(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0134159
(43) 공개일자 2022년10월05일

(51) 국제특허분류(Int. Cl.)
H04L 9/30 (2006.01) *H04L 9/06* (2006.01)
H04L 9/08 (2006.01)
(52) CPC특허분류
H04L 9/3026 (2013.01)
H04L 9/06 (2013.01)
(21) 출원번호 10-2021-0039458
(22) 출원일자 2021년03월26일
심사청구일자 2021년03월26일

(71) 출원인
인하대학교 산학협력단
인천광역시 미추홀구 인하로 100(용현동, 인하대학교)
(72) 발명자
이한호
인천광역시 연수구 컨벤시아대로252번길 30, 1505동 1701호(송도동, 송도 더샵 퍼스트파크 F15BL)
김용진
인천광역시 계양구 계산로 186, 107동 805호(계산동, 현대아파트)
두옹옥팝
인천광역시 미추홀구 인하로119번길 40, 102호 (용현동)
(74) 대리인
양성보

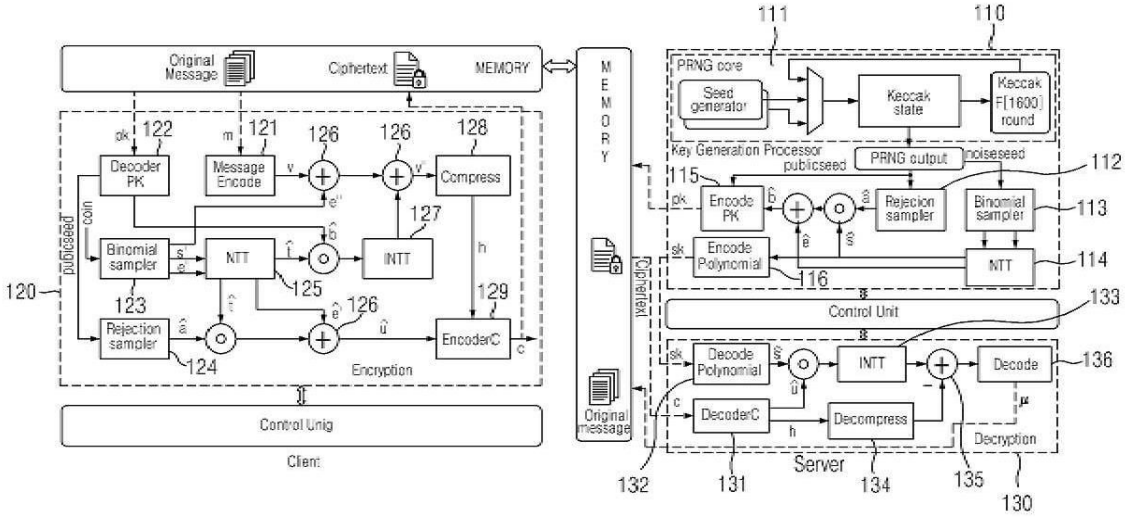
전체 청구항 수 : 총 8 항

(54) 발명의 명칭 **MDF 기반 NTT를 이용한 링-LWE 암호프로세서 장치 및 방법**

(57) 요 약

MDF 기반 NTT를 이용한 링-LWE 암호화 방법 및 장치가 제시된다. 본 발명에서 제안하는 MDF 기반 NTT를 이용한 링-LWE 암호화 장치는 PRNG 코어로부터 데이터를 입력 받아 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하고, NTT 연산을 수행한 뒤 공개키 및 비밀키를 생성하여 메모리에 저장하는 키생성 프로세서, 메모리로부터 저장된 공개키와 데이터를 수신하여 변형시키고, 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 및 INTT 연산을 수행한 뒤 암호문의 압축 과정을 거친 후, 암호문을 출력하는 암호화 프로세서 및 메모리로부터 저장된 비밀키와 암호문을 수신하여 암호문을 복원하고, 비밀키를 변형시키며, INTT 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 원본 메시지를 최종 출력하는 복호화 프로세서를 포함한다.

대 표 도



(52) CPC특허분류
H04L 9/0825 (2013.01)
H04L 2209/122 (2013.01)
H04L 2209/125 (2013.01)
H04L 2209/24 (2013.01)
H04L 2209/30 (2013.01)

이 발명을 지원한 국가연구개발사업
과제고유번호 1121011232
과제번호 2021R1A2C1011232
부처명 과학기술정보통신부
과제관리(전문)기관명 한국연구재단
연구사업명 기초연구사업
연구과제명 고신뢰성 고성능 격자-기반 양자내성암호 하드웨어 아키텍처 연구
기 여 율 1/2
과제수행기관명 인하대학교 산학협력단
연구기간 2021.03.01 ~ 2025.02.28
이 발명을 지원한 국가연구개발사업
과제고유번호 1415172944
과제번호 20010589
부처명 산업통상자원부
과제관리(전문)기관명 산업기술평가관리원
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

명 세 서

청구범위

청구항 1

PRNG 코어로부터 데이터를 입력 받아 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하고, NTT 연산을 수행한 뒤 공개키 및 비밀키를 생성하여 메모리에 저장하는 키생성 프로세서;

메모리로부터 저장된 공개키와 데이터를 수신하여 변형시키고, 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 및 INTT 연산을 수행한 뒤 암호문의 압축 과정을 거친 후, 암호문을 출력하는 암호화 프로세서; 및

메모리로부터 저장된 비밀키와 암호문을 수신하여 암호문을 복원하고, 비밀키를 변형시키며, INTT 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 원본 메시지를 최종 출력하는 복호화 프로세서

를 포함하는 암호화 장치.

청구항 2

제1항에 있어서,

키생성 프로세서는,

PRNG 코어로부터 데이터를 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성하는 거부 샘플러;

PRNG 코어로부터 데이터를 입력 받아 비밀키 생성과 암호화에 필요한 에러를 생성하는 이항 샘플러;

이항 샘플러의 출력을 입력 받아 NTT 연산을 수행하는 NTT 코어;

거부 샘플러 및 NTT 코어의 출력을 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 입력 받아 공개키를 생성하여 메모리에 저장하는 공개키 인코더; 및

NTT 코어의 출력을 입력 받아 데이터를 변형시켜 비밀키를 생성하여 메모리에 저장하는 다항식 인코더

를 포함하는 암호화 장치.

청구항 3

제1항에 있어서,

암호화 프로세서는,

메모리로부터 저장된 메세지를 수신하여 데이터를 변형시키는 메시지 인코더;

메모리로부터 저장된 공개키를 수신하여 공개키를 변형시키는 공개키 디코더;

공개키 디코더의 출력을 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성하는 거부 샘플러;

비밀키 생성과 암호화에 필요한 에러를 생성하는 이항 샘플러;

이항 샘플러의 출력을 입력 받아 NTT 연산을 수행하는 NTT 코어;

NTT 코어 및 공개키 디코더의 출력을 점 곱셈 연산한 값을 입력 받아 역-NTT 연산을 수행하는 INTT 코어;

이항 샘플러 및 메시지 인코더의 출력을 다항식 덧셈한 후 INTT 코어의 출력과 다항식 덧셈을 통해 각각의 계수를 더한 값을 입력 받아 암호문을 압축하는 컴프레스; 및

거부 샘플러 및 NTT 코어의 출력을 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 입력 받아 암호문을 출력하는 암호문 인코더

를 포함하는 암호화 장치.

청구항 4

제1항에 있어서,

복호화 프로세서는,

메모리로부터 저장된 암호문을 수신하여 암호문을 복원하는 암호문 디코더;

메모리로부터 저장된 비밀키를 수신하여 비밀키를 변형시키는 다항식 디코더;

암호문 디코더 및 다항식 디코더의 출력을 점 곱셈 연산한 값을 입력 받아 역-NTT 연산을 수행하는 INTT 코어;

암호문 디코더의 출력을 입력 받아 복원된 암호문을 압축해제하는 디컴프레스; 및

INTT 코어 및 디컴프레스의 출력을 다항식 뺄셈한 값을 입력 받아 원본 메시지를 최종 출력하는 디코더

를 포함하는 암호화 장치.

청구항 5

키생성 프로세서를 통해 PRNG 코어로부터 데이터를 입력 받아 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하고, NTT 연산을 수행한 뒤 공개키 및 비밀키를 생성하여 메모리에 저장하는 단계;

암호화 프로세서를 통해 메모리로부터 저장된 공개키와 데이터를 수신하여 변형시키고, 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 및 INTT 연산을 수행한 뒤 암호문의 압축 과정을 거친 후, 암호문을 출력하는 단계; 및

복호화 프로세서를 통해 메모리로부터 저장된 비밀키와 암호문을 수신하여 암호문을 복원하고, 비밀키를 변형시키며, INTT 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 원본 메시지를 최종 출력하는 단계

를 포함하는 암호화 방법.

