



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2025-0066933
(43) 공개일자 2025년05월14일

(51) 국제특허분류(Int. Cl.)

H04L 9/00 (2022.01)

(52) CPC특허분류

H04L 9/008 (2013.01)

(21) 출원번호 10-2023-0153062

(22) 출원일자 2023년11월07일

심사청구일자 2023년11월07일

(71) 출원인

인하대학교 산학협력단

인천광역시 미추홀구 인하로 100(용현동, 인하대학교)

(72) 발명자

이한호

인천광역시 연수구 컨벤시아대로252번길 30, 1505동 1701호(송도동, 송도 더샵 퍼스트파크 F15BL)

이재혁

서울특별시 노원구 초안산로1길 18, 208동 1203호(월계동, 월계주공2단지아파트)

(74) 대리인

양성보

전체 청구항 수 : 총 9 항

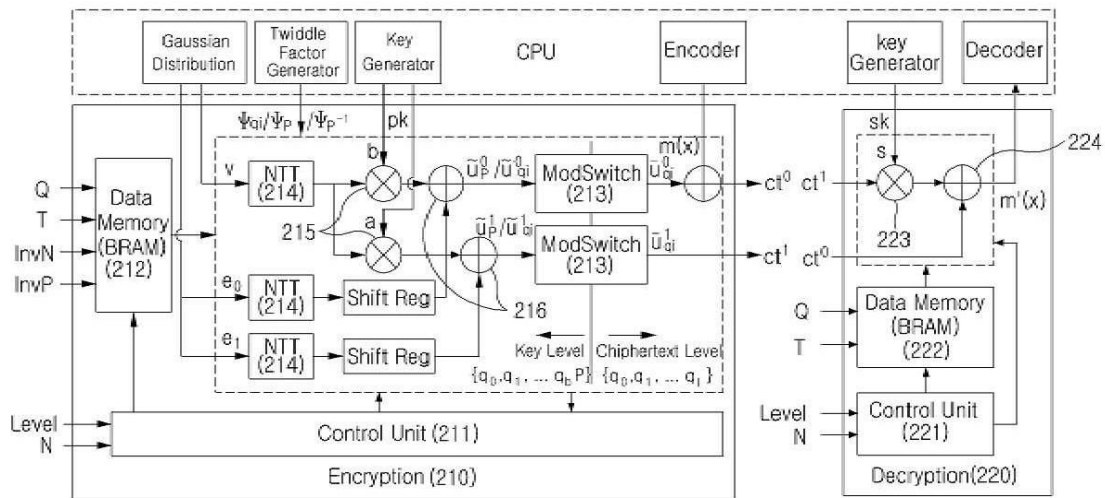
(54) 발명의 명칭 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 장치 및 방법

(57) 요약

동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 방법 및 시스템이 제시된다. 본 발명에서 제안하는 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 장치는 파라미터 길이, 모듈러스, 동형연산 가능 횟수를 가변적으로 지원하며 평문의 데이터를 암호화하여 암호문으로 변환하기 위한 암호화 처리 장치 및 암호화된 데이터

(뒷면에 계속)

대표도



터를 복호화하기 위한 복호화 처리 장치를 포함하고, 상기 암호화 처리 장치는 암호화를 위한 파라미터들을 임의로 생성하기 위한 에러를 생성하는 가우시안 분포 에러 생성기, 복수의 회전인자 및 역회전인자 값을 생성하는 회전인자 생성기, 암호화를 위한 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 키생성기, 평문을 생성하는 인코더 및 평문을 원본 메시지로 바꿔주는 디코더, 상기 키 생성기에서 생성되는 키 레벨의 데이터를 데이터 레벨로 변환해주는 모듈러 스위치, 사용자로부터 입력 받은 파라미터를 읽고 쓰는 데이터 메모리 및 사용자로부터 입력 받은 파라미터로부터 암호화 처리장치를 컨트롤 해주는 컨트롤 유닛을 포함한다.

이 발명을 지원한 국가연구개발사업

과제고유번호	1711193592
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	1/3
과제수행기관명	인하대학교 산학협력단
연구기간	2021.07.01 ~ 2028.12.31

이 발명을 지원한 국가연구개발사업

과제고유번호	1711188752
과제번호	2021R1A2C1011232
부처명	과학기술정보통신부
과제관리(전문)기관명	한국연구재단
연구사업명	기초연구사업
연구과제명	고신뢰성 고성능 격자-기반 양자내성암호 하드웨어 아키텍처 연구
기 여 율	1/3
과제수행기관명	인하대학교 산학협력단
연구기간	2021.03.01 ~ 2025.02.28

이 발명을 지원한 국가연구개발사업

과제고유번호	1711134703
과제번호	20210007790012003
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	방송통신산업기술개발사업
연구과제명	HW지원 프라이버시 보장 암호데이터 고속처리 기술 개발
기 여 율	1/3
과제수행기관명	인하대학교 산학협력단
연구기간	2021.04.01 ~ 2024.12.31

명세서

청구범위

청구항 1

파라미터 길이, 모듈러스, 동형연산 가능 횟수를 가변적으로 지원하며 평문의 데이터를 암호화하여 암호문으로 변환하기 위한 암호화 처리 장치; 및

암호화된 데이터를 복호화하기 위한 복호화 처리 장치

를 포함하고,

상기 암호화 처리 장치는,

암호화를 위한 파라미터들을 임의로 생성하기 위한 에러를 생성하는 가우시안 분포 에러 생성기;

복수의 회전인자 및 역회전인자 값을 생성하는 회전인자 생성기;

암호화를 위한 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 키생성기;

평문을 생성하는 인코더 및 평문을 원본 메시지로 바꿔주는 디코더;

상기 키 생성기에서 생성되는 키 레벨의 데이터를 데이터 레벨로 변환해주는 모듈러 스위치;

사용자로부터 입력 받은 파라미터를 읽고 쓰는 데이터 메모리; 및

사용자로부터 입력 받은 파라미터로부터 암호화 처리장치를 컨트롤 해주는 컨트롤 유닛

을 포함하는 암호화 -복호화 처리 장치.

청구항 2

제1항에 있어서,

상기 암호화 처리 장치는,

RNS(Residue Number System) 수체계에서 정의된 파라미터인 상기 복수의 파라미터 길이, 모듈러스, 동형연산 가능 횟수에 대해 재구성 가능한 암호화를 수행하기 위해 공개키와 데이터를 수신하여 컨트롤 유닛, 데이터 메모리, 모듈러 스위치를 통해 가변적으로 지원하며, NTT, 모듈러 덧셈기, 모듈러 곱셈기를 사용하여 암호화하는

암호화 -복호화 처리 장치.

청구항 3

제1항에 있어서,

상기 모듈러 스위치는,

RNS(Residue Number System) 수체계에서 정의된 파라미터인 상기 복수의 파라미터 길이와 동형연산 가능 횟수를 수신하여 컨트롤 유닛을 통해 연산 반복 횟수, 사이클의 크기를 판별하고, 수신된 키 레벨의 데이터를 컨트롤 유닛(Control Unit), 데이터 메모리(Data Memory), INTT, NTT, BRAM, 모듈러 덧셈기, 뺄셈기, 모듈러 곱셈기를 이용하여 데이터 레벨로 출력하는

암호화 -복호화 처리 장치.

청구항 4

제3항에 있어서,

상기 컨트롤 유닛은,

사용자로부터 입력 받은 파라미터를 판별하여 재구성형 CKKS-기반 암호화 처리 장치에서 동형연산 가능 횟수와 하나의 사이클의 길이를 결정하는

암호화-복호화 처리 장치.

청구항 5

제3항에 있어서,

상기 복호화 처리 장치는,

상기 모듈러 스위치로부터 복수의 파라미터를 갖는 가변적인 암호문과 비밀키를 수신하여 상기 복호화 처리 장치의 컨트롤 유닛, 데이터 메모리, 모듈러 덧셈기, 모듈러 곱셈기를 사용하여 원본 메시지로 변환하는

암호화-복호화 처리 장치.

청구항 6

가우시안 분포 에러 생성기를 통해 암호화를 위한 파라미터들을 임의로 생성하기 위한 에러를 생성하는 단계;

회전인자 생성기를 통해 암호화 처리에 필요한 복수의 회전인자 및 역회전인자 값을 생성하는 단계;

키 생성기를 통해 암호화를 위한 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 단계;

인코더를 통해 다항식을 수신하여 암호화를 위한 평문으로 부호화하는 단계; 및

NTT 코어, 모듈러 스위치, 모듈러 곱셈기, 모듈러 덧셈기를 포함하는 동형 암호화 처리 장치를 통해 파라미터, 공개키, 에러, 회전인자, 평문을 수신하여 암호화하는 단계

를 포함하는 동형암호를 위한 암호화 처리 방법.

청구항 7

제6항에 있어서,

상기 동형 암호화 처리 장치를 통해 파라미터, 공개키, 에러, 회전인자, 평문을 수신하여 암호화하는 단계는,

컨트롤 유닛을 통해 사용자로부터 입력 받은 파라미터를 판별하여 동형연산 가능 횟수와 하나의 사이클의 길이를 결정하는 단계;

컨트롤 유닛을 통해 회전인자 생성기 내에서 생성된 복수의 회전인자 및 역회전인자 값을 NTT/INTT와 모듈러 스위치로 입력하는 단계;

데이터 메모리를 통해 암호화의 특정 사이클 마다 파라미터를 읽고 쓰는 단계;

가우시안 분포 에러 생성기를 통해 생성된 에러에 대한 NTT 연산 수행하는 단계;

상기 NTT 연산 결과를 키 생성기에서 받은 공개 키와 모듈러 곱셈 연산 수행하는 단계;

상기 모듈러 곱셈 결과와 가우시안 분포 에러 생성기를 통해 생성된 에러의 NTT 연산 결과에 대한 모듈러 덧셈 연산 수행하는 단계;

모듈러 스위치를 통해 모듈러 덧셈 결과의 재구성 가능한 키 데이터를 데이터 레벨로 변환하는 단계; 및

모듈러 스위치 연산 결과와 인코더로부터 입력 받은 평문을 모듈러 덧셈 연산 통해 암호문으로 출력하는 단계

를 포함하는 동형암호를 위한 암호화 처리 방법.

청구항 8

제7항에 있어서,

상기 컨트롤 유닛을 통해 사용자로부터 입력 받은 파라미터를 판별하여 동형연산 가능 횟수와 하나의 사이클의 길이를 결정하는 단계는,

사용자로부터 입력 받은 파라미터를 컨트롤 유닛 내에서 Mux를 이용하여 파라미터 길이를 판별한 뒤, 각 모듈의 사이클 수를 파라미터 길이에 맞게 계산한 다음, 반복 동작으로 암호화 처리 연산을 반복하는

동형암호를 위한 암호화 처리 방법.

청구항 9

제7항에 있어서,

상기 데이터 메모리를 통해 암호화의 특정 사이클 마다 파라미터를 읽고 쓰는 단계는,

최초 사이클 동안 암호화 처리 연산의 수만큼 사용자로부터 입력 받은 파라미터를 컨트롤 유닛을 통해 데이터 메모리에 저장하고, 동형 암호화를 진행하며 해당 파라미터가 사용될 때 컨트롤 유닛에서 읽기 신호를 제어하여 특정 파라미터 값을 읽는

동형암호를 위한 암호화 처리 방법.

발명의 설명

기술 분야

[0001] 본 발명은 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 장치 및 방법에 관한 것이다.

