Adam's Bridge
Side-Channel Protection

SECURITY

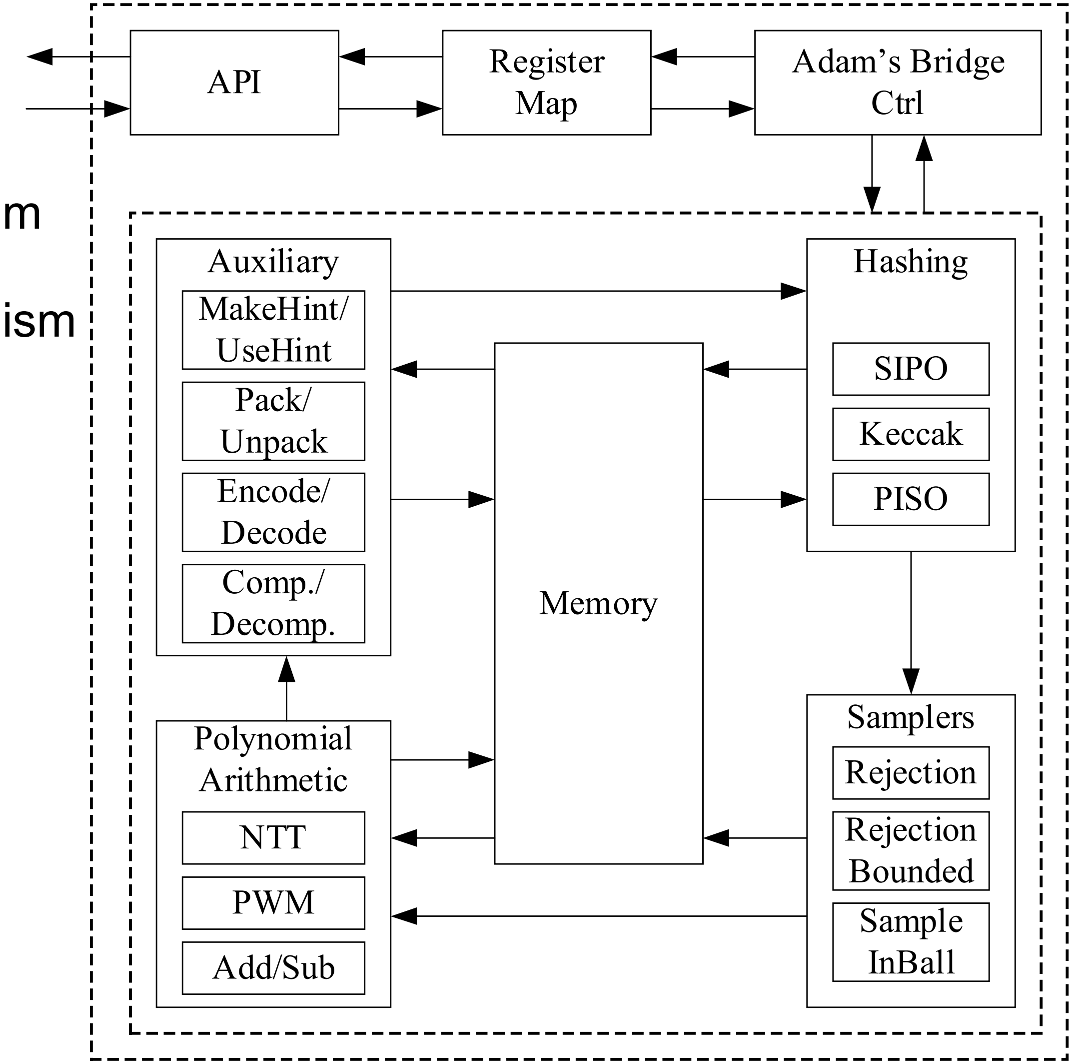
Mojtaba Bisheh-Niasar, Senior Security Architect, Microsoft

Emre Karabulut, Senior Security Architect, Microsoft

Adam’s Bridge Accelerator

- **PQC Accelerator**
  - Dilithium (ML-DSA) Digital Signature Algorithm
  - Kyber (ML-KEM) Key Encapsulation Mechanism
- **Two performance levels target:**
  - Embedded Architecture
  - High-Speed Architecture
- **Highest Security Level (Level-5)**
- **Embedded SCA countermeasures**
- **Open for public:**

  https://github.com/chipsalliance/adams-bridge



ML-KEM vs ML-DSA

ML-KEM		ML-DSA
	Lattice-Based scheme	
	NTT-based polynomial multiplication	
	NTT-friendly prime	
12-bit prime		23-bit prime
Incomplete NTT		Complete NTT
Pair-wise multiplication		Point-wise multiplication
	Keccak	
Binomial Sampling		Uniform Bounded Sampling
	Rejection Sampling	

