing to and reading from the character device dev/crypto
- Renamed pqcryptodev – custom kernel driver for performing PQC encryption/decryption
- Middleware allowing access to the hardware cryptographic modules from user-space applications

Measured Parameters – Bare Metal & Linux

- Public Key Operation (Encryption / Signature Verification) vs. Private Key Operation (Decryption / Signature Generation)
- End-to-end execution time
  - Input Transfer
  - Hardware Accelerator
  - Output Transfer
- Maximum clock frequency
- Operations per second
- Key generation time (in software)
- Key transfer time

All times measured using a hardware AXI timer with the accuracy of 1 cycle of 100 MHz clock = 10 ns

Examples of the First Results: QC-MDPC

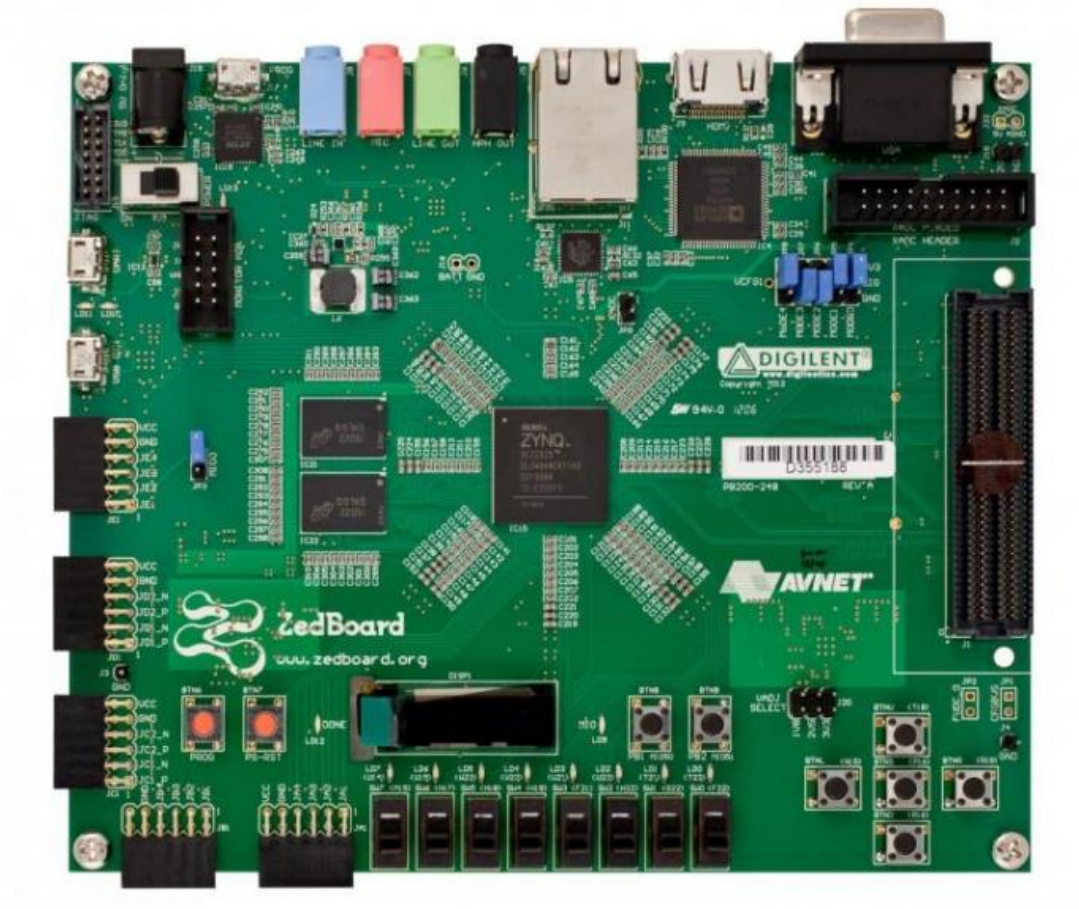
Bare Metal

Public Key Loading:	213 cycles =	2.1 µs
Encryption (end-to-end):	6407 cycles =	64.1 µs
Private Key Loading:	362 cycles =	3.6 µs
Decryption (end-to-end):	24048 cycles =	240.5 µs

Linux

			Linux Overhead
Public Key Loading:	747 cycles =	7.5 µs	350.7%
Encryption (end-to-end):	7014 cycles =	70.1 µs	9.5%
Private Key Loading:	1222 cycles =	12.2 µs	337.6%
Decryption (end-to-end):	26038 cycles =	260.4 µs	8.3%

ZYNQ Evaluation and Development Board – ZedBoard



Linux on the SD card
Price < 500 USD
Academic discounts

Remaining measurements in progress

To be reported at the upcoming FPGA conferences & ePrint

Thank you!

Comments?

Questions?

Suggestions?

ATHENa: http:/cryptography.gmu.edu/athena
CERG: http://cryptography.gmu.edu