



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2022년10월06일
(11) 등록번호 10-2451633
(24) 등록일자 2022년09월30일

(51) 국제특허분류(Int. Cl.)
H04L 9/00 (2022.01) G06F 17/10 (2006.01)
H04L 9/06 (2006.01)
(52) CPC특허분류
H04L 9/008 (2013.01)
G06F 17/10 (2013.01)
(21) 출원번호 10-2021-0175571
(22) 출원일자 2021년12월09일
심사청구일자 2021년12월09일
(56) 선행기술조사문헌
KR101952547 B1*
(뒷면에 계속)

(73) 특허권자
인하대학교 산학협력단
인천광역시 미추홀구 인하로 100(용현동, 인하대학교)
(72) 발명자
이한호
인천광역시 연수구 컨벤시아대로252번길 30, 150
5동 1701호(송도동, 송도 더샵 퍼스트파크 F15BL)
김지수
인천광역시 미추홀구 주안로 117-16, 101동 1304
호(주안동, 주안행복주택)
(74) 대리인
양성보

전체 청구항 수 : 총 7 항

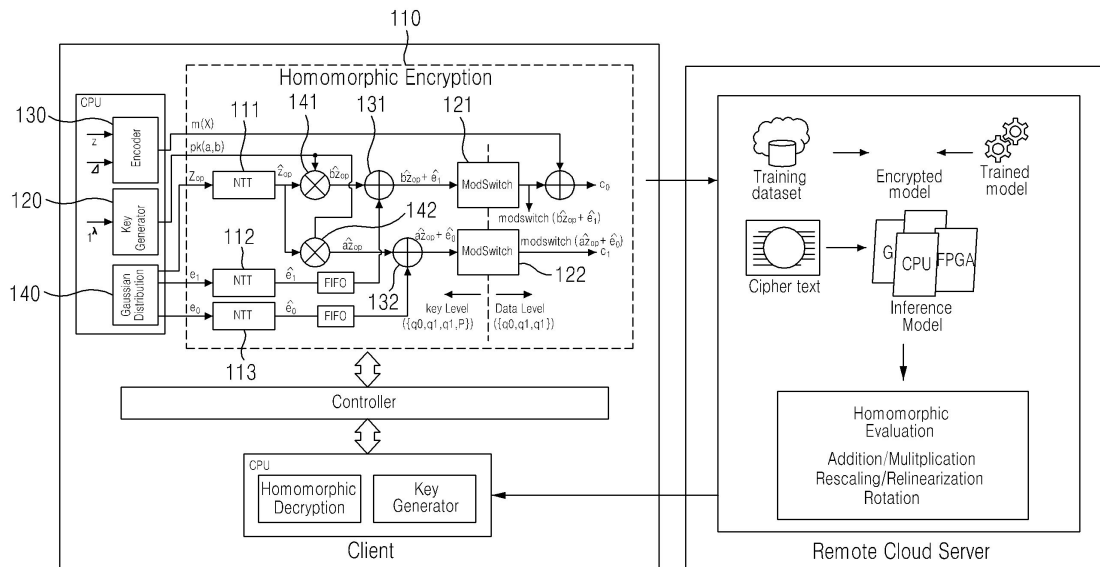
심사관 : 양종필

(54) 발명의 명칭 동형암호를 위한 암호화 처리 장치 및 방법

(57) 요약

동형암호를 위한 암호화 처리 장치 및 방법이 제시된다. 본 발명에서 제안하는 동형암호 장치는 암호 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 키 생성기(Key Generator), 다항식을 수신하여 암호화에 적합한 평문으로 부호화하는 부호화기(Encoder), 암호화에 필요한 에러를 생성하는 가우시안 분포 에러 생성기(Gaussian) (뒷면에 계속)

대표도



Distribution) 및 NTT 코어, 모듈러 곱셈기(Modular Multiplier), 모듈러 스위치(Modular Switch), 모듈러 덧셈기(Modular Addition)를 포함하고, 공개키 및 평문을 수신하여 암호화하는 동형 암호화(Homomorphic Encryption) 프로세서를 포함하고, 상기 동형 암호화 프로세서는 연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 축소하고, 암호화의 고속화를 위한 NTT 연산을 수행하는 NTT 코어, 상기 NTT 코어를 통해 연산된 결과들에 대하여 공개키와의 곱셈 연산을 수행하는 모듈러 곱셈기, 키(Key) 크기의 데이터를 평문 크기의 데이터로 축소하는 모듈러 스위치 및 상기 모듈러 스위치의 결과와 상기 부호화기를 통해 생성된 평문을 더하여 암호문을 생성하는 모듈러 덧셈기를 포함한다.

(52) CPC특허분류

H04L 9/0618 (2013.01)

H04L 2209/125 (2013.01)

(56) 선행기술조사문헌

KR1020180013064 A*

KR1020190138815 A*

KR1020210128303 A*

KR1020210146513 A*

*는 심사관에 의하여 인용된 문헌

이 발명을 지원한 국가연구개발사업

과제고유번호 1711139247

과제번호 IITP-2021-0-02052

부처명 과학기술정보통신부

과제관리(전문)기관명 정보통신기획평가원

연구사업명 대학ICT연구센터육성지원사업

연구과제명 스마트 모빌리티를 위한 인공지능 시스템반도체 핵심 기술 개발 및 인력 양성

기 여 율 1/2

과제수행기관명 인하대학교 산학협력단

연구기간 2021.07.01 ~ 2028.12.31

이 발명을 지원한 국가연구개발사업

과제고유번호 1415172944

과제번호 20010589

부처명 산업통상자원부

과제관리(전문)기관명 한국산업기술평가관리원

연구사업명 시스템반도체 핵심 IP 개발 사업

연구과제명 스마트인증과 데이터 유출의 무결성 보장을 위한 차세대 동형암호(HE)기반 IP개발

기 여 율 1/2

과제수행기관명 인하대학교 산학협력단

연구기간 2020.04.01 ~ 2022.12.31

명세서

청구범위

청구항 1

암호 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 키 생성기(Key Generator);

암호화를 위한 원문(original message) 및 스케일링 팩터(scaling factor)를 수신 받아 다항식을 이용하여 암호화를 위한 평문으로 부호화하는 부호화기(Encoder);

암호화에 필요한 에러를 생성하는 가우시안 분포 에러 생성기(Gaussian Distribution); 및

NTT 코어, 모듈러 곱셈기(Modular Multiplier), 모듈러 스위치(Modular Switch), 모듈러 덧셈기(Modular Addition)를 포함하고, 공개키 및 평문을 수신하여 암호화하는 동형 암호화(Homomorphic Encryption) 프로세서를 포함하고,

상기 동형 암호화 프로세서는,

연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 축소하고, 암호화의 고속화를 위한 NTT 연산을 수행하는 NTT 코어;

상기 NTT 코어를 통해 연산된 결과들에 대하여 공개키와의 곱셈 연산을 수행하는 모듈러 곱셈기;

키(Key) 크기의 데이터를 평문 크기의 데이터로 축소하는 모듈러 스위치; 및

상기 모듈러 스위치의 결과와 상기 부호화기를 통해 생성된 평문을 더하여 암호문을 생성하는 모듈러 덧셈기를 포함하고,

상기 모듈러 곱셈기는,

FPGA 기반의 암호시스템에서 DSP를 이용하여 곱셈기의 복잡도를 낮추기 위해 바렛 알고리즘을 이용한 정수 곱셈기를 이용하고,

모듈러 연산의 복잡도를 낮추기 위해 미리 정해진 파라미터 Q에 대한 보조 곱셈기 및 미리 계산된 T에 대한 보조 곱셈기를 이용함으로써 상기 바렛 알고리즘을 이용한 정수 곱셈기를 통해 곱셈된 값에 대한 쉬프트 연산과 덧셈 및 곱셈만으로 모듈러 연산을 수행하는

동형암호를 위한 암호화 처리 장치.

청구항 2

제1항에 있어서,

상기 NTT 코어는,

상기 가우시안 분포 에러 생성기를 통해 생성된 임의의 크기의 데이터가 RNS(Residue Number System) 방법으로 나뉜 값들에 대해 다항식 곱셈을 수행하기 위해 NTT 연산을 수행하는 파이프라인이 가능한 2-병렬 시스톱-어레이 NTT 코어이고,

상기 2-병렬 시스톱-어레이 NTT 코어는 CT(Cooley-Turkey), GS(Gentlemen-Sande)의 알고리즘을 사용하여 비트 역순과정을 생략하는

동형암호를 위한 암호화 처리 장치.

