LEIA: A 2.05mm^2 140mW Lattice Encryption Instruction Accelerator in 40nm CMOS

Shiming Song, Wei Tang, Thomas Chen, and Zhengya Zhang
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, MI 48105

***Abstract*—The recent rapid steps towards commercialization of quantum computing services have been calling for the deployment of post-quantum (PQ) security schemes at a large scale. This work presents a high-performance, programmable lattice encryption instruction accelerator named LEIA that supports all the recent popular ring learning with errors (ring-LWE) schemes, with parameter N from 64 to 2048, and q from 2 to $2^{32}-1$. LEIA is prototyped in a 2.05mm^2 40nm CMOS test chip. It achieves significant speedup and energy efficiency in core operations required by ring-LWE: above 1000× acceleration and 2000× better energy efficiency in number theoretic transform (NTT), and above 10× acceleration and 50× better energy efficiency in high-σ, high-precision discrete Gaussian (DG) generation over state-of-the-art designs using the most challenging published parameter settings. As the first silicon-proven lattice encryption accelerator, LEIA is the fastest, most energy-efficient, and most configurable solution to date.**

***Index Terms*—Crypto accelerator, post-quantum cryptography, number theoretic transform, discrete Gaussian generation**

I. Introduction

Almost all public key cryptosystems today, including Rivest-Shamir-Adleman (RSA) and elliptic curve cryptography (ECC), rely on the hardness of integer factorization and discrete logarithm, both of which are known to be efficiently solved by Shors algorithm performed on a quantum computer [1]. With quantum computers being closer to the public access than ever, a leap into post-quantum (PQ) security is not only timely but also necessary, as depicted in Fig. 1. The emerging lattice-based cryptography is widely regarded a quantum-resilient alternative to classical public key cryptosystems [2]. Ring learning with errors (ring-LWE) is the most-recognized

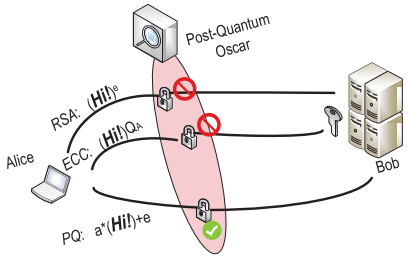

Fig. 1: Threat of quantum attacks calls for post-quantum cryptography.

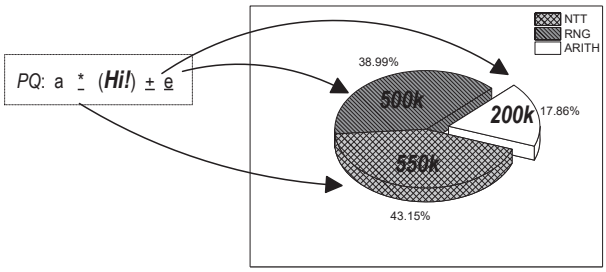

Fig. 2: Workload breakdown of NewHope [4] on Cortex-M4.

lattice-based cryptographic scheme and it holds promise towards practical fully homomorphic encryption [3].

Ring-LWE based cryptosystems are built using polynomial operations on cyclotomic rings of order N on R_q, where q is a prime number. To secure a message, cyclotomic ring polynomial multiplication is used to rotate the message, and discrete Gaussian (DG) samples are added to mask the message, as illustrated in Fig. 2. When N is a power of 2, cyclotomic ring polynomial multiplication can be done in $\mathcal{O}(N \log N)$ with number theoretic transform (NTT). A recent implementation of a ring-LWE based cryptosystem reveals that NTT and DG generation account for 43% and 39% of the workload, respectively [4]. To ensure high security, $N = 512$ and DG distribution with $\sigma = 215$ have been adopted in the latest designs [3], [5].