배경 기술

[0002] 동형암호(Homomorphic Encryption; HE)는 데이터를 암호화된 상태에서 연산할 수 있는 암호화 방법으로 클라우드 서버 내에 원본 데이터가 남지 않게 해주어 차세대 암호 알고리즘으로 주목 받고 있다. Ring-LWE(Learning With Error)는 에러를 삽입하여 데이터의 내용을 유추하기 어렵게 하는 암호화 기법으로 양자컴퓨터에 내성을 지니고 있어 주목 받고 있다. 동형암호의 BGV, BFV, CKKS 등의 알고리즘이 Ring-LWE로 만들어져 높은 보안성을 자랑한다.

[0003] 종래기술에서 제안된 CKKS 기법은 HEAAN(Homomorphic Encryption for Arithmetic of Approximate Numbers) 알고리즘의 일종으로 암호화된 결과의 근사치를 제공하는 근사연산 동형암호 아키텍처이다. CKKS 동형암호 알고리즘은 Ring-LWE의 방법을 사용하며 환(Ring) 내에서 계산된다. 동형연산을 수행할 수 있는 윗수인 레벨(Level)의 개념과 노후화된 암호문을 새 암호문으로 교환해주는 재부팅(Bootstrapping) 알고리즘을 도입하였다. 또한, 효율적인 계산을 위해 RNS(Residue Number System)를 통해 환 내에서 계산되는 Q값을 나눠 NTT(Number Theoretic Transform) 연산을 통해 다항식 곱셈들을 가속화한다.

[0004] 동형암호는 다른 Ring-LWE 암호와 다르게 복수의 표준을 지원한다. 동형암호 표준화기구(Homomorphic Encryption Standardization)에서 지정한 동형암호의 표준은 기구에서 지정한 수학적 정의를 만족하는 어떠한 수도 사용할 수 있고, 이는 동형암호 사용자가 정밀도, 연산윗수 등 사용자의 용도에 맞게 선택할 수 있다는 뜻이다. 동형암호 아키텍처는 사용자가 선택하는 다양한 길이와 모듈러스를 지원하는 하드웨어를 제공해야 한다. 따라서 동형암호 아키텍처는 다른 Ring-LWE 암호에서 목표로 하는 병렬화를 통한 빠른 성능 구현과 더불어 복수의 표준에 만족하는 다양한 길이, 다양한 모듈러스를 지원해야 한다. 따라서 사용자가 원하는 다양한 파라미터를 지원하면서 하드웨어 복잡도가 작은 완전동형 암호 아키텍처 및 암호시스템 연구가 필수적이다.

선행기술문헌

특허문헌

[0005] (특허문헌 0001) 한국공개특허 제10-2023-0123418호 (2023.08.23)

발명의 내용

해결하려는 과제

[0006] 본 발명이 이루고자 하는 기술적 과제는 동형암호 원리에 맞추어 파라미터(N, q, Level)에 대하여 가변성을 제공하여 하나의 하드웨어를 이용하여 복수의 표준을 지원하기 위한 동형암호를 위한 고성능 CKKS-기반 암호화-복호화 처리 장치의 하드웨어 구조 및 방법을 제공하는데 있다. 또한, 본 발명의 실시예에 따른 모듈러 연산기는 덧셈기, 뺄셈기 및 곱셈기로 이루어져 있으며 다양한 파라미터 q에 대응하도록 하고, 모듈러 스위치(Modular Switch)는 다양한 파라미터(N, q, Level)에 대해 대응 가능하며 데이터의 크기를 줄이도록 한다. 또한, NTT 다

항식 곱셈기는 MUX, DEMUX를 이용한 분기를 이용하여 다양한 파라미터(N, q)에 대해 대응하여 하드웨어 복잡성을 줄이고, 클럭 속도를 높이기 위해 2-병렬 MDF-기반 구조, 파이프라인 구조, 및 비트-역순(Bit-Reverse) 연산을 제거하는 방법을 제공하고자 한다.

과제의 해결 수단

- [0007] 일 측면에 있어서, 본 발명에서 제안하는 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 장치는 파라미터 길이, 모듈러스, 동형연산 가능 횟수를 가변적으로 지원하며 평문의 데이터를 암호화하여 암호문으로 변환하기 위한 암호화 처리 장치 및 암호화된 데이터를 복호화하기 위한 복호화 처리 장치를 포함하고, 상기 암호화 처리 장치는 암호화를 위한 파라미터들을 임의로 생성하기 위한 에러를 생성하는 가우시안 분포 에러 생성기, 복수의 회전인자 및 역회전인자 값을 생성하는 회전인자 생성기, 암호화를 위한 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 키생성기, 평문을 생성하는 인코더 및 평문을 원본 메시지로 바꿔주는 디코더, 상기 키 생성기에서 생성되는 키 레벨의 데이터를 데이터 레벨로 변환해주는 모듈러 스위치, 사용자로부터 입력 받은 파라미터를 읽고 쓰는 데이터 메모리 및 사용자로부터 입력 받은 파라미터로부터 암호화 처리장치를 컨트롤 해주는 컨트롤 유닛을 포함한다.
- [0008] 상기 암호화 처리 장치는 RNS(Residue Number System) 수체계에서 정의된 파라미터인 상기 복수의 파라미터 길이, 모듈러스, 동형연산 가능 횟수에 대해 재구성 가능한 암호화를 수행하기 위해 공개키와 데이터를 수신하여 컨트롤 유닛, 데이터 메모리, 모듈러 스위치를 통해 가변적으로 지원하며, NTT, 모듈러 덧셈기, 모듈러 곱셈기를 사용하여 암호화한다.
- [0009] 상기 모듈러 스위치는 RNS(Residue Number System) 수체계에서 정의된 파라미터인 상기 복수의 파라미터 길이와 동형연산 가능 횟수를 수신하여 컨트롤 유닛을 통해 연산 반복 횟수, 사이클의 크기를 판별하고, 수신된 키 레벨의 데이터를 컨트롤 유닛(Control Unit), 데이터 메모리(Data Memory), INTT, NTT, BRAM, 모듈러 덧셈기, 뺄셈기, 모듈러 곱셈기를 이용하여 데이터 레벨로 출력한다.
- [0010] 상기 컨트롤 유닛은 사용자로부터 입력 받은 파라미터를 판별하여 재구성형 CKKS-기반 암호화 처리 장치에서 동형연산 가능 횟수와 하나의 사이클의 길이를 결정한다.
- [0011] 또 다른 일 측면에 있어서, 본 발명에서 제안하는 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 방법은 가우시안 분포 에러 생성기를 통해 암호화를 위한 파라미터들을 임의로 생성하기 위한 에러를 생성하는 단계, 회전인자 생성기를 통해 암호화 처리에 필요한 복수의 회전인자 및 역회전인자 값을 생성하는 단계, 키 생성기를 통해 암호화를 위한 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 단계, 인코더를 통해 다항식을 수신하여 암호화를 위한 평문으로 부호화하는 단계 및 NTT 코어, 모듈러 스위치, 모듈러 곱셈기, 모듈러 덧셈기를 포함하는 동형 암호화 처리 장치를 통해 파라미터, 공개키, 에러, 회전인자, 평문을 수신하여 암호화하는 단계를 포함한다.

발명의 효과

- [0012] 본 발명의 실시예들에 따르면 다양한 파라미터(N, q, Level)에 대해 동작하는 CKKS 암호프로세서를 통하여 Ring-LWE 기반 RNS 수 체계에서 동작하는 동형암호를 위한 암호화 처리 장치에서 하드웨어 재사용성을 높이고 중복된 구조를 생략하여 하드웨어 복잡도를 낮추며 하드웨어를 이용하여 가속할 수 있다. 또한, 본 발명의 실시예에 따른 모듈러 연산기는 덧셈기, 뺄셈기 및 곱셈기로 이루어져 있으며 다양한 파라미터 q 에 대응하고, 모듈러 스위치(Modular Switch)는 다양한 파라미터(N, q, Level)에 대해 대응 가능하며 데이터의 크기를 줄일 수 있다. 또한, NTT 다항식 곱셈기는 MUX, DEMUX를 이용한 분기를 이용하여 다양한 파라미터(N, q)에 대해 대응하여 하드웨어 복잡성을 줄이고, 클럭 속도를 높이기 위해 2-병렬 MDF-기반 구조, 파이프라인 구조, 및 비트-역순(Bit-Reverse) 연산을 제거할 수 있다.

도면의 간단한 설명

- [0013] 도 1은 본 발명의 실시예에 따른 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 장치의 데이터 흐름을 설명하기 위한 도면이다.
- 도 2는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화-복호화 아키텍처의 구조를 나타내는 도면이다.
- 도 3은 본 발명의 실시예에 따른 Level과 N 에 따른 상세한 재구성형 CKKS-기반 암호화 아키텍처를 나타내는 도

면이다.

도 4는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 모듈러 스위치(Modular Switch) 아키텍처를 나타내는 도면이다.

도 5는 본 발명의 실시예에 따른 Level과 N에 따른 상세한 모듈러 스위치(Modular Switch) 아키텍처를 나타내는 도면이다.

도 6은 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 2-병렬 MDF-기반 재구성형 NTT 아키텍처를 나타내는 도면이다.

도 7은 본 발명의 실시예에 따른 NTT 아키텍처를 구성하는 PE1(Processing Element 1) 구조를 나타내는 도면이다.

도 8은 본 발명의 실시예에 따른 NTT 아키텍처를 구성하는 PE2(Processing Element 2) 구조를 나타내는 도면이다.

도 9는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 2-병렬 MDF-기반 재구성형 INTT 아키텍처를 나타내는 도면이다.

도 10은 본 발명의 실시예에 따른 INTT 아키텍처를 구성하는 PE1(Processing Element 1) 구조를 나타내는 도면이다.

도 11은 본 발명의 실시예에 따른 INTT 아키텍처를 구성하는 PE2(Processing Element 2) 구조를 나타내는 도면이다.

도 12는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 모듈러 곱셈기(Modular Multiplier)의 구조를 나타내는 도면이다.

도 13은 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 모듈러 덧셈기(Modular Addition)의 구조를 나타내는 도면이다.

도 14는 본 발명의 실시예에 따른 모듈러 스위치(Modular Switch)와 NTT 아키텍처를 구성하는 모듈러 뺄셈기(Modular Subtractor)의 구조를 나타내는 도면이다.

도 15는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처와 모듈러 스위치(Modular Switch)의 타이밍을 나타내는 도면이다.

도 16은 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처의 동작 방법을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

- [0014] 본 발명은 RNS 수체계에서 동작하는 Ring-LWE 기반의 동형암호를 위한 가변적인 파라미터에서 동작하는 재구성형 CKKS-기반 암호화-복호화 처리 장치 및 방법에 관한 것으로, 보다 상세하게는 다양한 암호문의 길이 N , RNS 수체계의 다양한 모듈러스 q , 동형연산 가능 횟수 레벨(Level) 지원이 가능하다.
- [0015] 본 발명의 실시예에 따른 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 장치는 RNS 수체계에서 동작하는 격자-기반의 Ring-LWE 기법의 동형암호에서 가변적인 파라미터를 지원하기 위해 사용한다. RNS 수체계의 Ring-LWE 기법의 동형암호는 용도에 따른 다양한 암호문의 길이 N 이 요구되며, RNS 수체계에서 동작함으로 다양한 소수(Prime Number)의 모듈러스 파라미터 q 를 요구한다. 또한 동형연산 가능 횟수인 Level은 최소 30 이상을 요구한다. 다양한 암호문의 길이 N , 모듈러스 q , 동형연산 가능 횟수 Level에 의하여 높은 보안성을 갖는다. 이하, 본 발명의 실시 예를 첨부된 도면을 참조하여 상세하게 설명한다.
- [0017] 도 1은 본 발명의 실시예에 따른 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 장치의 데이터 흐름을 설명하기 위한 도면이다.
- [0018] 본 발명의 실시예에 따른 동형암호는 복호화 또는 비밀키의 접근 없이 암호화된 데이터만을 이용해 연산할 수 있다. 비밀키를 함께 주지 않고 암호화 상태의 데이터만 클라우드 서버나 외부매체에 저장 및 계산하게 함으로

써 외부에서 정보가 유출되더라도 개인정보를 안전하게 보호할 수 있다. 개인정보보호에 대한 규제가 심한 금융이나 의료분야는 동형암호를 사용함으로써 데이터 장벽 없이 고성능 클라우드 시스템을 활용한 데이터 위탁처리 시스템을 구축할 수 있다.

