A $333.9uW$ $0.158mm^2$ Saber Learning with Rounding based Post-Quantum Crypto Accelerator

Archisman Ghosh*, J.M.B. Mera‡, Angshuman Karmakar‡, Debayan Das*, Santosh Ghosh†, Ingrid Verbauwhede‡ and Shreyas Sen*

*School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA

‡COSIC, KU Leuven, Belgium

†Intel Labs, Intel Corporation, Hillsboro, OR. USA

Abstract—**National Institute of Standard & Technology (NIST) is currently running a multiyearlong standardization procedure to select quantumsafe or postquantum cryptographic schemes to be used in the future. Saber is the only LWR based algorithm to be in the final of Round 3. This work presents a Saber ASIC which provides 1.37X power-efficient, 1.75× lower area, and 4× less memory implementation w.r.t. other SoA PQC ASIC. The energyhungry multiplier block is 1.5× energyefficient than SoA.**

I. INTRODUCTION

The arrival of large-scale quantum computers will break the security assurances of our current public-key cryptography. National Institute of Standard & Technology (NIST) is currently running a multi-year-long standardization procedure to select quantum-safe or post-quantum cryptographic schemes to be used in the future. Energy efficiency is an important criterion in the selection process. This paper presents the first Silicon verified ASIC implementation for Saber (LWR algorithm as proposed in [1], [2]), a NIST PQC Round 3 finalist candidate in the key-encapsulation mechanism (KEM) category. Fig. 1 briefly describes the learning with rounding (LWR) problem, which is hard to solve even in the presence of large quantum computers due to the noise generated from rounding. IC features are tabulated in Fig. 1. which also shows a simplified version of the Saber KEM scheme to establish a secret key between two communicating parties Alice and Bob. Due to learning with rounding, secret s is hard to guess based on publicly available data as shown in Fig.1.

II. ARCHITECTURE

Fig. 2 represents the architecture with self-explanatory named blocks to compute KeyGeneration, Encapsulation and Decapsulation of Saber. The blocks in the full chip architecture are equivalent to other lattice-based schemes, each of the operations involved in lattice-based cryptography are supported. The key innovation lies in the 1) multiplication and its implementation 2) the memory management and 3) interface of the co-processor. 1) Computationally, the most expensive and complex part of Saber is the multiplication of polynomials of length 256. Fastest method for such type of multiplication is the number theoretic transform (NTT), a special form of fast Fourier transforms for finite fields (used in other PQC algorithm [4]). However, Saber's parameters do not directly satisfy the above condition for applying NTT. Saber with Toom-Cook and Karatsuba based multiplications have
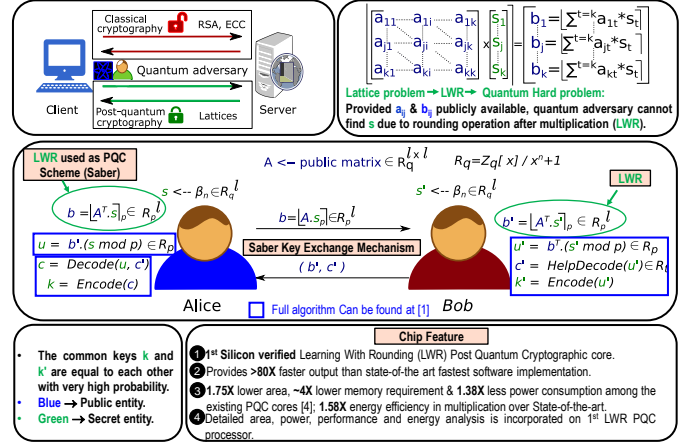

Fig. 1. Quantum resistant cryptographic core for IoT; Learning with rounding approach in lattice-based crypto; Key Exchange Mechanism (KEM) for a LWR based crypto algorithm (Saber); Chip features.