청구항 3

제1항에 있어서,

상기 모듈러 스위치는,

NTT 코어 및 INTT(Inverse NTT) 코어를 포함하고,

특수 파라미터(P)에 해당하는 데이터에 대해 INTT 연산을 수행한 후, 나머지 파라미터에 해당하는 데이터에 대해 NTT 연산을 수행한 후의 값에 대한 뺄셈을 수행하도록 함으로써, 곱셈 이후의 암호문의 데이터 크기를 축소하여 새로운 암호문 데이터로 생성하고, 상기 새로운 암호문에서의 오류의 크기를 감소시키는

동형암호를 위한 암호화 처리 장치.

청구항 4

제3항에 있어서,

상기 INTT 코어는,

다항식 입력에 대해 INTT 연산을 수행하는 파이프라인이 가능한 2-병렬 시스토크-어레이 INTT 코어인

동형암호를 위한 암호화 처리 장치.

청구항 5

삭제

청구항 6

키 생성기(Key Generator)를 통해 암호 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 단계;

가우시안 분포 에러 생성기(Gaussian Distribution)를 통해 암호화에 필요한 에러를 생성하는 단계;

부호화기(Encoder)를 통해 암호화를 위한 원문(original message) 및 스케일링 팩터(scaling factor)를 수신받아 다항식을 이용하여 암호화를 위한 평문으로 부호화하는 단계; 및

NTT 코어, 모듈러 곱셈기(Modular Multiplier), 모듈러 스위치(Modular Switch), 모듈러 덧셈기(Modular Addition)를 포함하는 동형 암호화(Homomorphic Encryption) 프로세서를 통해 공개키 및 평문을 수신하여 암호화하는 단계

를 포함하고,

상기 동형 암호화 프로세서를 통해 공개키 및 평문을 수신하여 암호화하는 단계는,

NTT 코어를 통해 연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 축소하는 단계;

모듈러 곱셈기를 통해 상기 NTT 코어에서 연산된 결과들에 대하여 공개키와의 곱셈 연산을 수행하는 단계;

모듈러 스위치를 통해 키(Key) 크기의 데이터를 평문 크기의 데이터로 축소하는 단계; 및

모듈러 덧셈기를 통해 상기 모듈러 스위치에서의 결과와 상기 부호화기를 통해 생성된 평문을 더하여 암호문을 생성하는 단계

를 포함하고,

상기 모듈러 곱셈기를 통해 상기 NTT 코어에서 연산된 결과들에 대하여 공개키와의 곱셈 연산을 수행하는 단계는,

FPGA 기반의 암호시스템에서 DSP를 이용하여 곱셈기의 복잡도를 낮추기 위해 바렛 알고리즘을 이용한 정수 곱셈기를 이용하고,

모듈러 연산의 복잡도를 낮추기 위해 미리 정해진 파라미터 Q에 대한 보조 곱셈기 및 미리 계산된 T에 대한 보조 곱셈기를 이용함으로써 상기 바렛 알고리즘을 이용한 정수 곱셈기를 통해 곱셈된 값에 대한 쉬프트 연산과 덧셈 및 곱셈만으로 모듈러 연산을 수행하는

동형암호를 위한 암호화 처리 방법.

청구항 7

제6항에 있어서,

상기 NTT 코어를 통해 연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 축소하는 단계는,

CT(Cooley-Turkey), GS(Gentlemen-Sande)의 알고리즘을 사용하여 비트 역순과정을 생략하고, 파이프라인이 가능한 2-병렬 시스토크-어레이 NTT 코어를 이용하여 상기 가우시안 분포 에러 생성기에서 생성된 임의의 크기의 데이터가 RNS(Residue Number System) 방법으로 나뉜 값에 대한 다항식 곱셈을 수행하기 위해 NTT 연산을 수행하는

동형암호를 위한 암호화 처리 방법.

청구항 8

제6항에 있어서,

상기 모듈러 스위치를 통해 키 크기의 데이터를 평균 크기의 데이터로 축소하는 단계는,

NTT 코어 및 INTT(Inverse NTT) 코어를 포함하는 모듈러 스위치를 이용하여 특수 파라미터(P)에 해당하는 데이터에 대해 INTT 연산을 수행한 후, 나머지 파라미터에 해당하는 데이터에 대해 NTT 연산을 수행한 후 값에 대한 뺄셈을 수행하도록 함으로써, 곱셈 이후의 암호문의 데이터 크기를 축소하여 새로운 암호문 데이터로 생성하고, 상기 새로운 암호문에서의 오류의 크기를 감소시키는

동형암호를 위한 암호화 처리 방법.

발명의 설명

기술 분야

[0001] 본 발명은 동형암호를 위한 암호화 처리 장치 및 방법에 관한 것이다.

배경 기술

[0002] 동형암호는 평문과 암호문이 같은 성질을 유지하는 기술로, 평문에 대한 연산 결과가 암호문에 대한 연산 결과와 같은 값을 가진다. 이런 특징을 이용하면, 암호화된 데이터를 복호화 없이 연산하는 암호로서, 암호화된 개인 정보를 풀어보지 않고도 통계분석이 가능하다. 암호화된 상태에서 통계처리나 검색, 기계학습(Machine Learning)이 가능하기 때문에 해커가 데이터를 해킹하더라도 암호를 해석할 수 없다. 이 때문에 의료나 금융 분야에서 개인 빅데이터 정보를 활용할 때 정보가 유출 되더라도 개인 식별이 불가능하다는 점에서 도용이 불가능한 "완전 암호화"로 불린다.

[0003] 현재 동형암호 기술은 미래 암호 기술의 핵심으로 다양한 응용 가능성 때문에 학계 및 산업계의 관심이 높으나, 수학적 난이도가 높고 기존 암호 체계에 비해 암호복호화 및 연산 속도가 느려, 실용화를 위해서는 고성능 동형암호 하드웨어 아키텍처 구현이 필요하다.

[0004] 동형암호를 위한 방법으로는 정수-기반 방법과 격자-기반 방법이 있으며, 현재까지는 격자-기반 방법들 중에 하나인 Ring-LWE 암호는 효율적인 하드웨어 아키텍처 구현이 가능하면서 낮은 하드웨어 복잡도를 가지는 반면에 암호화된 데이터 연산시에 연산 횟수가 증가함에 따라 오류가 증가하는 문제가 있어서, 연산 횟수에 제한이 있다.

[0005] 따라서 연산 횟수에 제한이 없으면서 하드웨어 복잡도가 작은 완전동형 암호 아키텍처 및 암호시스템 연구가 필수적이다. CKKS 기반 동형암호는 격자(Lattice)-기반 알고리즘으로서 Ring-LWE 기법을 사용한다. Ring-LWE는 링(Ring) 안에서 계산되며 다항식 곱셈이 주로 쓰이는데 가장 많은 시간자원을 사용하므로 다항식 연산속도를 향상시켜주는 고성능 NTT를 필요로 한다.

선행기술문헌

특허문헌

[0006] (특허문헌 0001) 한국 등록특허공보 제10-1600016호(2016.02.26)

발명의 내용

해결하려는 과제

- [0007] 본 발명이 이루고자 하는 기술적 과제는 CKKS 기반 동형암호를 위한 암호화 프로세서 구조를 2-병렬 시스톱-어레이(Systolic-array) NTT 코어를 사용하는 고성능 암호화 하드웨어 구조 및 방법을 제공하는데 있다. 또한, 곱셈기로서 바렛 모듈러 곱셈기(Barrett Modular Multiplier)를 이용하여 FPGA 기반의 암호시스템에서 DSP를 이용하여 곱셈기의 복잡도를 단순화하고, 미리 정해진 파라미터 Q 값과 미리 계산된 T를 이용하여 모듈러 연산을 간단한 쉬프트 연산과 덧셈 및 곱셈을 통해 단순화하고자 한다. 또한, 모듈러 스위치(ModSwitch)를 이용하여 데이터의 크기를 감소시키고, NTT 다항식 곱셈기의 하드웨어 복잡성을 줄이며, 클럭 속도를 높이기 위해 2-병렬 시스톱-어레이 구조, 파이프라인 구조, 및 비트-역순(Bit-Reverse) 연산을 제거하는 방법을 제공하고자 한다.

