



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2008-0099501
(43) 공개일자 2008년11월13일

(51) Int. Cl.

H03M 13/01 (2006.01) H03M 13/00 (2006.01)

(21) 출원번호 10-2007-0045111

(22) 출원일자 2007년05월09일

심사청구일자 2007년05월09일

(71) 출원인

한국전자통신연구원

대전 유성구 가정동 161번지

인하대학교 산학협력단

인천 남구 용현동 253 인하대학교

(72) 발명자

신중윤

대전 유성구 가정동 236-1 기숙사 신2-308

이한호

인천 연수구 옥련동 현대2차아파트 202-904

(뒷면에 계속)

(74) 대리인

리엔특허법인

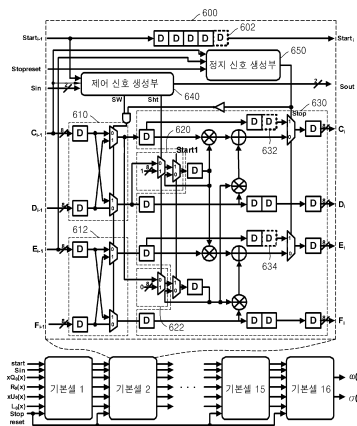
전체 청구항 수 : 총 6 항

(54) 고속 리드-솔로몬 복호기의 수정 유클리드 알고리즘 연산회로 및 그 방법

(57) 요약

고속 리드-솔로몬 복호기의 수정 유클리드 알고리즘 연산 회로 및 그 방법이 개시된다. 수정 유클리드 알고리즘의 각 기본셀의 교환 연산, 쉬프트 연산 및 다항식 연산을 제어하기 위한 제어신호 생성을 위한 유한체 상태 기계(Finite State Machine) 및 정지신호 생성을 위한 유한체 상태 기계를 이용함으로써, 종래의 차수 비교 및 계산 부분을 제거하여 저면적의 효율적인 리드-솔로몬 복호기의 구현이 가능하다.

대표도 - 도6



(72) 발명자

이승범

인천 남구 학익1동 민경빌라 101호

고제수

대전 유성구 전민동 엑스포아파트 302-1203

특허청구의 범위

청구항 1

신드롬 다항식을 기초로 상기 오류 값 다항식 및 오류 위치 다항식을 구하기 위한, 다수의 기본셀들로 구성된 시스토크-어레이 구조의 수정 유클리드 알고리즘 연산 회로에 있어서,

상기 기본셀들은,

다항식 C_{i-1} 의 값 및 다항식 C_{i-1} 와 D_{i-1} 의 차수를 기초로 결정되는 다항식 C_{i-1} 및 D_{i-1} 의 교환 연산 및/또는 쉬프트 연산의 수행여부에 대한 유한 상태 기계(finite state machine)를 기초로, 교환 연산 및/또는 쉬프트 연산에 대한 제어 신호를 발생하는 제어 신호 생성부; 및

상기 제어 신호에 따라 다항식 C_{i-1} , D_{i-1} , E_{i-1} , F_{i-1} 에 대한 교환 연산 또는/및 쉬프트 연산을 수행한 후 다항식 연산을 수행하는 연산부;를 각각 포함하고,

상기 기본셀들 중 첫 번째 기본셀의 입력 다항식 C_0 는 신드롬 다항식에 x 를 곱한 값, D_0 는 오류 정정 가능한 심볼 수의 2배의 차수를 가지는 값, E_0 는 x , 그리고, F_0 는 0이며, 상기 기본셀들 중 짝수 번째 기본셀들은 출력되는 다항식 C_i 및 D_i 의 차수를 1씩 낮추는 것을 특징으로 하는 수정 유클리드 알고리즘 연산 회로.

청구항 2

제 1항에 있어서,

상기 다항식 C_{i-1} 의 차수와 상기 다항식 E_{i-1} 의 차수에 따른 상태 전이를 나타내는 유한 상태 기계에서, 상기 다항식 C_{i-1} 의 차수가 상기 다항식 E_{i-1} 의 차수보다 작은 상태가 되면 정지 신호를 발생하는 정지신호 발생부;를 더 포함하고,

상기 기본셀들은 상기 정지신호가 발생하면, 출력 다항식 C_i 및 E_i 를 각각 오류 값 다항식 및 오류 위치 다항식으로 출력하고 연산을 종료하는 것을 특징으로 하는 수정 유클리드 알고리즘 연산 회로.

청구항 3

제 1항에 있어서, 상기 제어신호 발생부는,

상기 다항식 C_{i-1} 의 값이 0이면 쉬프트 연산에 대한 제어신호를 발생하고, 상기 다항식 C_{i-1} 의 값이 0이 아니고 상기 다항식 C_{i-1} 와 다항식 D_{i-1} 의 차수가 같으면 교환 연산에 대한 제어신호를 발생하는 조건을 만족하도록 각 상태에서의 제어신호 발생 조건을 기술한 유한 상태 기계를 기초로, 이전 기본셀의 상태와 입력 다항식 C_{i-1} 의 값에 따라 상태를 전이하고 해당하는 제어신호를 발생하는 것을 특징으로 하는 수정 유클리드 알고리즘 연산 회로.