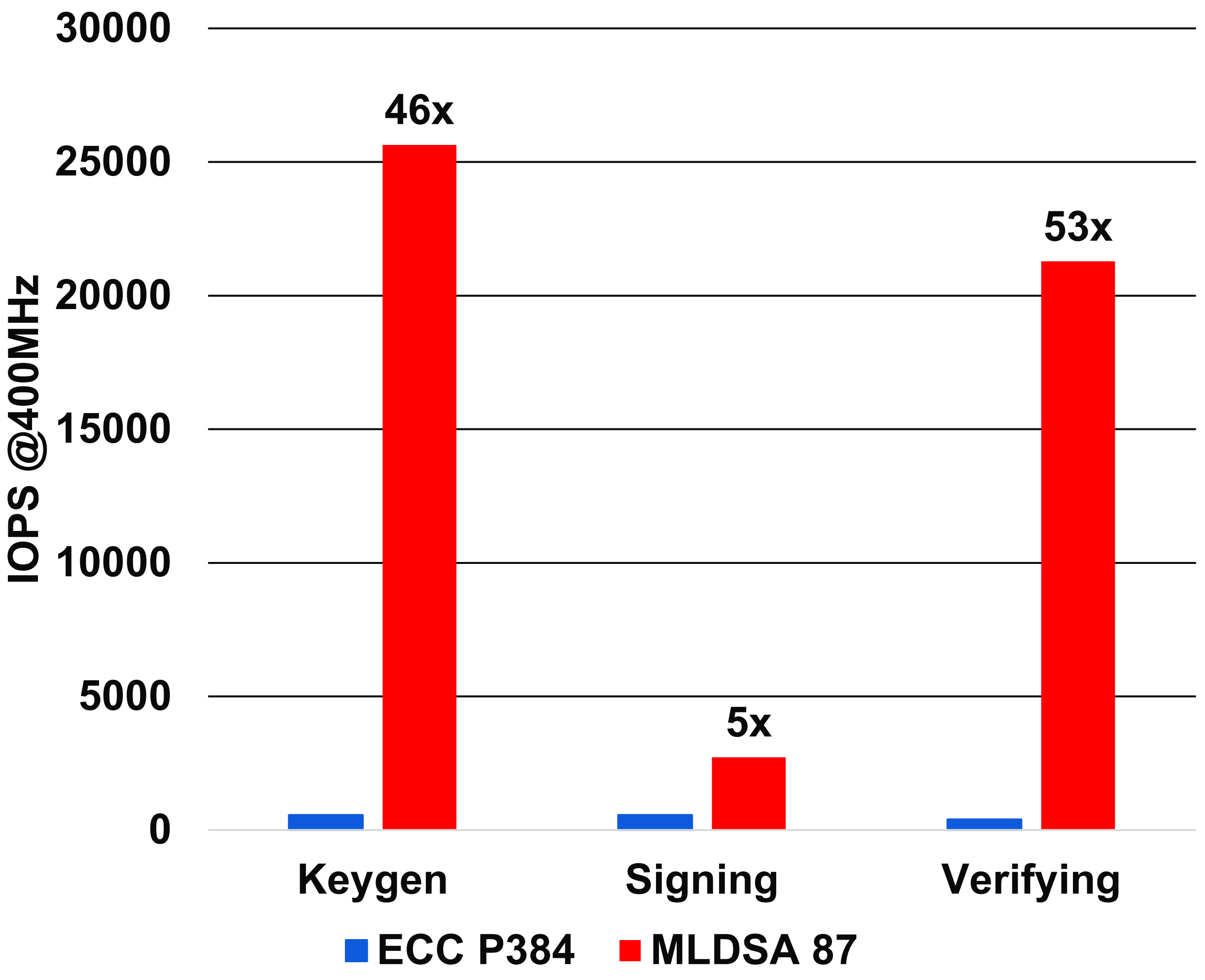
Key Size Comparison

Algorithm		Secret Key (Bytes)	Public Key (Bytes)	Signature Size (Bytes)
Dilithium	ML-DSA-87	4864	2592	4595
SPHINCS+	SPHINCS+-SHAKE256-256f-simple	128	64	49216
FALCON	FALCON-1024	2305	1793	1280
LMS	LMS256H15W4	1179648 = 36×215	64	524 (per leaf)
ECC	Secp384r1	48	96	96
RSA	RSA-2048	256	256	256

Algorithm		Decapsulation/ Secret Key (Bytes)	Encapsulation/ Public Key (Bytes)	Ciphertext Size (Bytes)	Shared Secret Key (Bytes)
Kyber	ML-KEM-1024	3168	1568	1568	32
ECC	Secp384r1	48	96	N/A	48
RSA	RSA-2048	256	256	256	N/A

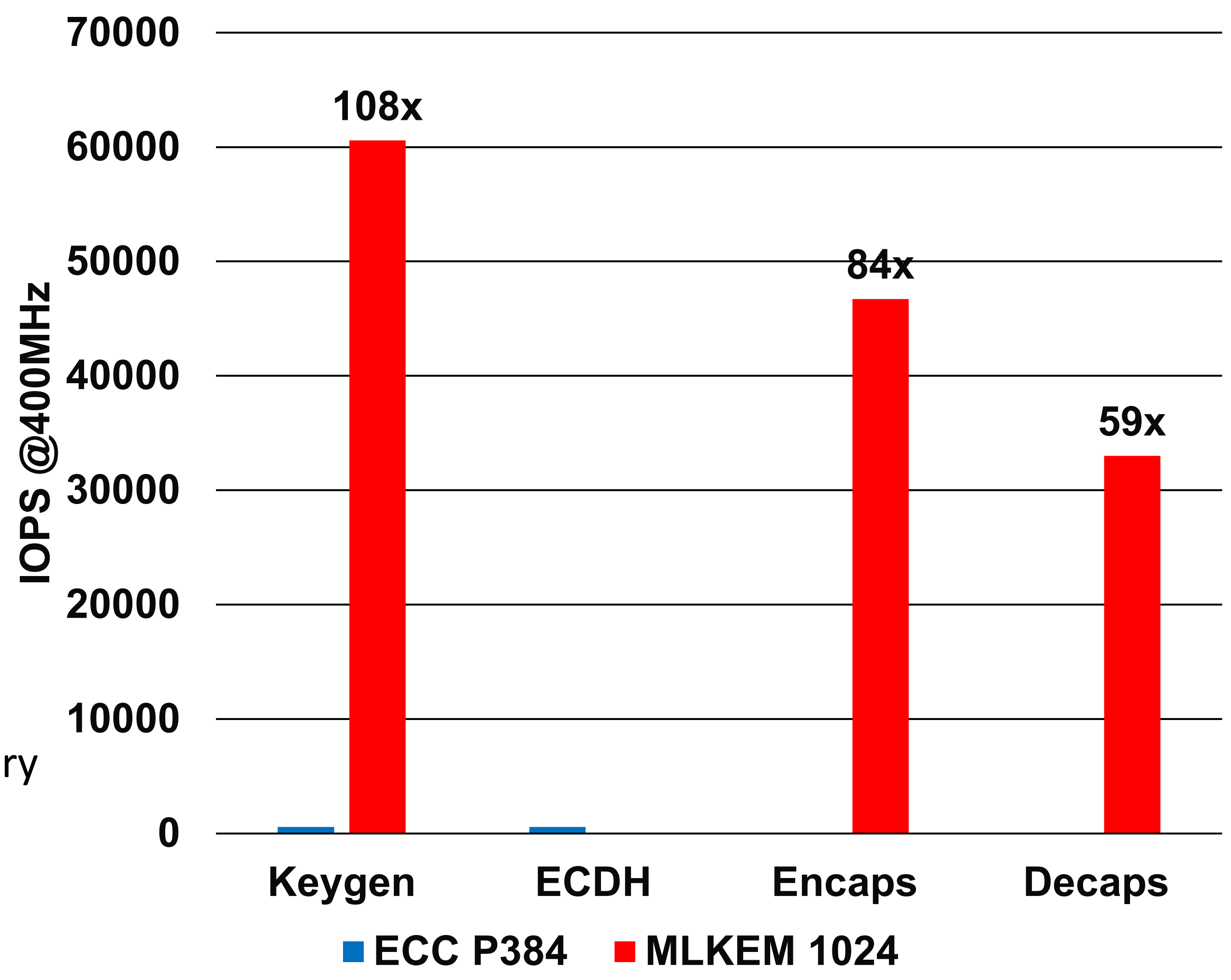
Adam's Bridge ML-DSA Specifications

- Signing in 36,700 cycles
  - ~ 92 usec @ 400 MHz
- Signing Rejection loop
  - Average: 3.85 signing rounds
  - ~ 367 usec @ 400 MHz
- 99.99% success: 31 Signing rounds
  - ~ 2.8 msec @ 400 MHz
- Comparison with Secp384r1:
  - ~ 1.8 msec @ 400 MHz
- Seed Caching Technique to save KV memory
  - KeyGen + Signing