과제의 해결 수단

- [0008] 일 측면에 있어서, 본 발명에서 제안하는 동형암호 장치는 암호 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 키 생성기(Key Generator), 다항식을 수신하여 암호화에 적합한 평문으로 부호화하는 부호화(Encoder) 프로세서, 암호화에 필요한 에러를 생성하는 가우시안 분포 에러 생성기(Gaussian Distribution) 및 NTT 코어, 모듈러 곱셈기(Modular Multiplier), 모듈러 스위치(Modular Switch), 모듈러 덧셈기(Modular Addition)를 포함하고, 공개키 및 평문을 수신하여 암호화하는 동형 암호화(Homomorphic Encryption) 프로세서를 포함하고, 상기 동형 암호화 프로세서는 연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 축소하고, 암호화의 고속화를 위한 NTT 연산을 수행하는 NTT 코어, 상기 NTT 코어를 통해 연산된 결과들에 대하여 공개키와의 곱셈 연산을 수행하는 모듈러 곱셈기, 키(Key) 크기의 데이터를 평문 크기의 데이터로 축소하는 모듈러 스위치 및 상기 모듈러 스위치의 결과와 상기 부호화기를 통해 생성된 평문을 더하여 암호문을 생성하는 모듈러 덧셈기를 포함한다.
- [0009] 본 발명의 실시예에 따른 NTT 코어는 상기 가우시안 분포 에러 생성기를 통해 생성된 임의의 크기의 데이터가 RNS(Residue Number System) 방법으로 나뉜진 반복적으로 이용되는 다항식 곱셈을 수행하기 위해 NTT 연산을 수행하는 파이프라인이 가능한 2-병렬 시스톱-어레이 NTT 코어이고, 상기 2-병렬 시스톱-어레이 NTT 코어는 CT(Cooley-Turkey), GS(Gentlemen-Sande)의 알고리즘을 사용하여 비트 역순과정을 생략할 수 있다.
- [0010] 본 발명의 실시예에 따른 모듈러 스위치는 NTT 코어 및 INTT(Inverse NTT) 코어를 포함하고, 특수 파라미터(P)에 해당하는 데이터에 대해 INTT 연산을 수행한 후, 나머지 파라미터에 해당하는 데이터에 대해 NTT 연산을 수행한 후 값에 대한 뺄셈을 수행하도록 함으로써, 곱셈 이후의 암호문의 데이터 크기를 축소하여 새로운 암호문 데이터로 생성하고, 상기 새로운 암호문에서의 오류의 크기를 감소시킨다.
- [0011] 본 발명의 실시예에 따른 INTT 코어는 다항식 입력에 대해 INTT 연산을 수행하는 파이프라인이 가능한 2-병렬 시스톱-어레이 INTT 코어일 수 있다.
- [0012] 본 발명의 실시예에 따른 모듈러 곱셈기는 바렛 알고리즘을 이용한 정수 곱셈기, 미리 정해진 파라미터 Q에 대한 보조 곱셈기, 미리 계산된 T에 대한 보조 곱셈기를 이용하여 곱셈된 값에 모듈러 연산을 수행한다.
- [0013] 또 다른 일 측면에 있어서, 본 발명에서 제안하는 동형암호 장치의 암호화 방법은 키 생성기(Key Generator)를 통해 암호 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 단계, 가우시안 분포 에러 생성기(Gaussian Distribution)를 통해 암호화에 필요한 에러를 생성하는 단계, 부호화기(Encoder)를 통해 다항식을 수신하여 암호화에 적합한 평문으로 부호화하는 단계 및 NTT 코어, 모듈러 곱셈기(Modular Multiplier), 모듈러 스위치(Modular Switch), 모듈러 덧셈기(Modular Addition)를 포함하는 동형 암호화(Homomorphic Encryption) 프로세서를 통해 공개키 및 평문을 수신하여 암호화하는 단계를 포함하고, 상기 동형 암호화 프로세서를 통해 공개키 및 평문을 수신하여 암호화하는 단계는 NTT 코어를 통해 연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 축소하는 단계, 모듈러 곱셈기를 통해 상기 NTT 코어에서 연산된 결과들에 대하여 공개키와의 곱셈 연산을 수행하는 단계, 모듈러 스위치를 통해 키(Key) 크기의 데이터를 평문 크기의 데이터로 축소하는 단계 및 모듈러 덧셈기를 통해 상기 모듈러 스위치에서의 결과와 상기 부호화기를 통해 생성된 평문을 더하여 암호문을 생성하는 단계를 포함한다.

발명의 효과

- [0014] 본 발명의 실시예들에 따르면 Ring-LWE 기반 CKKS 동형암호 프로세서에서 NTT 연산을 수행하는 과정에 있어서

데이터 메모리에 접근 없이 연속적인 연산을 통해 하드웨어 복잡도를 낮출 수 있다. CKKS 기반 동형암호를 위한 암호화 프로세서 구조를 2-병렬 시스톱-어레이(Systolic-array) NTT 코어를 사용하는 고성능 암호화 하드웨어 구조 및 방법을 제공하고, 곱셈기로서 바렛 모듈러 곱셈기(Barrett Modular Multiplier)를 이용하여 FPGA 기반의 암호시스템에서 DSP를 이용하여 곱셈기의 복잡도를 단순화할 수 있다. 또한, 미리 정해진 파라미터 Q 값과 미리 계산된 T를 이용하여 모듈러 연산을 간단한 쉬프트 연산과 덧셈 및 곱셈을 통해 단순화할 수 있다. 또한, 모듈러 스위치(ModSwitch)를 이용하여 데이터의 크기를 감소시키고, NTT 다항식 곱셈기의 하드웨어 복잡성을 줄이며, 2-병렬 시스톱-어레이 구조, 파이프라인 구조, 및 비트-역순(Bit-Reverse) 연산을 제거하는 방법을 제공하여 클럭 속도를 높일 수 있다.

도면의 간단한 설명

[0015]

도 1은 본 발명의 일 실시예에 따른 CKKS 기반 동형암호 장치의 구성을 설명하기 위한 도면이다.

도 2는 본 발명의 일 실시예에 따른 NTT 코어를 이용한 CKKS 기반 동형암호 장치의 동형 암호화 프로세서의 구조를 나타내는 도면이다.

도 3은 본 발명의 일 실시예에 따른 NTT 코어와 INTT 코어를 이용한 모듈러 스위치(Modular Switch)의 구조를 나타내는 도면이다.

도 4는 본 발명의 일 실시예에 따른 2-병렬 시스톱-어레이 NTT 코어의 구조를 나타내는 도면이다.

도 5는 본 발명의 일 실시예에 따른 2-병렬 시스톱-어레이 INTT 코어의 구조를 나타내는 도면이다.

도 6은 본 발명의 일 실시예에 따른 2-병렬 시스톱-어레이 NTT 코어의 PE1(Processing Element 1)의 구조를 나타내는 도면이다.

도 7은 본 발명의 일 실시예에 따른 NTT 코어의 PE2(Processing Element 2)의 구조를 나타내는 도면이다.

도 8은 본 발명의 일 실시예에 따른 모듈러 곱셈기(Modular Multiplier; MM)의 구조를 나타내는 도면이다.

도 9는 본 발명의 일 실시예에 따른 정수 곱셈기(IntMult)의 구조를 나타내는 도면이다.

도 10은 본 발명의 일 실시예에 따른 Q에 대한 보조 곱셈기(HalfMult_Q)의 구조를 나타내는 도면이다.

도 11은 본 발명의 일 실시예에 따른 T에 대한 보조 곱셈기(HalfMult_T)의 구조를 나타내는 도면이다.

도 12는 본 발명의 일 실시예에 따른 동형암호의 암호화 프로세서에서의 데이터 흐름을 설명하기 위한 도면이다.

도 13은 본 발명의 일 실시예에 따른 CKKS 기반 동형암호 장치의 암호화 프로세서의 암호화 방법을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0016]

본 발명은 NTT(Number Theoretic Transform) 코어를 이용한 안전한 데이터 저장과 효율적인 데이터 처리를 가지는 Ring-LWE(Learning With Error)를 이용한 Large N에 대한 CKKS-기반 동형암호(Homomorphic Encryption; HE)의 암호화 처리 장치 및 방법에 관한 것으로, 보다 상세하게는 파이프라인 구조를 위한 방식 중 높은 데이터 처리율 대비 낮은 하드웨어 면적을 갖는 시스톱-어레이 방식을 적용하여 동형 암호화 프로세서의 데이터 메모리 접근을 줄여 하드웨어 복잡도를 단순화하는 방법에 대한 것이다.