NTT follows a FFT-like bufferfly compute graph, and a large NTT presents a computational bottelenck. A state-of-the-art $N = 512$ NTT (512-point NTT) requires 1ms and 220μJ per NTT operation [5]. Mapping a large NTT to hardware is further complicated by complex routing. High-σ DG generation presents another bottleneck, because DG generation is serial and a high-σ distribution requires a high numerical precision. A state-of-the-art $\sigma = 215$ DG generator takes 3ms and 140μJ to generate one DG sample [3]. A high-σ, high-precision DG generation also requires a large memory that scales exponentially with precision, e.g., a 256-bit DG generator requires almost 1Mb of memory. The performance and efficiency NTT and DG generation currently do not meet the requirements of practical applications.

We present LEIA, the first ring-LWE instruction accelerator

978-1-5386-2483-8/18/$31.00 ©2018 IEEE

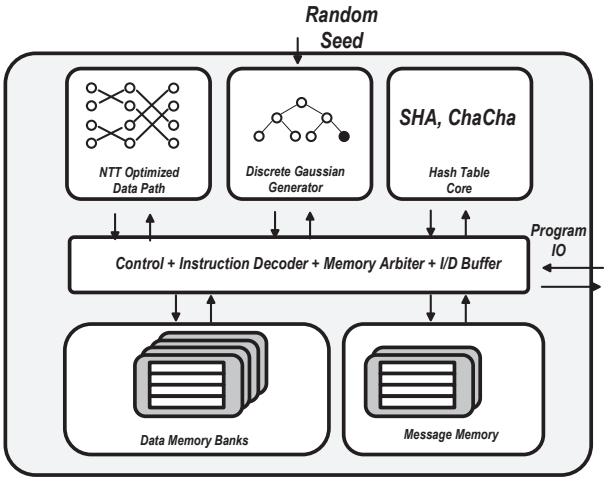

Fig. 3: Top-level architecture of LEIA.

to support a set of common ring-LWE based security applications including digital signature and key exchange. A 2.05mm^2 40nm CMOS LEIA chip achieves 1.6μs and 96nJ per NTT operation, and 25μs and 2.6μJ per DG sample generation, demonstrating improvements over the state-of-the-art [3], [5] by order(s) of magnitude.

II. LEIA Architecture and Chip Implementation

LEIA contains two core blocks, an NTT-optimized datapath (or NTT core) and a DG generator, as shown in Fig. 3. To support a large N using a compact area, the NTT core is designed in a 3-stage architecture to achieve the optimal balance between latency and size. To be future-proof, the NTT core accommodates N up to 2048 and q up to $2^{32}-1$. To improve performance while minimizing memory, the DG generator is parallelized by utilizing a two-layer memory hierarchy with an efficient compression. The DG generator supports high σ that uses up to 256 bits of precision. An even higher $10\times$ σ extension is obtainable by iterating between NTT and DG following Micciancio's extension scheme [6] to enable up to 2Kb security applications. The 32-bit arithmetic and memory blocks on LEIA support all published parameters.

LEIA also contains a hash-table core, a central controller with instruction decoder and memory arbiter, data memory banks for the NTT core and DG generator, and a message memory to store input and output messages for processing. LEIA is programmed by instructions to implement common security applications.

A. Three-Stage Configurable NTT Core

The NTT computation resembles FFT, and it is done using stages of integer butterfly operations. We choose decimation-in-time for NTT and decimation-in-frequency for inverse NTT (INTT) both of which reuse the same hardware. The NTT architecture is constructed in hierarchical stages (Fig. 4(a)) to support N up to 2048 while minimizing routing complexity.