[0019] 본 발명의 실시예에 따른 CKKS Scheme은 4세대 완전동형암호 및 근사연산 동형암호 알고리즘으로, 암호화된 결과의 근사치를 제공한다. 근사연산 동형암호는 계산 결과에 유의미한 영향을 미치지 않는 소수점 아래 값을 버리는 근사연산방식을 사용함으로써 동형암호의 연산속도를 개선한 암호 알고리즘을 말한다. 결과의 근사값이 제공된다는 점에서 $Dec(Ct_1 \times Ct_1) \approx Dec(Ct_1) \times Dec(Ct_0), Dec(Enc(m)) \approx m$ 와 같이 계산된다.

[0020] 본 발명의 실시예에 따른 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 장치의 데이터 흐름도에서 키 생성기(Key Generator)는 보안 파라미터를 수신하여 암호화에 이용할 수 있는 공개키, 비밀키 및 평가키를 생성하는 키 생성 프로세서이다.

[0021] 클라이언트로부터 Private 데이터를 수신하여 다항식 형식으로 변환한 데이터는 암호화에 적합한 형식으로 변환하는 인코더를 거친 후 암호화(Encryption)를 거치게 된다. 2-병렬 MDF-기반 NTT 코어를 사용하여 높은 데이터 처리율과 낮은 복잡도를 가지는 Ring-LWE 기반 CKKS 암호화 장치는 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. 이 암호화 처리 장치의 결과는 클라우드 서버로 전달되어 다양한 동형 연산을 진행한 결과를 다시 클라이언트 측으로 수신한다. 클라이언트 측으로 수신된 암호문은 복호화(Decryption)을 통해 원본 데이터로 변환하고, 디코더(Decoder)를 통해 원래의 Private 데이터 결과를 얻게 된다. 이 과정에서 동형암호의 특성으로 인하여 동형 연산의 결과와 원본 데이터의 연산 결과는 같다.

[0022] 본 발명의 실시예에 따른 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 처리 장치는 임의의 데이터를 암호화하기 위해 데이터를 다항식으로 변환하는 인코딩 단계를 거치며, 반대로 암호문을 복호화한 이후에는 원래의 데이터로 되돌리는 디코딩 단계를 가진다. 코딩 단계는 숫자 여러 개로 구성된 평문 벡터를 하나의 다항식으로 치환하는 다중메시지 패킹(Multiple message packing), 암호화하기 위한 데이터의 유효숫자를 지정하는 자릿수 조정(Scaling process) 과정의 두 단계로 이루어져 있다. 다중메시지 패킹과정은 다항식 하나에 가능한 많은 데이터를 포함함으로써 암호문이 사용하는 메모리의 낭비를 줄이고, 또한 여러 데이터를 동시에 암호화할 수 있는 형식을 제공하여 암호화의 속도를 향상한다. n개의 복소수 데이터 $D_1, D_2, \dots, D_n$ 이 있을 때, $2n-1$ 차 다항식 $m'(x)$ 은 식 (1)과 같다.

$$z_1 = e^{\pi i / 2\pi}, z_2 = e^{3\pi i / 2\pi}, \dots, z_n = e^{(2n-1)\pi i / 2\pi}$$

[0023] $m'(z_1) = D_1, m'(z_2) = D_2, \dots, m'(z_n) = D_n$ 식(1)

[0024] 본 발명의 실시예에 따른 자릿수 조정과정은 스케일링 팩터(Scaling factor) ρ 를 정하여 근사 데이터를 만드는 과정이다. 다중메시지 패킹과정을 거친 다항식 $m'(x)$ 에 ρ 를 곱한다. 이후 $\rho \cdot m'(x)$ 의 각 계수를 소수점 아랫자리에서 반올림하여 근사다항식 $m(x) = [\rho \cdot m'(x)]$ 을 얻는다. 결론적으로 데이터를 인코딩한 값인 $m(x) \bmod q$ 을 얻게 된다.

[0025] 본 발명의 실시예에 따른 디코더는 다항식 $m(x)$ 에 Scaling factor ρ 의 역수 ρ^{-1} 를 구하여, 그 역수를 평문 다항식에 곱한다. 그 뒤 다항식의 변수에 $z_1, z_2, \dots, z_n$ 을 대입하여 메시지 벡터 $\bar{m} = (\rho^{-1}m(z_1), \rho^{-1}m(z_2), \dots, \rho^{-1}m(z_n))$ 을 얻는다. 결과로 나오는 메시지 벡터는 소수점 이하를 반올림함으로써 정수 성분의 벡터를 얻는다.

[0026] 본 발명의 실시예에 따른 키생성 알고리즘은 안전성 변수 λ 가 주어졌을 때 정수 L, Scaling factor ρ , $q_L = \rho^L q$, 정수 P를 고르고, 암호화, 복호화, 연산에 필요한 키를 생성한다. 비밀키, 공개키, 계산키의 생성은 각각 식(2)와 같다. 여기서 $s(x)$ 는 환 $\mathbb{Z}[x]/x^{2n} + 1$ 내에 있으며, 해밍 가중치(Hamming weight)가 h인 무작

위 다항식, $a(x)$ 는 무작위, $e(x)$ 는 각 항의 계수가 표준편차 σ 인 이산가우스분포(Gaussian Distribution)를 따라 고른다.

$$sk = (-s(x), 1)$$

$$pk = (a(x), a(x)s(x) + e(x)) \in (R/q_L R)^2$$

$$evk = (a_i(x), a'_i(x)s(x) + e'_i(x) + P^{i+1} \cdot s^2(x)) \pmod{P \cdot q_L} \quad (2)$$

본 발명의 실시예에 따른 암호화 알고리즘은 공개키 $pk = (a(x), a(x)s(x) + e(x)) \in (R/q_L R)^2$ 를 사용하며, 다항식 $v(x)$ 의 모든 항은 계수가 -1, 0, 1중 하나이다. 계수가 0인 항은 n 개, 계수가 1과 -1인 항은 $n/2$ 개다. 두 예제 $e_0(x)$, $e_1(x)$ 은 각 항의 계수가 표준편차 σ 인 이산 가우스 분포를 따라 고른다. 식(3)은 암호화하기 위한 식이다.

$$\begin{aligned} ct &= (v(x)a(x) + e_0(x), v(x)b(x) + m(x) + e_1(x)) \pmod{q_L} \\ &= (c_0(x), c_1(x)) \end{aligned} \quad (3)$$

본 발명의 실시예에 따른 복호화 알고리즘은 비밀키 $sk = (-s(x), 1)$ 을 사용하며, 암호문 $ct = (c_0(x), c_1(x))$ 를 복호화하기 위해 암호문과 비밀키의 내적값을 구한다. 식(4)은 복호화 알고리즘의 식이다. 암호문을 복호화하면 본래의 평문의 근사다항식을 복호문으로 얻는다. 이 근사다항식과 본래 평문 다항식의 계수들은 같은 유효숫자를 가진다.

$$m(x) \equiv \langle ct, sk \rangle \equiv -c(x)s(x) + d(x) \pmod{q} \quad (4)$$

본 발명의 실시예에 따른 CKKS 암호 알고리즘은 평문의 근사값 및 그 유효숫자를 암호화하는 알고리즘이고, 유효숫자의 덧셈과 곱셈을 통해 동형성질이 구현된다. 같은 레벨을 가지는 두 암호문 $ct_1 = (c_1(x), d_1(x)) \pmod{q_t}$, $ct_2 = (c_2(x), d_2(x)) \pmod{q_t}$ 이 주어졌을 때, 동형 덧셈은 처음 두 암호문과 같은 레벨을 가지는 새로운 암호문이 된다. 식(5)는 동형 덧셈 연산의 식이다.

$$ct_{add} = (c_1(x) + c_2(x), d_1(x) + d_2(x)) \pmod{q_t} \quad (5)$$

본 발명의 실시예에 따른 두 암호문 $ct_1 = (c_1(x), d_1(x)) \pmod{q_t}$, $ct_2 = (c_2(x), d_2(x)) \pmod{q_t}$ 이 주어졌을 때, 동형 곱셈은 일차적으로는 처음 두 암호문과 같은 레벨을 가지는 새로운 암호문을 얻는다. 식(6)은 동형 곱셈의 식을 나타낸다. 앞서 곱셈 알고리즘을 통해 얻는 레벨이 l 인 새 암호문 $ct = (c(x), d(x)) \pmod{q_t}$ 을 그대로 복호화할 경우, 원래의 두 암호문의 평문의 곱셈에 p 를 곱한 값의 근사값이 나오게 된다. 이를 조정하기 위해 암호문

$ct \in (R/q_t)^2$ 의 각 계수를 $-\frac{q_t}{2} < \text{계수} \leq \frac{q_t}{2}$ 인 정수를 취한 뒤 실수 p 로 나누어 실계수 다항식으로 이루어진 새 암호문 $\frac{1}{p} \cdot ct$ 를 모듈러스 q_{t-1} 과 함께 얻는다. 그 후 새 모듈러스 q_{t-1} 에 대해 조정된 레벨이 $l-1$ 인 암호문 $ct_{r2} = [p^{-1} \cdot ct] \in (R/q_{t-1})^2$ 을 얻는다. 동형 곱셈은 조정된 레벨이 $l-1$ 인 암호문 ct_{r2} 를 주어진 두 암호문의 곱셈으로 한다.

$$f(x) = c_1(x) \cdot c_2(x)$$

$$g(x) = c_1(x) \cdot d_2(x) + c_2(x) \cdot d_1(x)$$

$$h(x) = d_1(x) \cdot d_2(x) \quad (6)$$

본 발명의 실시예에 따른 암호문 간의 연산, 특히 곱셈이 반복될 경우 새 암호문들의 모듈러스는 작아지게 되는데, 결과적으로 새 암호문들은 덧셈과 곱셈을 할 수 있는 횟수가 줄어든다. 이를 노후화된 암호문이라 한다. 평문을 갖 암호화한 암호문은 연산 가능한 횟수가 최대인 상태인데, 이처럼 연산횟수가 최대 상태인 암호문을 새 암호문이라고 한다. 재부팅(Bootstrapping)이란 노후화된 암호문을 새 암호문으로 교환해주는 알고리즘이다. CKKS 암호 알고리즘에서 사용하는 재부팅의 원리는 노후화된 암호문을 복호화한 뒤 다시 암호화하는 것이다. 단, 이 과정은 모두 암호화된 상태로 수행해야 하며 이를 재암호화(Recryption)이라고 한다. 재암호화는 암호문의 복호화와 복호문의 암호화 과정을 모두 덧셈과 곱셈으로 표현해야 한다. 또한, 재암호화가 발생시키는 노후화가 본래의 암호문에 영향을 미치지 않도록, 별도의 키를 이용하여 재암호화 해야 한다. CKKS 암호 알고리즘에서는 곱셈구현 단계에서 재부팅 알고리즘 고려하고 있으며, 곱셈 후 줄어든 암호문 공간을 q_t 로 확장하는 것으로 구현되어 있다.

