NTT 연산을 위한 병렬 트위들 팩터 생성기 구조

엄유진, 양희훈, 유호영*
충남대학교 전자공학과
e-mail : yjeom.cas@gmail.com, hhyang.cas@gmail.com, hyyoo@cnu.ac.kr

Parallel Twiddle Factor Generator for NTT

Yujiin Eom, Heehun Yang, and Hoyoung Yoo*
Department of Electronics Engineering
Chungnam National University

Abstract

This paper proposes a parallel TF (Twiddle Factor) generator that produces p TFs simultaneously to enhance the efficiency of NTT (Number Theoretic Transform) hardware. With the rise of cloud computing, the importance of Fully Homomorphic Encryption (FHE) has grown, as it allows unlimited operations on encrypted data, ensuring secure and reliable data processing. Implementing NTT is crucial for polynomial multiplication in FHE hardware, but the increasing number of TFs with more NTT inputs extends initialization time. To address this, we introduce a parallel TF generator. Evaluations on a Kintex Ultrascale KU040 board using Xilinx Vivado 2021.2 show that our TF generator can be optimized for hardware resources, comparing ROM-based TF storage methods.

I. Introduction

최근 클라우드 컴퓨팅의 발전으로 데이터를 보호하기 위해 안전한 정보처리 시스템의 중요성이 강조되고 있다. 이에 따라 대량의 정보를 안전하게 저장하고 처리하기 위한 효과적인 암호화 알고리즘이 필수적으로 요구된다. 다양한 암호화 기술 중 FHE(Fully Homomorpgic Encryption)는 원본 데이터의 노출 없이 안전하게 데이터를 처리할 수 있게 하는 기술로써 주목받고 있다. FHE는 암호화된 데이터 상태에서 무한한 연산을 가능하게 함으로써 데이터 처리의 신뢰성을 높이는 데 기여한다[1]. 이러한 특성은 데이터 변조나 무단 접근의 위험을 최소화하여 데이터 처리 시스템의 보안성을 강화할 수 있다. 그러나 FHE는 큰 크기의 다항식을 이용한 제한 없는 연산을 구현해야 한다는 병목 현상을 가지고 있다. 이 병목 현상은 NTT(Number Theoretic Transform)을 구현함으로써 해결할 수 있다[2]. NTT는 주어진 다항식의 계수를 주파수 영역으로 변환하여 다항식의 곱셈을 빠르게 수행할 수 있게 한다.

입력 다항식을 주파수 영역으로 변환할 때에는 각 입력에 Twiddle Factor(TF)를 곱하여 변환을 수행한다. 입력의 수만큼 TF를 필요로 하기 때문에 하드웨어 상에서 큰 차수의 NTT 연산 시 TF의 저장공간이 커져 하드웨어 리소스를 많이 소모하는 문제가 발생하게 된다. 또한, 연산에 필요한 각 TF를 생성하고 저장하는 시간은 입력의 개수만큼 기하급수적으로 증가하게 되며 이에 따라 FHE 하드웨어의 초기화 시간이 길어지는 문제를 야기한다. 이러한 문제를 해결하기 위해서 TF를 NTT 연산에 맞춰 on-the-fly로 생성하거나[3], 회로 내부에서

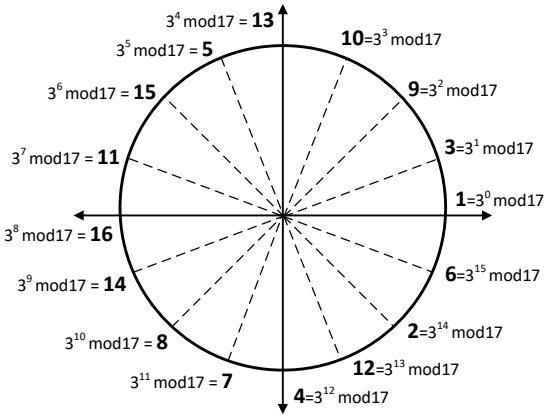

그림 1. N=16일 때 NTT에 대한 원시근

TF를 생성[4]하는 등 다양한 방식으로 연구되고 있다. 본 논문은 효율적인 FHE 하드웨어 시스템을 구축하는데 기여할 수 있는 TF 생성기인 병렬 TF 생성기 구조를 제안한다. 제안하는 구조는 NTT 내부에서 TF를 동시에 생성하여 FHE 하드웨어의 초기화 시간 문제를 개선할 수 있다.