shown that implementations using these algorithms can be as fast as NTT multiplications. A brief Toom-Cook multiplication is shown in Fig. 2 along with the complete architecture. Since Saber needs the final result C(x) to be reduced by $x^{256} + 1$, striding Toom-Cook multiplication is adapted for memory efficiency (In this case, instead of splitting the polynomial A(x) (B(x) similarly) as $A_0(x) = a_0 + a_1x + ... + a_{63}x^{63}$, $A_1(x) = a_{64} + a_{65}x + ... + a_{127}x^{63}$,. . . , $A_3(x) = a_{192} + a_{193}x + ... + a_{255}x^{63}$ during the evaluation stage of classical Toom-Cook multiplication, the polynomial A(x) is split with striding of 4 as $A_0(x) = a_0 + a_4x + ... + a_{252}x^{63}$, $A_1(x) = a_1 + a_5x + ... + a_{253}x^{63}$, . . , $A_3(x) = a_2 + a_6x + ... + a_{255}x^{63}$, and resulting polynomial can be stored reduced by $x^{64} + 1$ reducing the memory requirement to half with respect to most efficient software Toom-Cook implementation [3]). Optimization of multiplication block (most power-hungry block) results in 1.37X power-efficient, 1.75X lower area and 4X less memory requirement w.r.t. other SoA PQC ASIC [4]. Our point multiplication is 1.58X energy efficient than state-of-the art NTT based multiplier design, which proves strongest candidacy of Saber in NIST PQC competition.

It should be noted that previous implementations as of now are either for LWE based scheme [5], or post silicon is not verified [6], [7]. Fig.3 explains the overview of the module that performs vector-vector multiplication in our accelerator. This

Saber IC Architecture

Input from outside

Serial to parallel Converter

High Speed Serial interface

Saber Accelerator

PolyMul Cache

VVectorMul

PolyMul

bus

Low Power Low Leakage Dual Port SRAM (8KB) **Data bus:** 64 bit

SHA3/Shake

BinomialSampler

AddPack

Unpack

AddRound

Verify

CMOV

CopyWords

CMD Controller

micro-instructions

✓Standard blocks (lower energy w.r.t multiplier)used in Saber.
✓Bus base architecture ensures reusability of functions across KeyGen, Encapsulation, Decapsulation.

Offset2	Offset1	Op.
23 .... 13 .... 3 0		

Op: 4 Bit Opcode
Offset$n$: Memory offset @Port$n$

0000: Mem Wr.
0001; Mem Rd
0010: SHA/SHAKE
0011: Sampler
0100: Addpack
0101: AddRound
0110: Unpack

Different Opcode
1001: Verify
1010: Copy
1011: CMOV
1100: SHAKE from outside

✗ Multiplier is most Area, Power & Energy hungry block
✗ NTT/FFT structure cannot be used due to Saber parameters.
✓ **Toom Cook** Multiplier for faster multiplication (FPGA, [8]).
✓ **Striding Toom Cook** for Memory efficiency.
✓ **1st ASIC Implementation** of LWR-based algorithm & Toom-Cook mult..

Classic Toom-Cook 4

Evaluation

A

$0 \quad n/4-1 \quad 3n/4-1 \quad n-1$

B

$0 \quad n/4-1 \quad 3n/4-1 \quad n-1$

A_0 ... A_6 B_0 ... B_6

$0 \quad n/4-1$ ... $0 \quad n/4-1$ $0 \quad n/4-1$ ... $0 \quad n/4-1$

Point Multiplication

C_0 ... C_6

$0 \quad 2n/4-2$ ... $0 \quad 2n/4-2$

Interpolation

Sequential memory access

C

$0 \quad$ Not Ring reduced $\quad 2n-2$

Classical Toom Cook [8]

• 4 Way classical Toom-Cook approach.
• Sequential memory access.
• Not ring reduced [3]
• 2X Cache requirement

Striding Toom-Cook 4

Evaluation

A

$0 1 2 3 4$ ... $n-1$

B

$0 1 2 3 4$ ... $n-1$

A_0 ... A_6 B_0 ... B_6

$0 \quad n/4-1$ ... $0 \quad n/4-1$ $0 \quad n/4-1$ ... $0 \quad n/4-1$

Point Multiplication

C_0 ... C_6

$0 \quad n/4-1$ ... $0 \quad n/4-1$

Interpolation

Strided memory access

C

$0 1 2 3 4$ ... $0 \quad$ Ring reduced $\quad n-1$

Striding Toom Cook, This work

• 4 Way striding Toom-Cook approach.
• Strided memory access.
• Ring reduced [3].
• Less memory requirement as cache.
• ASIC implementation .