청구항 6

제5항에 있어서,

키생성 프로세서를 통해 PRNG 코어로부터 데이터를 입력 받아 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하고, NTT 연산을 수행한 뒤 공개키 및 비밀키를 생성하여 메모리에 저장하는 단계는,

거부 샘플러를 통해 PRNG 코어로부터 데이터를 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성하고,

이항 샘플러를 통해 PRNG 코어로부터 데이터를 입력 받아 비밀키 생성과 암호화에 필요한 에러를 생성하고,

NTT 코어를 통해 이항 샘플러의 출력을 입력 받아 NTT 연산을 수행하고,

거부 샘플러 및 NTT 코어의 출력을 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 공개키 인코더를 통해 입력 받아 공개키를 생성하여 메모리에 저장하고,

다항식 인코더를 통해 NTT 코어의 출력을 입력 받아 데이터를 변형시켜 비밀키를 생성하여 메모리에 저장하는

암호화 방법.

청구항 7

제5항에 있어서,

암호화 프로세서를 통해 메모리로부터 저장된 공개키와 데이터를 수신하여 변형시키고, 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 및 INTT 연산을 수행한 뒤 암호문의 압축 과정을 거친 후, 암호문을 출력하는 단계는,

메시지 인코더를 통해 메모리로부터 저장된 메세지를 수신하여 데이터를 변형시키고,

공개키 디코더를 통해 메모리로부터 저장된 공개키를 수신하여 공개키를 변형시키고,

거부 샘플러를 통해 공개키 디코더의 출력을 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성하고,

이항 샘플러를 통해 비밀키 생성과 암호화에 필요한 에러를 생성하고,

NTT 코어를 통해 이항 샘플러의 출력을 입력 받아 NTT 연산을 수행하고,

NTT 코어 및 공개키 디코더의 출력을 점 곱셈 연산한 값을 INTT 코어를 통해 입력 받아 역-NTT 연산을 수행하고,

이항 샘플러 및 메시지 인코더의 출력을 다항식 덧셈한 후 INTT 코어의 출력과 다항식 덧셈을 통해 각각의 계수를 더한 값을 컴프레스를 통해 입력 받아 암호문을 압축하고,

거부 샘플러 및 NTT 코어의 출력을 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 암호문 인코더를 통해 입력 받아 암호문을 출력하는

암호화 방법.

청구항 8

제5항에 있어서,

복호화 프로세서를 통해 메모리로부터 저장된 비밀키와 암호문을 수신하여 암호문을 복원하고, 비밀키를 변형시키며, INTT 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 원본 메시지를 최종 출력하는 단계는,

암호문 디코더를 통해 메모리로부터 저장된 암호문을 수신하여 암호문을 복원하고,

다항식 디코더를 통해 메모리로부터 저장된 비밀키를 수신하여 비밀키를 변형시키고,

암호문 디코더 및 다항식 디코더의 출력을 점 곱셈 연산한 값을 INTT 코어를 통해 입력 받아 역-NTT 연산을 수행하고,

디컴프레스를 통해 암호문 디코더의 출력을 입력 받아 복원된 암호문을 압축해제하고,

INTT 코어 및 디컴프레스의 출력을 다항식 뺄셈한 값을 디코더를 통해 입력 받아 원본 메시지를 최종 출력하는

암호화 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 NTT(Number Theoretic Transform) 코어를 이용한 높은 데이터 처리율과 낮은 복잡도를 가지는 Ring-LWE(Learning With Error) 암호프로세서 장치 및 방법에 관한 것이다.

배 경 기 술

[0002] RSA(Rivest, Shamir, and Adleman)와 ECC(Elliptic Curve Cryptosystem) 암호시스템은 이산로그나 인수분해와 같이 다항시간에 풀 수 없는 어려운 문제들에 기반을 두어 설계되었다. 그러나, 이러한 문제들은 Shor가 제안한 양자컴퓨터를 사용하는 알고리즘을 사용하면 다항시간 내에 해결될 수 있다. 따라서, 보다 강력한 암호시스템 혹은 차세대 양자 암호시스템이 요구되고 있으며, NIST(National Institute of Standards and Technology)에서 표준화를 진행 중에 있다. 미래의 양자컴퓨터 시대를 대비한 양자내성암호로 꼽히는 격자-기반 암호 알고리즘은 주로 LWE(Learning With Error) 기법을 사용한다. LWE는 격자-기반 암호 구현 시 에러를 삽입하여 행렬식의 해를 찾기 어렵도록 하는 기법이다. 하지만 계산시에 키가 커지는 단점과 연산속도 문제점 때문에 환(Ring) 안에서 계산하는 Ring-LWE(Learning With Error) 방법이 제안되었다.

[0003] Ring-LWE 암호는 격자-기반 암호 알고리즘으로서 Ring-LWE 기법을 사용한다. Ring-LWE는 환(Ring) 안에서 계산되며 다항식 곱셈이 주로 쓰이는데 가장 많은 시간자원을 사용하므로 다항식 연산속도를 향상시켜주는 고성능 NTT를 필요로 한다.

선행기술문헌

특허문헌

[0004] (특허문헌 0001) 한국 등록특허공보 제10-2040106호(2019.10.29)

발명의 내용

해결하려는 과제

[0005] 본 발명이 이루고자 하는 기술적 과제는 높은 데이터 처리율과 낮은 연산 복잡도를 가지는 Ring-LWE 기반 암호 시스템을 위해, 4-병렬 MDF(Multi-path Delay Feedback) 기반 NTT 코어를 사용하는 효율적인 방법 및 장치를 제공하는데 있다. 또한 보안수준을 높이기 위해 파라미터 n=1024을 기반으로 설계되었다. 또한 암호학적 해쉬 알고리즘인 SHA-3(Secure Hash Algorithm-3)를 이용하여 알고리즘의 출력 길이를 확장할 수 있는 함수인 SHAKE128과 SHAKE256을 사용하고 샘플링 과정에 대한 효율적 구현의 어려움과 타이밍 공격으로부터 보호하기 위해 중심 이항 분포(Centered Binomial Distribution)를 에러 분포로 사용하는 방법과 하드웨어 구조를 제안한다. 또한, NTT 다항식 곱셈기의 하드웨어 복잡성을 줄이고 클럭 속도를 높이기 위해 4-병렬 MDF 구조, 파이프라인 구조, 그리고 비트-역순(Bit-Reverse) 연산을 제거하는 방법을 제안한다.

과제의 해결 수단

[0006] 일 측면에 있어서, 본 발명에서 제안하는 MDF 기반 NTT를 이용한 링-LWE 암호화 장치는 PRNG 코어로부터 데이터를 입력 받아 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하고, NTT 연산을 수행한 뒤 공개키 및 비밀키를 생성하여 메모리에 저장하는 키생성 프로세서, 메모리로부터 저장된 공개키와 데이터를 수신하여 변형시키고, 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 및 INTT 연산을 수행한 뒤 암호문의 압축 과정을 거친 후, 암호문을 출력하는 암호화 프로세서 및 메모리로부터 저장된 비밀키와 암호문을 수신하여 암호문을 복원하고, 비밀키를 변형시키며, INTT 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 원본 메시지를 최종 출력하는 복호화 프로세서를 포함한다.

[0007] 키생성 프로세서는 PRNG 코어로부터 데이터를 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성하는 거부 샘플러, PRNG 코어로부터 데이터를 입력 받아 비밀키 생성과 암호화에 필요한 에러를 생성하는 이항 샘플러, 이항 샘플러의 출력을 입력 받아 NTT 연산을 수행하는 NTT 코어, 거부 샘플러 및 NTT 코어의 출력을 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 입력 받아 공개키를 생성하여 메모리에 저장하는 공개키 인코더 및 NTT 코어의 출력을 입력 받아 데이터를 변형시켜 비밀키를 생성하여 메모리에 저장하는 다항식 인코더를 포함한다.

[0008] 암호화 프로세서는 메모리로부터 저장된 메세지를 수신하여 데이터를 변형시키는 메시지 인코더, 메모리로부터 저장된 공개키를 수신하여 공개키를 변형시키는 공개키 디코더, 공개키 디코더의 출력을 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성하는 거부 샘플러, 비밀키 생성과 암호화에 필요한 에러를 생성하는 이항 샘플러, 이항 샘플러의 출력을 입력 받아 NTT 연산을 수행하는 NTT 코어, NTT 코어 및 공개키 디코더의 출력을 점 곱셈 연산한 값을 입력 받아 역-NTT 연산을 수행하는 INTT 코어, 이항 샘플러 및 메시지 인코더의 출력을 다항식 덧셈한 후 INTT 코어의 출력과 다항식 덧셈을 통해 각각의 계수를 더한 값을 입력 받아 암호문을 압축하는 컴프레스 및 거부 샘플러 및 NTT 코어의 출력을 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 입력 받아 암호문을 출력하는 암호문 인코더를 포함한다.

[0009] 복호화 프로세서는 메모리로부터 저장된 암호문을 수신하여 암호문을 복원하는 암호문 디코더, 메모리로부터 저장된 비밀키를 수신하여 비밀키를 변형시키는 다항식 디코더, 암호문 디코더 및 다항식 디코더의 출력을 점 곱셈 연산한 값을 입력 받아 역-NTT 연산을 수행하는 INTT 코어, 암호문 디코더의 출력을 입력 받아 복원된 암호문을 압축해제하는 디컴프레스 및 INTT 코어 및 디컴프레스의 출력을 다항식 뺄셈한 값을 입력 받아 원본 메시지를 최종 출력하는 디코더를 포함한다.