본 발명의 실시예에 따른 CKKS 동형암호 알고리즘에서는 안전성을 고려하여 파라미터의 예시로 다음과 같이 기술한다. $\lambda = 128$ 일 때는 $N = 4096$, $\log_2(q_t) = 71$, $h = 64$, $\sigma = 3.2$ 의 파라미터를 제안하고, $\lambda = 192$ 일 때는 $N = 4096$, $\log_2(q_t) = 41$, $h = 64$, $\sigma = 3.2$ 의 파라미터를, $\lambda = 256$ 일 때는 $N = 8192$, $\log_2(q_t) = 53$, $h = 64$, $\sigma = 3.2$ 의 파라미터를 제안한다.

도 2는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화-복호화 아키텍처의 구조를 나타내는 도면이다.

도 2는 본 발명의 실시예에 따른 동형암호 표준화기구에서 제안한 복수의 표준에 맞는 $2^{12} \leq N \leq 2^{16}$ 인 다양한 길이(N)의 다항식을 지원하고 q, Level을 가변적으로 지원하는 동형암호를 위한 재구성형 CKKS-기반 암호화-복호화 아키텍처의 구조를 나타내는 도면이다.

본 발명의 실시예에 따른 동형암호를 위한 CKKS-기반 암호화 처리 장치(210)는 컨트롤 유닛(Control Unit)(211), 데이터 메모리(Data Memory)(212), 모듈러 스위치(Modular Switch)(213), 복수의 NTT 아키텍처(214), 모듈러 곱셈기(Modular Multiplier)(215), 모듈러 덧셈기(Modular Addition)(216), 모듈러 뺄셈기(Modular Subtractor), 및 INTT(Inverse NTT) 아키텍처를 포함하고 CKKS-기반 복호화 처리 장치(220)는 컨트롤 유닛(221), 데이터 메모리(222), 모듈러 곱셈기(223), 모듈러 덧셈기 아키텍처(224)를 포함한다.

본 발명의 실시예에 따른 동형암호를 위한 CKKS-기반 암호화 처리 장치(210)는 파라미터 길이(N), 모듈러스(q), 동형연산 가능 횟수(Level)을 가변적으로 지원하며 평문의 데이터를 암호화하여 암호문으로 변환한다.

본 발명의 실시예에 따른 동형암호를 위한 CKKS-기반 암호화 처리 장치(210)는 사용자로부터 파라미터 N, Level을 받아 컨트롤해주는 컨트롤 유닛(Control Unit)(211), 사용자에게 파라미터 Q, T, InvN, InvP를 받아 저장해주는 데이터 메모리(Data Memory)(212), 파라미터 N, q, Level이 가변적인 모듈러 스위치(Modular Switch)(213), 파라미터 N, Level이 가변적인 복수의 NTT 아키텍처(214), 파라미터 q가 가변적인 모듈러 곱셈기(Modular Multiplier)(215), 모듈러 덧셈기(Modular Addition)(216)를 포함한다.

본 발명의 실시예에 따른 가우시안 분포 에러 생성기(Gaussian Distribution)는 암호화시에 필요한 변수들을 임의로 생성하는 에러 생성 프로세서이다.

본 발명의 실시예에 따른 회전인자 생성기(Twiddle Factor Generator)는 다수의 회전인자(Twiddle Factor)(Ψ_{q^t})와 역회전인자(Inverse Twiddle Factor)(Ψ_P^{-1})를 생성해주는 장치이다.

본 발명의 실시예에 따른 키 생성기(Key Generator)는 보안 파라미터를 수신하여 암호화에 이용할 수 있는 공개키, 비밀키 및 평가키를 생성하는 키 생성 프로세서이다.

본 발명의 실시예에 따른 암호화 인코더(Encoder)는 다항식을 수신하여 암호화에 적합하게 변화시켜주고, 평문을 출력하여 암호화 결과 값과 모듈러 덧셈 연산을 하여 암호문을 만들어내는 인코더이다.

- [0053] 본 발명의 실시예에 따른 복호화 디코더(Decoder)는 두 암호문을 수신하여 비밀키와 모듈러 곱셈 연산과 모듈러 덧셈 연산을 하여 만들어진 평문을 원본 메시지로 변환하는 디코더이다.
- [0054] 본 발명의 실시예에 따른 NTT 코어(214)는 $N = 2^{12} \sim 2^{16}$ 차 다항식과 $q \leq 64$ 비트(Bit)내의 데이터에 대해 NTT 연산을 수행하는 2-병렬 MDF-기반 NTT 코어 장치이다.
- [0055] 본 발명의 실시예에 따른 모듈러 곱셈기(215)는 예러의 NTT 연산 결과와 공개키를 곱한 뒤 모듈러 연산을 진행한 결과를 출력할 수 있는 장치이다.
- [0056] 본 발명의 실시예에 따른 모듈러 덧셈기(216)는 모듈러 곱셈기(215)의 결과에 예러의 NTT 연산 후의 결과를 더하는 장치이다. 또한 이후 모듈러 스위치(213)의 결과와 인코더를 통해 변형해둔 평문을 더해 암호문을 출력하는 장치이다.
- [0057] 본 발명의 실시예에 따른 모듈러 스위치(Modular Switch)(213)는 키 크기의 데이터를 평문 크기의 데이터로 줄여주는 장치이다.
- [0058] 도 2를 참조하자면, 인코더는 Δ (Scaling factor)와 z (Original message)를 이용하여 암호화 처리 장치에서 사용할 수 있는 평문(Plaintext)로 변환하는 장치를 포함한다. 키 생성기는 1^λ 을 입력으로 암호화 처리 장치에서 사용할 수 있는 공개키(Public key), 비밀키(Secret key), 평가키(Evaluation key)를 생성하는 장치를 포함한다. 가우시안 분포 예러 생성기에서 생성되는 Z_{op} 는 $\{-1, 0, 1\}$, e_0, e_1 는 $\{-10 \sim 10\}$ 의 정수를 생성하는 프로세서이다. 가우시안 분포 예러 생성기의 출력을 빠른 다항식 곱셈 방법인 NTT를 이용하기 위해 RNS(Residue Number System)로 사용자가 원하는 모듈러스의 크기 $\{q_0, q_1, \dots, q_l\}$ 로 나뉘어 연산한다. RNS는 64비트가 넘는 큰 정수의 계산의 복잡성을 개선하기 위하여 사용하였다. 큰 수 하나를 RNS 기저 $\{q_0, q_1, \dots, q_l\}$ 나눈 나머지를 이용하는 연산이다. 나뉜 값은 빠른 다항식 곱셈을 이용하기 위해 NTT 연산을 수행한다. NTT 연산은 식(7)처럼 나타낼 수 있다.
- [0059]
$$A_i = \sum_{j=0}^{n-1} a_j w_n^{ij} \bmod q, \text{ for } i = 0, 1, \dots, n-1 \quad (7)$$
- [0060] 또한 Ring-LWE의 모든 연산은 $R_q = \mathbb{Z}_q[x]/f(x)$ 의 조건을 만족시키면서 동작하므로 다항식 $a(x)$ 와 $b(x)$ 는 식(8)과 같이 나타낼 수 있다.
- [0061]
$$a(x) = a_0 + a_1x + a_2x^2 + \dots + a_{n-1}x^{n-1} \quad (8-1)$$
- [0062]
$$b(x) = b_0 + b_1x + b_2x^2 + \dots + b_{n-1}x^{n-1} \quad (8-2)$$
- [0063] 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)로 이 프로세서에서는 $2^{12} \leq N \leq 2^{16}$ 의 가변적인 크기를 가진다. Q 는 $q \equiv 1 \pmod{2N}$ $q \equiv 1 \pmod{2N}$ 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 다항식간의 점(Point)별 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 $\{q_0, q_1, \dots, q_l\}$ 의 크기에 해당하는 키 크기의 데이터를 $\{q_0, q_1, \dots, q_{l-1}\}$ 크기처럼 특정 파라미터(Special prime)을 줄여 평문의 크기와 동일하게 만들어준다.
- [0064] 본 발명의 실시예에 따른 가변적인 길이의 N , q , Level을 지원하는 재구성형 CKKS-기반 암호화 아키텍처는 사용자로부터 원하는 파라미터 셋(Parameter set)($Q, T, \text{InvN}, \text{InvP}, \text{Level}, N$)을 입력 받아 파라미터에 맞게 가변적으로 암호화가 진행된다. Q 는 모듈러스 값이며 Level + 2의 개수만큼 사용자로부터 입력 받아 데이터 메모리에 저장한다. 예를 들어 Level이 2라면 (q_0, q_1, q_2, P) 총 4개의 모듈러스 값을 입력 받게 된다. T 는 Ratio 값

으로 $T = \left\lfloor \frac{2^{2N}}{Q} \right\rfloor$ 로 계산되며 가변적인 Barrett Modular Multiplication을 위해 입력 받는다. InvN 과 InvP 는 모듈러 역수(modular inverse)로 각각 식(9)을 만족하는 값을 가진다.

$$N \times \text{InvN} \bmod Q = 1$$

$$P \times \text{InvP} \bmod Q = 1 \quad (9)$$

Level(≥ 1)은 동형연산 가능 횟수로 동형연산시 늘어나는 노이즈(Noise)들을 관리할 수 있다. N은 다항계수정도(Poly Modulus Degree)로 $2^{12} \leq N \leq 2^{16}$ 의 범위를 지원한다. 사용자로부터 받은 Parameter set(Q, T, InvN, InvP)은 각각 데이터 메모리(Data Memory)에 저장된다. 데이터 메모리는 FPGA내의 BRAM을 사용하여 구현하였으며, 동형암호화를 진행하며 필요한 순간마다 컨트롤 유닛에서 읽기(Read) 신호를 통해 값을 읽는다. Parameter set(Level, N)은 컨트롤 유닛으로 들어가며, 컨트롤 유닛에서 Level과 N의 값에 따라 각기 다른 조합으로 암호화 모듈을 구성한다.

본 발명의 실시예에 따른 예러, 키, 회전인자, 평문을 생성하여 FPGA 내에서 구현할 수 있도록 메모리에 저장한다. 저장한 예러값 Z_{op} , e_0 , e_1 을 각각의 NTT_q 연산을 수행하고 Z_{op} 의 결과와 공개 키를 모듈러 곱셈기를 이용하여 곱한다. 그 후 곱셈의 결과와 각각의 e_0 , e_1 값들을 더해지게 되며 그 결과를 모듈러 스위치(Modular Switch) 연산을 수행하게 된다. 모듈러 스위치 모듈의 결과값에 인코딩된 평문 $m(X)$ 을 더하여 암호문 Ct_0 와 Ct_1 의 값을 얻게 된다. CPU 내에서 생성된 회전인자(psi) / 역회전인자(inverse psi) 값은 컨트롤 유닛을 통해 NTT와 모듈러 스위치로 입력이 들어간다.

도 3은 본 발명의 실시예에 따른 Level과 N에 따른 상세한 재구성형 CKKS-기반 암호화 아키텍처를 나타내는 도면이다.

레벨(Level)이 증가할수록 파라미터 q에 대한 암호화 연산이 한 번 더 반복되는 것을 알 수 있다. 따라서 Level은 재구성형 CKKS-기반 암호화 아키텍처의 깊이(Depth)와 같은 성분이라고 할 수 있다. 이러한 생각에서 착안해낸 아이디어는 반복문을 이용해 Level에 따라 암호화 연산을 반복해주면 Level에 대해 가변적으로 암호화가 가능하다는 것이다.