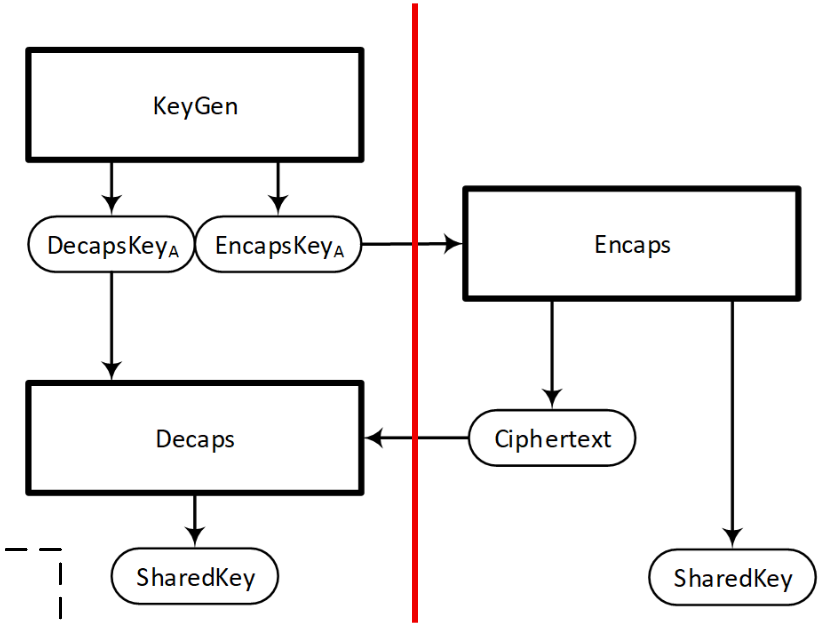


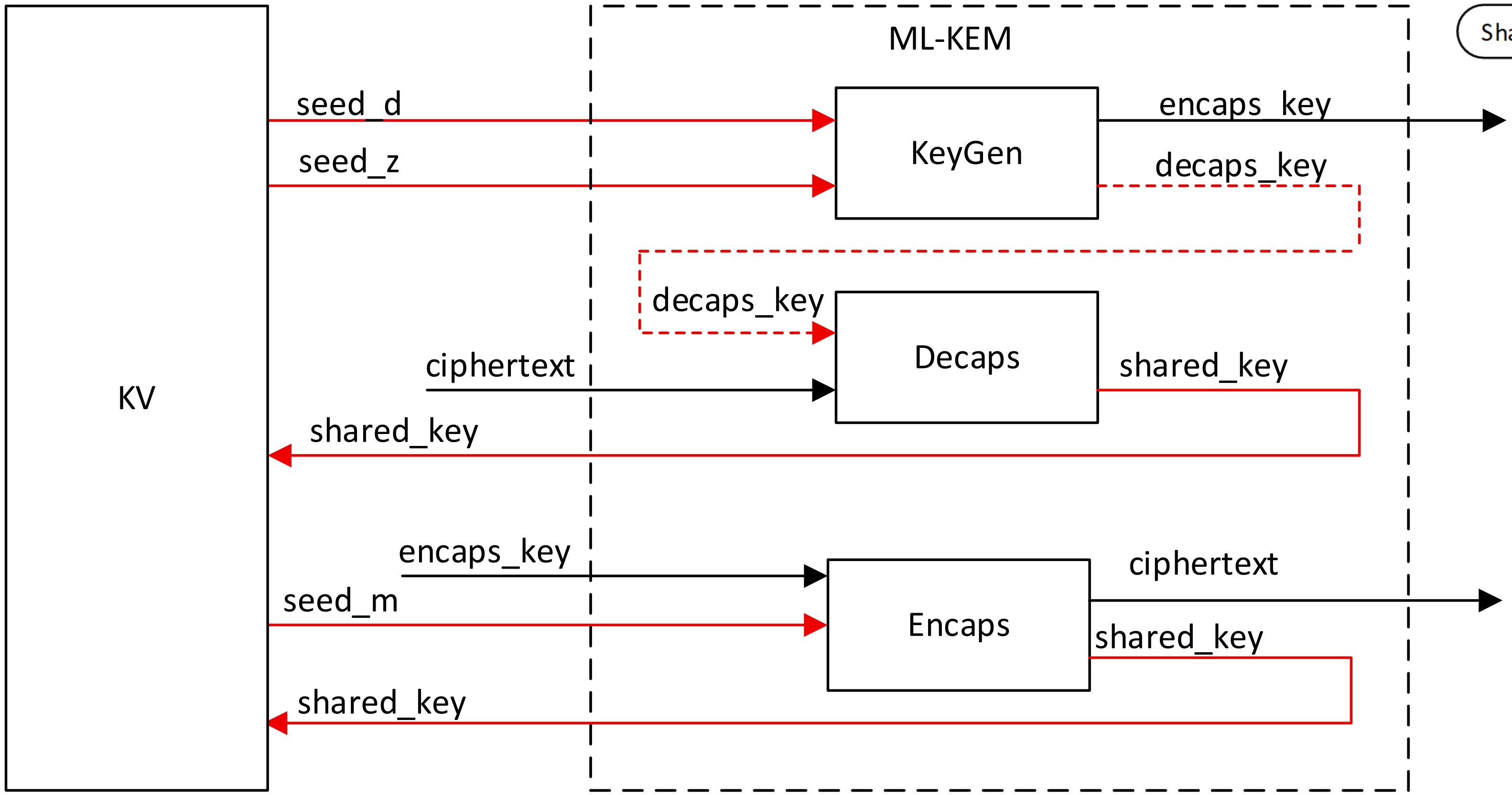
Adam's Bridge ML-KEM Specifications

- KeyGen in 6,600 cycles
  - ~ 17 usec @ 400 MHz
- Encapsulation in 8,500 cycles
  - ~ 21 usec @ 400 MHz
- Decapsulation in 12,100 cycles
  - ~ 30 usec @ 400 MHz
- Comparison with Secp384r1:
  - ~ 1.8 msec @ 400 MHz
- Seed Caching Technique to save KV memory
  - KeyGen + Decaps



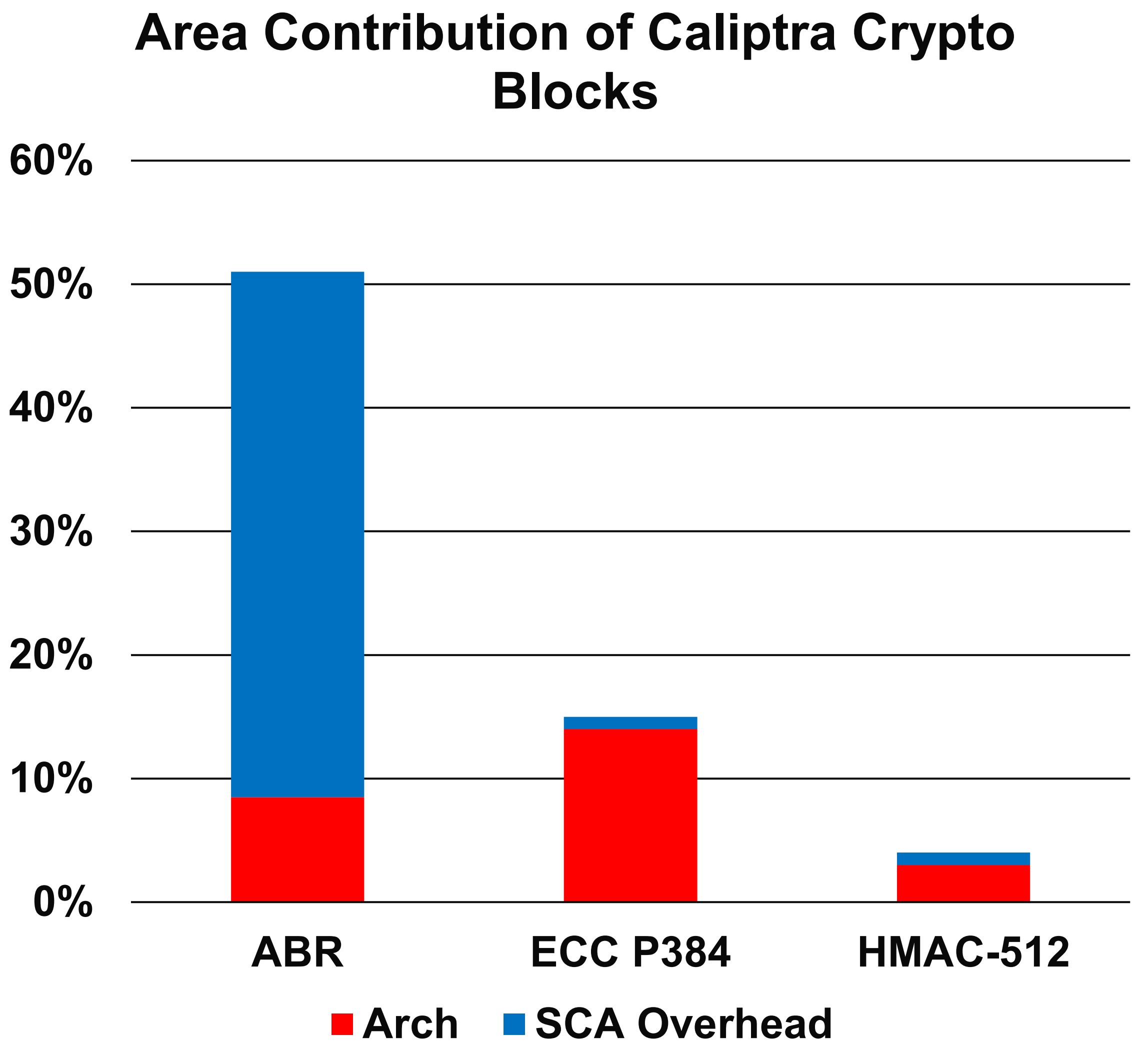
ML-KEM KV interface





Adam's Bridge Protection Cost

- ABR area including ML-DSA-87 and ML-KEM-1024 is around 3.4x of ECC P384
- The masked design results in:
  - 6x area overhead
  - 37% latency overhead MLDSA Signing
  - 10% Latency overhead MLKEM
- Compared to ECC P384, that needs 1.5x latency overhead with negligible area impact