Binomial Sampler

Pseudo-Random from SHAKE

7 ⋮ 4 → Hamming Weight

3 ⋮ 0 → Hamming Weight

⊖

Output Buffer

8

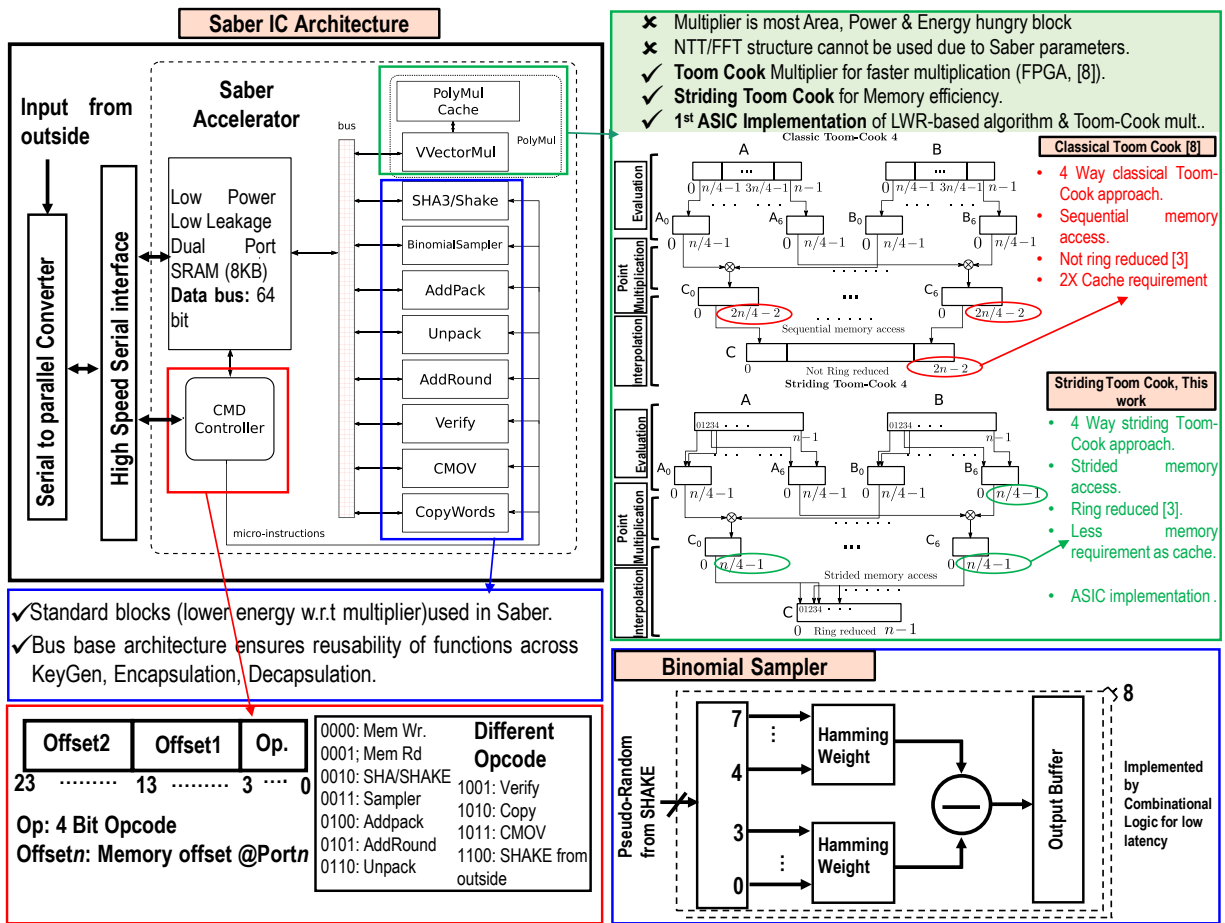Implemented by Combinational Logic for low latency

Fig. 2. Saber (NIST LWR based PQC Key Encapsulation Mechanism finalist) co-processor; New Toom-Cook multiplication architecture; OpCode format; Vector Sampler.