청구항 4

신드롬 다항식을 기초로 상기 오류 값 다항식 및 오류 위치 다항식을 구하기 위한, 다수의 기본셀들로 구성된 시스토크-어레이 구조의 수정 유클리드 알고리즘 연산 방법에 있어서,

다항식 C_{i-1} 의 값 및 다항식 C_{i-1} 와 D_{i-1} 의 차수를 기초로 결정되는 다항식 C_{i-1} 및 D_{i-1} 의 교환 연산 및/또는 쉬프트 연산의 수행여부에 대한 유한 상태 기계(finite state machine)를 기초로, 교환 연산 및/또는 쉬프트 연산에 대한 제어 신호를 발생하는 단계;

상기 제어 신호에 따라 다항식 C_{i-1} , D_{i-1} , E_{i-1} , F_{i-1} 에 대한 교환 연산 또는/및 쉬프트 연산을 수행한 후 다항식 연산을 수행하는 단계; 및

상기 제어신호 발생 단계 및 상기 연산 수행 단계를 반복하여 수행하며, 짝수 번째 연산 수행단계에서는 연산 결과로서 출력되는 다항식 C_i 및 D_i 의 차수를 1씩 낮추는 단계;를 포함하고,

상기 기본셀들 중 첫 번째 기본셀의 입력 다항식 C_0 는 신드롬 다항식에 x 를 곱한 값, D_0 는 오류 정정 가능한 심볼 수의 2배의 차수를 가지는 값, E_0 는 x , 그리고, F_0 는 0인 것을 특징으로 하는 수정 유클리드 알고리즘 연산 방법.

청구항 5

제 4항에 있어서,

상기 다항식 C_{i-1} 의 차수와 상기 다항식 E_{i-1} 의 차수에 따른 상태 전이를 나타내는 유한 상태 기계에서, 상기 다항식 C_{i-1} 의 차수가 상기 다항식 E_{i-1} 의 차수보다 작은 상태가 되면 정지 신호를 발생하는 단계;

상기 정지신호가 발생하면, 현재의 출력 다항식 C_i 및 E_i 를 각각 오류 값 다항식 및 오류 위치 다항식으로 출력하기 위해 연산을 종료하고 다항식의 차수를 조정하는 단계;를 더 포함하는 것을 특징으로 하는 수정 유클리드 알고리즘 연산 방법.

청구항 6

제 4항에 있어서, 상기 제어신호 발생 단계는,

상기 다항식 C_{i-1} 의 값이 0이면 쉬프트 연산에 대한 제어신호를 발생하고, 상기 다항식 C_{i-1} 의 값이 0이 아니고 상기 다항식 C_{i-1} 와 다항식 D_{i-1} 의 차수가 같으면 교환 연산에 대한 제어신호를 발생하는 조건을 만족하도록 각 상태에서의 제어신호 발생 조건을 기술한 유한 상태 기계를 기초로, 이전 기본셀의 상태와 입력 다항식 C_{i-1} 의 값에 따라 상태를 전이하고 해당하는 제어신호를 발생하는 단계;를 포함하는 것을 특징으로 하는 수정 유클리드 알고리즘 연산 방법.

명세서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

- <10> 본 발명은 고속 리드-솔로몬 복호기의 수정 유클리드 알고리즘 연산 회로 및 그 방법에 관한 것이다.
- <11> 일반적으로, 리드-솔로몬 코드는 마그네틱, 광 저장매체, 유선 및 위성 통신 등 다양한 응용분야에 널리 쓰이는 Forward Error Correction(FEC) 기술이다. 리드-솔로몬 부호를 나타내는 전형적인 형태는 $RS(n,k,t)$ 이다. 여기서, n 은 부호 심볼 수, k 는 정보 심볼 수, t 는 오류 정정 가능 심볼 수를 나타내며, $t=(n-k)/2$ 이다. 따라서 $RS(255,239)$ 에서 $t=8$ 이다.
- <12> 도 1은 종래의 리드-솔로몬 복호기의 구조를 도시한 도면이다.
- <13> 도 1을 참조하면, 신드롬 다항식 계산 블록(Syndrome Polynomial Computation, 이하, SC)은 신드롬 다항식 $S(x)$ 를 생성하고 수신된 코드워드(codeword)의 오류 패턴을 표현한다. 신드롬 다항식 $S(x)$ 는 리드-솔로몬 복호기의 두 번째 부분인 KES(Key-Equation Solver) 블록에서 사용된다. KES 블록은 키 등식(Key equation) $S(x)\sigma(x)=\omega(x)\bmod x^{2t}$ 을 해결하기 위해 유클리드 알고리즘, 수정 유클리드 알고리즘 또는 벨르캄프-메시(Berlecamp-Massay) 알고리즘 등을 이용한다. 이러한 알고리즘은 오류 위치 다항식(error locator polynomial) $\sigma(x)$ 와 오류값 다항식(error value polynomial) $\omega(x)$ 을 구하기 위하여 사용된다.
- <14> 유클리드 알고리즘은 유한체(Galois-field) 연산의 역원이 사용되므로 이를 위한 LUT(Look Up Table)가 ROM의 형태로 필요하다. 그러나 수정 유클리드 알고리즘은 LUT를 필요로 하지 않으므로 LUT 사용으로 인한 속도 지연을 줄일 수 있고 시스톱릭-어레이(systolic-array) 구조로 구현이 가능하여 벨르캄프-메시 알고리즘에 비해 파이프라인화(pipelined)가 용이하여 쉽게 고속 구현을 할 수 있다.
- <15> $\sigma(x)$ 와 $\omega(x)$ 의 두 다항식은 친서치(Chien Search) 블록과 포니(Formey) 알고리즘 연산 블록을 이용하여 오류의 위치에 대응하는 오류들의 크기 값을 구하기 위해 사용된다.

<16> KES 블록에서 사용되는 수정된 유클리드 알고리즘은 최고차항을 소거하여 반복적으로 차수를 줄이는 방식으로 오류 위치 및 오류 값 다항식을 구하기 위해 다음 수학식 1과 같이 초기값을 정한다. 여기서, $S(x)$ 는 앞서 언급한 신드롬 다항식이다.