[0017]

본 발명의 실시예에 따른 CKKS 기반 동형암호는 격자-기반 암호 알고리즘으로서 Ring-LWE 기법을 사용한다. Ring-LWE는 링(Ring) 안에서 계산되며 다항식 곱셈이 주로 쓰이는데 가장 많은 시간자원을 사용함으로 다항식 연산속도를 향상시켜 주는 고성능 NTT를 필요로 한다. 또한 메시지를 포함하는 데이터 다음에 노이즈(Noise)에 해당하는 데이터를 추가하여 암호화 연산을 진행한 후 모듈러 스위치 연산을 이용하여 노이즈에 해당하는 크기의 데이터를 줄이는 방식으로 근사 산술 연산의 정확도를 높인다. 또한 Ring-LWE 기법을 이용하여 에러의 크기를 늘리지 않아 암호화하려는 메시지의 정밀도를 유지할 수 있다.

[0018]

본 발명의 실시예에 따르면, 비밀 키 $< c, sk > = m + e \pmod{q}$ 형식의 암호 해독 구조를 지게 된다. 여기서 c는 암호문(Ciphertext)이며, sk는 비밀 키(Secret Key), m은 암호화하고자 하는 메시지, e는 에러(Error)를 의미한다. 여기서 e는 안전한 암호화를 보장하기 위해 삽입된 작은 에러이다. 에러가 메시지에 비해 충분히 작다

면 이 예러는 m 의 유효한 메시지를 파괴하지 않고 대략적인 산술로 본래의 메시지를 대체할 수 있다.

- [0019] 제안하는 NTT 코어를 사용한 CKKS 기반 동형암호 아키텍처는 높은 데이터 처리율을 얻기 위해 파이프라인이 가능한 구조인 2-병렬 구조를 사용한다. 또한, 연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 줄여 하드웨어 복잡도 대비 높은 데이터 처리율을 가지는 NTT 코어, NTT 연산된 결과들을 공개키와 곱셈 연산을 수행하는 모듈러 곱셈기(Modular Multiplier), 키(Key) 크기의 데이터를 평문 크기의 데이터로 줄여주는 모듈러 스위치(Modular Switch), 평문을 더해 암호문을 만드는 모듈러 덧셈기(Modular Addition)를 포함한다. 이하, 본 발명의 실시 예를 첨부된 도면을 참조하여 상세하게 설명한다.
- [0021] 도 1은 본 발명의 일 실시예에 따른 CKKS 기반 동형암호 장치의 구성을 설명하기 위한 도면이다.
- [0022] 제안하는 CKKS 기반 동형암호 장치는 동형 암호화(Homomorphic Encryption) 프로세서(110), 키 생성기(Key Generator)(120), 부호화(Encoder) 프로세서(130) 및 가우시안 분포 에러 생성기(Gaussian Distribution)(140)를 포함한다.
- [0023] 본 발명의 실시예에 따른 키 생성기(120)는 암호 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성한다.
- [0024] 본 발명의 실시예에 따른 부호화기(130)는 암호문을 부호화한다. 본 발명의 실시예에 따른 부호화기(130)는 다항식을 수신하여 암호화에 적합한 평문으로 변화시켜주는 인코더이다.
- [0025] 본 발명의 실시예에 따른 가우시안 분포 에러 생성기(140)는 암호화에 필요한 변수들을 임의로 생성하여 에러를 생성한다.
- [0026] 본 발명의 실시예에 따른 동형 암호화 프로세서(110)는 공개키 및 평문을 수신하여 암호화를 수행한다.
- [0027] 본 발명의 실시예에 따른 동형 암호화 프로세서(110)는 NTT 코어(111, 112, 113), 모듈러 곱셈기(Modular Multiplier)(141, 142), 모듈러 스위치(Modular Switch)(121, 122), 모듈러 덧셈기(Modular Addition)(131, 132)를 포함한다.
- [0028] 본 발명의 실시예에 따른 NTT 코어(111, 112, 113)는 연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 축소하고, 암호화의 고속화를 위한 NTT 연산을 수행한다.
- [0029] 본 발명의 실시예에 따른 NTT 코어(111, 112, 113)는 상기 가우시안 분포 에러 생성기(140)를 통해 생성된 임의의 크기의 데이터가 RNS(Residue Number System) 방법으로 나뉜 반복적으로 이용되는 다항식 곱셈을 수행하기 위해 NTT 연산을 수행한다.
- [0030] 본 발명의 실시예에 따른 NTT 코어(111, 112, 113)는 16384차 다항식을 수신할 수 있는 입력에 대해 NTT 연산을 수행하는 파이프라인이 가능한 2-병렬 시스톱-어레이 NTT 코어일 수 있다. 또한, 본 발명의 실시예에 따른 2-병렬 시스톱-어레이 NTT 코어는 CT(Cooley-Turkey), GS(Gentlemen-Sande)의 알고리즘을 사용하여 비트 역순 과정을 생략할 수 있다.
- [0031] 본 발명의 실시예에 따른 모듈러 곱셈기(141, 142)는 연산된 결과들에 대하여 공개키와의 곱셈 연산을 수행한다. 다시 말해, 여러 입력 데이터에 대해 NTT 연산을 수행한 결과에 대해 공개키를 곱하는 장치이다.
- [0032] 본 발명의 실시예에 따른 모듈러 곱셈기(141, 142)는 바렛 알고리즘을 이용한 정수 곱셈기, 미리 정해진 파라미터 Q 에 대한 보조 곱셈기, 미리 계산된 T 에 대한 보조 곱셈기를 이용하여 곱셈된 값에 모듈러 연산을 수행한다.
- [0033] 본 발명의 실시예에 따른 모듈러 스위치(121, 122)는 NTT 코어 및 INTT(Inverse NTT) 코어를 포함하고, 키(Key) 크기의 데이터를 평문 크기의 데이터로 축소한다.
- [0034] 본 발명의 실시예에 따른 모듈러 스위치(121, 122)는 특수 파라미터(P)에 해당하는 데이터에 대해 INTT 연산을 수행한 후, 나머지 파라미터에 해당하는 데이터에 대해 NTT 연산을 수행한 후 값에 대한 뺄셈을 수행하도록 함으로써, 곱셈 이후의 암호문의 데이터 크기를 축소하여 새로운 암호문 데이터로 생성하고, 상기 새로운 암호문에서의 오류의 크기를 감소시킬 수 있다.
- [0035] 본 발명의 실시예에 따른 모듈러 스위치(121, 122)의 INTT 코어는 16384차 다항식을 수신할 수 있는 다항식 입력에 대해 INTT 연산을 수행하는 파이프라인이 가능한 2-병렬 시스톱-어레이 INTT 코어일 수 있다.
- [0036] 본 발명의 실시예에 따른 모듈러 덧셈기(131, 132)는 모듈러 스위치(121, 122)의 결과와 부호화기(130)를 통해 생성된 평문을 더하여 암호문을 생성한다. 또한, 모듈러 곱셈기(141, 142)의 결과에 에러의 NTT 연산 후의 결과

를 더한다.

[0037] 도 1를 참조하면, 본 발명의 실시예에 따른 부호화기(다시 말해, 인코더)(130)는 Δ (scaling factor)와 z (original message)를 이용하여 암호화 프로세서에서 사용할 수 있는 평문(Plaintext)으로 변환할 수 있다. 키 생성은 1^{λ} 을 입력으로 암호화 프로세서에서 사용할 수 있는 공개키(Public key), 비밀키(Secret key), 평가키(Evaluation key)를 생성할 수 있다.

[0038] 본 발명의 실시예에 따른 가우시안 분포 에러 생성기(140)는, Z_{op} 는 $\{-1, 0, 1\}$, e_0, e_1 는 $\{-10 \sim 10\}$ 의 정수를 생성할 수 있다. 가우시안 분포 에러 생성기(140)의 출력에 대해 빠른 다항식 곱셈 방법인 NTT를 이용하기 위해 RNS(Residue Number System) 방법으로, $\{60, 48, 48, 48, 60\}$ 으로 나뉘어 연산한다. RNS는 64비트가 넘는 큰 정수의 계산의 복잡성을 개선하기 위하여 사용하였다. 큰 수 하나를 RNS 기저 $\{q_0, q_1, \dots, q_i\}$ 로 나눈 나머지를 이용하는 연산이다. 나뉘진 값은 빠른 다항식 곱셈을 이용하기 위해 NTT 연산을 수행한다. NTT 연산은 수학적 1처럼 나타낼 수 있다.

[0039]
$$A_i = \sum_{j=0}^{n-1} a_j w_n^{ij} \bmod q, \text{ for } i = 0, 1, \dots, n-1$$
 수학적 1

[0040] 또한, 본 발명의 실시예에 따른 Ring-LWE의 모든 연산은 $R_q = \mathbb{Z}_q[x]/f(x)$ 의 조건을 만족시키면서 동작하므로 다항식 $a(x)$ 와 $b(x)$ 는 수학적 2과 같이 나타낼 수 있다.