[0010] 또 다른 일 측면에 있어서, 본 발명에서 제안하는 MDF 기반 NTT를 이용한 링-LWE 암호화 방법은 키생성 프로세서를 통해 PRNG 코어로부터 데이터를 입력 받아 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하고, NTT 연산을 수행한 뒤 공개키 및 비밀키를 생성하여 메모리에 저장하는 단계, 암호화 프로세서를 통해 메모리로부터 저장된 공개키와 데이터를 수신하여 변형시키고, 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 및 INTT 연산을 수행한 뒤 암호문의 압축 과정을 거친 후, 암호문을 출력하는 단계 및 복호화 프로세서를 통해 메모리로부터 저장된 비밀키와 암호문을 수신하여 암호문을 복원하고, 비밀키를 변형시키며, INTT 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 원본 메시지를 최종 출력하는 단계를 포함한다.

발명의 효과

[0011] 본 발명의 실시예들에 따르면 Ring-LWE 암호프로세서에 있어서 에러를 생성하는 샘플링 과정에서 효율적 구현과 타이밍 공격으로부터 보호하기 위해 중심 이항 분포를 에러 분포로 사용하며, 병렬 처리를 통해서 지연 시간을 줄이고 높은 데이터 처리량을 가질 수 있다. 또한, NTT 코어에 있어서 파이프라인을 이용한 4-병렬 MDF-기반으로 설계하여 높은 클럭 속도를 제공할 수 있다.

도면의 간단한 설명

[0012] 도 1은 본 발명의 일 실시예에 따른 MDF 기반 NTT를 이용한 Ring-LWE 암호화 장치의 구성을 나타내는 도면이다.

도 2는 본 발명의 일 실시예에 따른 키생성 프로세서를 설명하기 위한 도면이다.

도 3은 본 발명의 일 실시예에 따른 암호화 프로세서를 설명하기 위한 도면이다.

도 4는 본 발명의 일 실시예에 따른 복호화 프로세서를 설명하기 위한 도면이다.

도 5는 본 발명의 일 실시예에 따른 거부 샘플러를 설명하기 위한 도면이다.

도 6은 본 발명의 일 실시예에 따른 이항 샘플러를 설명하기 위한 도면이다.

도 7은 본 발명의 일 실시예에 따른 4-병렬 MDF-기반 NTT 코어의 구조를 나타내는 도면이다.

도 8은 본 발명의 일 실시예에 따른 NTT 코어의 PE1(Processing Element 1) 구조를 나타내는 도면이다.

도 9는 본 발명의 일 실시예에 따른 NTT 코어의 PE2(Processing Element 1) 구조를 나타내는 도면이다.

도 10은 본 발명의 일 실시예에 따른 NTT 코어의 MR(Modulo Reduction) 구조를 나타내는 도면이다.

도 11은 본 발명의 일 실시예에 따른 MDF 기반 NTT를 이용한 Ring-LWE 암호화 방법을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0013] 본 발명은 NTT(Number Theoretic Transform Core) 코어를 이용한 높은 데이터 처리율과 낮은 복잡도를 가지는 Ring-LWE(Learning With Error) 암호프로세서 장치 및 방법에 관한 것으로, 더욱 상세하게는, 파이프라인 구조를 위한 방식 중 높은 데이터 처리율 대비 낮은 하드웨어 면적을 갖는 MDF(Multi-path Delay Feedback) 방식을 적용하여 Ring-LWE 암호프로세서의 면적은 줄이고 데이터 처리율은 높이는 방법에 관한 것이다.

[0014] Ring-LWE 암호는 격자-기반 암호 알고리즘으로서 Ring-LWE 기법을 사용한다. Ring-LWE는 환(Ring) 안에서 계산되며 다항식 곱셈이 주로 쓰이는데 가장 많은 시간자원을 사용하므로 다항식 연산속도를 향상시켜주는 고성능 NTT를 필요로 한다. 제안된 고성능 NTT 아키텍처는 높은 데이터 처리율을 얻기 위해 파이프라인이 가능한 구조인 4-병렬 MDF(Multi-path Delay Feedback) 구조 방법을 사용하였다. 이하, 본 발명의 실시 예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0016] 도 1은 본 발명의 일 실시예에 따른 MDF 기반 NTT를 이용한 Ring-LWE 암호화 장치의 구성을 나타내는 도면이다.

[0017] 제안하는 MDF 기반 NTT를 이용한 Ring-LWE 암호화 장치는 키생성 프로세서(110), 암호화 프로세서(120) 및 복호화 프로세서(130)를 포함한다.

[0018] 키생성 프로세서(110)는 PRNG 코어로부터 데이터를 입력 받아 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하고, NTT 연산을 수행한 뒤 공개키 및 비밀키를 생성하여 메모리에 저장한다.

[0019] 본 발명의 실시예에 따른 키생성 프로세서(110)는 PRNG 코어(111), 거부 샘플러(112), 이항 샘플러(113), NTT 코어(114), 공개키 인코더(115) 및 다항식 인코더(116)를 포함할 수 있다.

[0020] 본 발명의 실시예에 따른 PRNG 코어(111)는 해쉬 알고리즘으로 거부 샘플러(112)와 이항 샘플러(113)의 입력에 사용되는 장치이다.

[0021] 본 발명의 실시예에 따른 거부 샘플러(112)는 PRNG 코어(111)로부터 데이터를 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성할 수 있다.

[0022] 본 발명의 실시예에 따른 이항 샘플러(113)는 PRNG 코어(111)로부터 데이터를 입력 받아 비밀키 생성과 암호화에 필요한 에러를 생성할 수 있다.

[0023] 본 발명의 실시예에 따른 NTT 코어(114)는 파라미터 n =1024 다항식을 수신할 수 있는 입력에 대해 NTT 연산을 수행하는 4-병렬 MDF-기반 NTT 코어 장치일 수 있다.

[0024] 본 발명의 실시예에 따른 다항식 인코더(116)는 이항 샘플러의 결과인 다항식을 수신하여 암호화에 사용할 수 있는 비밀키로 만들어주는 인코더이다.

[0025] 본 발명의 실시예에 따른 키생성 프로세서(110)에 대하여 도 2를 참조하여 더욱 상세히 설명한다.

[0027] 암호화 프로세서(120)는 메모리로부터 저장된 공개키와 데이터를 수신하여 변형시키고, 메모리로부터 받아온 공개키를 이용하여 필요한 에러를 생성하며, NTT 및 INTT 연산을 수행한 뒤 암호문의 압축 과정을 거친 후, 암호문을 출력한다.

[0028] 본 발명의 실시예에 따른 암호화 프로세서(120)는 메시지 인코더(121), 공개키 디코더(122), 이항 샘플러(123), 거부 샘플러(124), NTT 코어(125), 덧셈기(126) INTT 코어(127), 컴프레스(128) 및 암호문 인코더(129)를 포함한다.

[0029] 본 발명의 실시예에 따른 메시지 인코더(121)는 데이터를 수신하여 암호화 프로세서(120)에 적합하게 데이터를 변형시켜주는 인코더이다.

[0030] 본 발명의 실시예에 따른 공개키 디코더(122)는 공개키에 대한 다항식을 수신하여 암호화에 적합하게 데이터의 변형시켜주는 디코더이다.

[0031] 본 발명의 실시예에 따른 INTT 코어(126)는 파라미터 n =1024 다항식을 수신할 수 있는 입력에 대해 역 NTT 연산을 수행하는 4-병렬 MDF-기반 INTT 코어일 수 있다.

[0032] 본 발명의 실시예에 따른 컴프레스(128)는 암호화된 데이터의 크기를 줄여주는 구조이다.

[0033] 본 발명의 실시예에 따른 암호문 인코더(129)는 다항식을 수신하여 암호화에 적합하게 데이터를 변형시켜주는 인코더이다.

[0034] 본 발명의 실시예에 따른 암호화 프로세서(120)에 대하여 도 3를 참조하여 더욱 상세히 설명한다.

[0036] 복호화 프로세서(130)는 메모리로부터 저장된 비밀키와 암호문을 수신하여 암호문을 복원하고, 비밀키를 변형시키며, INTT 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 원본 메시지를 최종 출력한다.

[0037] 본 발명의 실시예에 따른 복호화 프로세서(130)는 암호문 디코더(131), 다항식 디코더(132), INTT 코어(133), 디컴프레스(134), 뺄셈기(135) 및 디코더(136)를 포함한다.

[0038] 본 발명의 실시예에 따른 디컴프레스(134)는 압축과정에서 줄어든 데이터의 크기를 복원하는 구조이다.

[0039] 본 발명의 실시예에 따른 디코더(136)는 복호화 프로세서의 연산 값을 변형하여 최종적으로 복호화된 데이터를 출력하는 장치일 수 있다.

[0040] 본 발명의 실시예에 따른 복호화 프로세서(130)에 대하여 도 4를 참조하여 더욱 상세히 설명한다.