본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처는 N이 증가할수록 NTT(311, 312, 313)와 곱셈기(321, 322), 덧셈기(331, 332), 모듈러 스위치(Modular Switch)(340) 등 각 모듈의 암호화를 한번 진행하는데 걸리는 사이클(Cycle)의 수가 일정한 규칙 $f(N) = 2N + 45\log_2 N + 2$ [Cycles]으로 변하는 것을 알 수 있다. 따라서 N은 재구성형 CKKS-기반 암호화 아키텍처의 너비(Width)와 같은 성분이라고 볼 수 있다. 이러한 생각에서 착안해낸 아이디어는 Mux를 이용해 N에 따라 각 모듈이 걸리는 사이클 수만큼 컨트롤 유닛에서 다르게 제어하게 되면 $2^{12} \leq N \leq 2^{16}$ 이내의 그 어떤 N이라도 암호화가 가능하다는 것이다. 따라서 Mux를 이용해 N의 값을 판별한 후에, 반복문을 통해 Level+2만큼 암호화를 반복해주면 그 어떠한 N과 Level에 대해서도 재구성 가능하게 동작할 수 있다. 또한, 첫 번째 암호화 연산의 경우 모듈러 스위치내의 첫 번째 INTT 연산을 거쳐 BRAM에 저장된 후 출력이 되지 않기에 모듈러 스위치를 기점으로 암호문의 크기가 스페셜 프라임(Special Prime)의 비트 수만큼 작아지게 된다. 이는 키 레벨(Key Level)의 데이터에서 스페셜 프라임의 데이터를 제거한 데이터 레벨(Data Level)로 변환해주는 모듈러 스위치의 특징이다.

도 4는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 모듈러 스위치(Modular Switch) 아키텍처를 나타내는 도면이다.

도 4를 참조하면, 본 발명의 실시예에 따른 도 1과 도 2에서의 가변적인 길이의 N, q, Level을 지원하는 모듈러 스위치(Modular Switch) 아키텍처 블록도를 나타낸다. 상위 모듈인 재구성형 CKKS-기반 암호화 아키텍처와 마찬가지로 데이터 메모리(Data Memory)와 컨트롤 유닛(Control Unit)을 가지고 있어, 상위 모듈로부터 파라미터 셋(Parameter set)(Level, N)을 받으면 컨트롤 유닛 내에서 Mux를 이용해 N의 값을 판별한 뒤에 각 모듈의 사이클

수를 N에 맞게 계산한 다음, 반복 동작으로 Level+1 만큼 모듈러 스위치 연산을 반복해주어 N과 Level에 가변 적이게 동작하게 한다. Parameter set(Q, T, InvN, InvP) 역시 상위 모듈로부터 받게 되면 컨트롤 유닛에서 BRAM으로 구현한 데이터 메모리에 쓰기(Write) 신호를 제어하여 저장 후, 모듈러 스위치 연산 중 해당 파라미터 가 필요한 모듈이 있다면 컨트롤 유닛에서 읽기(Read) 신호를 제어하여 특정 파라미터만 읽게 된다.

[0076] 본 발명의 실시예에 따른 모듈러 스위치 아키텍처에서 가장 첫 번째로 보이는 $INTT_P$ 모듈(410)은 모듈러 스위치 연산 중 최초 1회만 동작하며, $INTT$ 모듈의 $outvld$ 가 On 되었을 때 $P > 1$ 값과 모듈러 덧셈(420) 연산을 하여 BRAM(430)에 쓰게다. 이때도 역시 컨트롤 유닛에서 BRAM의 Write 신호를 제어하여, Address 0번지부터 순차적으로 저장하게 된다. 이후, BRAM에 저장된 값들은 매번 ModSwich 연산마다 읽어와 연산에 쓰이게 된다. 첫 번째 모듈러 스위치 연산이 종료 후 BRAM(430)에 저장되어 있던 데이터들을 읽어 $mod q_i$ 연산을 수행한다. 그 후 나온 결과값과 $P > 1 mod q_i$ 을 모듈러 뺄셈(440) 연산을 통해 환 내에 존재할 수 있도록 한다. 그런 다음 NTT_{q_i} (450) 연산을 수행하게 된다. 이때에도 컨트롤 유닛에서 컨트롤 신호로 각 모듈러스에 맞는 Psi 값을 NTT 모듈 (450)에 입력해주게 된다. NTT 연산 결과 나온 값은, 모듈러 스위치 두 번째 입력 벡터들부터 함께 모듈러 뺄셈 (460) 연산을 진행하게 된다. 그 후 제일 마지막 단의 모듈러 곱셈기(470)를 통해 $P^{-1} mod q_0$ 값(즉, InvP)과 연산 후 결과값이 출력되게 된다. 여기까지 모듈러 스위치의 벡터 단위의 연산횟수 한번 진행한 것이므로, 총 (Level + 1)번의 반복을 통해 (Level + 1)개의 출력 벡터 값들을 얻을 수 있다.

[0078] 도 5는 본 발명의 실시예에 따른 Level과 N에 따른 상세한 모듈러 스위치(Modular Switch) 아키텍처를 나타내는 도면이다.

[0079] 본 발명의 실시예에 따른 재구성형 모듈러 스위치(Modular Switch) 아키텍처 역시 N이 증가할수록 NTT 와 곱셈기, 뺄셈기 등 각 모듈의 모듈러 스위치 연산을 한번 진행하는데 걸리는 사이클 수가 일정한 규칙 $f(N) = 2N + 45\log_2 N + 2$ [Cycles]으로 변하는 것을 알 수 있다. 따라서 N은 재구성형 모듈러 스위치 아키텍처의 Width 와 같은 성분이라고 볼 수 있다. 이러한 생각에서 착안해낸 아이디어는 Mux를 이용해 N에 따라 각 모듈이 걸리는 사이클 수만큼 컨트롤 유닛에서 다르게 제어하게 되면 $2^{12} \leq N \leq 2^{16}$ 이내의 그 어떤 N이라도 모듈러 스위치 연산이 가능하다는 것이다. 따라서 Mux를 이용해 N의 값을 판별한 후에, 반복문을 통해 Level + 1만큼 모듈러 스위치 연산을 반복해주면 그 어떠한 N과 Level에 대해서도 재구성 가능하게 동작할 수 있다.

[0081] 도 6은 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 2-병렬 MDF-기반 재구성형 NTT 아키텍처를 나타내는 도면이다.

[0082] 본 발명의 실시예에 따른 NTT 다항식 곱셈기 장치는 NTT 아키텍처를 이용하여 고성능 저복잡도의 다항식 곱셈 연산을 수행하는 장치로 PE1(Processing Element 1), PE2 (Processing Element 2), MUX를 포함한다. PE1과 PE2 는 입력 받은 다항식을 CT(Cooley-Tukey) 알고리즘으로 버터플라이(Butterfly) 연산을 수행하며 MUX는 분기를 통해 최대 16개의 단계를 결정하는 방식으로 다항식의 길이 파라미터 N에 대하여 $N=2^{12}, 2^{13}, 2^{14}, 2^{15}, 2^{16}$ 의 N을 가변적으로 지원한다.

[0083] 본 발명의 실시예에 따른 복수의 NTT 아키텍처는 복수의 길이(N)와 모듈러스(q)를 갖는 가변적인 파라미터의 다항식을 수신하여 NTT 형태의 다항식으로 변환한다.

[0084] 본 발명의 실시예에 따른 복수의 NTT 아키텍처 각각은 PE1과 PE2, PE 내에서 회전인자(Twiddle Factor)와 다항식의 계수를 곱해주는 모듈러 곱셈기(Modular Multiplier), CT 알고리즘의 버터플라이 연산을 수행하는 모듈러 덧셈기(Modular Adder)와 모듈러 뺄셈기(Modular Subtractor)를 포함하며, 회전인자를 저장하기 위한 메모리 구조를 포함한다.

[0085] 본 발명의 실시예에 따른 복수의 NTT 아키텍처는 반복적으로 이용되는 다항식 곱셈을 수행하기 위해 NTT 연산을 수행하는 파이프라인이 가능한 2-병렬 MDF-기반 NTT 아키텍처이고, 상기 2-병렬 MDF-기반 NTT 아키텍처는 CT(Cooley-Tukey) 알고리즘을 사용하여 비트 역순과정을 생략할 수 있다.

[0086] 본 발명의 실시예에 따른 복수의 NTT 아키텍처는 MUX를 이용한 단계 조절을 통하여 복수의 길이(N)에 대한 가변성을 제공하며, 파라미터 N에 대하여 PE의 반복 횟수를 조절함으로써 효율적으로 동작한다.

[0087] 본 발명의 실시예에 따른 복수의 NTT 아키텍처 각각의 PE1은 모듈러 곱셈기를 이용하여 회전인자와 다항식 계수를 곱하고, 모듈러 덧셈기, 모듈러 뺄셈기를 이용하여 CT 알고리즘을 수행하며, FIFO(First-In-First-Output)를 이용한 피드백(Feed-back) 구조로 효율적으로 다항식을 NTT 변환한다. FIFO의 깊이는 각 단계에 효율적인 2의 승수로 설정된다.

[0088] 본 발명의 실시예에 따른 복수의 NTT 아키텍처 각각의 PE2는 모듈러 곱셈기를 이용하여 회전인자와 다항식 계수를 곱하고, 모듈러 덧셈기, 모듈러 뺄셈기를 이용하여 CT 알고리즘을 수행하며, 파이프라인이 가능한 2-병렬 구조로 FIFO에 의한 피드백 없이 동작한다.

[0089] 본 발명의 실시예에 따른 복수의 NTT 아키텍처 각각의 메모리 구조는 각 단계에서 필요한 회전인자에 개수에 맞춘 독립적인 메모리 뱅크(Bank)로 이루어지며, CT 알고리즘, GS(Gentleman-Sande) 알고리즘에서 비트 역순 과정을 생략하기 위한 순서로 저장된다.

[0090] 본 발명의 실시예에 따른 NTT 아키텍처는 빠른 다항식 곱셈을 이용하기 위해 NTT 연산을 수행한다. NTT 연산은 식 (10)과 같이 나타낼 수 있다:

$$A_i = \sum_{j=0}^{n-1} a_j w_n^{ij} \bmod q, \text{ for } i = 0, 1, \dots, n-1 \quad (10)$$

[0092] 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 식(11)과 같이 나타낼 수 있다:

$$a(x) = a_0 + a_1x + a_2x^2 + \dots + a_{n-1}x^{n-1} \quad (11-1)$$

$$b(x) = b_0 + b_1x + b_2x^2 + \dots + b_{n-1}x^{n-1} \quad (11-2)$$

[0095] 벡터(Vector) $\mathbf{a} = a_0, \dots, a_{n-1}$ 는 다항식 $a(x)$ 의 계수이며, 숫자 n 은 2의 자승이다. $q \equiv 1 \pmod{2N}$ 을 만족하는 소수(Prime number)이다. 수학적 1에서 n 과 w 는 $n \cdot n^{-1} \equiv 1 \bmod q$, $w \cdot w^{-1} \equiv 1 \bmod q$ 의 조건을 만족하는 숫자로 구성된다. 모듈러 곱셈기는 $AB = A \cdot B \pmod{q}$ 와 같은 연산을 수행하며 다항식 간의 포인트-와이즈(Point-wise) 곱셈을 진행하며 정해진 q 값에 대해 모듈러 연산을 진행한다.