module is micro-programmed to perform the vector-vector polynomial multiplication in Saber using a single polynomial signed multiplier. From an algorithmic point of view, the polynomial multiplier provides the first hardware implementation of the striding version of Toom-Cook multiplication, which is a compact variant of the well-known Toom-Cook multiplication. Detailed data path for evaluation, multiplication & interpolation is presented in Fig.3. The evaluation data path is equivalent to other Toom-Cook versions [8]. In addition, we also use the lazy interpolation technique which helps in reducing latency, hence reducing energy consumption, and improving performance. 7 parallel point-multiplier with local memory (utilized by distributed SRAM in the chip micrograph) significantly improves the performance (7X latency improvement) at the expense of negligible accelerator area. This memory needs to allocate 2 polynomials of 64 coefficients X16 bits each (A & C), 1 polynomial of 64 coefficients of 8 bits each (B) with dual port reading capabilities for faster approach. The point-multiplier can instantiate several multiply-and-accumulate (MAC) units to further parallelize the point-value multiplication. Using 4 multipliers gives similar performance to [4] while providing improvement in power,

area, and energy. 2) For achieving the memory efficiency offered by striding Toom-Cook, the polynomial multiplication becomes a convolution where the expanded result is negatively wrapped. This nega-cyclic convolution can be achieved by performing the complement of two every time a coefficient that would lie out of the bounds shall be used. To do so, accounting for the fact that the length of the polynomials is 64, the sixth bit of the coefficient index can be used to perform or not the complement of two operation of every coefficient being read or written from or to memory, while only the lesser significant bits of the coefficient index are used to generate the memory address. 3) This highly area, energy and memory-efficient accelerator can be integrated with bigger SoCs/ resource constraint IoT devices readily using the high-speed serial interface. Interactive instruction set based architecture re-uses basic building blocks to reduce area overhead and makes it compatible with processors running in SoC. The 24-bit custom instruction set supports 4-bit opcodes (to support 11 operations) and two 10-bit address offsets to access 8KB data memory (dual port 1K address x 64-bit data bus). Binomial sampler is implemented using combinational logic as it does not incur high area.
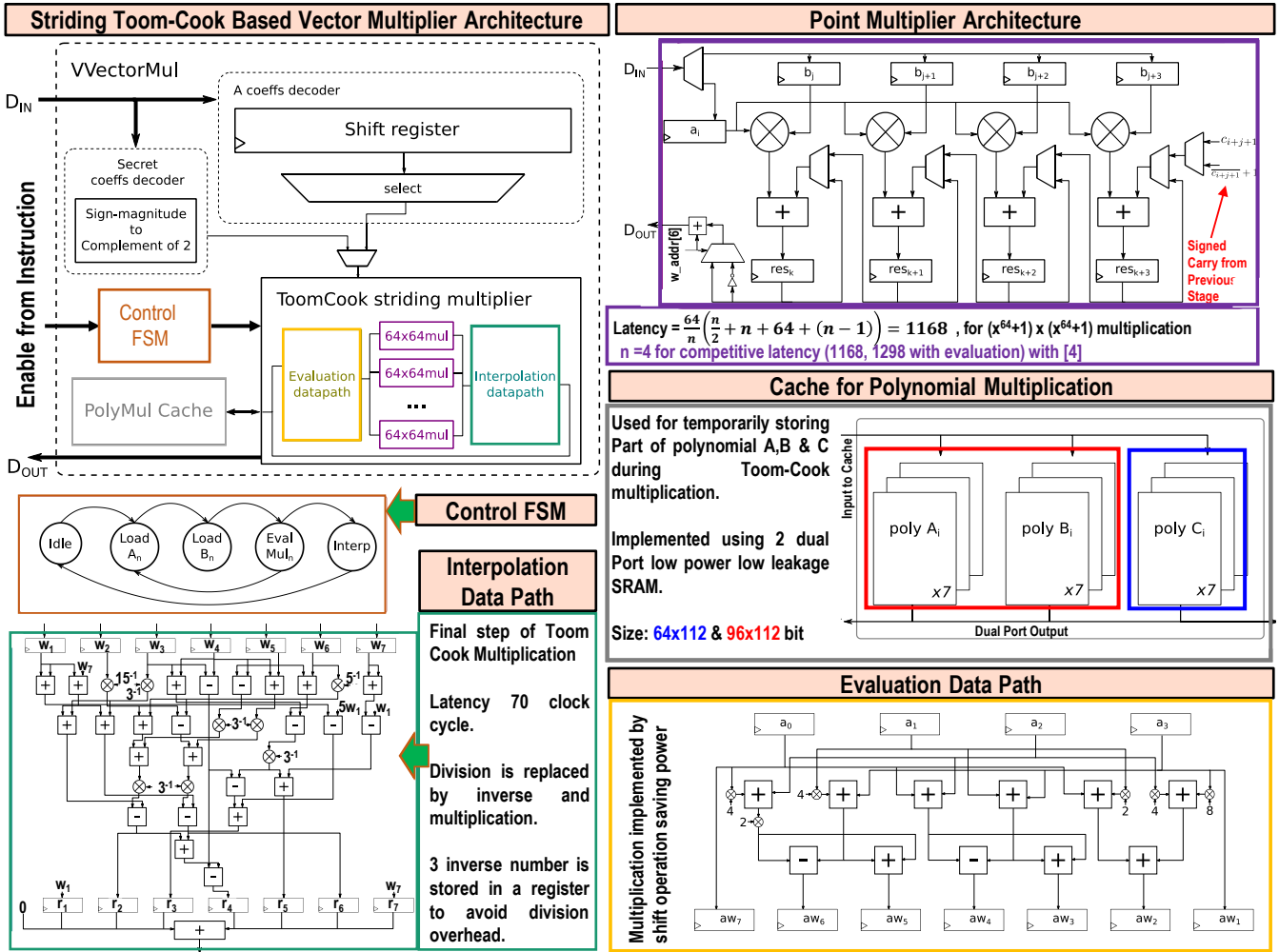
Striding Toom-Cook Based Vector Multiplier Architecture