[0041]
$$a(x) = a_0 + a_1x + a_2x^2 + \dots + a_{n-1}x^{n-1}$$
 수학적 2-1

[0042]
$$b(x) = b_0 + b_1x + b_2x^2 + \dots + b_{n-1}x^{n-1}$$
 수학적 2-2

[0044] 벡터(Vector) $\mathbf{a} = a_0, \dots, a_{n-1}$ 는 다항식 $a(x)$ 의 계수이다. 숫자 n 은 1의 n 제곱근(root of unity)으로 본 발명의 실시예에 따른 프로세서에서는 $n=16384$ 의 크기를 가진다. Q 는 $q \equiv 1 \pmod{2N}$ 을 만족하는 소수(prime number)이다. 수학적 1에서 n 과 w 는 $n * n^{-1} \equiv 1 \bmod q, w * w^{-1} \equiv 1 \bmod q$ 의 조건을 만족하는 숫자로 구성된다. 모듈러 곱셈기는 $AB = A * B \pmod{q}$ 와 같은 연산을 수행하며 다항식 간의 점별 곱셈을 진행하며 정해진 q 값에 대해 모듈러 연산을 진행한다. 모듈러 덧셈기는 $AB = A+B \pmod{q}$ 와 같은 연산을 수행하며 다항식간의 점별 덧셈을 진행하며 정해진 q 값에 대해 모듈러 연산을 진행한다. 모듈러 스위치는 $\{60, 48, 48, 48, 60\}$ 의 크기에 해당하는 키 크기의 데이터를 $\{60, 48, 48, 48\}$ 크기와 같은 특정 파라미터(special prime)로 줄여 평문의 크기와 동일하게 만들어준다.

[0046] 도 2는 본 발명의 일 실시예에 따른 NTT 코어를 이용한 CKKS 기반 동형암호 장치의 동형 암호화 프로세서의 구조를 나타내는 도면이다.

[0047] 제안하는 CKKS 기반 동형암호 장치의 동형 암호화 프로세서는 NTT 코어를 사용하여 연산한 결과와 공개키를 곱하는 모듈러 곱셈기, 키 데이터의 크기를 암호문 데이터 크기로 줄여주는 모듈러 스위치, 모듈러 스위치 연산의 결과에 평문을 더하는 모듈러 덧셈기를 포함한다. 이러한 동형 암호화 프로세서의 결과는 서버로 전달되어 다양한 연산을 진행한 후, 결과를 다시 클라이언트 측으로 수신한다.

[0048] 파라미터는 각각 $q_0 = 2^{60} - 2^{18} + 2^{16} + 2^{15} + 1$ (60비트), $q_1 = 2^{48} - 2^{20} + 1$ (48비트), $q_2 = 2^{48} - 2^{19} + 2^{17} + 1$ (비트), $q_3 = 2^{48} - 2^{18} + 2^{16} + 2^{15} + 1$ (48비트), $P = 2^{60} - 2^{17} + 2^{15} + 1$ (60비트)를 가지고 수행된다. 이 경우 Level = 3으로 암호화 후에 연산될 수 있는 횟수에 해당한다. P 에 해당하는 파라미터는 모듈러 스위치 이후에 사라지는 파라미터이다.

[0050] 도 3은 본 발명의 일 실시예에 따른 NTT 코어와 INTT 코어를 이용한 CKKS 모듈러 스위치(Modular Switch)의 구조를 나타내는 도면이다.

[0051] 본 발명의 실시예에 따른 모듈러 스위치는 특수 파라미터 P 에 대해 INTT(Inverse Number Theoretic Transform)를 수행한다. INTT는 GS(Gentlemen-Sande) 알고리즘을 이용하며 수학적 3으로 나타낸다.

[0052]
$$a_i = n^{-1} \sum_{j=0}^{n-1} A_j w_n^{-ij} \bmod q, \text{ for } i = 0, 1, \dots, n-1$$
 수학적 3

- [0053] 본 발명의 실시예에 따른 INTT의 연산 결과에 모듈러 덧셈기를 이용하여 $P \gg 1$ 를 더한다. 그 결과를 바렛(Barrett) 모듈러 알고리즘을 이용하여 각 파라미터에 맞게 모듈러 연산을 진행한다. 이후 모듈러 뺄셈기를 이용하여 $P \gg 1 \bmod q$ 의 값인 각 파라미터에 해당하는 값을 뺀다. 각 파라미터에 해당하는 NTT 연산을 입력 값에서 뺄셈을 진행한 뒤 $P^{-1} \bmod q$ 를 모듈러 곱셈기를 통해 진행하여 결과를 도출한다. 이 결과는 이전의 입력과 다르게 {60, 48, 48, 48}의 크기가 도출된다.
- [0055] 도 4는 본 발명의 일 실시예에 따른 2-병렬 시스토크-어레이 NTT 코어의 구조를 나타내는 도면이다.
- [0056] 본 발명의 실시예에 따른 2-병렬 시스토크-어레이 NTT 코어의 모든 데이터는 2-병렬 구조로 연산을 수행하며, NTT 코어는 PE1과 PE2로 구성된다. 입력은 2-path를 사용하고 각 경로(path)에서는 PE1구조를 이용하여 연산하며, 각 경로는 스테이지(stage) 13까지 8192-point BU(Butterfly Unit)부터 2-point BU까지 연산하고 스테이지(stage) 14에서는 2-point BU 연산을 위한 PE2 구조 1개가 사용된다.
- [0058] 도 5는 본 발명의 일 실시예에 따른 2-병렬 시스토크-어레이 INTT 코어의 구조를 나타내는 도면이다.
- [0059] 본 발명의 실시예에 따른 INTT 또한 상술된 NTT와 동일하게 PE1과 PE2로 구성되어 있으며 NTT와 반대로 연산을 수행하게 된다.
- [0061] 도 6은 본 발명의 일 실시예에 따른 2-병렬 시스토크-어레이 NTT 코어의 PE1(Processing Element 1)의 구조를 나타내는 도면이다.
- [0062] 본 발명의 실시예에 따른 PE1은 모듈러 곱셈기, 모듈러 덧셈기, 모듈러 뺄셈기, FIFO 레지스터, 모듈러 리덕션(Modular Reduction)으로 구성되어 있다. 입력된 데이터는 모듈러 곱셈기를 통해 ROM의 형태로 저장된 ω^i 값과 곱셈 및 모듈러 연산을 한다. 해당 연산을 수행한 후에 먼저 입력된 값은 FIFO를 통해 입력된 값과 BU(Butterfly Unit) 연산을 수행하여 링(Ring) 내의 값을 도출하기 위해 모듈러 리덕션을 진행 후 결과를 출력한다.
- [0064] 도 7은 본 발명의 일 실시예에 따른 NTT 코어의 PE2(Processing Element 2)의 구조를 나타내는 도면이다.
- [0065] 본 발명의 실시예에 따른 PE2는 PE1과 다르게 FIFO가 없고 바로 BU연산을 수행하게 된다.
- [0067] 도 8은 본 발명의 일 실시예에 따른 모듈러 곱셈기(Modular Multiplier; MM)의 구조를 나타내는 도면이다.
- [0068] 본 발명의 실시예에 따른 모듈러 곱셈기는 바렛 알고리즘(Barrett Algorithm)을 이용한다. 하드웨어 복잡도를 줄이기 위해 특정 Q 값을 사용하였으며, 입력 A, B는 정수 곱셈기(IntMult)를 이용하여 $A*B$ 의 결과를 갖게 된다. 그 결과는 각각 모듈러 연산과 쉬프트 연산을 이용하여 T와 Q의 보조 곱셈기를 통해 연산한 결과와 덧셈 혹은 뺄셈을 진행한다. 또한 Q보다 큰 값에 한하여 Q 값을 뺄셈한 결과를 도출한다.
- [0070] 도 9은 본 발명의 일 실시예에 따른 정수 곱셈기(IntMult)의 구조를 나타내는 도면이다.
- [0071] 본 발명의 실시예에 따른 정수 곱셈기는 FPGA 사용시 하드웨어 복잡도를 줄이기 위해 DSP를 활용하여 입력을 27비트 및 18비트 단위로 곱셈을 수행한다. 하나의 스테이지에서는 27비트x18비트의 곱셈을 수행한다.
- [0073] 도 10은 본 발명의 일 실시예에 따른 Q에 대한 보조 곱셈기(HalfMult_Q)의 구조를 나타내는 도면이다.
- [0074] 본 발명의 실시예에 따르면, 파라미터 $q = 2^{60} - 2^{17} + 2^{15} + 1$ 의 경우에는 입력 A에 대해 결과 $B = (A \ll 60) - (A \ll 17) + (A \ll 15) + A$ 처럼 연산을 수행한다.
- [0076] 도 11은 본 발명의 일 실시예에 따른 T에 대한 보조 곱셈기(HalfMult_T)의 구조를 나타내는 도면이다.
- [0077] 본 발명의 실시예에 따르면, 파라미터 $q = 2^{60} + 2^{17} - 2^{15} - 1$ 의 경우에는 입력 A에 대해 결과 $B = (A \ll 60) + (A \ll 17) - (A \ll 15) - A$ 처럼 연산을 수행한다.
- [0079] 도 12는 본 발명의 일 실시예에 따른 동형암호의 암호화 프로세서에서의 데이터 흐름을 설명하기 위한 도면이다.
- [0080] 도 12를 참조하면, 제안하는 동형암호 장치의 동형 암호화 프로세서의 모듈러 곱셈기를 통해 NTT 코어를 사용하여 연산한 결과와 공개키를 곱하고, 모듈러 스위치를 통해 키 데이터의 크기를 암호문 데이터 크기로 축소하며,