II. Background

다항식 곱셈은 현대 암호학에서 기본적인 연산으로, 많은 암호 시스템에서 사용된다. 그러나 이는 $O(N^2)$의 시간 복잡도를 가져 큰 다항식에 대해서는 비효율적이다. 이를 해결하기 위해 사용할 수 있는 것이 NTT이다. NTT는 다항식 곱셈의 시간 복잡도를 $O(N \log N)$으로 줄이는 것이 가능하다. FFT(Fast Fourier Transform)와 NTT 모두 시간영역의 데이터를 주파수 영역으로 변환하여 연산한다는 점은 같다. 하지만 FFT는 복소 평면에서 연산을 수행하는 반면, NTT는 단위 원에 기반한 유한 링에서 연산을 수행한다. NTT의 동작은 입력과 N번째 원시 단위근인 ω와의 곱셈으로 이루어진다. ω는 식 1을 만족하며, TF라고 부른다.

$$\omega^N \equiv 1 \bmod q \quad (1)$$

식 1에서 q는 ω와 서로소인 자연수이다. 그림 1은 TF가 식 1을 만족하는 단위원에 기반한 원시근임을 보인다. TF를 사용한 NTT 변환은 식 2와 같이 표현된다.

$$A_i = \sum_{j=0}^{N-1} a_j \omega^{ij} \quad mod\ q \quad (2)$$

식 2의 A_i는 변환된 데이터의 i번째 출력 값으로, 주파수 영역에서의 다항식 계수를 나타낸다. a_j는 입력 데이터의 j번째 값으로, 시간 영역에서의 다항식 계수를 나타낸다. N은 변환할 데이터의 총 개수로, 다항식의 차수와 동일하다. 식 2에서 NTT의 입력 개수만큼 TF를 필요로 하게 되므로, 효율적인 TF 생성 방법이

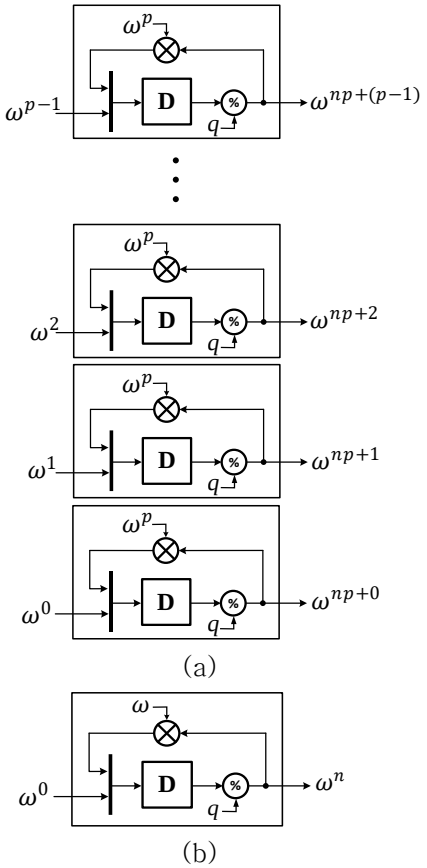

그림2. (a) 제안하는 병렬 트위들 팩터 생성기
(b) 단일 트위들 팩터 생성기

필수적이다. NTT에서 TF를 사용하는 방법 중 하나인 ROM 기반 방법은 NTT 처리 크기가 커질수록 더 많은 ROM 공간이 필요하여 전력 소모와 칩 면적이 커지는 단점이 있다. 예를 들어 그림 1과 같은 경우, 17로 모듈러하여 17 미만의 데이터를 저장하는 16개의 TF를 저장해야 한다. 데이터의 최대 크기는 5bit이므로 16*5bit 크기의 메모리를 필요로 하게 된다. NTT 입력 개수가 2^{10}, 2^{11} 또는 2^{12}과 같이 큰 경우, 필요한 TF 개수도 동일하게 증가하게 되며 메모리 크기와 전력 소모가 기하급수적으로 커진다. 그러므로 NTT의 효율성을 극대화하기 위한 TF를 생성하는 방법이 필요하며, 본 논문에서 제안하는 TF 생성기는 이를 효과적으로 구현 가능하다.

Ⅲ. Proposed Design

본 논문은 효율적인 TF 생성을 위해 그림 2의 (a)와 같은 병렬 TF 생성기 구조를 제안한다. 해당 구조는 p개의 병렬 TF 생성기로 구성되며, 각 생성기는 서로 다른 TF를 계산한다. 생성기 내부에서는 지연 소자 D Flip-Flop과 모듈러 곱셈

표 1. ROM 구조의 합성 결과

		ROM		
		$N=2^{13}$	$N=2^{14}$	$N=2^{15}$
Area	LUT	18K	36K	72K
	FF	–	–	–
	DSP	–	–	–
Latency		–		
Throughput		1		

연산을 포함하는 피드백 루프를 사용한다. ω의 n번째 각 생성기의 초기 값은 ω^0, ω^1, ω^2, ..., ω^{p-1}로 설정된다. 이후 각 사이클에서 출력은 p번째 TF인 ω^p와 곱해져 q로 모듈러 연산을 거친다. 단위근의 성질에 따라 각 생성기는 현재 번째에 p를 더한 위치의 TF 값을 생성하게 된다. 이에 따라 p개의 TF를 동시에 생성 가능하다.

그림 2의 (b)는 p가 1일 때에 해당하는 단일 TF 생성기의 하드웨어 구조를 보여준다. 초기 입력은 ω^0을 나타내는 1로 설정된다. 이후 각 사이클에서 ω가 곱해지고 q로 모듈러 연산을 거친 후 출력된다. 해당 과정은 반복되며 ω, ω^2, ω^3, ..., ω^{N-1}의 순서로 출력된다.