VVectorMul

A coeffs decoder

Shift register

select

Secret coeffs decoder

Sign-magnitude to Complement of 2

Control FSM

ToomCook striding multiplier

Evaluation datapath — 64x64mul, 64x64mul, ..., 64x64mul — Interpolation datapath

PolyMul Cache

D_{IN}, D_{OUT}

Enable from Instruction

Control FSM

Idle → Load A_n → Load B_n → Eval Mul_n → Interp

Interpolation Data Path

Final step of Toom Cook Multiplication

Latency 70 clock cycle.

Division is replaced by inverse and multiplication.

3 inverse number is stored in a register to avoid division overhead.

Point Multiplier Architecture

D_{IN}, a_i, b_j, b_{j+1}, b_{j+2}, b_{j+3}

D_{OUT}, w_addr[6]

res_k, res_{k+1}, res_{k+2}, res_{k+3}

c_{i+j+1}, $\overline{c_{i+j+1}} + 1$

Signed Carry from Previous Stage

Latency $= \frac{64}{n}\left(\frac{n}{2} + n + 64 + (n-1)\right) = 1168$, for $(x^{64}+1)$ x $(x^{64}+1)$ multiplication

n =4 for competitive latency (1168, 1298 with evaluation) with [4]

Cache for Polynomial Multiplication

Used for temporarily storing Part of polynomial A,B & C during Toom-Cook multiplication.

Implemented using 2 dual Port low power low leakage SRAM.

Size: **64x112** & **96x112** bit

poly A_i (x7), poly B_i (x7), poly C_i (x7)

Input to Cache

Dual Port Output

Evaluation Data Path

Multiplication implemented by shift operation saving power

a_0, a_1, a_2, a_3

aw_7, aw_6, aw_5, aw_4, aw_3, aw_2, aw_1

Fig. 3. Vector multiplier architecture; Control FSM for multiplier; Interpolation data path; Point multiplication; Cache access; Evaluation data path.