Introduction to Side-channel attacks in PQC

- The security of PQCs against quantum computing does not guarantee a security against the side-channel attacks (SCAs)
- Allow extracting secret values from the implementation characteristics such as execution time, power consumption, and electromagnetic radiation
- More than 10 different attacks on CRYSTALS-DILITHIUM implementations in addition to our vulnerability discoveries
- An extreme case: the secrets can be can extracted with a single execution

Third PQC Standardization Conference

The NIST Post-Quantum Cryptography Standardization Process has entered the third ph
alternate candidates are being considered for standardization. NIST held the third NIST
2021 to discuss various aspects of these candidates, and to obtain valuable feedback fo
of the 15 finalists and alternates, was invited to give a short update on their algorithm.

The conference was held virtually.

Call for Papers

Agenda (includes links to on-demand videos)

On-Demand Videos

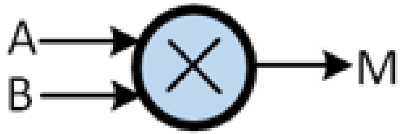
- Session I - Welcome and Candidate Updates
- Session II - Security I
- Session III - Hardware
- Session IV - NIST/DHS Talk and Side Channels
- Session V - Applications
- Session VI - Candidate Updates

Challenges

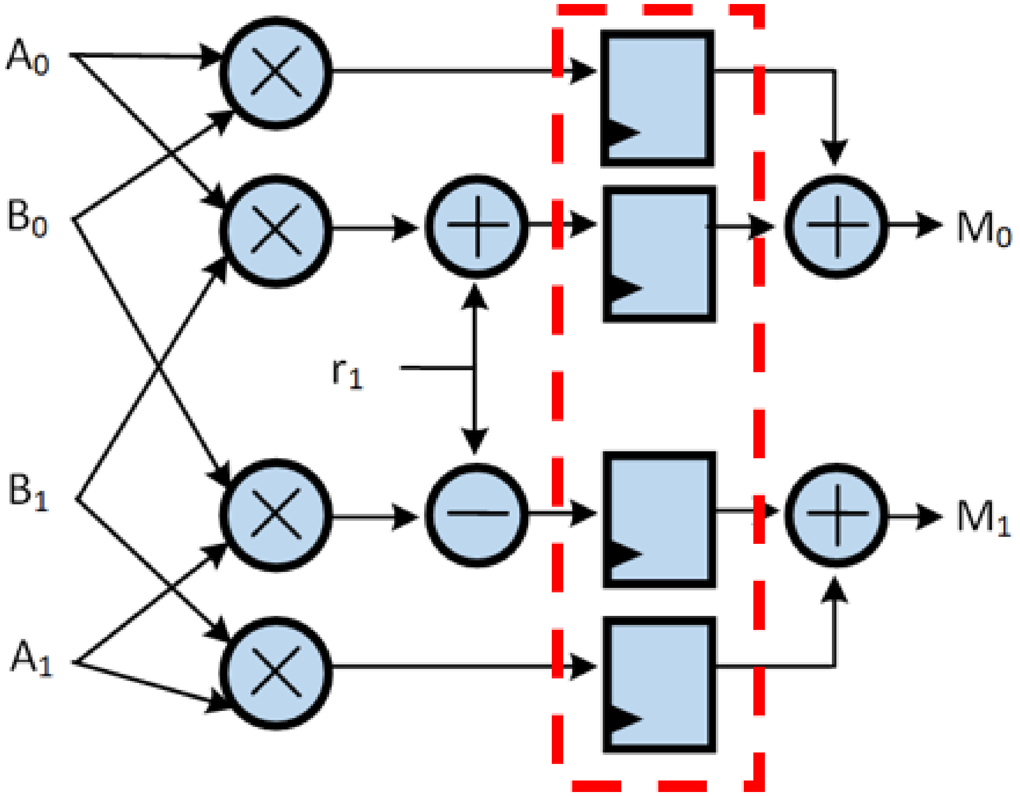
- SCA can break AES-256 in a few seconds
- PQC introduces unique operations (NTT, sampler, etc.), meaning… new vulnerabilities
- Existing solutions are not trivial to extend
- Side-channel attacks target **implementation**, not algorithm!
- Cost
    - Existing solutions are expensive (area-delay overhead up to ≈ 4,569x)