[0042] 도 1를 참조하면, 키 생성과정으로 PRNG 코어를 통해서 생성된 시드(Seed)를 이항 샘플링과정과 거부 샘플링과정으로 출력하고 인코딩 과정을 통해서 최종적으로 공개키와 비밀키를 생성한다. 암호화 과정은 키 생성 모듈에서 공개키를 전달 받고 원하는 정보를 암호화하는 과정으로 전달 받은 공개키를 디코딩을 통해서 Publicseed와 $\hat{b}$ 으로 출력하고 키 생성 과정과 동일하게 이항 샘플링과정과 거부 샘플링과정을 갖는다. 원하는 정보를 메시지 인코더를 통해서 타인이 확인할 수 없게 암호화하고 암호문의 크기를 줄이는 압축 과정을 갖는다. 최종적으로 다항식 인코더를 통해서 암호문을 출력으로 갖는다. 복호화 구조는 암호문과 비밀키를 전달받아 연산을 수행한다. 암호문은 디코더 과정을 통해서 $\hat{u}$ 와 h 로 분리되고 h 는 압축해제를 통해서 v' 로 분할된다. 비밀키는 다항식

디코더을 통해서 $\hat{s}$를 출력하고 $\hat{u}$와 연산을 하며 디코딩 과정을 통해서 원본 정보로 복원된다.

[0043] Ring-LWE의 모든 연산은 $R_q = Z_q[x]/f(x)$의 조건을 만족시키면서 동작하므로 다항식 a(x)와 b(x)는 수학식 1과 같이 나타낼 수 있다.

$$a(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1}$$

[0044] $$b(x) = b_0 + b_1x + b_2x^2 + \ldots + b_{n-1}x^{n-1}$$ 수학식 1

[0045] 다항식 곱셈기에 사용되는 NTT 연산은 수학식 2와 같이 나타낼 수 있다.

$$A_i = \sum_{j=0}^{n-1} a_j w_n^{ij} \bmod q, \text{ for } i = 0, 1, \ldots, n-1$$

[0046] 수학식 2

[0047] 벡터(Vector) $a = (a_0, \ldots, a_{n-1})$는 다항식 a(x)의 계수이다. n은 다항식의 계수로 1의 n 제곱근(root of unity)의 값을 가진다. 또한w는 위의 수학식에서 사용되는 각 다항식마다 곱해지는 회전인자(twiddle factor)이며, n과 w는 $n * n^{-1} \equiv 1 \bmod q$, $w * w^{-1} \equiv 1 \bmod q$의 조건을 만족하는 숫자로 구성된다. q는 Ring-LWE 암호에서 사용되는 소수이다.

[0048] 그리고 다항식 곱셈기에 사용되는 INTT(Inverse Number Theoretic Transform) 연산은 수학식 3과 같이 나타낸다.

$$a_i = n^{-1} \sum_{j=0}^{n-1} A_j w_n^{-ij} \bmod q, \text{ for } i = 0, 1, \ldots, n-1$$

[0049] 수학식 3

[0050] 이항 샘플러는 Ring-LWE의 에러 분포에 대해 매개 변수 k=8의 중심 이항분포 k를 사용하며 연산은 수학식 4와 같이 나타낸다.

[0051] $$(b_i, b_i' \in \{0,1\}) \rightarrow R_q = \sum_{i=0}^{k-1} b_i - b_i'$$ 수학식 4

[0053] 도 2는 본 발명의 일 실시예에 따른 키생성 프로세서를 설명하기 위한 도면이다.

[0054] 본 발명의 실시예에 따른 키생성 프로세서(200)는 PRNG 코어(210), 거부 샘플러(220), 이항 샘플러(230), NTT 코어(240), 공개키 인코더(250) 및 다항식 인코더(260)를 포함할 수 있다.

[0055] 거부 샘플러(220)는 PRNG 코어로부터 데이터를 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성한다.

[0056] 이항 샘플러(230)는 PRNG 코어로부터 데이터를 입력 받아 비밀키 생성과 암호화에 필요한 에러를 생성한다.

[0057] NTT 코어(240)는 이항 샘플러의 출력을 입력 받아 NTT 연산을 수행한다.

[0058] 공개키 인코더(250)는 거부 샘플러 및 NTT 코어의 출력을 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 입력 받아 공개키를 생성하여 메모리에 저장한다.

[0059] 다항식 인코더(260)는 NTT 코어의 출력을 입력 받아 데이터를 변형시켜 비밀키를 생성하여 메모리에 저장한다.

[0060] 본 발명의 실시예에 따르면, PRNG 코어(210)는 SHAKE128과 SHAKE256의 알고리즘 구현에 사용되며 SHAKE는 SHA-3 Keccak알고리즘 중 XOF으로 출력이 가변 길이를 가지는 해쉬 알고리즘이다. PRNG 코어(210)를 통해서 생성된 PRNG 출력이 각각 거부 샘플러(220)와 이항 샘플러(230)로 입력이 되고 이항 샘플러(230)의 출력은 각각 e와 s가 되어 NTT 코어(240)에 입력된다. NTT 코어(240)의 결과로 나온 e와 s는 거부 샘플러(220)의 출력 값과 함께 공개키 인코더(250)에 입력되고 출력으로 공개키를 생성하며 s는 따로 나와서 다항식 인코더(260)를 통해 비밀키를 생성한다.

[0062] 도 3은 본 발명의 일 실시예에 따른 암호화 프로세서를 설명하기 위한 도면이다.

[0063] 본 발명의 실시예에 따른 암호화 프로세서(300)는 메시지 인코더(310), 공개키 디코더(320), 이항 샘플러(330), 거부 샘플러(340), NTT 코어(350), 점 곱셈기(351, 352), 덧셈기(360), INTT 코어(370), 컴프레스(380) 및 암호문 인코더(390)를 포함한다.

[0064] 메시지 인코더(310)는 메모리로부터 저장된 메세지를 수신하여 데이터를 변형시킨다.

[0065] 공개키 디코더(320)는 메모리로부터 저장된 공개키를 수신하여 공개키를 변형시킨다.

[0066] 거부 샘플러(340)는 공개키 디코더의 출력을 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성한다.

[0067] 이항 샘플러(330)는 비밀키 생성과 암호화에 필요한 에러를 생성한다.

[0068] NTT 코어(350)는 이항 샘플러(330)의 출력을 입력 받아 NTT 연산을 수행한다.

[0069] INTT 코어(370)는 NTT 코어(350) 및 공개키 디코더(320)의 출력을 점 곱셈기(351)를 통해 점 곱셈 연산한 값을 입력 받아 역-NTT 연산을 수행한다.

[0070] 컴프레스(380)는 이항 샘플러(330) 및 메시지 인코더(310)의 출력을 덧셈기(360)를 통해 다항식 덧셈한 후 INTT 코어(370)의 출력과 다항식 덧셈을 통해 각각의 계수를 더한 값을 입력 받아 암호문을 압축한다.

[0071] 암호문 인코더(390)는 거부 샘플러(340) 및 NTT 코어(350)의 출력을 점 곱셈기(352)를 통해 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 입력 받아 암호문을 출력한다.

[0072] 본 발명의 실시예에 따르면, 원본 메시지를 입력 받아 메시지 인코더를 통해 인코딩되고 키생성 프로세서에서 전달 받은 공개키를 디코딩을 통해 publicseed와 $\hat{b}$ 으로 출력하고 키생성 프로세서에서의 과정과 동일하게 이항 샘플링과정과 거부 샘플링과정을 갖는다. NTT 코어와 점 곱셈 연산을 한 후 다항식 덧셈을 통해서 각각의 계수를 더하고 암호문의 크기를 줄이는 압축 과정을 갖는다. 최종적으로 암호문 인코더를 통해 암호문이 출력된다.

[0074] 도 4는 본 발명의 일 실시예에 따른 복호화 프로세서를 설명하기 위한 도면이다.

[0075] 본 발명의 실시예에 따른 복호화 프로세서(400)는 암호문 디코더(410), 다항식 디코더(420), 점 곱셈기(421), INTT 코어(430), 디컴프레스(440), 뺄셈기(441) 및 디코더(450)를 포함한다.

[0076] 암호문 디코더(410)는 메모리로부터 저장된 암호문을 수신하여 암호문을 복원한다.

[0077] 다항식 디코더(420)는 메모리로부터 저장된 비밀키를 수신하여 비밀키를 변형시킨다.

[0078] INTT 코어(430)는 암호문 디코더(410) 및 다항식 디코더(420)의 출력을 점 곱셈기(421)를 통해 점 곱셈 연산한 값을 입력 받아 역-NTT 연산을 수행한다.

[0079] 디컴프레스(440)는 암호문 디코더(410)의 출력을 입력 받아 복원된 암호문을 압축해제한다.

[0080] 디코더(450)는 INTT 코어(430) 및 디컴프레스(440)의 출력을 뺄셈기(441)를 통해 다항식 뺄셈한 값을 입력 받아 원본 메시지를 최종 출력한다.

[0081] 본 발명의 실시예에 따르면, 복호화 프로세서(400)는 비밀키를 이용해서 전달 받은 암호문을 원래 정보로 복원하는 과정을 가지며 전달 받은 암호문과 비밀키는 다항식 디코더를 거치고 나서 점 곱셈연산을 한 뒤에 INTT 코어에 입력된다. 입력된 값은 다시 압축해제 과정을 거친 암호문과 각각의 다항식의 계수 뺄셈 연산을 하고 디코더를 통해 최종적으로 원본 메시지를 출력한다.