III. MEASUREMENT RESULTS

Fig. 4 shows area breakdown and area comparison with SoA PQC accelerator. This accelerator consumes 1.75X lowest area with respect to other PQC accelerators (40nm) despite the disadvantages of process (65nm). Clock cycle requirement for Key Generation, Encapsulation and Decapsulation is 80X, 78X and 66X improvement with respect to SoA software. Striding memory access approach reduces the memory requirement to 10KB, 4X lower than other SoA PQC core [4]. All the mechanism of KEM (keygen, encapsulation and decapsulation) are observed to be finished within 89, 117 & 146 us respectively (80X improvement over fastest software implementation, no hardware implementation till date). Power and energy consumption are presented in Fig. 5. Highly pipelined architecture results in maximum frequency of 160MHz at 1.1V VDD. Full core consumes only 334uW at 10MHz and 0.7V while being fully operational. It consumes the lowest energy at 40MHz and 0.7V VDD. Key generation, encapsulation and decapsulation consume 444.1, 579.4 & 724.5 uJ energy for entire operation in the above-mentioned operating point. A comparison table with other state-of-the-art implementations

[5], [6], [7] (LWE based) is presented in Fig. 6. This is the first silicon-verified implementation of LWR PQC core. 1.58X lesser energy consumption of unit multiplication with respect to SoA NTT multiplication proves the credibility of Saber (LWR PQC) as a finalist of NIST PQC competition final round.

REFERENCES

[1] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. Saber: Module-lwr based key exchange, cpa-secure encryption and cca-secure kem. Cryptology ePrint Archive, Report 2018/230, 2018.

[2] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 719–737. Springer, 2012.

[3] Jose Maria Bermudo Mera, Angshuman Karmakar, and Ingrid Verbauwhede. Time-memory trade-off in toom-cook multiplication: an application to module-lattice based cryptography. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(2):222–244, 2020.
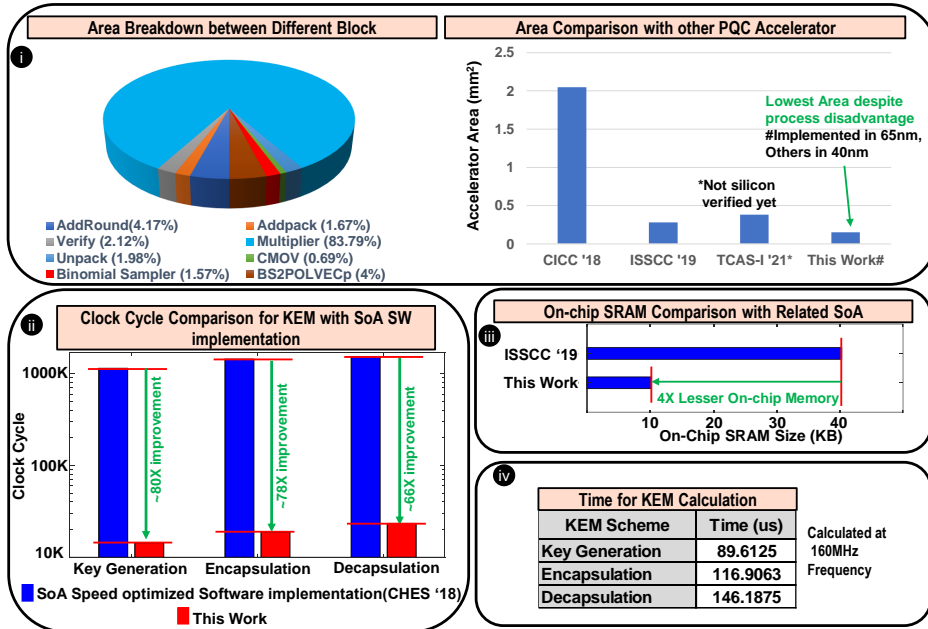
Fig. 4. i) Area breakdown & comparison with State-of-the-art PQC implementation; ii) Clock cycle comparison with fastest SoA implementation on Cortex-M4; iii) On-chip memory comparison with SoA Learning With Error (LWE) PQC algorithms; iv) Time for KEM calculation.
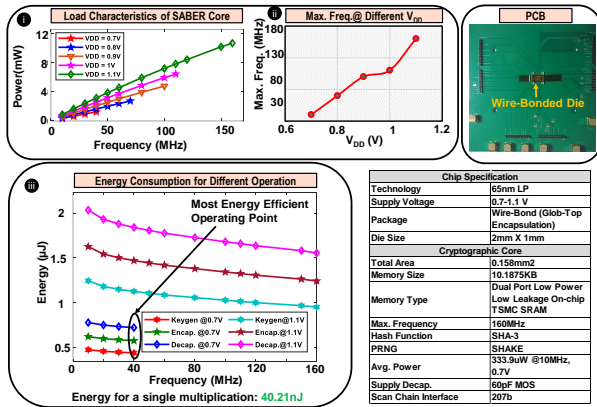

Fig. 5. i) Load Characteristics of Saber; ii) Maximum frequency; iii) Energy consumption; PCB & Chip specification.