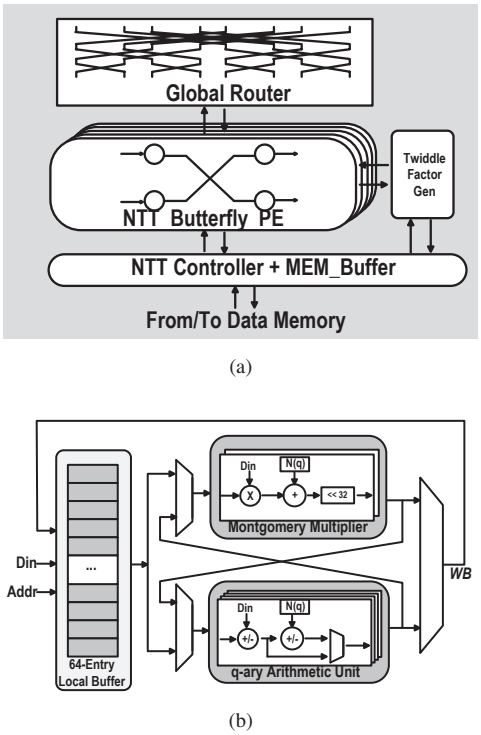

Fig. 4: (a) NTT-optimized datapath, (b) NTT PE.

An NTT PE (Fig. 4(b)) performs two butterfly operations using two Montgomery multipliers, four q-ary integer arithmetic units for modulo-q adds or subtracts, and a 4-port scratch pad to provide efficient butterfly connections. The size of the scratch pad determines the size of NTT that a PE can compute locally. To save area, we limit the scratch pad to 64 entries to support up to 64-point NTT locally.

Beyond 64-point NTT, we use multiple NTT PEs and routers to exchange data between PEs (Fig. 4(a)). However, larger NTTs result in larger routers and inenfficient global wiring. Therefore, we limit routing to span only a group of 8 NTT PEs to support up to 256-point NTT. Above 256-point inputs (power-of-2) are divided into vectors of 256, and implemented in vector processing by the 8 PEs.

To sum up, the NTT computation is divided to three stages to support efficient reconfiguration: short NTT up to 64 points using only local processing, intermediate NTT up to 256 points using routers, and large NTT up to 2048 points using vector processing. The three-stage architecture balances local storage and global routing, as well as data routing and processing delay, resulting in nearly optimal tradeoff between latency and size as shown in Fig. 6. The NTT architecture produces 32 points/cycle. Further doubling the PE count improves latency of a 256-point NTT by only 37%, showing fast diminishing returns. The NTT-optimized datapath comes with an instruction set listed in Table I to support programming.

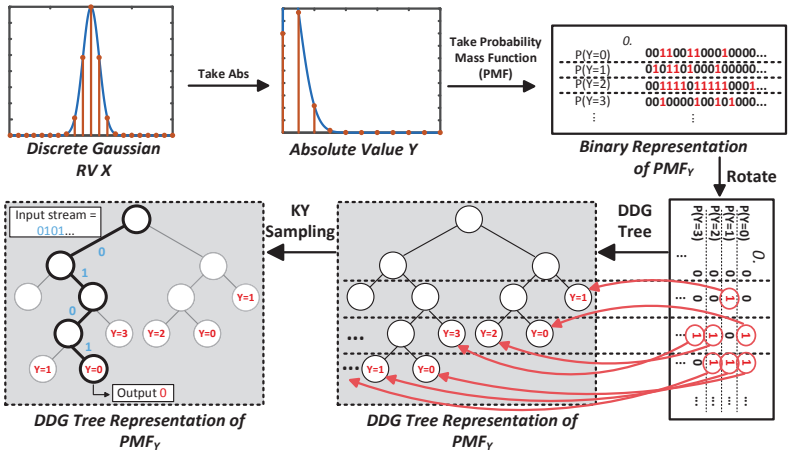

Fig. 5: Construction of DDG tree and operation of KY Sampling.

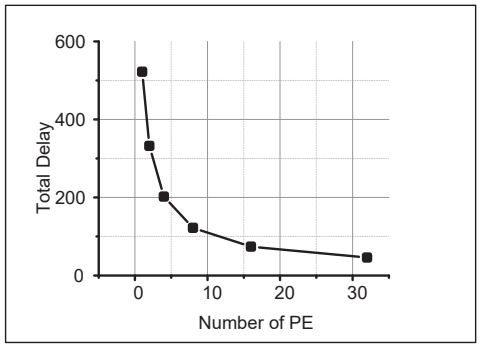

Fig. 6: PE count and latency of computing a 256-point NTT.