[0096] 본 발명의 실시예의 INTT 아키텍처는 다항식 곱셈 이후 NTT 형태의 다항식에서 원래 형태의 다항식으로 변환하기 위하여 INTT 연산을 수행한다. INTT 연산은 식(12)과 같이 나타낼 수 있다:

$$a_i = n^{-1} \sum_{j=0}^{n-1} A_j w_n^{-ij} \bmod q, \text{ for } i = 0, 1, \dots, n-1 \quad (12)$$

[0098] 다항식의 곱셈 연산은 식 (1), (2), (3)을 이용하여 아래의 식(13)로 표현할 수 있다:

$$a(x) * b(x) = INTT(NTT(a) \cdot NTT(b)) \quad (13)$$

[0100] 식(13)의 좌변의 다항식 곱셈은 합성곱(Convolution) 연산으로 $O(N^2)$ 의 시간복잡도를 가지나, 우변의 NTT 기반의 다항식 곱셈은 포인트-와이즈 곱셈 연산으로 $O(N \log N)$ 의 시간 복잡도를 가져 고성능 저복잡도의 구조를 갖는다.

[0102] 도 7은 본 발명의 실시예에 따른 NTT 아키텍처를 구성하는 PE1(Processing Element 1) 구조를 나타내는 도면이다.

[0103] NTT 다항식 곱셈기 장치는 NTT 아키텍처의 CT 알고리즘을 수행하는 장치로 모듈러 곱셈기, 모듈러 덧셈기, 모듈러 뺄셈기, 메모리, FIFO를 포함한다.

[0104] 본 발명의 실시예에 따른 NTT의 PE1은 하나의 데이터 경로를 FIFO를 이용한 피드백 구조를 이용하여, 메모리에

서 회전인자(Twiddle Factor)를 수신하여 입력 받은 다항식의 계수와 모듈러 곱셈기를 통하여 곱하고, 모듈러 덧셈기 및 모듈러 뺄셈기를 이용하여 CT 알고리즘을 수행한다. 각 단계의 PE1의 FIFO의 깊이는 단계 1에서 32768개, 단계 2에서 16384개, 단계 3에서 8192개로 1/2씩 줄어든다.

[0106] 도 8은 본 발명의 실시예에 따른 NTT 아키텍처를 구성하는 PE2(Processing Element 2) 구조를 나타내는 도면이다.

[0107] NTT 다항식 곱셈기 장치는 NTT 아키텍처의 CT 알고리즘을 수행하는 장치로 모듈러 곱셈기, 모듈러 덧셈기, 모듈러 뺄셈기, 메모리를 포함한다.

[0108] 본 발명의 실시예에 따른 NTT의 PE2은 2개의 데이터 경로를 이용하여 2개의 다항식의 계수를 입력 받아, 메모리에서 회전인자(Twiddle Factor)를 수신하여 입력 받은 다항식의 계수와 모듈러 곱셈기를 통하여 곱하고, 모듈러 덧셈기 및 모듈러 뺄셈기를 이용하여 CT 알고리즘을 수행한다.

[0110] 도 9는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 2-병렬 MDF-기반 재구성형 INTT 아키텍처를 나타내는 도면이다.

[0111] NTT 다항식 곱셈기 장치는 INTT 아키텍처를 이용하여 고성능 저복잡도의 다항식 곱셈 연산을 수행하는 장치로 PE1, PE2, DEMUX를 포함한다. PE1과 PE2는 입력 받은 다항식을 GS 알고리즘으로 Butterfly 연산을 수행하며 DEMUX는 분기를 통해 최대 16개의 단계를 결정하는 방식으로 다항식의 길이 파라미터 N 에 대하여 $N=2^{12}$, 2^{13} , 2^{14} , 2^{15} , 2^{16} 의 N 을 가변적으로 지원한다.

[0113] 도 10은 본 발명의 실시예에 따른 INTT 아키텍처를 구성하는 PE1(Processing Element 1) 구조를 나타내는 도면이다.

[0114] NTT 다항식 곱셈기 장치의 INTT 아키텍처의 GS 알고리즘을 수행하는 장치로 모듈러 곱셈기, 모듈러 덧셈기, 모듈러 뺄셈기, 메모리, FIFO를 포함한다.

[0115] 본 발명의 실시예에 따른 INTT의 PE1은 하나의 데이터 경로를 FIFO를 이용한 피드백 구조를 이용하여, 모듈러 덧셈기 및 모듈러 뺄셈기를 이용하여 GS 알고리즘을 수행하고, 메모리에서 회전인자(Twiddle Factor)를 수신하여 입력 받은 다항식의 계수와 모듈러 곱셈기를 통하여 곱한다. 각 단계의 PE1의 FIFO의 깊이는 단계 1에서 1개, 단계 2에서 2개, 단계 3에서 4개로 2배씩 늘어난다.

[0117] 도 11은 본 발명의 실시예에 따른 INTT 아키텍처를 구성하는 PE2(Processing Element 2) 구조를 나타내는 도면이다.

[0118] NTT 다항식 곱셈기 장치의 INTT 아키텍처의 GS 알고리즘을 수행하는 장치로 모듈러 곱셈기, 모듈러 덧셈기, 모듈러 뺄셈기, 메모리를 포함한다.

[0119] 본 발명의 실시예에 따른 INTT의 PE2은 2개의 데이터 경로를 이용하여 2개의 다항식의 계수를 입력 받아, 메모리에서 회전인자(Twiddle Factor)를 수신하여 입력 받은 다항식의 계수와 모듈러 곱셈기를 통하여 곱하고, 모듈러 덧셈기 및 모듈러 뺄셈기를 이용하여 GS 알고리즘을 수행한다.

[0121] 도 12는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 모듈러 곱셈기(Modular Multiplier)의 구조를 나타내는 도면이다.

[0122] 본 발명의 실시예에 따른 모듈러 곱셈기는 바렛 모듈러(Barrett Modular) 알고리즘을 이용한다. 가변적인 파라미터 q 를 값을 사용하였으며, 입력 A , B 는 정수 곱셈기를 이용하여 $A*B$ 의 결과를 갖게 된다. 그 결과는 복잡도를 낮추기 위한 분할-정복 알고리즘을 통해 T 와 q 의 보조 곱셈기에 입력되고, 연산한 결과와 덧셈 혹은 뺄셈을 진행한다. 또한 Q 보다 큰 값에 한하여 Q 값을 뺄셈한 결과를 도출한다.

- [0124] 도 13은 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처를 구성하는 모듈러 덧셈기(Modular Addition)의 구조를 나타내는 도면이다.
- [0125] 본 발명의 실시예에 따른 모듈러 덧셈기는 모듈러 연산의 일종으로 입력 in1, in2의 덧셈 연산의 결과를 q보다 작은 링(Ring)으로 축소시켜주는 역할을 한다. 유한체 내의 존재하는 in1, in2에 대하여 비교 연산을 통해 모듈러 덧셈 연산을 수행한다.
- [0127] 도 14는 본 발명의 실시예에 따른 모듈러 스위치(Modular Switch)와 NTT 아키텍처를 구성하는 모듈러 뺄셈기(Modular Subtractor)의 구조를 나타내는 도면이다.
- [0128] 본 발명의 실시예에 따른 모듈러 뺄셈기는 모듈러 연산의 일종으로 입력 in1, in2의 뺄셈 연산의 결과를 q보다 작은 링(Ring)으로 축소시켜주는 역할을 한다. 유한체 내의 존재하는 in1, in2에 대하여 비교 연산을 통해 모듈러 뺄셈 연산을 수행한다.
- [0130] 도 15는 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처와 모듈러 스위치(Modular Switch)의 타이밍을 나타내는 도면이다.
- [0132] 도 16은 본 발명의 실시예에 따른 재구성형 CKKS-기반 암호화 아키텍처의 동작 방법을 설명하기 위한 흐름도이다.
- [0133] 제안하는 재구성형 CKKS-기반 암호화 아키텍처의 동작 방법은 가우시안 분포 에러 생성기(Gaussian Distribution)를 통해 암호화 처리에 필요한 에러(Z_{op} , e_0 , e_1)를 임의로 생성하는 단계, 회전인자 생성기(Twiddle Factor Generator)를 통해 암호화 처리에 필요한 다수의 회전인자(Ψ_{q1})와 역회전인자(Ψ_p^{-1})를 생성하는 단계, 키 생성기(Key Generator)를 통해 암호 파라미터를 수신하여 공개키, 비밀키 및 평가키를 생성하는 단계, 부호화기(Encoder)를 통해 다항식을 수신하여 암호화에 적합한 평문으로 부호화하는 단계 및 NTT 코어, 모듈러 스위치(Modular Switch), 모듈러 곱셈기(Modular Multiplier), 모듈러 덧셈기(Modular Addition)를 포함하는 동형 암호화(Homomorphic Encryption) 프로세서를 통해 Parameter, 공개키, 에러, Twiddle Factor, 평문을 수신하여 암호화하는 단계를 포함한다.
- [0134] 상기 동형 암호화 처리 장치를 통해 파라미터, 공개키, 에러, 회전인자, 평문을 수신하여 암호화하는 단계는 컨트롤 유닛(Control Unit)을 통해 Parameter(N, Level)을 판별하여 암호화 처리의 반복횟수, 한 사이클의 길이 f(N)을 결정하는 단계(1610), 컨트롤 유닛(Control Unit)을 통해 회전인자 생성기(Twiddle Factor Generator) 내에서 생성된 회전인자(Ψ_{q1})와 역회전인자(Ψ_p^{-1}) 값을 NTT/INTT와 모듈러 스위치(Modular Switch) 모듈로 입력하고, 데이터 메모리(Data Memory)를 통해 암호화의 특정 사이클 마다 Parameter(q, T, InvN, InvP)를 읽고 쓰는 단계(1620), 가우시안 분포 에러 생성기(Gaussian Distribution)를 통해 받은 에러 값(Z_{op} , e_0 , e_1)의 NTT 연산 수행하는 단계(1630), Z_{op} 의 NTT 연산 결과를 키 생성기에서 받은 Public Key(PK_0 , PK_1)와 모듈러 곱셈 연산 수행하는 단계(1640), 모듈러 곱셈 결과와 e_0 , e_1 의 NTT 연산 결과를 모듈러 덧셈 연산 수행하는 단계(1650), 모듈러 스위치(Modular Switch)를 통해 모듈러 덧셈 결과 $\tilde{ap}/\tilde{aq}$ 의 Parameter(N, q, Level)가 재구성 가능한 키 레벨의 데이터를 데이터 레벨로 변환해주는 단계(1660); 모듈러 스위치 연산 결과와 인코더로부터 받은 평문 m(X)를 모듈러 덧셈 연산 통해 암호문 출력하는 단계(1680)를 포함한다.
- [0135] 상기 컨트롤 유닛을 통해 Parameter(N, Level)을 판별하여 암호화 처리의 반복횟수, 한 사이클의 길이 f(N)을 결정하는 단계는 Parameter set(N, Level)을 받으면 컨트롤 유닛(Control Unit)내에서 Mux를 이용해 N의 값을 판별한 뒤에 각 모듈의 사이클 수를 N에 맞게 $f(N) = 2N + 45\log_2 N + 2$ [Cycles]로 계산한 다음, 반복 동작으로 Level + 2만큼 암호화 처리 연산을 반복해주어 N과 Level에 재구성 가능하게 동작한다.
- [0136] 상기 컨트롤 유닛(Control Unit)을 통해 회전인자 생성기(Twiddle Factor Generator) 내에서 생성된 회전인자

(Ψ_{q1})와 역회전인자(Ψ_p^{-1}) 값을 NTT/INTT와 모듈러 스위치(Modular Switch) 모듈로 입력이 들어가는 단계는 NTT/INTT와 모듈러 스위치(Modular Switch)가 회전인자(Ψ_{q1})와 역회전인자(Ψ_p^{-1}) 값을 쓰는(Write) 특정 사이클에서 컨트롤 유닛(Control Unit)을 통해 BRAM에 값을 저장한다.