[0083] 도 5는 본 발명의 일 실시예에 따른 거부 샘플러를 설명하기 위한 도면이다.

[0084] 본 발명의 일 실시예에 따른 거부 샘플러는 처음 입력 값을 통해서 PRNG 코어의 출력을 받고 하위 16비트씩 MUX에 입력되며 5q 보다 작으면 RAM에 해당하는 값을 저장하고 값이 5q 보다 크면 저장하지 않는다.

[0085] 본 발명의 일 실시예에 따르면, 해당 동작을 계속해서 64번 반복한 뒤에 RAM의 주소 값이 64보다 크면 IN 값으

로 다시 PRNG 코어 연산을 하고 그렇지 않으면 PRNG 코어의 출력이 다시 입력으로 들어가게 된다. 해당 동작은 총 1024개의 16비트가 저장될 때까지 계속해서 연산을 반복할 수 있다.

[0087] 도 6은 본 발명의 일 실시예에 따른 이항 샘플러를 설명하기 위한 도면이다.

[0088] 본 발명의 일 실시예에 따른 이항 샘플러는 PRNG 출력으로 얻은 1024비트를 32비트씩 순차적으로 연산을 하게 되며 4-병렬 구조로 설계되었다. 이항 샘플러는 Ring-LWE의 에러 분포에 대해 매개 변수 k=8의 중심 이항분포 k를 사용한다. k=8일 때의 8비트를 1로 설정된 비트의 수를 반환하는 함수인 HW(Hamming weight)를 통해 계산한다. 결과 값과 q값은 MUX를 통해 연결되어 q값을 기준으로 값을 선택해서 출력한다.

[0090] 도 7은 본 발명의 일 실시예에 따른 4-병렬 MDF-기반 NTT 코어의 구조를 나타내는 도면이다.

[0091] 본 발명의 일 실시예에 따르면, 모든 데이터는 4-병렬 구조로 연산을 수행하며, NTT 코어는 PE1, PE2로 구성될 수 있다. 4-병렬 기반이므로 4-path를 사용하고 각각의 path에서는 PE1 구조를 이용해서 연산하며, 각 path는 stage-8까지 256-point BU(Butterfly Unit)부터 2-point BU까지 연산하고 stage-9와 stage-10 에서는 2-point BU 연산을 위한 PE2가 2개씩 사용될 수 있다.

[0093] 도 8은 본 발명의 일 실시예에 따른 NTT 코어의 PE1(Processing Element 1) 구조를 나타내는 도면이다.

[0094] 본 발명의 일 실시예에 따른 PE1 Addition, 28비트 MR(Modular Reduction), FIFO 레지스터, BU로 구성되어 있다. a + bw, a - bw 와 같은 BU 연산을 수행하는 과정 중 입력된 데이터는 MR1 블록에서 연산된 후에 제어신호에 의해 ROM형식으로 저장된 ω^i 값과 곱셈 연산을 한다. 해당 연산을 수행한 후에 환 안의 값으로 줄여주는 MR2 블록을 지나서 BU 연산을 하게 된다.

[0096] 도 9는 본 발명의 일 실시예에 따른 NTT 코어의 PE2(Processing Element 1) 구조를 나타내는 도면이다.

[0097] 본 발명의 일 실시예에 따르면, MDF-기반 구조에만 사용된 PE2는 PE1과 다르게 FIFO 레지스터가 없고 바로 BU 연산을 하게 된다.

[0099] 도 10은 본 발명의 일 실시예에 따른 NTT 코어의 MR(Modulo Reduction) 구조를 나타내는 도면이다.

[0100] 본 발명의 일 실시예에 따른 NTT 코어의 연산은 환(ring) 안에서 동작해야 하므로 매 연산마다 환 안의 값으로 줄여주는 MR 연산을 필요로 한다. t값은 ((x<<1) + (x<<4) + (x<<6) + (x<<8) + (x<<10) + (x<< 12) + (x<<14) >>28)이며 t_2는 Reduction을 위해 입력 받은 x값에서 (t<<12)+(t<<13)+t 값을 뺀다. 계산된 t_2값이 q값인 12289보다 크면 12289을 빼주고 그렇지 않으면 그대로 t_2값을 내보낸다.

[0102] 도 11은 본 발명의 일 실시예에 따른 MDF 기반 NTT를 이용한 Ring-LWE 암호화 방법을 설명하기 위한 흐름도이다.

[0103] 본 발명의 일 실시예에 따른 MDF 기반 NTT를 이용한 Ring-LWE 암호화 방법은 키생성 프로세서를 통해 PRNG 코어로부터 데이터를 입력 받아 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하고, NTT 연산을 수행한 뒤 공개키 및 비밀키를 생성하여 메모리에 저장하는 단계(1110), 암호화 프로세서를 통해 메모리로부터 저장된 공개키와 데이터를 수신하여 변형시키고, 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 및 INTT 연산을 수행한 뒤 암호문의 압축 과정을 거친 후, 암호문을 출력하는 단계(1120) 및 복호화 프로세서를 통해 메모리로부터 저장된 비밀키와 암호문을 수신하여 암호문을 복원하고, 비밀키를 변형시키며, INTT 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 원본 메시지를 최종 출력하는 단계(1130)를 포함한다.

[0104] 단계(1110)에서, 키생성 프로세서를 통해 PRNG 코어로부터 데이터를 입력 받아 공개키 생성 및 비밀키 생성과

암호화에 필요한 에러를 생성하고, NTT 연산을 수행한 뒤 공개키 및 비밀키를 생성하여 메모리에 저장한다.

[0105] 단계(1110)에서, 거부 샘플러를 통해 PRNG 코어로부터 데이터를 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성하고, 이항 샘플러를 통해 PRNG 코어로부터 데이터를 입력 받아 비밀키 생성과 암호화에 필요한 에러를 생성한다.

[0106] NTT 코어를 통해 이항 샘플러의 출력을 입력 받아 NTT 연산을 수행하고, 거부 샘플러 및 NTT 코어의 출력을 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 공개키 인코더를 통해 입력 받아 공개키를 생성하여 메모리에 저장한다. 이후, 다항식 인코더를 통해 NTT 코어의 출력을 입력 받아 데이터를 변형시켜 비밀키를 생성하여 메모리에 저장한다.

[0107] 단계(1120)에서, 암호화 프로세서를 통해 메모리로부터 저장된 공개키와 데이터를 수신하여 변형시키고, 공개키 생성 및 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 및 INTT 연산을 수행한 뒤 암호문의 압축 과정을 거친 후, 암호문을 출력한다.

[0108] 단계(1120)에서, 메시지 인코더를 통해 메모리로부터 저장된 메세지를 수신하여 데이터를 변형시키고, 공개키 디코더를 통해 메모리로부터 저장된 공개키를 수신하여 공개키를 변형시킨다.

[0109] 거부 샘플러를 통해 공개키 디코더의 출력을 입력 받아 공개키 생성과 암호화에 필요한 에러를 생성하고, 이항 샘플러를 통해 비밀키 생성과 암호화에 필요한 에러를 생성한다.

[0110] NTT 코어를 통해 이항 샘플러의 출력을 입력 받아 NTT 연산을 수행하고, NTT 코어 및 공개키 디코더의 출력을 점 곱셈 연산한 값을 INTT 코어를 통해 입력 받아 역-NTT 연산을 수행한다.

[0111] 이항 샘플러 및 메시지 인코더의 출력을 다항식 덧셈한 후 INTT 코어의 출력과 다항식 덧셈을 통해 각각의 계수를 더한 값을 컴프레스를 통해 입력 받아 암호문을 압축하고, 거부 샘플러 및 NTT 코어의 출력을 점 곱셈 연산한 후 다항식 덧셈을 통해 각각의 계수를 더한 값을 암호문 인코더를 통해 입력 받아 암호문을 출력한다.

[0112] 단계(1130)에서, 복호화 프로세서를 통해 메모리로부터 저장된 비밀키와 암호문을 수신하여 암호문을 복원하고, 비밀키를 변형시키며, INTT 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 원본 메시지를 최종 출력한다.

[0113] 단계(1130)에서, 암호문 디코더를 통해 메모리로부터 저장된 암호문을 수신하여 암호문을 복원하고, 다항식 디코더를 통해 메모리로부터 저장된 비밀키를 수신하여 비밀키를 변형시킨다.

[0114] 디코더 및 다항식 디코더의 출력을 점 곱셈 연산한 값을 INTT 코어를 통해 입력 받아 역-NTT 연산을 수행하고, 디컴프레스를 통해 디코더의 출력을 입력 받아 복원된 암호문을 압축해제한다. 이후, INTT 코어 및 디컴프레스의 출력을 다항식 뺄셈한 값을 디코더를 통해 입력 받아 원본 메시지를 최종 출력한다.

[0116] 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.

[0117] 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.