How to Hide SCA Leakage

SCA Vulnerable Multiplier

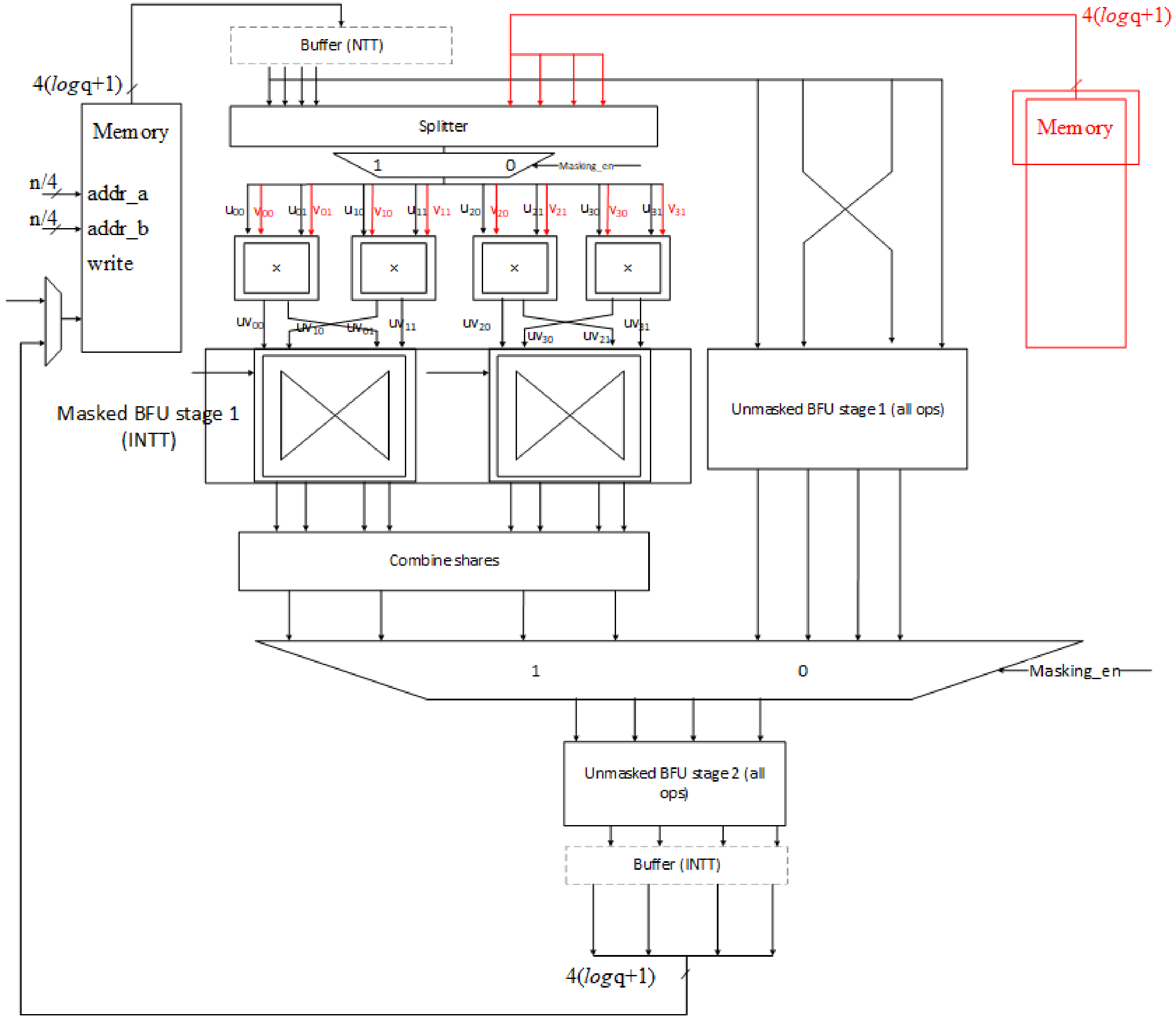


SCA Secure Multiplier



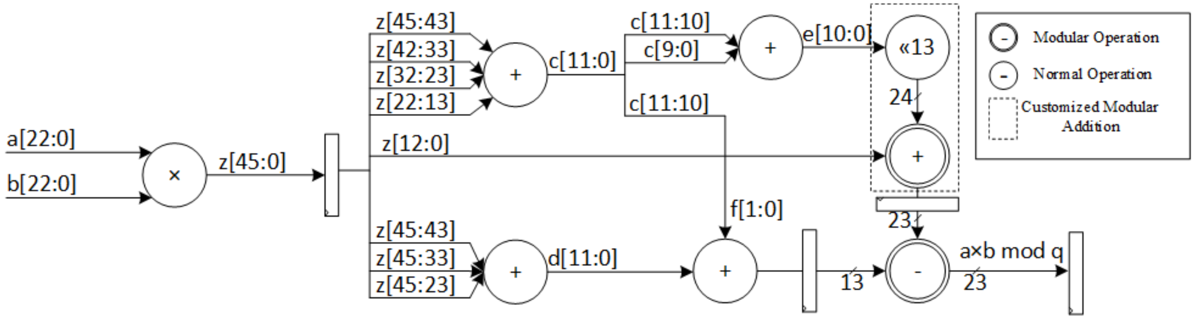
Masking + Shuffling in Adam's Bridge – NTT, PWM, PWA, PWS

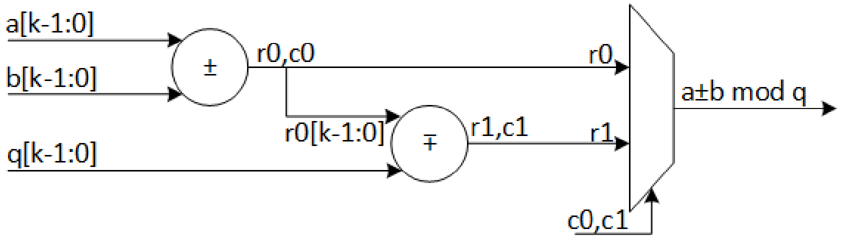
- Hybrid approach
- Masked PWM + 1st stage masked INTT + shuffling
- Subsequent stages of INTT operation are only shuffled
- 4x latency overhead on one stage of INTT

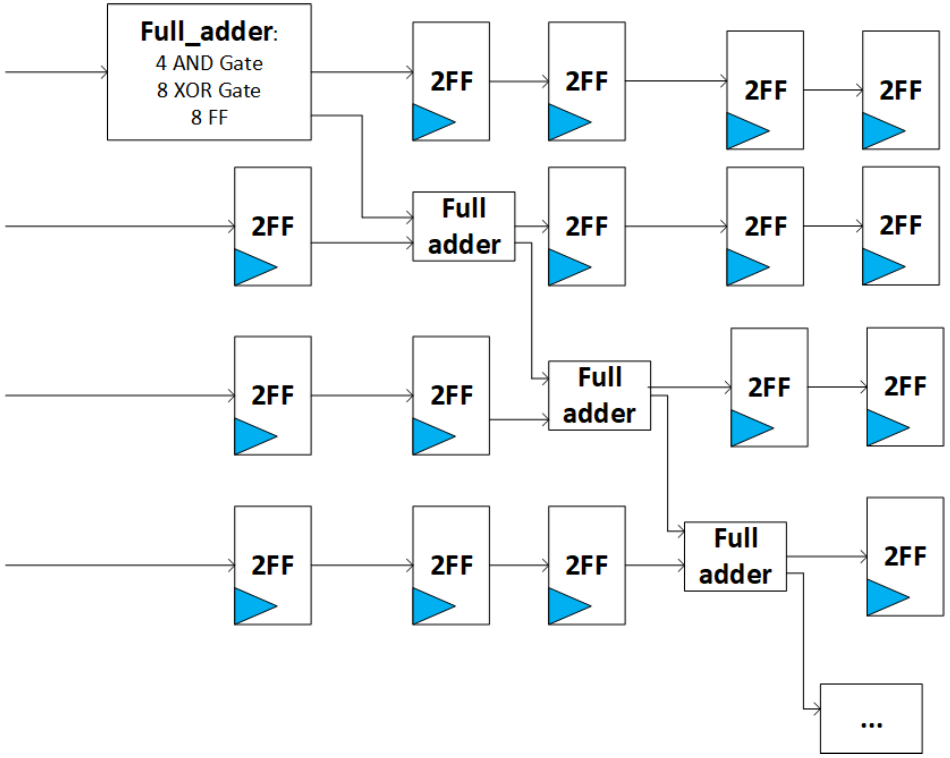


Masked Boolean Domain Overhead on Reduction

Efficient architecture for un-masked design



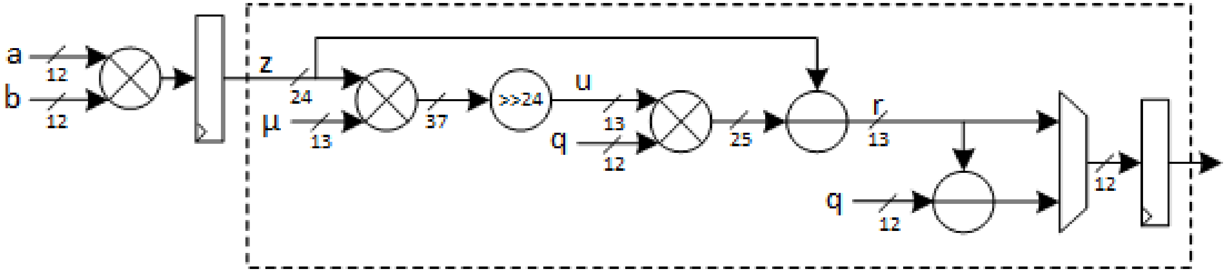


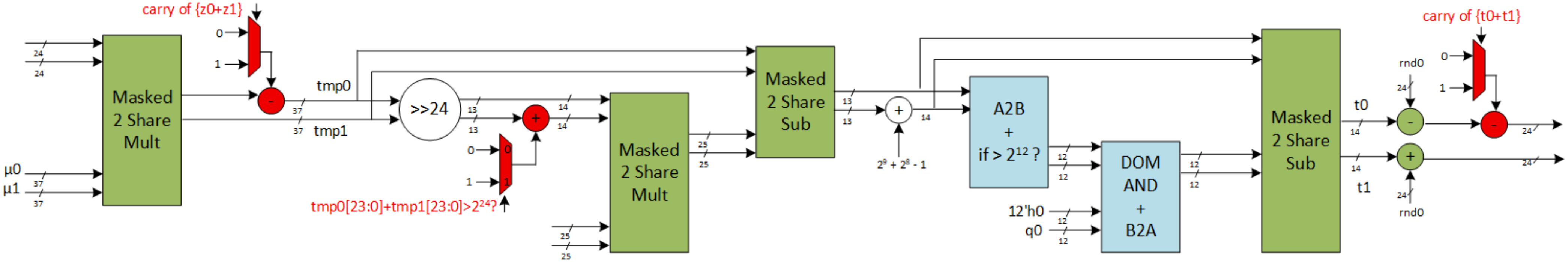


Masking Optimization (Barret Reduction) with 2.1 Version

- Removed A2B/B2A conversions → saved area & reduced latency
- Fully pipelined Barrett reduction → higher throughput
- Replaced lazy reduction MUX → Barrett reduction after Karatsuba