수학식 1

<17> $R_0(x)=x^{2t}$, $Q_0(x)=S(x)$, $L_0(x)=0$, $U_0(x)=1$

<18> 수학식 2 및 수학식 3은 반복적으로 i 번 수행될 $R_i(x)$, $Q_i(x)$, $L_i(x)$, $U_i(x)$ 에 대해 나타낸 것이다. a_i 와 b_i 는 각각 $R_i(x)$, $Q_i(x)$ 의 최고차항의 계수이고 다항식 $R_i(x)$, $Q_i(x)$, $L_i(x)$, $U_i(x)$ 의 값은 다항식 $R_{i-1}(x)$, $Q_{i-1}(x)$ 의 차수에 따라 값이 결정된다. $\deg(R_i(x))$ 와 $\deg(Q_i(x))$ 는 각각 다항식 $R_i(x)$, $Q_i(x)$ 의 차수를 나타낸다. $\deg(R_i(x)) < \deg(L_i(x))$ 가 되면 반복 연산을 멈추게 되고 이때의 $R_i(x)$ 와 $L_i(x)$ 가 각각 오류 값 다항식과 오류 위치 다항식이 된다.

수학식 2

$$R_i(x)=[\sigma_{i-1}b_{i-1}R_{i-1}(x)-\bar{\sigma}_{i-1}a_{i-1}Q_{i-1}(x)]-x^{l_{i-1}}[\sigma_{i-1}a_{i-1}Q_{i-1}(x)-\bar{\sigma}_{i-1}b_{i-1}R_{i-1}(x)]$$

$$Q_i(x)=\sigma_{i-1}Q_{i-1}(x)-\bar{\sigma}_{i-1}R_{i-1}(x)$$

<19>

수학식 3

$$L_i(x)=[\sigma_{i-1}b_{i-1}L_{i-1}(x)-\bar{\sigma}_{i-1}a_{i-1}U_{i-1}(x)]-x^{l_{i-1}}[\sigma_{i-1}a_{i-1}U_{i-1}(x)-\bar{\sigma}_{i-1}b_{i-1}L_{i-1}(x)]$$

$$U_i(x)=\sigma_{i-1}U_{i-1}(x)-\bar{\sigma}_{i-1}L_{i-1}(x)$$

<20>

수학식 4

$$l_{i-1} = \text{Deg}(R_{i-1}) - \text{Deg}(Q_{i-1})$$

$$\sigma_{i-1} = \begin{cases} 1, & \text{if } l_{i-1} \geq 0 \\ 0, & \text{if } l_{i-1} < 0 \end{cases}$$

<21>

수학식 5

$$R_0(x)=x^{2t}, Q_0(x)=xS(x), L_0(x)=0, U_0(x)=x$$

<22>

<23> 수정 유클리드 알고리즘은 위의 수학식 2부터 수학식 4까지 반복적으로 연산을 하는 것이고, 종래의 방식은 수학식 4의 두 다항식 $R_{i-1}(x)$, $Q_{i-1}(x)$ 의 차수를 계산하고 l_{i-1} 을 구하여 수학식 2와 수학식 3이 빠르게 연산될 수 있도록 제어 신호를 만든다. 제어 신호는 반복 연산의 종료 조건인 $\deg(R_i(x)) < \deg(L_i(x))$ 에 따른 정지(stop) 신호도 포함하고 있다.

<24> 도 2a 및 도 2b는 상기 수학식 2부터 수학식 4까지의 다항식 연산을 처리하는 수정 유클리드 알고리즘의 기본 셀과 시스템 어레이 방식으로 연결된 연산회로를 도시한 도면이다.

<25> 수식연산을 하기 위해 실제로 기본 셀 PE1(200)의 입력은 수학식 5가 된다. $\deg(R_{i-1}(x)) \geq \deg(Q_{i-1}(x))$ 이상이면 수식 연산을 수행하고 교환 신호는 0이 되고 그렇지 않으면 교환 신호는 1이 된다. 교환 신호가 1인 경우 멀티플렉서에 의해 $R_{i-1}(x)$ 와 $Q_{i-1}(x)$ 그리고 $L_{i-1}(x)$ 와 $U_{i-1}(x)$ 는 서로 교환되고 $\deg(R_{i-1}(x))$ 과 $\deg(Q_{i-1}(x))$ 도 함께 교환된다. 멀티플렉서의 출력단은 각각 수식 연산을 수행하여 각각 $R_i(x)$, $Q_i(x)$, $L_i(x)$, $U_i(x)$ 가 된다. t 개의

오류가 발생한 경우 $2t$ 개의 셀에서 수식 연산을 수행한 결과인 $R_{2t}(x)$, $L_{2t}(x)$ 가 각각 오류 값 다항식과 오류 위치 다항식이 된다. t 개 미만의 오류가 발생한 경우, 즉 멀티플렉서의 출력단 $Q_i(x)$ 의 첫 계수가 0인 경우 $Q_i(x)$ 와 $U_i(x)$ 의 값은 오류 값 다항식과 오류 위치 다항식이 되고 $\deg(Q_i(x)) = \deg(Q_{i-1}(x)) - 1$ 로 구한 $\deg(Q_i(x))$ 가 t 보다 작은지 비교하여 $2t$ 개의 기본 셀 연산이 끝난 후 $Q_{2t}(x)$ 와 $U_{2t}(x)$ 의 차수가 각각 $t-1$ 과 t 가 되도록 쉬프트(shift) 연산이 수행된다.