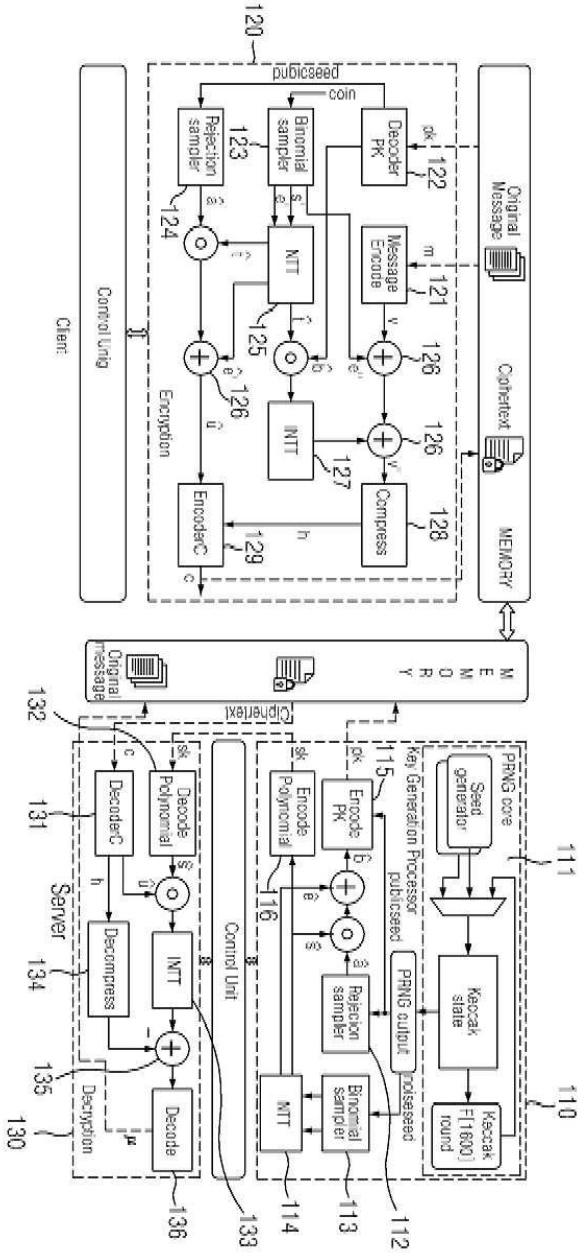
[0118] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic 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.

[0119] 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.

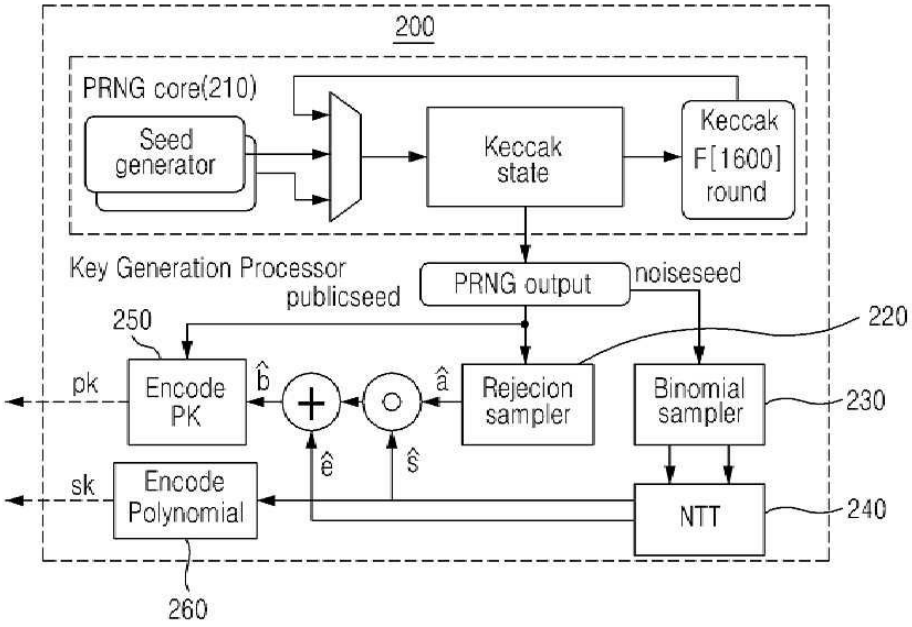
[0120] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

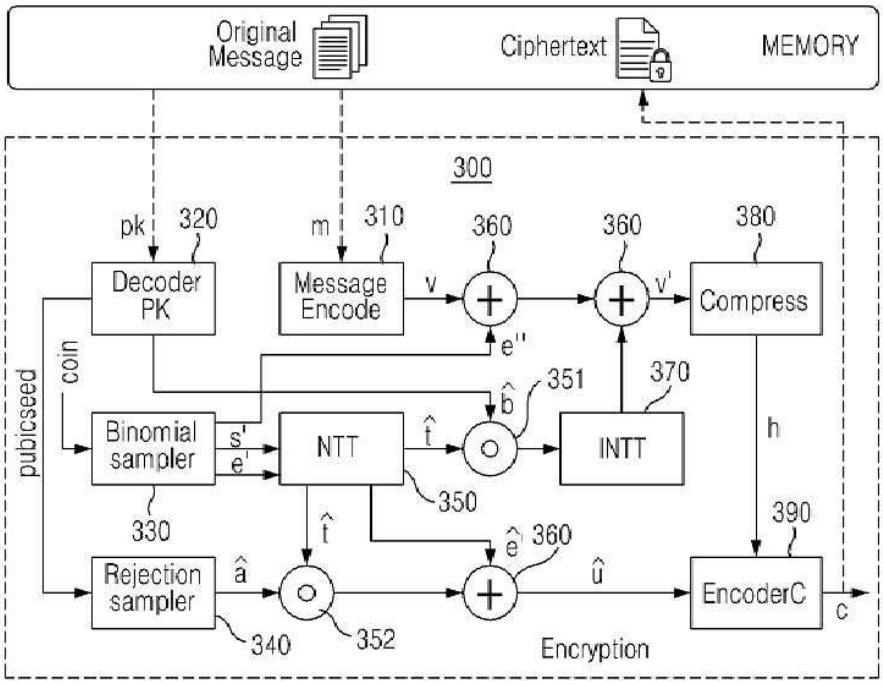
도면1


Client
Control Unit
Encryption
publicseed
coin
Rejection sampler
Binomial sampler
Decoder PK
Message Encode
NTT
INTT
EncoderC
Compress
Original Message
Ciphertext
MEMORY
Server
Decryption
DecoderC
Decode Polynomial
Decompress
Decode
Control Unit
Key Generation Processor
Encode Polynomial
Encode PK
PRNG core
Seed generator
Keccak state
Keccak F[1600] round
PRNG output
noiseseed

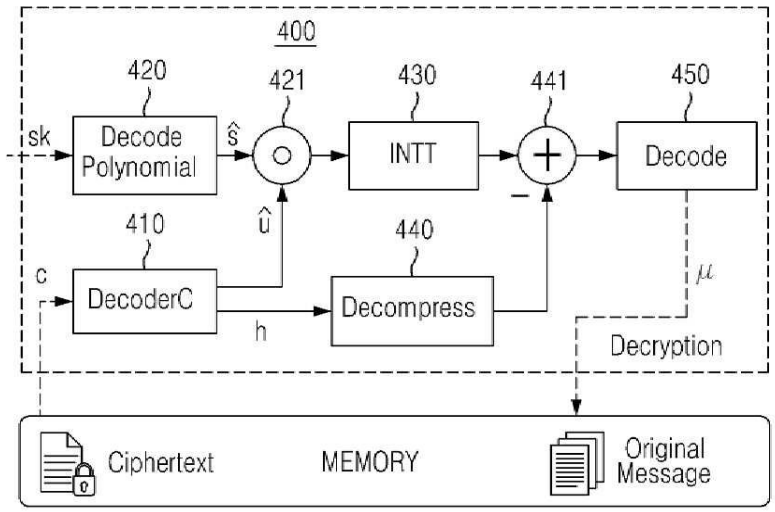
도면2


200
PRNG core(210)
Seed generator
Keccak state
Keccak F[1600] round
Key Generation Processor
publicseed
PRNG output
noiseseed
250
220
pk
Encode PK
Rejecion sampler
Binomial sampler
230
sk
Encode Polynomial
NTT
240
260

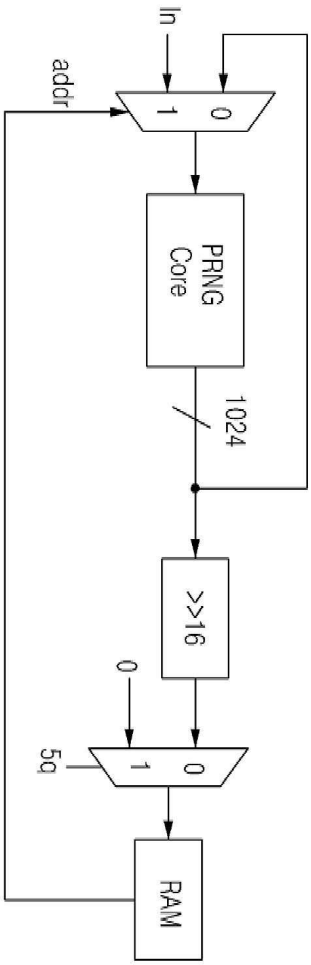
도면3


Original Message
Ciphertext
MEMORY
300
pk
320
m
310
360
360
380
Decoder PK
Message Encode
v
v'
Compress
e''
coin
pubicseed
351
370
Binomial sampler
s'
e'
NTT
INTT
h
350
330
390
Rejection sampler
340
352
360
EncoderC
c
Encryption