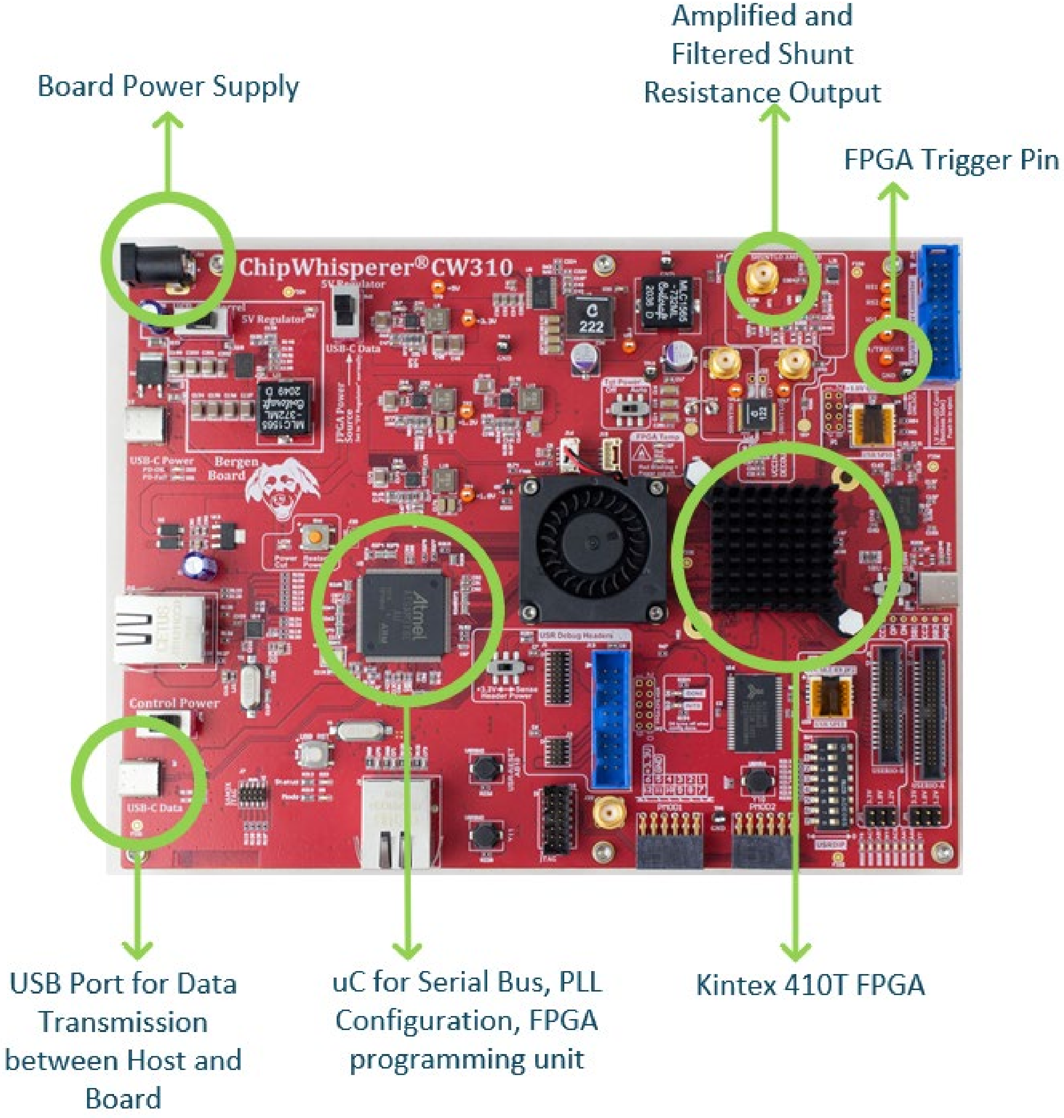
Efficient architecture for masked design



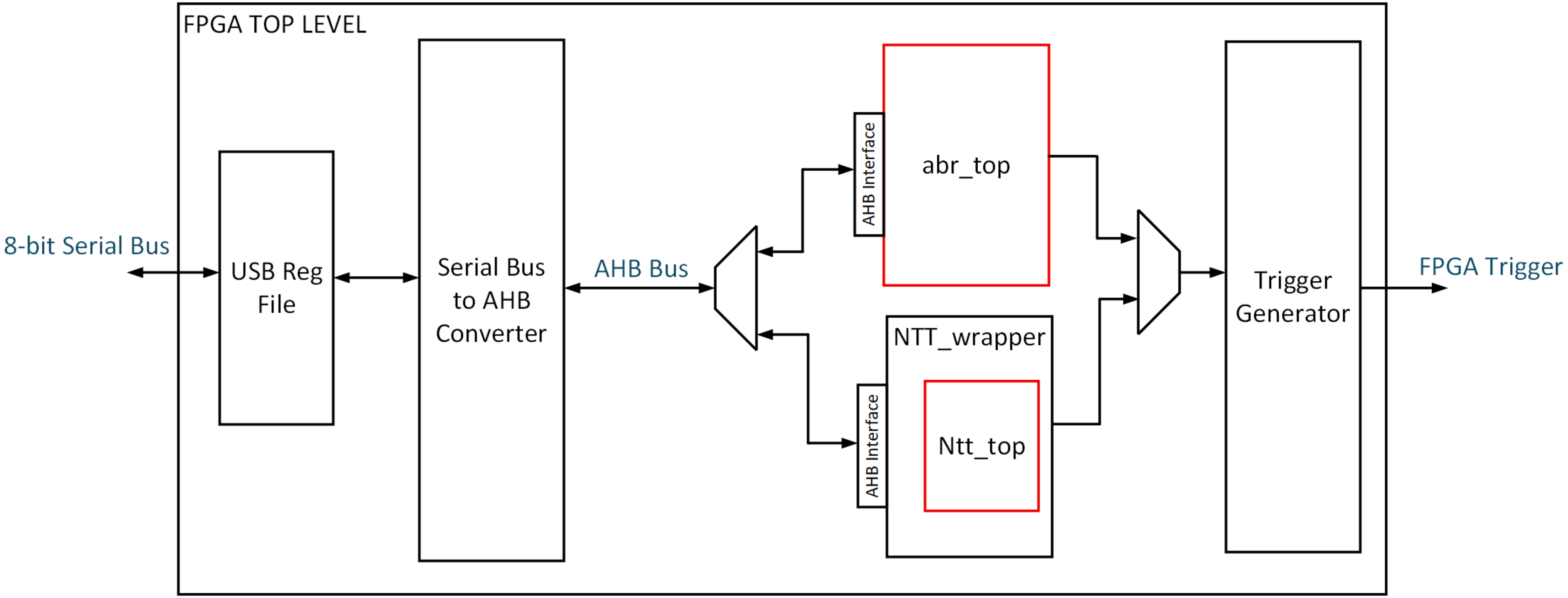


SCA Victim Device

- **FPGA Bitstream Generation**
  - Created from Adam’s Bridge (ABR) RTL fi
- **Microcontroller (uC) Setup**
  - Programs the FPGA with the ABR bitstrea
  - Configures the PLL
  - Manages data transmission between HOS and FPGA
- **FPGA Execution**
  - Runs the ABR logic
  - Generates a pulse signal on the Trigger P to indicate ABR operation start
- **Measurement Phase**
  - Oscilloscope captures voltage drop across shunt resistor for side-channel analysis