- <26> 도 3은 도 2b의 시스톱릭 어레이 방식에서 기본 셀의 출력을 다시 입력으로 재귀하여 하나의 기본 셀을 반복적으로 사용하는 구조를 도시한 도면이다.
- <27> 기본 셀의 입력에서 출력까지의 클럭 지연을 m 이라 하면 출력이 다시 입력으로 재귀하기 위해 $2t-m$ 개의 레지스터를 거쳐야 하므로 m 이 작을수록 쉬프트 레지스터(shift-register)의 수가 증가한다. 16개의 기본 셀을 사용하는 도 2b에 비해 하나의 기본 셀을 사용함으로써 하드웨어 면적을 상당히 줄일 수 있지만, 매번 쉬프트 레지스터를 거쳐야 하므로 높은 클럭 지연이 발생한다.
- <28> 상기에 설명한 도 2, 도 3의 수정 유클리드 알고리즘 연산회로는 수학적 1부터 수학적 4를 수행하는 기본 셀로 구성되고, 기본 셀은 각각 다항식의 차수를 계산하고 비교하는 연산회로가 포함되어 있어 많은 면적을 차지한다.
- <29> 도 4는 차수 계산이 필요없는(degree computationless) 수정 유클리드 알고리즘의 연산회로를 도시한 도면이고, 도 5는 도 4의 차수 계산이 필요 없는 수정 유클리드 알고리즘의 제어 신호 생성을 위한 각 경우별 다항식의 연산을 도시한 도면이다.
- <30> 최종 출력이 나오기까지 $2t$ 의 클럭 지연이 소요되고 $3t+2$ 개의 셀이 사용되었으며 면적 효율적인 구조로 제안되었다. 그러나 차수의 계산 및 비교를 하는 연산회로는 없지만 블록을 파이프라인화 하기 어려워 빠른 클럭 속도(clock speed)를 얻기가 힘들다. 그리고 $2t$ 클럭 이후 블록의 출력으로 오류 값 다항식과 오류 위치다항식이 나오기 전까지 새 입력을 처리할 수 없어서, KES블록 한 개를 공유한 4-채널 병렬 구조에 적용하면 신드롬 계산 블록당 클럭 지연(clock latency)이 선형으로 증가한다.

발명이 이루고자 하는 기술적 과제

- <31> 본 발명이 이루고자 하는 기술적 과제는, 종래의 수정 유클리드 연산회로의 기본셀에 포함된 차수 계산 및 비교 부분을 제거하고 데이터 스트림 처리에 효율적이고 파이프라인화하기 쉬운 고속 저면적의 수정 유클리드 연산회로 및 그 방법을 제공하는 데 있다.

발명의 구성 및 작용

- <32> 상기의 기술적 과제를 달성하기 위한, 본 발명에 따른 수정 유클리드 연산회로의 일 실시예는, 신드롬 다항식을 기초로 상기 오류 값 다항식 및 오류 위치 다항식을 구하기 위한, 다수의 기본셀들로 구성된 시스톱릭-어레이 구조의 수정 유클리드 알고리즘 연산 회로에 있어서, 상기 기본셀들은, 다항식 C_{i-1} 의 값 및 다항식 C_{i-1} 와 D_{i-1} 의 차수를 기초로 결정되는 다항식 C_{i-1} 및 D_{i-1} 의 교환 연산 및/또는 쉬프트 연산의 수행여부에 대한 유한 상태 기계(finite state machine)를 기초로, 교환 연산 및/또는 쉬프트 연산에 대한 제어 신호를 발생하는 제어 신호 생성부; 및 상기 제어 신호에 따라 다항식 C_{i-1} , D_{i-1} , E_{i-1} , F_{i-1} 에 대한 교환 연산 또는/및 쉬프트 연산을 수행한 후 다항식 연산을 수행하는 연산부;를 각각 포함하고, 상기 기본셀들 중 첫 번째 기본셀의 입력 다항식 C_0 는 신드롬 다항식에 x 를 곱한 값, D_0 는 오류 정정 가능한 심볼 수의 2배의 차수를 가지는 값, E_0 는 x , 그리고, F_0 는 0이며, 상기 기본셀들 중 짝수 번째 기본셀들은 출력되는 다항식 C_i 및 D_i 의 차수를 1씩 낮추는 것을 특징으로 한다.
- <33> 상기의 기술적 과제를 달성하기 위한, 본 발명에 따른 수정 유클리드 연산 방법의 일 실시예는, 신드롬 다항식을 기초로 상기 오류 값 다항식 및 오류 위치 다항식을 구하기 위한, 다수의 기본셀들로 구성된 시스톱릭-어레이 구조의 수정 유클리드 알고리즘 연산 방법에 있어서, 다항식 C_{i-1} 의 값 및 다항식 C_{i-1} 와 D_{i-1} 의 차수를 기초로 결정되는 다항식 C_{i-1} 및 D_{i-1} 의 교환 연산 및/또는 쉬프트 연산의 수행여부에 대한 유한 상태 기계(finite state machine)를 기초로, 교환 연산 및/또는 쉬프트 연산에 대한 제어 신호를 발생하는 단계; 상기 제어 신호에 따라

다항식 C_{i-1} , D_{i-1} , E_{i-1} , F_{i-1} 에 대한 교환 연산 또는/및 쉬프트 연산을 수행한 후 다항식 연산을 수행하는 단계; 및 상기 제어신호 발생 단계 및 상기 연산 수행 단계를 반복하여 수행하며, 짝수 번째 연산 수행단계에서는 연산 결과로서 출력되는 다항식 C_i 및 D_i 의 차수를 1씩 낮추는 단계;를 포함하고, 상기 기본셀들 중 첫 번째 기본셀의 입력 다항식 C_0 는 신드롬 다항식에 x 를 곱한 값, D_0 는 오류 정정 가능한 심볼 수의 2배의 차수를 가지는 값, E_0 는 x , 그리고, F_0 는 0이다.