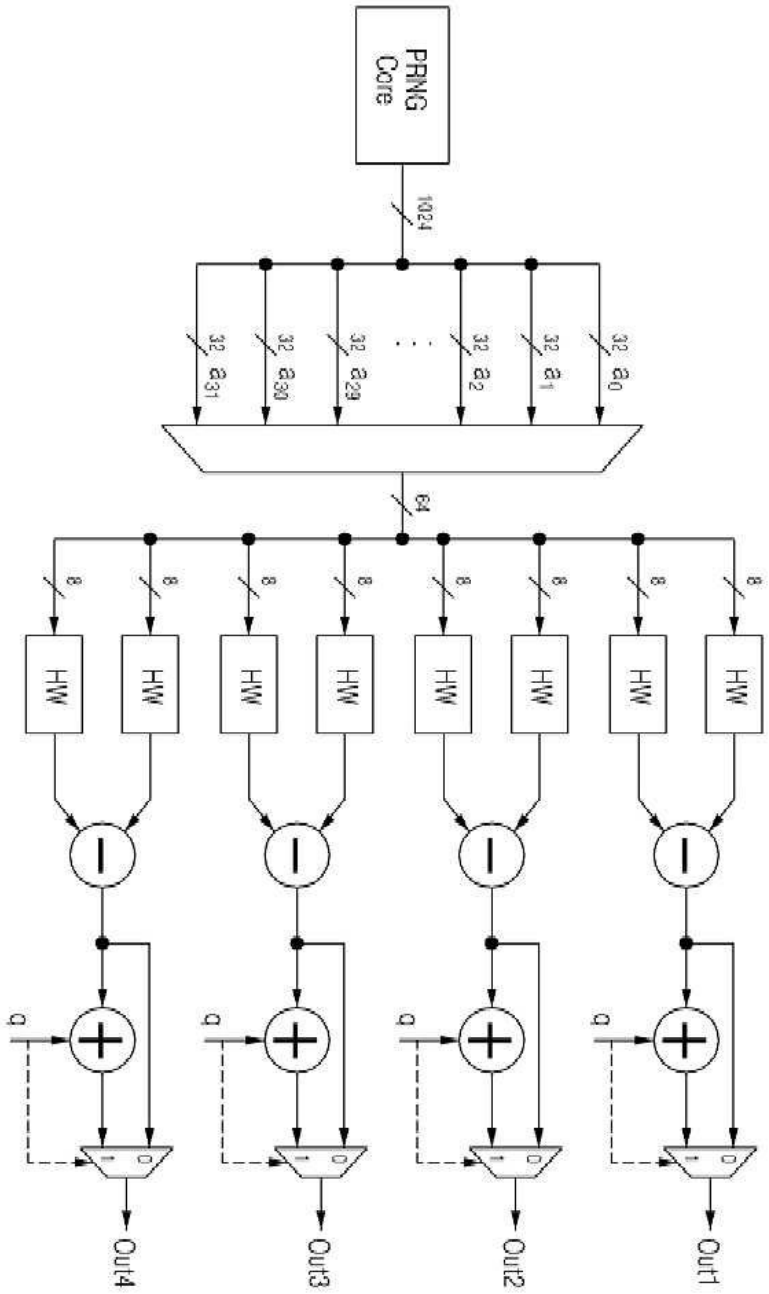
도면4


400
420
421
430
441
450
sk
Decode Polynomial
INTT
Decode
410
440
c
DecoderC
Decompress
Decryption
Ciphertext
MEMORY
Original Message

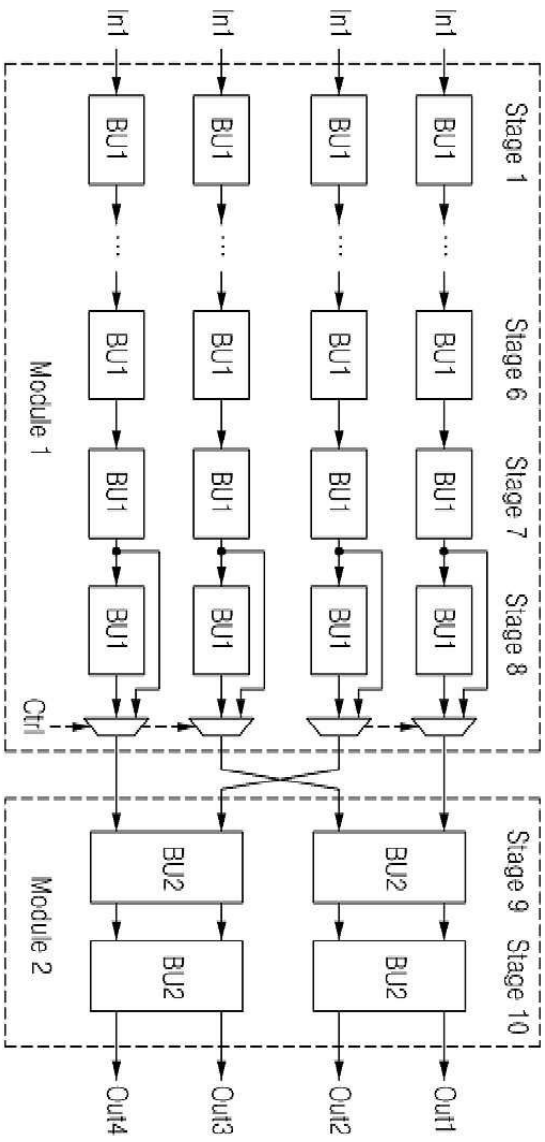
도면5


In
addr
PRNG Core
1024
>>16
0
5q
RAM

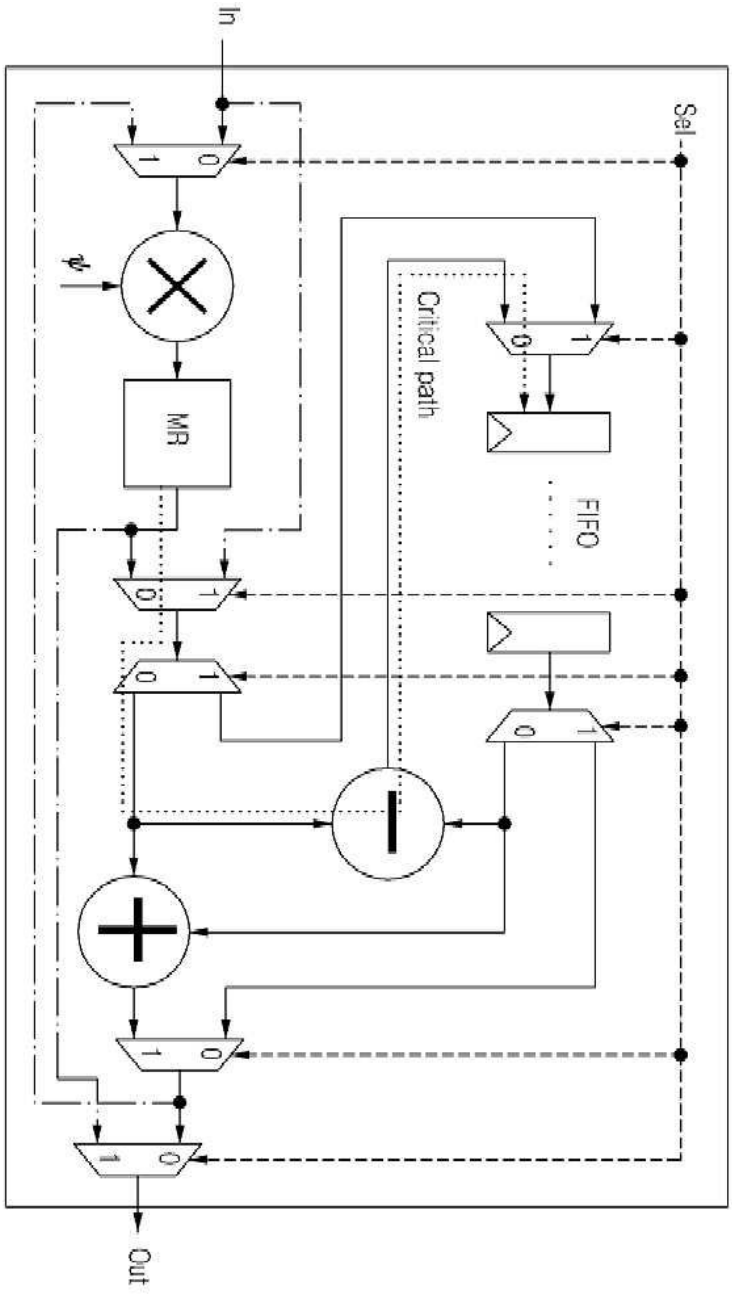
도면6


PRNG Core
1024
32 a31
32 a30
32 a29
32 a2
32 a1
32 a0
64
8
HW
q
Out4
Out3
Out2
Out1