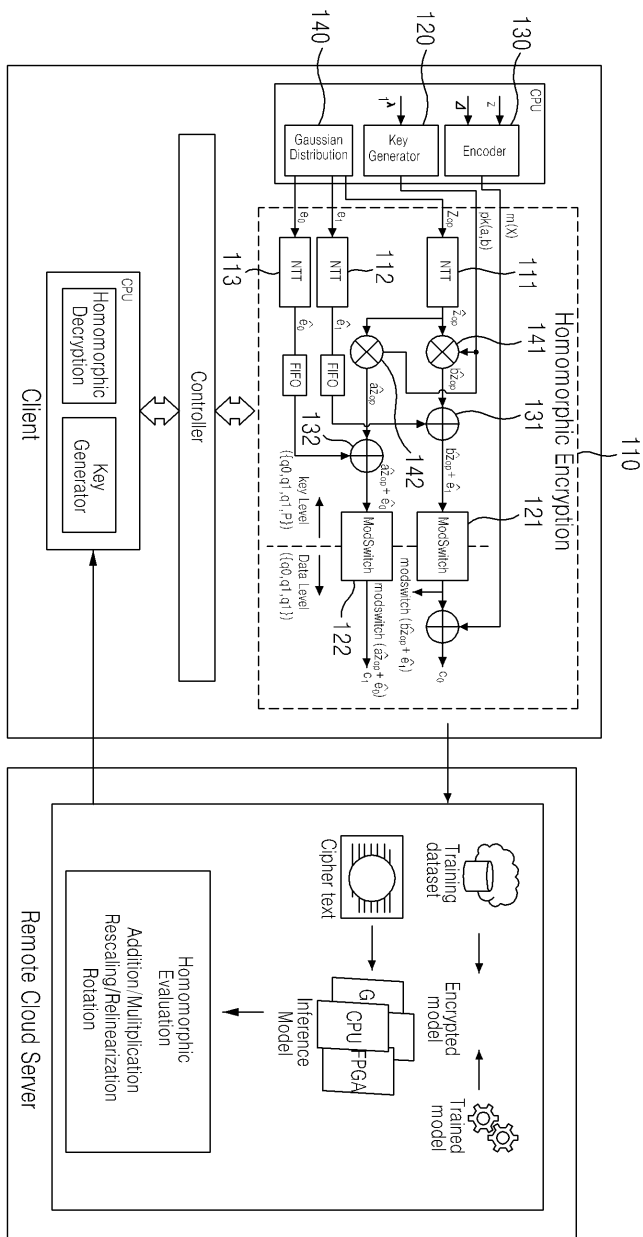
모듈러 덧셈기를 통해 모듈러 스위치 연산의 결과에 평문을 더하는 과정의 데이터 흐름을 나타낸다. 이러한 동형 암호화 프로세서의 결과는 서버로 전달되어 다양한 연산을 진행한 후, 결과를 다시 클라이언트 측으로 수신한다.

- [0082] 도 13은 본 발명의 일 실시예에 따른 CKSS 기반 동형암호 장치의 암호화 프로세서의 암호화 방법을 설명하기 위한 흐름도이다.
- [0083] 제안하는 CKSS 기반 동형암호 장치의 암호화 프로세서의 암호화 방법은 키 생성기(Key Generator)를 통해 암호 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하고, 가우시안 분포 에러 생성기(Gaussian Distribution)를 통해 에러를 생성하며, 부호화기(Encoder)를 통해 암호문을 부호화하는 단계(1310) 및 NTT 코어, 모듈러 곱셈기(Modular Multiplier), 모듈러 스위치(Modular Switch), 모듈러 덧셈기(Modular Addition)를 포함하는 동형 암호화(Homomorphic Encryption) 프로세서를 통해 공개키 및 평문을 수신하여 암호화하는 단계(1320)를 포함한다.
- [0084] 단계(1310)에서, 본 발명의 실시예에 따른 키 생성기는 암호 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성한다.
- [0085] 본 발명의 실시예에 따른 부호화기는 다항식을 수신하여 암호화에 적합한 평문으로 바꾸어준다.
- [0086] 본 발명의 실시예에 따른 가우시안 분포 에러 생성기는 암호화에 필요한 변수들을 임의로 생성하여 에러를 생성한다.
- [0087] 단계(1320)에서, NTT 코어, 모듈러 곱셈기(Modular Multiplier), 모듈러 스위치(Modular Switch), 모듈러 덧셈기(Modular Addition)를 포함하는 동형 암호화(Homomorphic Encryption) 프로세서를 통해 공개키 및 평문을 수신하여 암호화한다.
- [0088] 동형 암호화 프로세서를 통해 공개키 및 평문을 수신하여 암호화하는 단계(1320)는 NTT 코어를 통해 연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 축소하는 단계(1321), 모듈러 곱셈기를 통해 상기 NTT 코어에서 연산된 결과들에 대하여 공개키와의 곱셈 연산을 수행하는 단계(1322), 모듈러 스위치를 통해 키(Key) 크기의 데이터를 평문 크기의 데이터로 축소하는 단계(1323) 및 모듈러 덧셈기를 통해 상기 모듈러 스위치에서 생성된 평문 크기의 데이터에 평문을 더하여 암호문을 생성하는 단계(1324)를 포함한다.
- [0089] 단계(1321)에서, 본 발명의 실시예에 따른 NTT 코어는 연산의 결과를 레지스터(Register)에 저장하여 메모리의 접근을 축소하고, 암호화의 고속화를 위한 NTT 연산을 수행한다.
- [0090] 본 발명의 실시예에 따른 NTT 코어는 상기 가우시안 분포 에러 생성기를 통해 생성된 임의의 크기의 데이터가 RNS(Residue Number System) 방법으로 나뉜 값들에 대해 다항식 곱셈을 수행하기 위해 NTT 연산을 수행한다.
- [0091] 본 발명의 실시예에 따른 NTT 코어는 16384차 다항식을 수신할 수 있는 입력에 대해 NTT 연산을 수행하는 파이프라인이 가능한 2-병렬 시스톱릭-어레이 NTT 코어일 수 있다. 또한, 본 발명의 실시예에 따른 2-병렬 시스톱릭-어레이 NTT 코어는 CT(Cooley-Turkey), GS(Gentlemen-Sande)의 알고리즘을 사용하여 비트 역순과정을 생략할 수 있다.
- [0092] 단계(1322)에서, 본 발명의 실시예에 따른 모듈러 곱셈기는 연산된 결과들에 대하여 공개키와의 곱셈 연산을 수행한다. 다시 말해, 여러 입력 데이터에 대해 NTT 연산을 수행한 결과에 대해 공개키를 곱하는 장치이다.
- [0093] 본 발명의 실시예에 따른 모듈러 곱셈기는 바렛 알고리즘을 이용한 정수 곱셈기, 미리 정해진 파라미터 Q에 대한 보조 곱셈기, 미리 계산된 T에 대한 보조 곱셈기를 이용하여 곱셈된 값에 모듈러 연산을 수행한다.
- [0094] 단계(1323)에서, 본 발명의 실시예에 따른 모듈러 스위치는 NTT 코어 및 INTT(Inverse NTT) 코어를 포함하고, 키(Key) 크기의 데이터를 평문 크기의 데이터로 축소한다.
- [0095] 본 발명의 실시예에 따른 모듈러 스위치는 특수 파라미터(P)에 해당하는 데이터에 대해 INTT 연산을 수행한 후, 나머지 파라미터에 해당하는 데이터에 대해 NTT 연산을 수행한 후 값에 대한 뺄셈을 수행하도록 함으로써, 곱셈 이후의 암호문의 데이터 크기를 축소하여 새로운 암호문 데이터로 생성하고, 상기 새로운 암호문에서의 오류의 크기를 감소시킬 수 있다.
- [0096] 본 발명의 실시예에 따른 모듈러 스위치의 INTT 코어는 16384차 다항식을 수신할 수 있는 다항식 입력에 대해 INTT 연산을 수행하는 파이프라인이 가능한 2-병렬 시스톱릭-어레이 INTT 코어일 수 있다.