<34> 이하에서, 첨부된 도면들을 참조하여 본 발명에 따른 수정 유클리드 알고리즘의 연산 회로 및 그 방법에 대해 상세히 설명한다.

<35> 도 6은 본 발명에 따른 수정 유클리드 알고리즘의 연산 회로를 도시한 도면이다.

<36> 본 발명은 기존의 차수 계산 및 비교 부분을 제거하기 위해 수정 유클리드 알고리즘의 수학식 2 내지 수학식 4를 다음 수학식 6 내지 수학식 8의 형태로 변형한다.

수학식 6

$$R_i(x) = b_{i-1}R_{i-1}(x) - x^{|l_{i-1}|}a_{i-1}Q_{i-1}(x)$$

$$Q_i(x) = Q_{i-1}(x)$$

<37>

수학식 7

$$R_i(x) = a_{i-1}Q_{i-1}(x) - x^{|l_{i-1}|}b_{i-1}R_{i-1}(x)$$

$$Q_i(x) = R_{i-1}(x)$$

<38>

수학식 8

$$\text{if}(l_{i-1} < 0) \text{ SWAP}(R_{i-1}(x), Q_{i-1}(x))$$

$$R_i(x) = b_{i-1}R_{i-1}(x) - x^{|l_{i-1}|}a_{i-1}Q_{i-1}(x)$$

$$Q_i(x) = Q_{i-1}(x)$$

<39>

수학식 9

$$\text{if}(l_{i-1} < 0) \text{ SWAP}(R_{i-1}(x), Q_{i-1}(x))$$

$$L_i(x) = b_{i-1}L_{i-1}(x) - x^{|l_{i-1}|}a_{i-1}U_{i-1}(x)$$

$$U_i(x) = U_{i-1}(x)$$

<40>

<41> σ_{i-1} 가 1인 경우 수학식 2는 수학식 6의 형태가 되고 σ_{i-1} 가 0인 경우 수학식 2는 수학식 7의 형태가 된다. l_{i-1} 이 음수인 경우 $R_{i-1}(x)$ 와 $Q_{i-1}(x)$ 를 서로 바꾸어 수학식 6에 적용하면 $R_{i-1}(x)_{\text{new}}$ 는 $Q_{i-1}(x)$ 이고 $Q_{i-1}(x)_{\text{new}}$ 는 $R_{i-1}(x)$ 이고 a_{i-1_new} 는 $Q_{i-1}(x)$ 의 최고차항의 계수 b_{i-1} 이고 b_{i-1_new} 는 $R_{i-1}(x)$ 의 최고차항의 계수 a_{i-1} 이므로 $R_{i-1}(x)$ 와 $Q_{i-1}(x)$ 를 수학식 7에 적용한 것과 같은 값을 가지게 된다. 즉 수학식 6과 수학식 7을 수학식 8의 형태로 표현할 수 있고 같은 방법으로 수학식 9를 나타낼 수 있다.

<42> 도 6에서 기본 셀의 연산과 관계하여 수학식 8의 연산은 쉬프트 연산, 다항식 연산, 교환 연산의 세 가지로 분류될 수 있다. 즉, 기본 셀의 제어 신호는 세 가지가 된다. 도 6는 도 2a의 다항식 연산 블록(polynomial arithmetic block)과 비슷한 형태로 구성되어 있고, 도 6의 기본 셀은 x 를 곱하는 쉬프트 연산과 두 입력 다항식의 최고차항의 계수를 서로 상대방의 다항식과 곱하여 더하는 수식 연산을 동시에 할 수 없고 하나의 셀은 두

변 이상의 쉬프트 연산을 수행할 수 없다.

<43> 그리고 쉬프트 연산을 수행할 다항식은 멀티플렉스의 출력단에 위치해야 한다. l_{i-1} 을 이용하지 않고 세 가지의 제어 신호를 생성하기 위해 다항식의 입력 패턴을 이용한다. 세 가지의 입력패턴은 다음의 수학식 11부터 수학식 13까지로 나타낸다.

수학식 10

$$C_0(x) = xQ_0(x), \quad D_0(x) = R_0(x), \quad E_0(x) = x, \quad F_0(x) = 0$$

<44>

수학식 11

$$R_0(x) = x^{2t}$$

$$Q_0(x) = S_{2t-1}x^{2t-1} + S_{2t-2}x^{2t-2} + \dots + S_1x^1 + S_0 \quad (S_i \neq 0, \quad 0 \leq i \leq 2t-1)$$

<45>

수학식 12

$$R_0(x) = x^{2t}$$

$$Q_0(x) = S_{2t-1}x^{2t-1} + S_{2t-2}x^{2t-2} + \dots + S_1x^1 + S_0 \quad \begin{cases} S_i = 0, & 2t-1-k \leq i \leq 2t-1, \quad k \geq 0, \quad k \text{은 정수} \\ S_i \neq 0, & \text{otherwise} \end{cases}$$

<46>

수학식 13

$$R_0(x) = x^{2t}$$

$$Q_0(x) = S_{2t-1}x^{2t-1} + S_{2t-2}x^{2t-2} + \dots + S_1x^1 + S_0 \quad \begin{cases} S_i = 0, & 2t-1-m \leq i < 2t-1, \quad m > 0, \quad m \text{은 정수} \\ S_i \neq 0, & \text{otherwise} \end{cases}$$