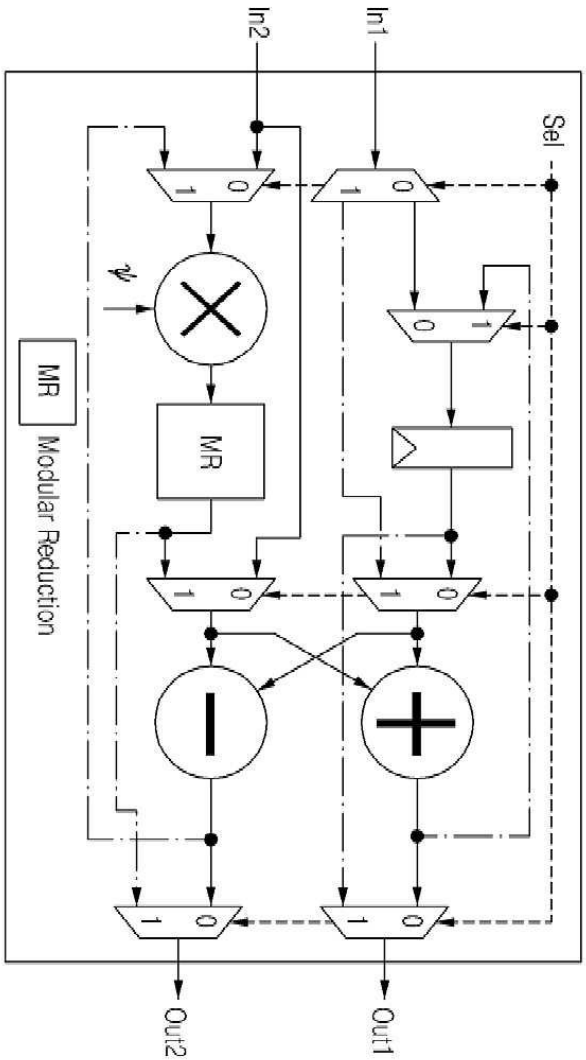
도면7


In1
In1
In1
In1
BU1
Stage 1
Stage 6
Stage 7
Stage 8
Module 1
Ctrl
BU2
Stage 9
Stage 10
Module 2
Out4
Out3
Out2
Out1

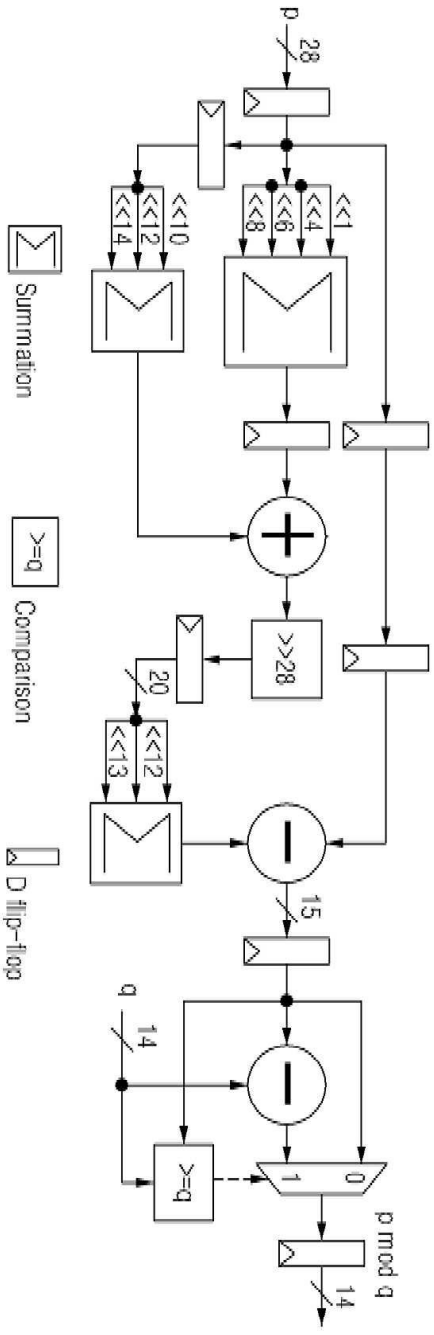
도면8



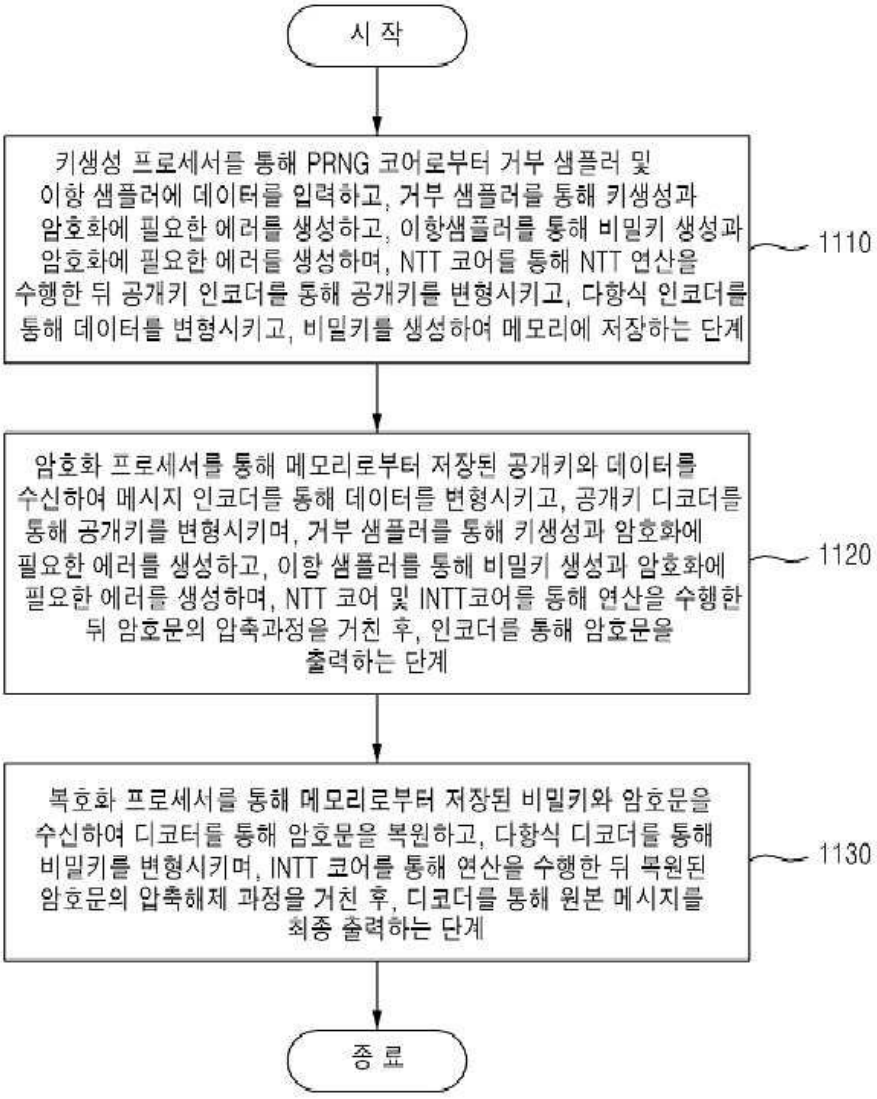
도면9



도면10


p
28
<<14
<<12
<<10
<<8
<<6
<<4
<<1
Summation
>=q
Comparison
D flip-flop
>>28
20
<<13
<<12
15
q
14
>=q
1
0
p mod q
14

도면11


시 작
키생성 프로세서를 통해 PRNG 코어로부터 거부 샘플러 및 이항 샘플러에 데이터를 입력하고, 거부 샘플러를 통해 키생성과 암호화에 필요한 에러를 생성하고, 이항샘플러를 통해 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 코어를 통해 NTT 연산을 수행한 뒤 공개키 인코더를 통해 공개키를 변형시키고, 다항식 인코더를 통해 데이터를 변형시키고, 비밀키를 생성하여 메모리에 저장하는 단계
1110
암호화 프로세서를 통해 메모리로부터 저장된 공개키와 데이터를 수신하여 메시지 인코더를 통해 데이터를 변형시키고, 공개키 디코더를 통해 공개키를 변형시키며, 거부 샘플러를 통해 키생성과 암호화에 필요한 에러를 생성하고, 이항 샘플러를 통해 비밀키 생성과 암호화에 필요한 에러를 생성하며, NTT 코어 및 INTT코어를 통해 연산을 수행한 뒤 암호문의 압축과정을 거친 후, 인코더를 통해 암호문을 출력하는 단계
1120
복호화 프로세서를 통해 메모리로부터 저장된 비밀키와 암호문을 수신하여 디코터를 통해 암호문을 복원하고, 다항식 디코더를 통해 비밀키를 변형시키며, INTT 코어를 통해 연산을 수행한 뒤 복원된 암호문의 압축해제 과정을 거친 후, 디코더를 통해 원본 메시지를 최종 출력하는 단계
1130
종 료