Fig. 7. Chip micrograph

	Cortex-M4[7]	CICC'18[5]	ISSCC'19[4]	TCAS-Iª[6]	This work
Technology	-	40nm	40nm	65nm	65nm
Supply Voltage	3-5	0.9	0.68-1.1	1.1	0.7-1.1
Frequency (MHz)	100	300	12-72	400	40-160
Total Processor Area (mm²)	-	2.05	0.28	0.38	0.158
Supported Lattice Crypto Primitives	All	Ring-LWE	Ring-LWE & Module-LWE	Module-LWR	Module-LWR
Supported Lattice Parameters	All	N: 64-2048 q: 32-bit conf.	N: 64-2048 q: 24-bit conf.	N: 256 q: 13-bit	N: 256 q: 13-bit
Average Power	-	140mW	519uW	35-41mW	333.9uW
Individual Multiplier Performance					
Multiplier Cycle	65459	160	1288	81+pipeline	1298*
Energy (nJ)	40.1 x 10³	31	63.4	-	40.21

ª Not silicon verified. Results reported in simulation. *Interpolation needs 70 clock cycle, which happens once in 3 multiplication, Evaluation clock cycle added.

Fig. 6. Comparison of 1st Saber ASIC with Cortex-M4 software and other hardware post-quantum cryptography accelerators.
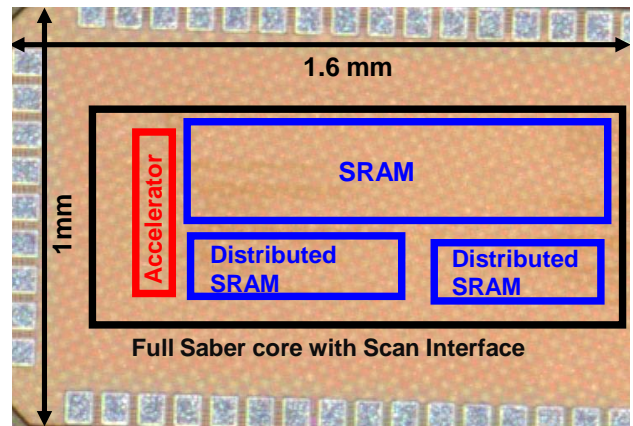
[4] Utsav Banerjee, Abhishek Pathak, and Anantha P Chandrakasan. 2.3 an energy-efficient configurable lattice cryptography processor for the quantum-secure internet of things. In *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*, pages 46–48. IEEE, 2019.

[5] Shiming Song, Wei Tang, Thomas Chen, and Zhengya Zhang. Leia: A 2.05 mm 2 140mw lattice encryption instruction accelerator in 40nm cmos. In *2018 IEEE Custom Integrated Circuits Conference (CICC)*, pages 1–4. IEEE, 2018.

[6] Yihong Zhu, Min Zhu, Bohan Yang, Wenping Zhu, Chenchen Deng, Chen Chen, Shaojun Wei, and Leibo Liu. Lwrpro: An energy-efficient configurable crypto-processor for module-lwr. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 68(3):1146–1159, 2021.

[7] Angshuman Karmakar, Jose Maria Bermudo Mera, Sujoy Sinha Roy, and Ingrid Verbauwhede. Saber on arm cca-secure module lattice-based key encapsulation on arm. Cryptology ePrint Archive, Report 2018/682, 2018.

[8] Jose Maria Bermudo Mera, Furkan Turan, Angshuman Karmakar, Sujoy Sinha Roy, and Ingrid Verbauwhede. Compact domain-specific co-processor for accelerating module lattice-based kem. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2020.