<47>

<48> $Q_0(x)$ 의 차수는 항상 $R_0(x)$ 의 차수보다 작으므로 도 6의 기본 셀 1은 수학식 10의 입력을 가진다. 수학식 11의 경우는 S_i 의 값이 모두 0이 아닌 경우이다. 이는 수학식 8의 연산에서 $R_0(x)$ 의 최고차항만을 소거하는 것을 의미한다. 즉 $R_1(x)$ 는 $Q_1(x)(=Q_0(x))$ 의 차수와 같음을 의미한다. 다항식 수학식 8을 적용하여 $R_2(x)$ 와 $Q_2(x)$ 를 구할 수 있고, $R_2(x)$ 가 $R_1(x)$ 의 최고차항만을 소거한 형태라면 $R_2(x)$ 와 $Q_2(x)$ 를 서로 교환(swap)한 후 다시 수학식 11과 같은 입력 패턴이 된다. 만약 $R_2(x)$ 가 $R_1(x)$ 의 최고차항뿐만 아니라 두 번째 항까지 소거하였다면 $R_2(x)$ 와 $Q_2(x)$ 는 서로 교환된 후 수학식 12의 입력 패턴이 된다. 이처럼 $R_i(x)$ 의 최고차항을 소거하는 과정에서 $R_i(x)$ 보다 $Q_i(x)$ 의 차수가 커지게 되어 두 다항식을 스왑하는 경우를 새 연산의 시작이라고 한다.

<49> 도 6의 기본 셀과 관련하여 $C_1(x)$ 과 $D_1(x)$ 는 각각 $R_1(x)$ 와 $Q_1(x)$ 가 되고, $C_2(x)$ 과 $D_2(x)$ 는 각각 $R_2(x)$ 와 $Q_2(x)$ 가 된다. 다항식 연산 또는 쉬프트 연산을 수행하는 경우 $D_i(x)$ 의 출력은 $C_i(x)$ 의 출력에 비해 1클럭 지연되는데, 이는 두 가지의 역할을 수행한다. 첫 번째는 수학식 8에서 덧셈연산을 위해 $Q_{i-1}(x)$ 에 곱해진 x 를 소거하는 것이고, 두 번째는 $R_{i-1}(x)$ 와 $Q_{i-1}(x)$ 를 스왑하기 전에 미리 $R_{i-1}(x)$ 에 x 를 곱하여 다항식 연산을 쉬프트 연산 없이 바로 수행할 수 있게 한다.

<50> 도 7에서 S0상태는 연산이 처음으로 시작되거나 새 연산이 시작 될 때의 상태를 나타내고 출력은 Sw 신호와 Sht 신호이다. 도 7의 출력은 기본 셀의 제어신호 Sw 및 Sht와 같고 Sw 신호가 1일때 교환 연산, Sht 신호가 1일때 쉬프트 연산, Sht 신호가 0일때 다항식 연산이 된다. S0 상태에서 두 입력 다항식의 차수가 같은지 아닌지 여부에 따라 S1 또는 S2 상태로 천이되고 두 입력 다항식의 차수가 같은지 아닌지를 나타내는 입력 신호는 SI이다.

수학식 11의 경우는 두 입력 다항식의 차수가 같으므로 SI는 1이되고 S1상태로 천이된다. 이 경우 연산은 다항식 연산이므로 Sht 신호는 0이되고, $C_0(x)$ 는 $xQ_0(x)$ 이므로 다항식 연산을 위해 Sw 신호는 1이 된다. $R_1(x)$ 가 $R_0(x)$ 에서 최고차항만을 소거한 경우라면 다항식 연산을 수행하고, 도 7에서 S1 상태에서 SI가 1일 때의 제어신호가 기본 셀 2에 적용된다. 계속해서 $R_i(x)$ 가 $R_{i-1}(x)$ 의 최고차항만을 소거하는 경우 도 7의 상태는 S0-S1을 반복하게 된다.

<51> 수학식 12의 입력 패턴을 가지는 경우 기본 셀 1의 입력인 $xQ_0(x)$ 의 차수가 $R_0(x)$ 의 차수보다 낮으므로 $xQ_0(x)$ 에 x 를 곱하는 연산을 수행한다. 그러므로 도 7에서 초기 S0상태에서 SI입력이 0인 경우 S2상태가 되고 쉬프트 연산에 대한 제어신호가 출력된다. 기본 셀 2의 두 입력 다항식 $x^2Q_0(x)$ 와 $R_0(x)$ 의 차수가 같으면 다항식 연산을 수행한다. $Q_1(x)(=Q_0(x))$ 에 x 를 곱한 다항식과 $R_1(x)$ 의 차수가 같으면 다항식 연산을 수행하고, 그렇지 않으면 $R_1(x)$ 와 $Q_1(x)$ 의 차수가 같은지를 비교하여 같으면 다항식 연산을 수행하고 그렇지 않으면 $R_1(x)$ 의 차수가 $Q_1(x)$ 의 차수보다 낮으므로 교환연산을 수행한다. k 가 1인 경우의 상태천이는 S0-S2-S3-S1-S1이다.

<52> 수학식 13의 입력 패턴을 가지는 경우 기본 셀 1은 수학식 11일 때와 같게 된다. 하지만 이때의 $R_1(x)$ 의 차수는 $Q_1(x)$ 의 차수보다 낮으므로 $R_1(x)$ 와 $Q_1(x)$ 에 대하여 다항식 연산이 아닌 교환 연산을 수행한다. 교환 연산과 다항식 연산은 동시에 수행 가능하지만 다항식 연산과 쉬프트 연산은 동시에 수행될 수 없으므로 스왑하기 이전에 바로 다항식 연산을 수행하도록 쉬프트 연산을 수행한다. 그러므로 도 7에서 S1 상태의 SI 입력이 0인 경우 쉬프트 연산에 대한 제어신호가 출력된다. 기본 셀 2의 연산이 끝난 후 S0상태에 있게 된다.