TABLE I: Instruction Set for the NTT-Optimized Datapath

Instruction	Cycle Count	Function
REG_RD	16	Load data from meory
REG_ST	16	Write data to memory
ADD/SUB	20	Modulo vector add/subtract
NTT	160	256-point NTT
INTT	200	256-point INTT
MULT	20	Modulo vector multiply
DGREAD	20	Load DDG

B. Parallel DG Generation Using Compact Memory

A DG generator produces DG samples based on a binary random source. LEIA's DG generator is designed based on Knuth-Yao (KY) Sampling that is known to be the fastest method for high-σ random sample generation [7]. Given a DG distribution, its numerical probability mass function (PMF) can be represented in a discrete distribution generation (DDG) tree, as shown in Fig. 5. KY Sampling traverses the DDG tree from the root, taking one binary random bit as input each time to decide either left or right branch to take next. It outputs a sample of the DG distribution when it reaches a non-empty node (NEN). The DG generator also supports binomial noise generation used in NewHope [4].

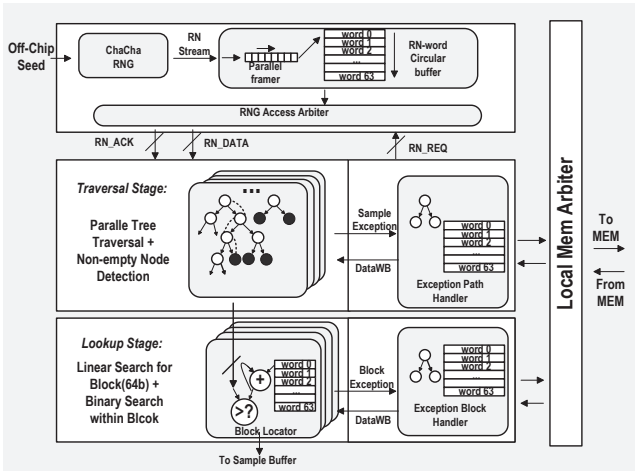

Fig. 7: Top-level architecture of parallel KY sampler.

A naive implementation of the DG generator is slow and costly. Tree traversal is serial in nature, and storing the DDG tree for a 256-bit precision distribution requires almost 1Mb memory. If DG generation is parallelized, the DDG tree memory needs to be duplicated or additional ports added, both of which incur prohibitive costs in area and power.

We design a fast and light-weight KY Sampling architecture based on a key insight: each level down the DDG tree represents a higher precision of the DG distribution, and the probability of having to traverse deeper into the tree decreases approximately exponentially. This insight leads to a parallel KY Sampling architecture that consist of four sampling modules and a two-level cache hierarchy, as shown in Fig. 7. A sampling module operates in two steps: 1) tree traversal step that takes binary random bits as inputs and eventually detects an NEN in the DDG tree; and 2) sample look-up step that loads the row in L1 cache that corresponds to the stage of the NEN, and binary searches the position of the NEN in the row. The

TABLE II: Comparison with State-of-the-Art

q	Distribution	Reference	N	Platform	Frequency (MHz)	NTT cycles	INTT cycles	DG cycles	NTT Power (mW)	DG Power (mW)	NTT Energy (uJ)	DG Energy (uJ)
12289	DG[a], σ= 215	[5]	512	CortexM4	180	508624	508624	935925	78	78	220.4	405.6
		[3]	512	ATxmega128	32	516971	468090	105153	45	45	727.0	147.9
			256			194145	174023	53023	45	45	273.0	73.2
		This Work	**512**	**40nm CMOS**	**300**	**492**	**572**	**7603.2**[b]	**58.9**	**101.2**	**0.096**	**2.6**
			256			**160**	**200**	**3801.6**[b]	**58.9**	**101.2**	**0.031**	**1.3**
	Binomial, Φ_{16}	[4]	512	Cortex-M4	168	87,223	97,789	54,332	72	72	37.4	23.3
		This Work		**40nm CMOS**	**300**	**492**	**572**	**3703.8**	**58.9**	**101.2**	**0.096**	**1.25**
7681	DG[a], $\sigma = \frac{11.32}{\sqrt{2\pi}}$	[2], [7]	256	Virtex-6	313	667	1048	805	-	-	-	-
				ASIC synthesis	500							
		This Work		**40nm CMOS**	**300**	**160**	**200**	**262**	**58.9**	**101.2**	**0.031**	**0.088**