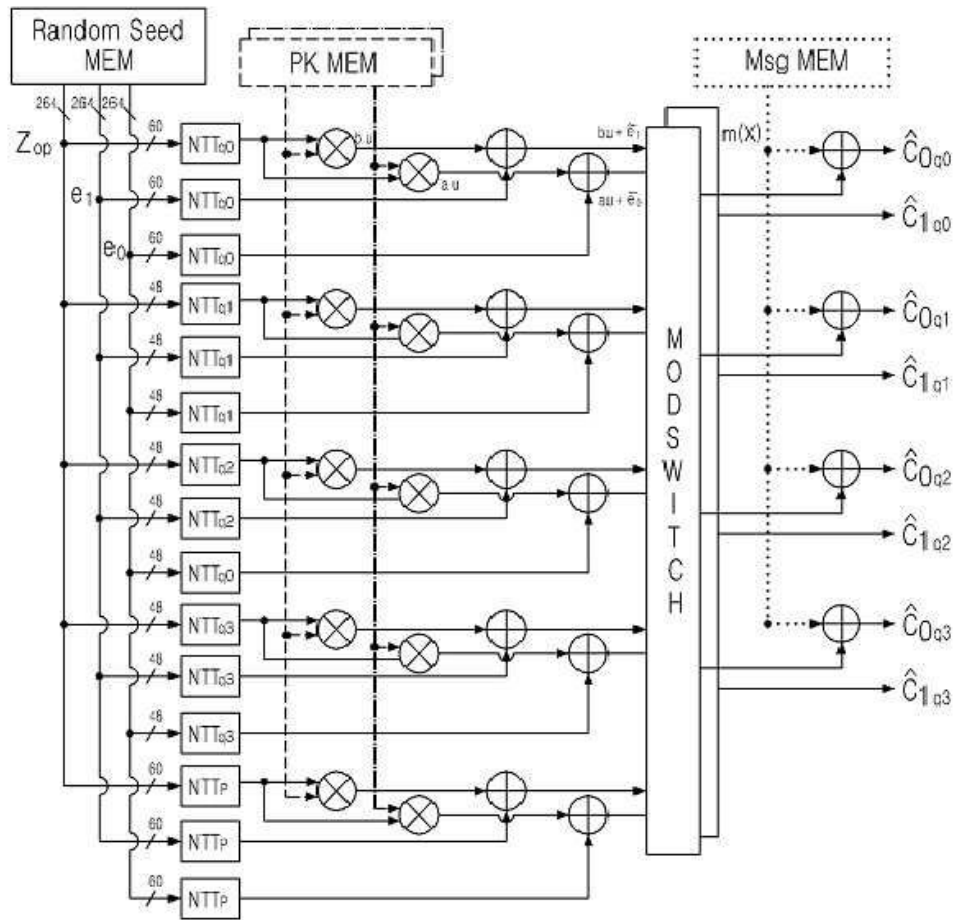
- [0097] 단계(1324)에서, 본 발명의 실시예에 따른 모듈러 덧셈기는 모듈러 스위치의 결과와 부호화기를 통해 생성된 평문을 더하여 암호문을 생성한다. 또한, 모듈러 곱셈기의 결과에 에러의 NTT 연산 후의 결과를 더한다.
- [0099] 이상에서 설명된 장치는 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성요소는, 예를 들어, 프로세서, 콘트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPA(field programmable array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 콘트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.
- [0100] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치에 구체화(embody)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.
- [0101] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다.
- [0102] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.
- [0103] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