본 논문에서 제안하는 병렬 TF 생성기는 확장 가능하며, 동시에 생성할 수 있는 TF개수인 p의 수를 조정 가능하다. 이에 따라 NTT 하드웨어에서의 리소스 활용 최적화에 기여할 수 있다.

Ⅳ. Experimental Results

본 논문에서는 제안하는 병렬 TF 생성기의 성능을 평가하기 위해 Xilinx Vivado 2021.2 환경에서 실험을 진행하였다. 실험 대상 보드는 Kintex Ultrascale KU040 평가보드이다. 표 1, 2는 ROM 방식과 제안된 병렬 TF 생성기의 합성 결과를 보여주며, 입력 데이터 수 N의 크기에 따른 ROM의 면적과 병렬 개수 p에 따른 하드웨어 면적을 비교한다. 또한, 실험 결과로 TF 생성기의 지연 시간과 출력량을 제시한다.

표 1에 따르면, ROM 방식은 입력 N이 증가함에 따라 저장해야 하는 TF의 개수가 증가하여 ROM의 크기도 함께 증가한다. 반면, 표 2는 제안된 병렬 TF 생성기는 입력 데이터 수 N과 무관하게 병렬 개수 p가 증가함에 따라 면적이 증가함을 보인다. p를 1, 2, 4로 2배씩 증가시킬 경우 사용되는 LUT도 2배로 증가하며, Flip-Flop과 DSP 소자도 증가하는 경향을 보인다. ROM의 출력은 1 clock-cycle당 하나의 TF를 출력하는 반면, 제안하는 구조는 p가 증가할수록 전체 TF를 생성하는 시간은 감소하고, 생성된 TF의 출력량은 병렬 개수 p에 비례하여 증가한다. 결론적으로, 제안하는 구조는 병렬 구현

표 2. 제안하는 병렬 TF 구조의 합성 결과

		p=1	p=2	p=4	p
Area	LUT	13K	26K	52K	p*13K
	FF	17	81	209	p^2*17
	DSP	34	68	136	p*34
Latency		$2^{\log N}$	$2^{\log N-1}$	$2^{\log N-2}$	$2^{\log N-\log p}$
Throughput		1	2	4	p

개수 p에 따라 하드웨어 면적이 증가하며 TF 생성 속도와 출력 속도도 증가한다. 이는 하드웨어 리소스에 따라 회로의 면적과 TF 생성량 간의 trade-off 관계에서 최적화된 p값을 선택할 수 있음을 보인다.

Ⅴ. Conclusion

본 논문에서는 NTT에서 사용하는 TF를 효율적으로 생성하기 위한 병렬 TF 생성기 구조를 제안한다. NTT의 입력 크기가 증가함에 따라 필요한 TF의 개수도 늘어나며, 기존의 ROM 방식을 사용할 경우 증가하는 TF를 저장하는 데 많은 하드웨어 리소스가 필요하다. 이에 대한 해결책으로, TF를 계산하여 생성하는 병렬 TF 생성기를 제안한다. 제안하는 TF 생성기는 병렬 구조를 채택하여 여러 개의 TF 값을 동시에 생성할 수 있다. 기존의 ROM 방식과 제안된 TF 생성기의 합성을 진행하고 성능 평가를 실시하였다. 제안된 병렬 TF 생성기는 병렬 개수 p에 따라 면적, 지연시간, 출력량이 변하며, 면적과 출력의 trade-off 관계를 최적화함으로써 NTT 하드웨어에 리소스에 맞춰 효율적인 면적으로 TF 생성기를 구현할 수 있다. 또한, 제안된 구조는 새로운 TF 값을 즉각적으로 생성할 수 있으며, 다양한 NTT 입력 값에 대해 유연하게 사용 가능하다. 따라서, 이는 효율적인 NTT 하드웨어를 구현하는 데 유용하다.

Acknowledgments

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2022R1A58026986), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (2022-0-01170), National R&D Program through the NRF funded by Ministry of Science and ICT(2020M3H2A1078119).

참고문헌

[1] Intel Corporations, Intel StrongArm SA-1110 Microprocessor Develper's Manual, June 2000. M. Rella, S, Ardianto, N. Phap, and L. Hanho, "A Bootstrapping-Capable Configurable NTT Architecture for Fully Homomorphic Encryption," IEEE Access, April 2024.

[2] Phap Duong-Ngoc, Sunmin Kwon, Donghoon Yoo, and Hanho Lee, "Area-Efficient Number Theoretic Transform Architecture for Homomorphic Encryption," IEEE Transactions on Circuits and Systems I: Regular Papers, December 2022.

[3] Ghada Alsuhli1 , Hani Saleh,, Mahmoud Al-Qutayri, Baker Mohammad,, and Thanos Stouraitis, "Efficient Twiddle Factor Generation for Post Quantum Cryptography FALCON-based Number Theoretic Transform," TechRxiv, April 2024.

[4] Chulwoo Lee, Hanyoung Lee, Phap Duong-Ngoc, Hanho Lee, "Twiddle Factor Generator Architecture for Number Theoretic Transform," 2023 20th International SoC Design Conference (ISOCC), October 2023.