[0137] 상기 데이터 메모리(Data Memory)를 통해 암호화의 특정 사이클마다 Parameter(q , T , $InvN$, $InvP$)를 읽고 쓰는 단계는 사용자로부터 받은 Parameter set(Q , T , $InvN$, $InvP$)을 초기 (Level+2) [cycles] 동안 Level + 2의 개수만큼 사용자로부터 입력 받아 컨트롤 유닛(Control Unit)에서 쓰기 신호를 제어하여 데이터 메모리에 저장하고, 동형 암호화를 진행하며 해당 파라미터가 필요한 순간마다 컨트롤 유닛에서 읽기신호를 제어하여 특정 파라미터 값을 읽는다.

[0138] 본 발명의 실시예에 따른 복호화 처리 장치는 복수의 Parameter(N , q , Level)를 갖는 가변적인 암호문과 비밀키를 수신하여 컨트롤 유닛(Control Unit), 데이터 메모리(Data memory)를 모듈러 덧셈기, 모듈러 곱셈기를 사용하여 원본 메시지로 변환한다.

[0140] 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, FPGA(field programmable gate 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.

[0141] 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, 가상장치(virtual equipment), 컴퓨터 저장 매체 또는 장치에 구체화(embodiment)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.

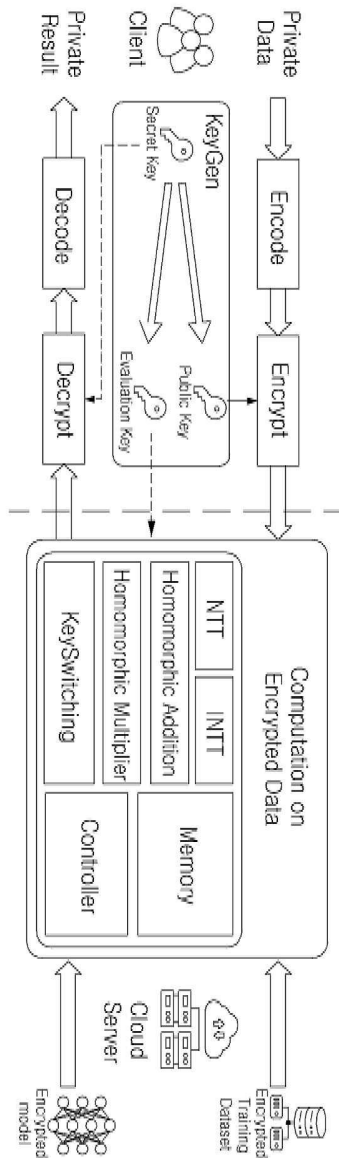
[0142] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic 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.

[0143] 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.

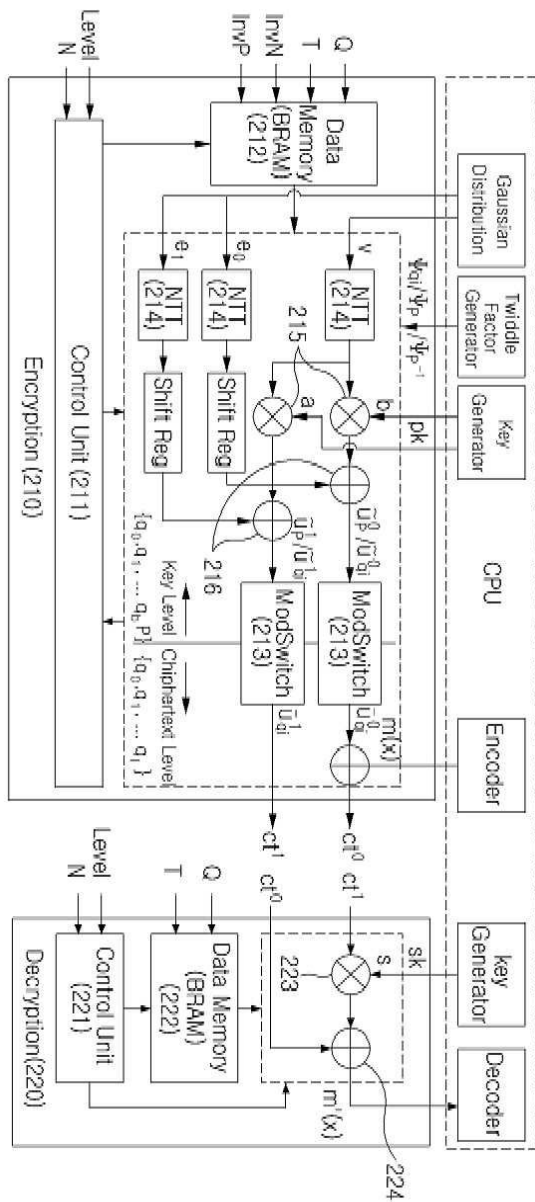
[0144] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