SCA Victim Device Hardware Architecture



SCA Experimental Flow

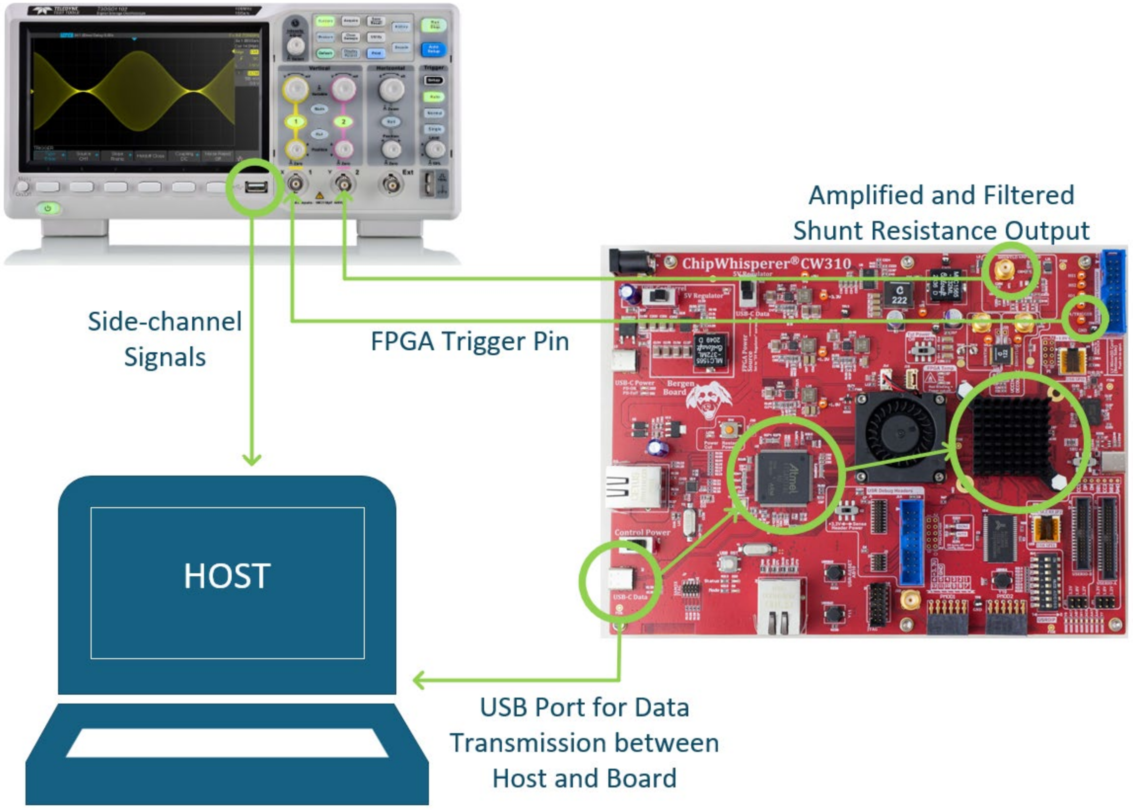
Initialization Phase
HOST

- Programs the FPGA with the ABR/NTT bitstream
- Configures PLL to generate a 10 MHz clock
- Sets up the oscilloscope:
  - Defines buffer size **N**
  - Specifies trigger channel and capture channel for **N** samples
  - Configures FPGA sampling rate (156MS/s)

Fork: Parallel Execution Begins

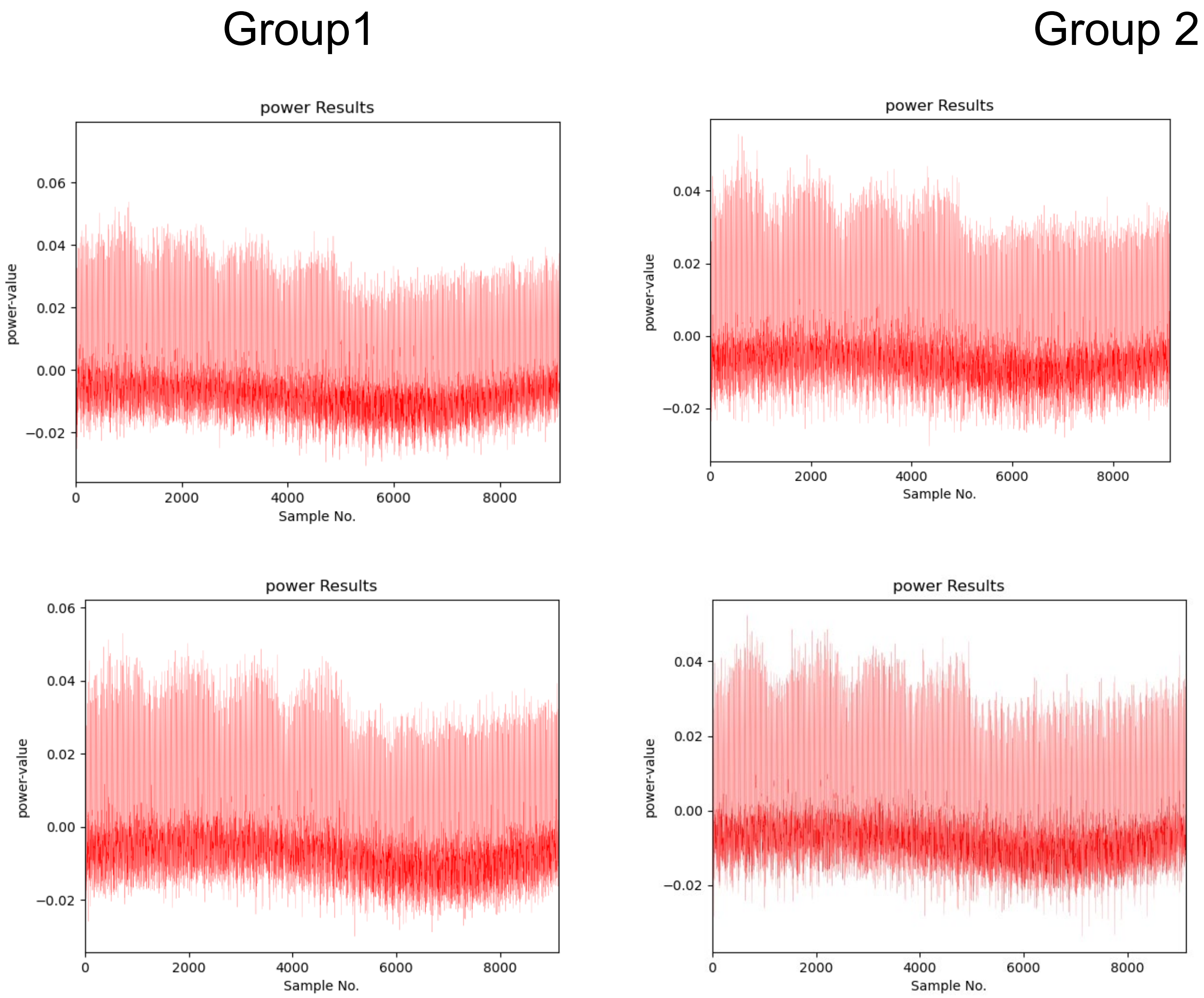
- **Branch: HOST**
  - Sends ABR inputs (e.g., seed, message)
  - Issues the **START** command
  - Waits for the **DONE** signal from FPGA
  - Retrieves **N** samples from the oscilloscope buffer
- **Branch: FPGA**
  - Waits for the **START** signal
  - Asserts the **Trigger Pin**
  - Executes the ABR operation
  - Asserts the **DONE** signal
- **Branch: Oscilloscope**
  - Waits for trigger signal on the configured channel
  - Captures **N** samples from the designated channel

Join: Synchronization Point



SCA Assessment Theory

- Can SCA traces reveal information about the operations being performed?
- Are your traces data-dependent?
- One simple method to investigate this is by creating two distinct groups:
    - **Group 1**: SCA traces captured using a fixed input for every execution. This helps establish a baseline for power consumption patterns when the input remains constant.
    - **Group 2**: SCA traces captured using randomly generated inputs for each execution. This group reflects how power consumption varies with different inputs.
- By comparing these two groups, you can assess whether the traces exhibit data-dependent behavior and potentially leak sensitive information.