[a]Discrete Gaussian distribution
[b]Average performance over 100,000 samples

TABLE III: LEIA Chip Summary

Technology	40nm GP
Core Area	2.05mm^2
Frequency	300MHz
Supply Voltage	0.9V
Power[a]	140mW (1 DG module enabled)
	216.5mW (4 DG modules enabled)
NTT Throughput	2.2M OP/s[b]
DG Throughput	2 samples/s (average)
Data Memory	195kB
DDG Memory	65kB
Message Memory	80kB
q	2 to 2^{32}-1
N	64 to 2048
σ	1 to 1500
RNG Scheme	KY Sampling, ChaCha20
Hash Function	SHA-256, SHA-512

[a]Benchmarked based on NewHope [4]
[b]1 OP/s = 1 NTT operation per second

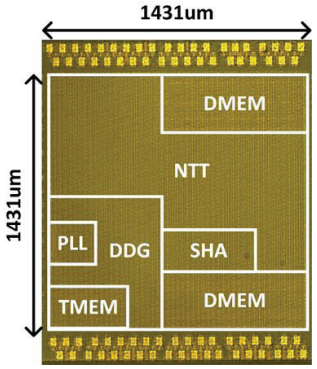

Fig. 8: Microphotograph of LEIA test chip.

position of the NEN is the DG sample. Each module employs a compact, fully search-able L1 cache that stores only the first 6 levels of the DDG tree to accelerate local tree traversal and look-up. Within 1% cache miss rate, an exception handler accesses the L2 cache. An arbiter serializes the highly unlikely case of L2 cache access contention by multiple modules.

The numerical PMFs of the DG distribution are stored in caches as bit-planes. The high-σ, 256-bit PMFs are 95% sparse (i.e., zeroes) in the binary representation. We create a compression scheme by dividing a bit-plane into 16-bit vertical slices, and removing all-zero slices. The simple compression scheme saves 93% memory. For distributions of larger σ, the KY Sampling is designed to skip the top levels of the DDG tree to maintain an average 2 cycles per DG sample generation.

III. Measurement Results and Comparison

LEIA was protoyped in a 2.05mm^2 40nm CMOS test chip. The measured performance of the LEIA chip at 0.9V in room temperature is compared with recently published ring-LWE implementations on different hardware platforms in Table II. LEIA is compatible with all the referenced schemes. LEIA's NTT accelerator demonstrates above 1000$\times$ speedup over the state-of-the-art NTT design [5], and its energy efficiency is 2000$\times$ better using the most challenging published parameter setting ($q = 12289$, $N = 256$). LEIA's DG generator achieves more than 10$\times$ improvement over the state-of-the-art DG design [3], and its energy efficiency is 50$\times$ better using the most challenging published parameter setting (DG of $\sigma = 215$). Note that most of the previous ring-LWE implementations were designed for narrow parameter ranges, while our design is compatible with all of them to support the widest range of applications. Table III shows the summary of the LEIA chip. The chip microphotograph is shown in Fig. 8.

References

[1] R. de Clercq, et al., in *DATE*, 2015.
[2] I. Verbauwhede, et al., in *ISSCC*, 2015.
[3] Liu, et al., *ACM Trans. Embed. Comput. Syst.*, September 2017.
[4] E. Alkim, et al., in *SPACE*, 2016.
[5] T. Oder, et al., in *DAC*, 2014.
[6] Micciancio, et al., in *CRYPTO*, 2017.
[7] Roy, et al., in *CHES*, 2014.