<53> 위에서 언급한 규칙을 이용하여 도 7의 FSM이 완성되었고 $k < 8$ 라고 가정하였다. 그러므로 S16 상태는 채널 전송 중 오류가 발생하지 않아 신드롬 값이 0임을 의미한다. t 개의 오류가 발생한 경우 오류 위치 다항식과 오류 값 다항식은 $2t$ 개 기본 셀 연산을 수행한 후 얻을 수 있다. 하지만 $v(< t)$ 개의 오류가 발생한 경우 $2t$ 보다 적은 수의 기본 셀 연산으로 오류 위치 다항식과 오류 값 다항식을 얻게 된다. 오류 값 다항식의 차수가 오류 위치 다항식의 차수보다 낮다는 것을 착안하여 FSM으로 다항식의 연산의 종료를 나타내는 정지(stop) 신호를 생성하였다.

<54> 도 8은 기본 셀의 두 입력 $C_{i-1}(x)$ 와 $E_{i-1}(x)$ 중 먼저 0이 아닌 값이 들어오는 다항식이 어떤 다항식인지를 확인하고 $E_{i-1}(x)$ 의 값이 먼저 들어오면 상태는 초기 S0에서 S2상태로 이동하고 정지 리셋 신호가 들어올 때까지 S2상태를 유지한다. 반면에 $C_{i-1}(x)$ 의 값이 먼저 들어오거나 동시에 들어오면 S1상태로 이동하고 마찬가지로 정지 리셋 신호가 들어올 때까지 S1상태를 유지한다.

<55> 도 8의 입력은 $C_{i-1}(x)$ 와 $E_{i-1}(x)$ 이 0이 아닌 경우에 각각 1이고, '-'기호는 해당 입출력이 상태가 천이되는데 영향을 주지 않는다는 것을 의미한다. 정지 신호가 1이 되면 기본 셀은 쉬프트 연산만을 수행하나 $2t$ 개의 기본 셀을 수행한 후의 $C_{2t}(x)$ 와 $E_{2t}(x)$ 가 올바른 값이 되려면 추가적으로 고려해야 할 사항이 있다. 예를 들어 기본 셀 6의 출력이 오류 값 다항식과 오류 위치 다항식이지만, 일반적으로 기본 셀 6의 $C_6(x)$ 의 차수는 12이고 $C_{2t}(x)$ 의 차수가 7이 되려면 두 개의 기본 셀 당 1번의 차수를 낮춰야 하므로 짝수 셀의 레지스터(2)는 제거된다.

<56> 도 6을 참조하면, 본 발명에 따른 수정 유클리드 연산 회로의 각 기본셀(600)은 크게 교환 연산부(610), 쉬프트 연산부(620), 다항식 연산부(630), 제어신호 발생부(640) 및 정지신호 발생부(650)로 구성된다.

<57> 제어신호 발생부(640)는 도 7에 도시된 것과 같은 유한 상태 기계를 기초로 제어 신호를 발생한다. 구체적으로, 제어신호 발생부(640)는 이전 기본셀로부터 전달받은 유한 상태 기계에서의 상태값(Sin)과 이전 기본셀의 출력 다항식 C_{i-1} 을 기초로 유한 상태 기계에서 해당하는 상태로 전이한다. 상태 전이시 해당하는 연산 제어신호를 발생한다.

<58> 정지신호 발생부(650)는 도 8에 도시된 것과 같은 유한 상태 기계를 기초로 정지 신호를 발생한다. 구체적으로, 정지신호 발생부(650)는 이전 기본셀의 출력 다항식 C_{i-1} 및 D_{i-1} 를 기초로 상태를 전이한다. 정지신호가 발생하면 그 기본셀의 출력이 곧 오류위치 다항식 및 오류 값 다항식이 된다.

<59> 각 연산부(610,620,630)는 제어신호 발생부(640) 및 정지신호 발생부(650)의 제어신호에 따라 각각 연산을 수행하며, 특히 짝수 번째 기본셀은 도 6에서 점선으로 표시된 레지스터(602,632,634)가 제거되어 차수를 낮춘다.

<60> 이제까지 본 발명에 대하여 그 바람직한 실시예들을 중심으로 살펴보았다. 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자는 본 발명이 본 발명의 본질적인 특성에서 벗어나지 않는 범위에서 변형된 형태로 구현될 수 있음을 이해할 수 있을 것이다. 그러므로 개시된 실시예들은 한정적인 관점이 아니라 설명적인 관점에서 고려되어야 한다. 본 발명의 범위는 전술한 설명이 아니라 특허청구범위에 나타나 있으며, 그와 동등한 범위 내에 있는 모든 차이점은 본 발명에 포함된 것으로 해석되어야 할 것이다.