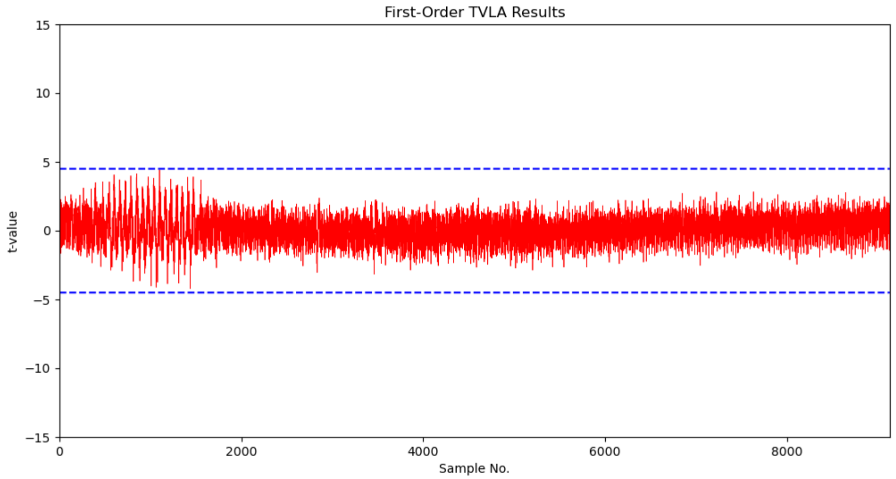


Understanding TVLA

- TVLA is a statistical method used to detect **side-channel leakage**
- **Welch's t-test** is applied to two sets of power traces:
  - **Group 1**: Fixed input
  - **Group 2**: Randomized input
- A significant difference in means (|t| > 4.5) indicates **potential leakage**.
- **Testing Modes**
- **Fixed vs Random (Non-specific Leakage Test)**
  Detects general leakage without targeting specific operations.
- **Group 1/2 Classification (Specific Leakage Test)**
  Categorizes traces based on known sensitive intermediate values (e.g., secret key bits).

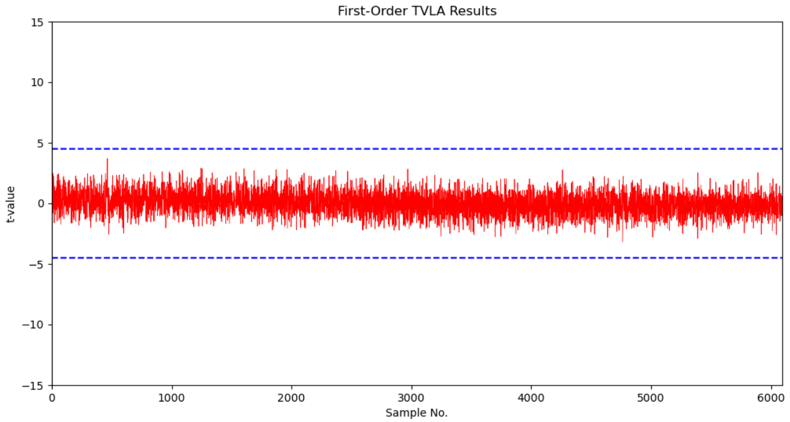
$$t = \frac{\mu_1 - \mu_2}{\sqrt{\frac{\sigma_1^2}{N_1} + \frac{\sigma_2^2}{N_2}}}$$

Not Secure

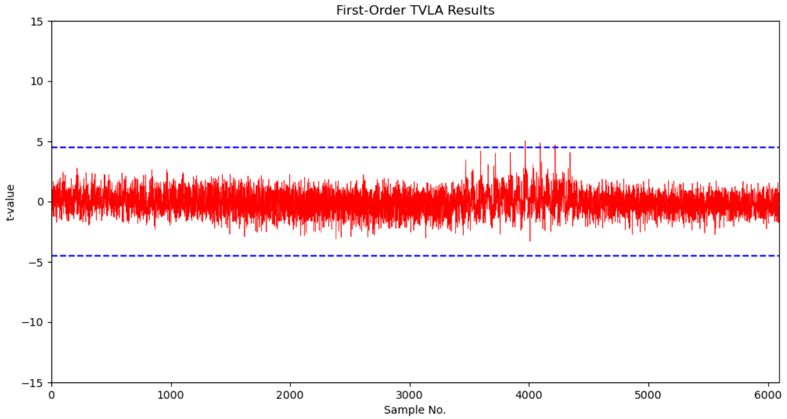


Adam's Bridge is Secure—TVLA results for 1M Traces

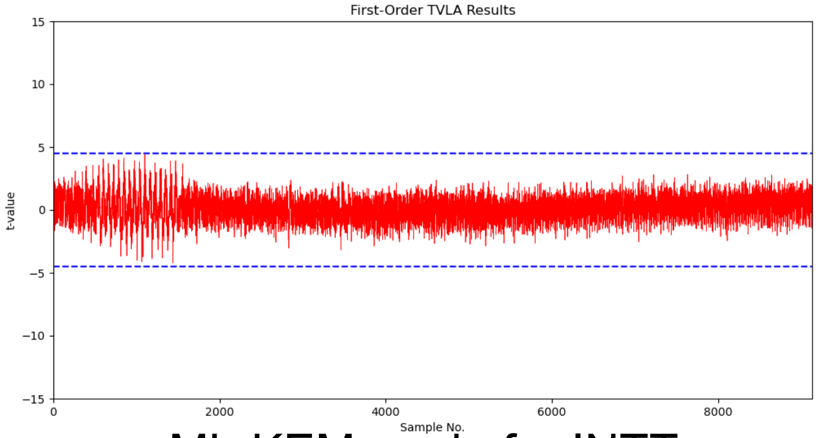
The t-scores consistently remain within the threshold (±4.5), demonstrating the empirical security of our masked design.



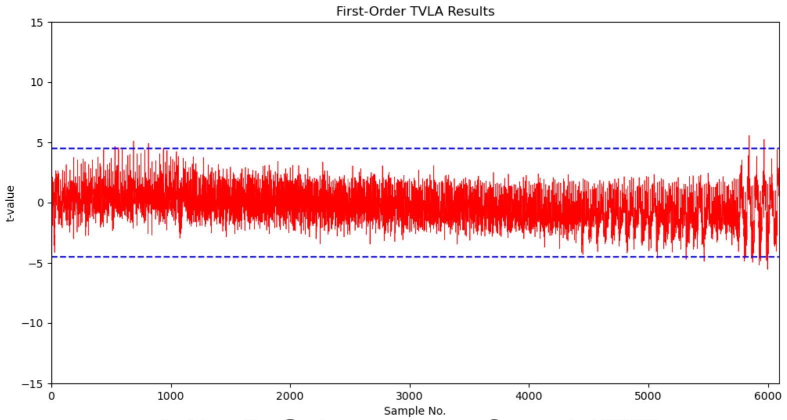
ML-KEM mode for PWM



ML-DSA mode for PWM



ML-KEM mode for INTT



ML-DSA mode for INTT

Thank you!

OPEN
Compute
Project®