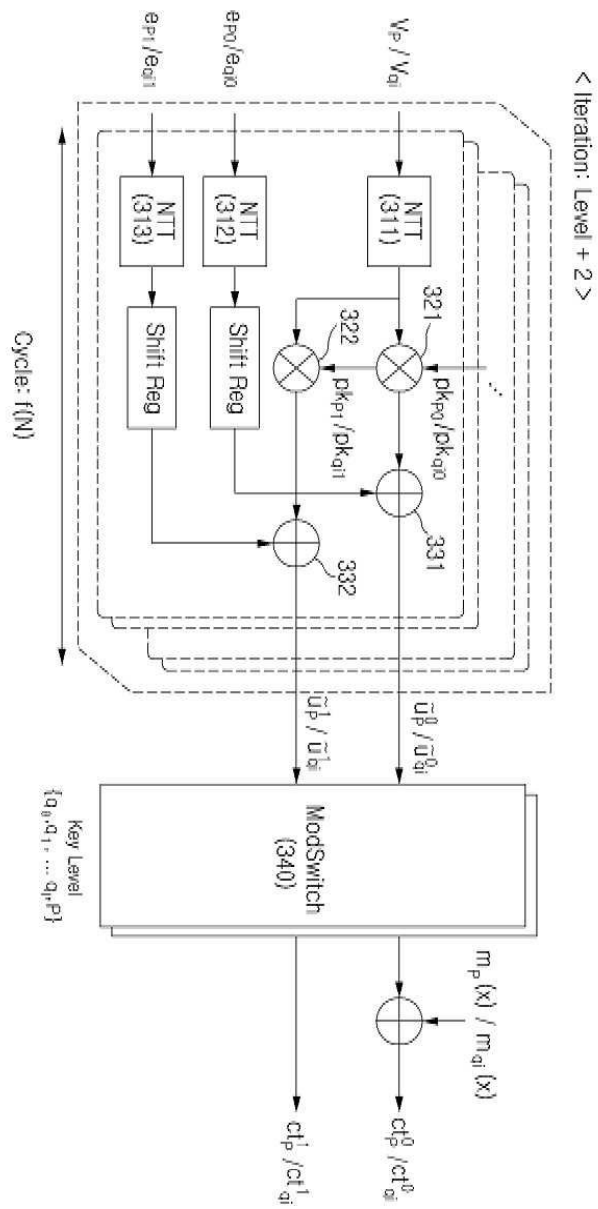
도면1



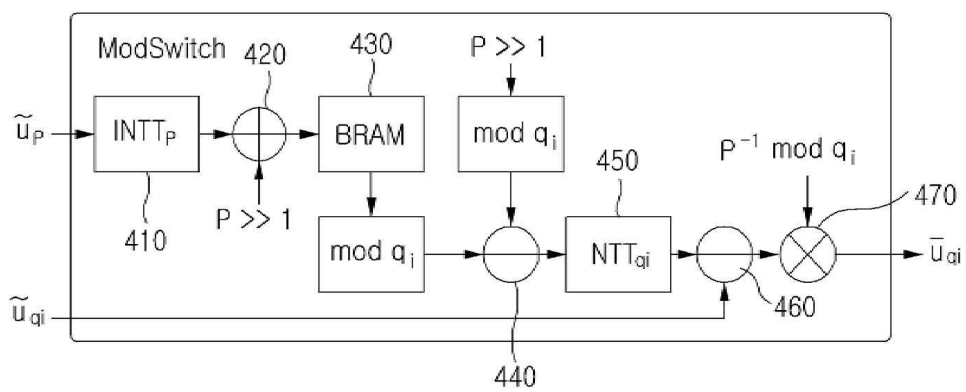
도면2



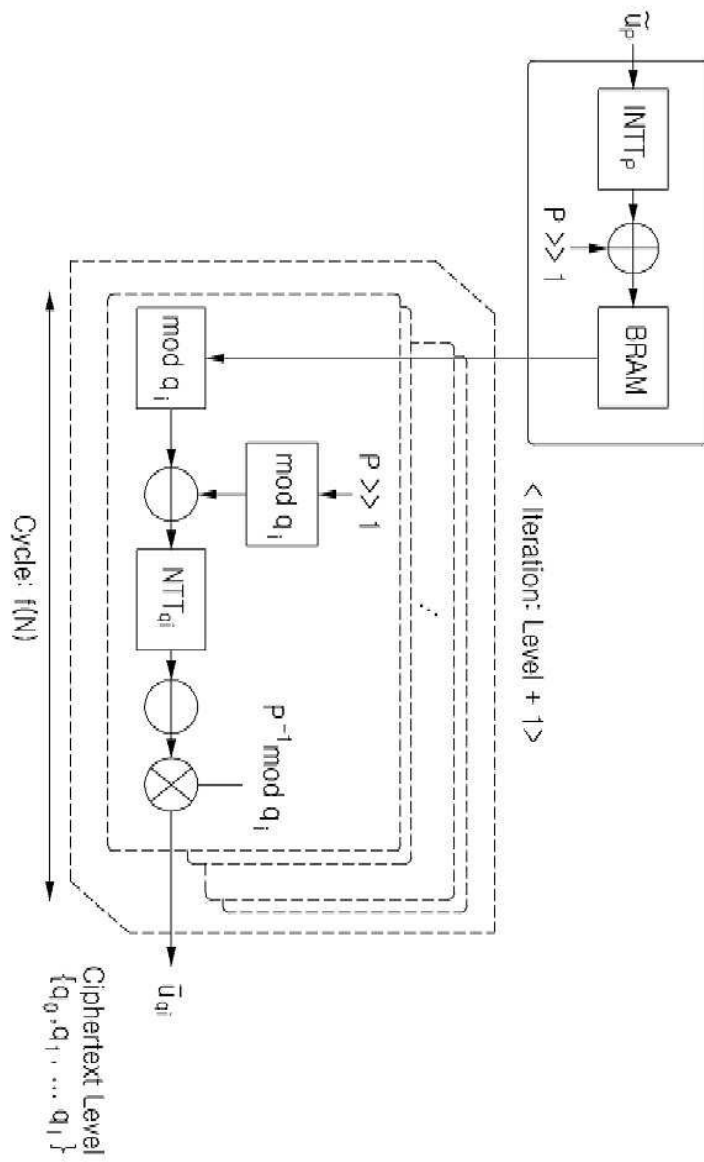
도면3



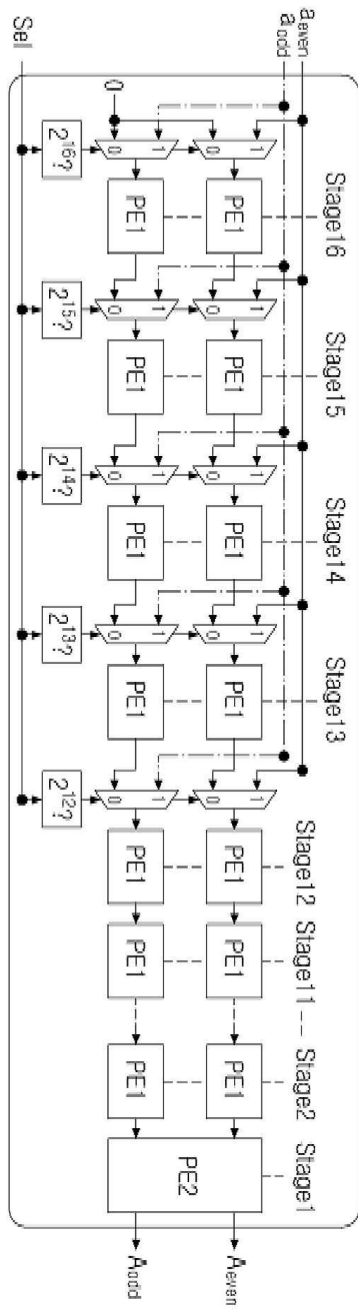
도면4



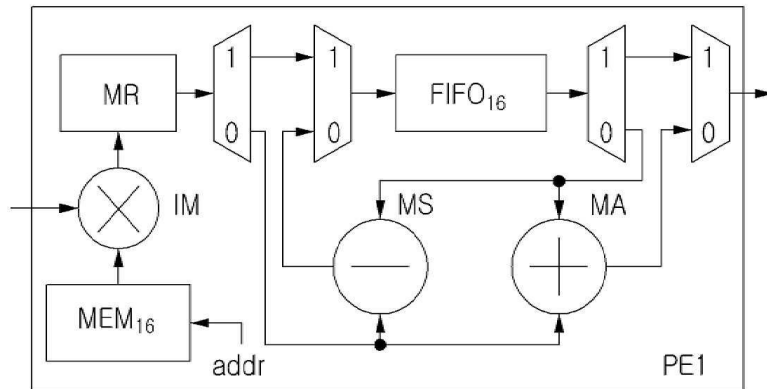
도면5



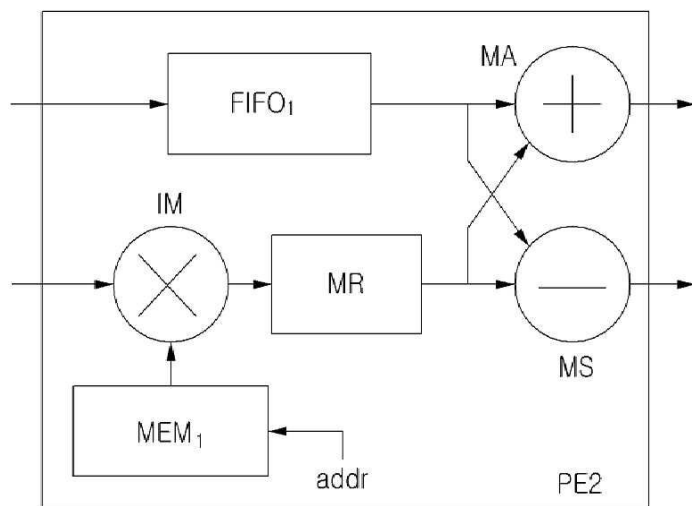
도면6



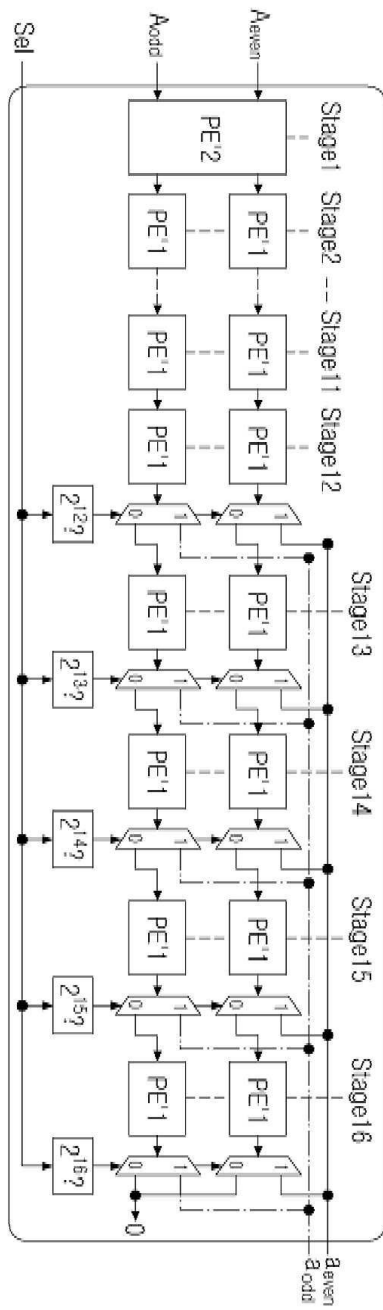
도면7



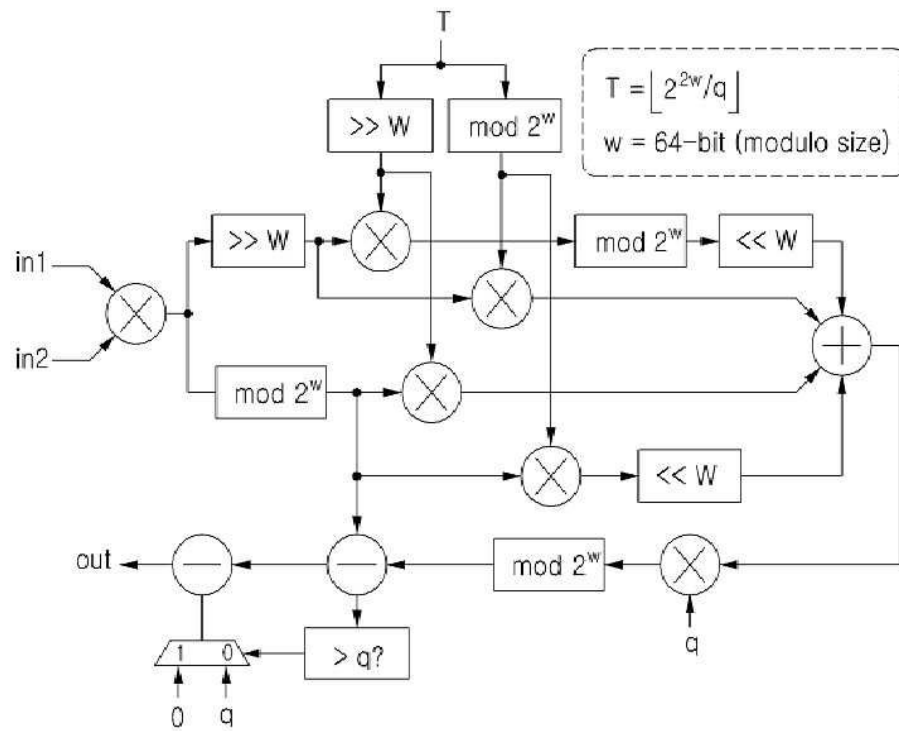
도면8



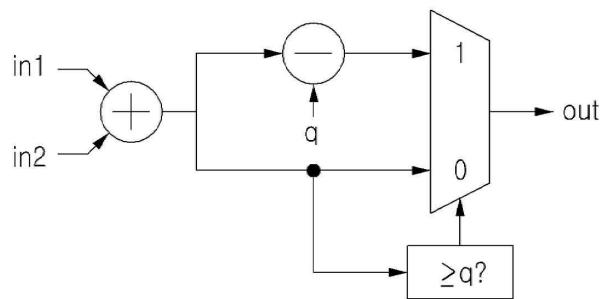
도면9



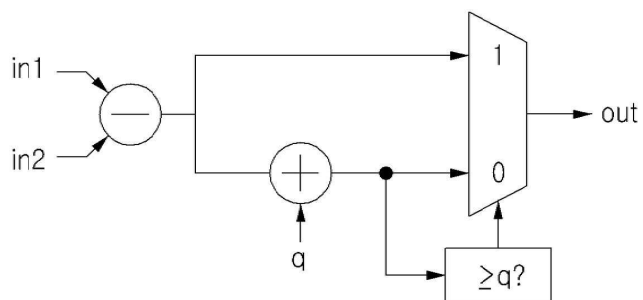
도면12



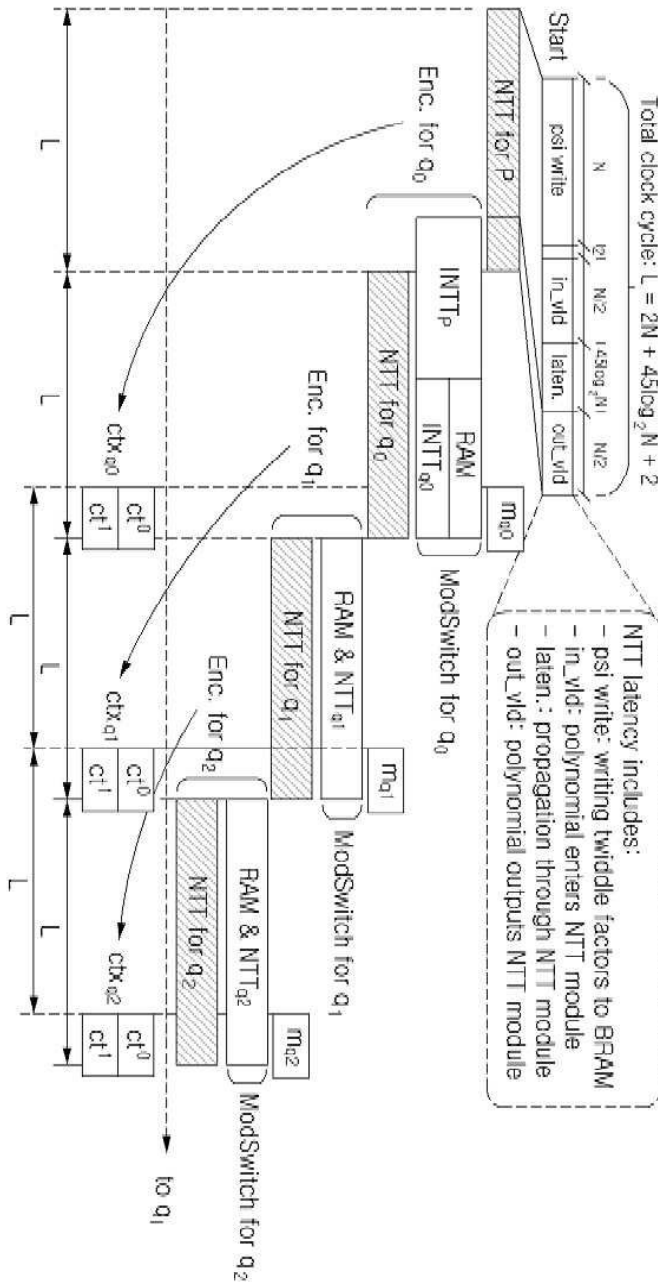
도면13



도면14



도면15



도면16