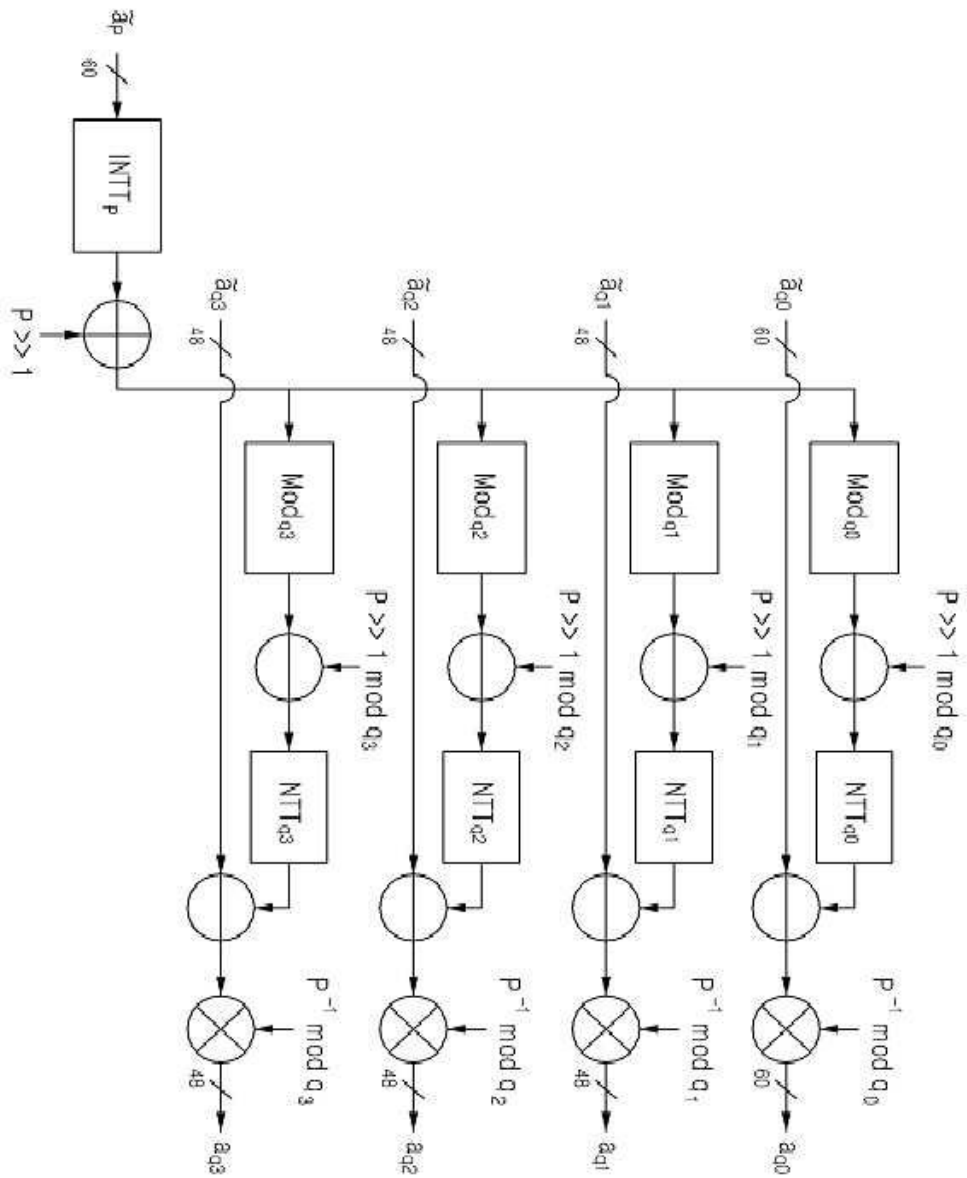
도면1



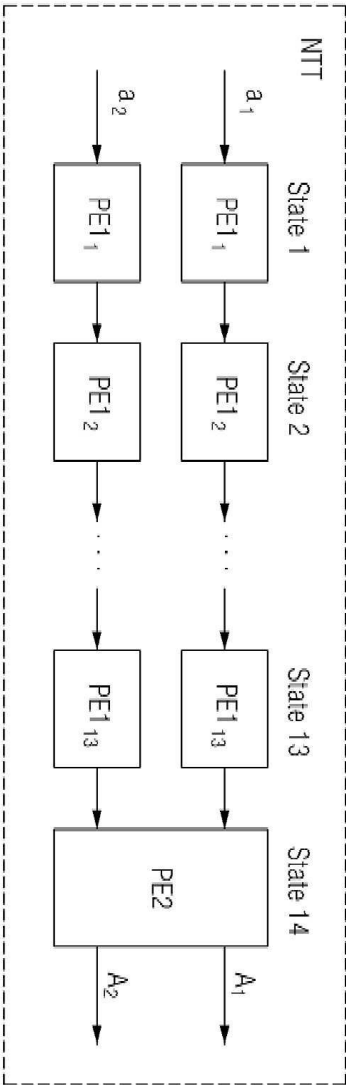
도면2



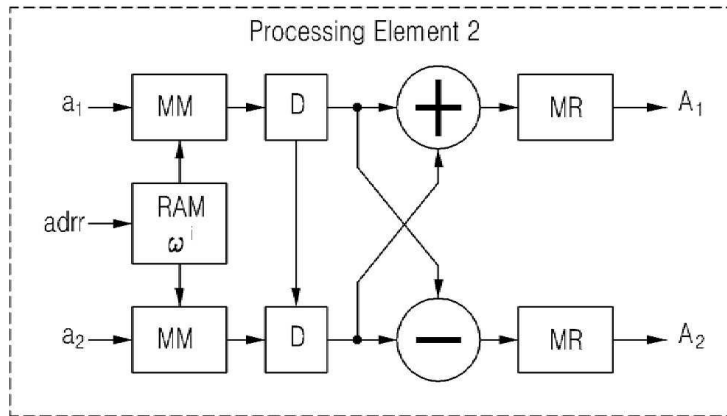
도면3



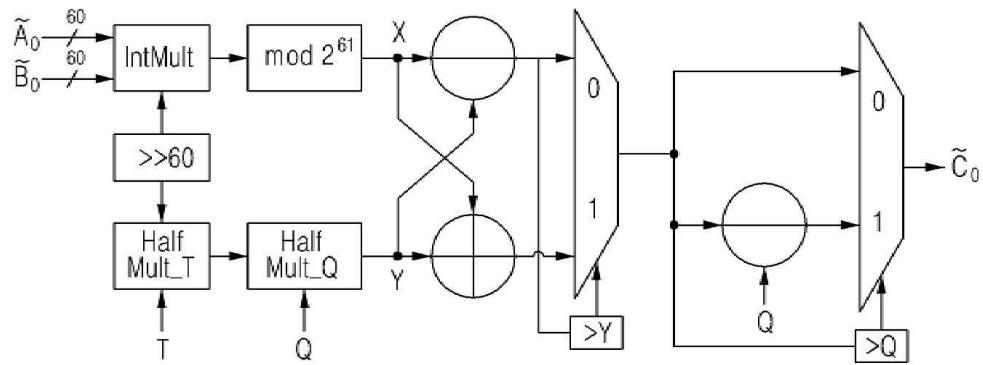
도면4



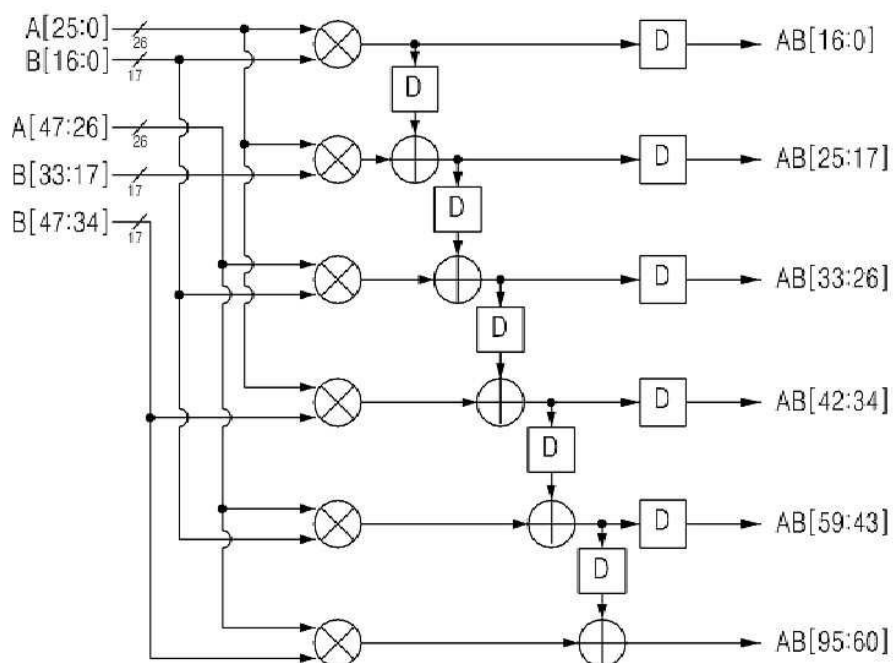
도면7



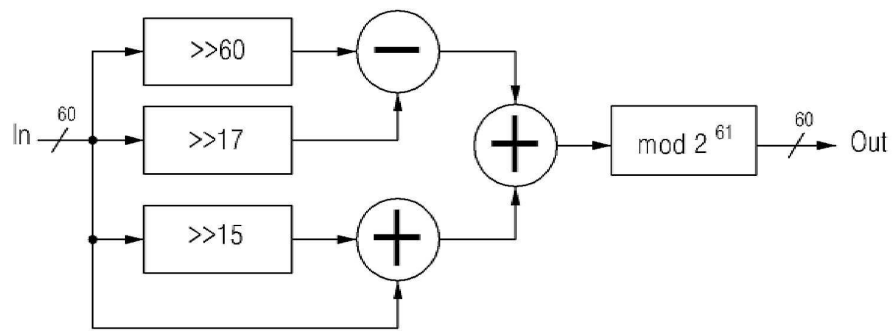
도면8



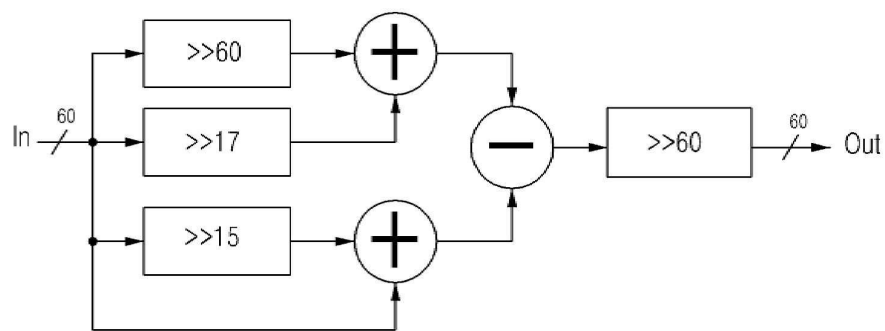
도면9



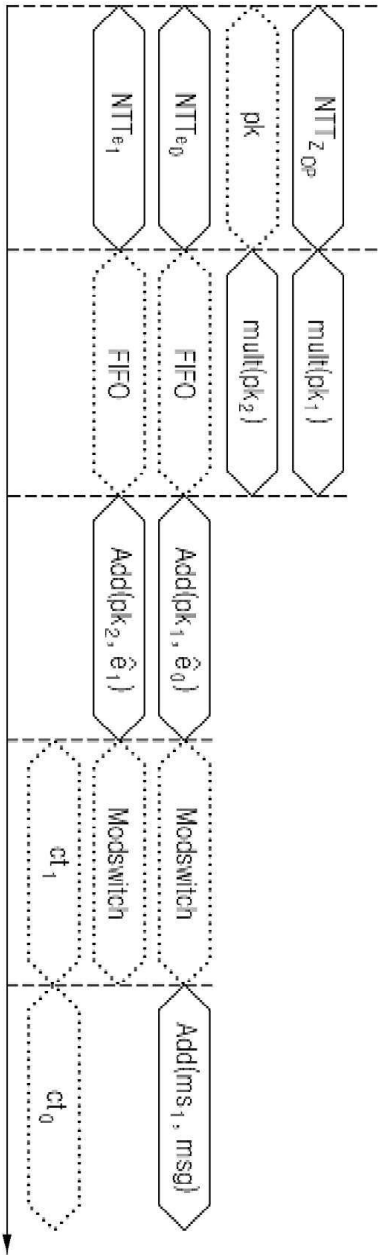
도면10



도면11



도면12



도면13