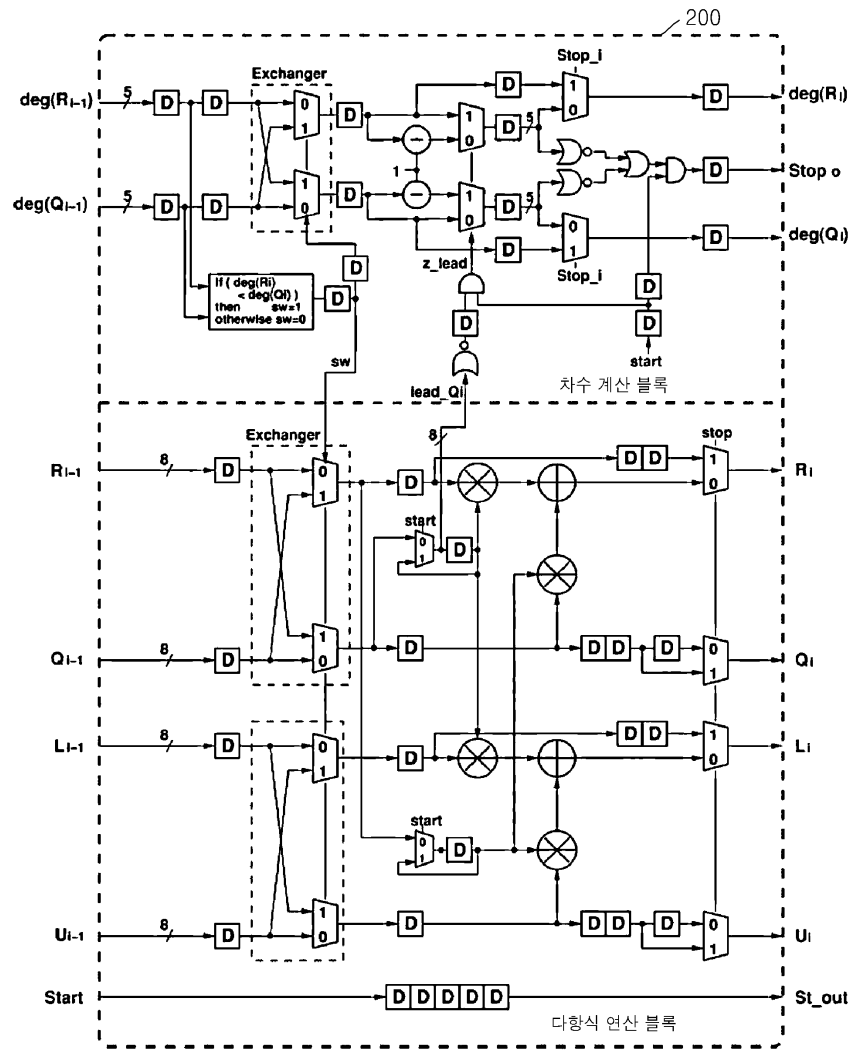
발명의 효과

<61> 본 발명에 따르면, 종래의 수정 유클리드 알고리즘 연산회로의 기본 셀에 포함된 차수 비교 및 계산 부분을 제거함으로써 데이터 스트림 처리에 효율적이고 파이프라인화하기 쉬우면서 하드웨어 복잡도를 감소시킬 수 있다.

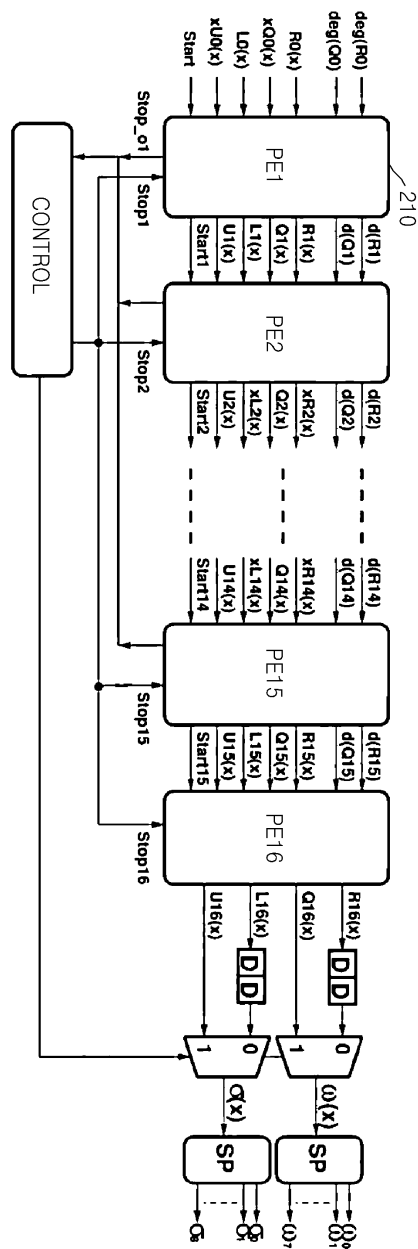
도면의 간단한 설명

- <1> 도 1은 종래의 리드-솔로몬 복호기의 구조를 도시한 도면,
- <2> 도 2a는 종래의 수정 유클리드 알고리즘의 기본 셀을 도시한 도면,
- <3> 도 2b는 도 2a의 기본 셀들이 시스토크 어레이 방식으로 연결된 연산회로를 도시한 도면,
- <4> 도 3은 도 2b의 시스토크 어레이 방식에서 기본 셀의 출력을 다시 입력으로 재귀하여 하나의 기본 셀을 반복적으로 사용하는 구조를 도시한 도면,
- <5> 도 4는 차수 계산이 필요없는(degree computationless) 수정 유클리드 알고리즘의 연산회로를 도시한 도면,
- <6> 도 5는 도 4의 차수 계산이 필요 없는 수정 유클리드 알고리즘의 제어 신호 생성을 위한 각 경우별 다항식의 연산을 도시한 도면,
- <7> 도 6은 본 발명에 따른 수정 유클리드 알고리즘의 연산 회로를 도시한 도면,
- <8> 도 7은 연산 제어 신호 생성을 위한 유한 상태 기계(FSM:Finite State Machine)을 도시한 도면, 그리고,
- <9> 도 8은 정지 신호 생성을 위한 유한 상태 기계를 도시한 도면이다.

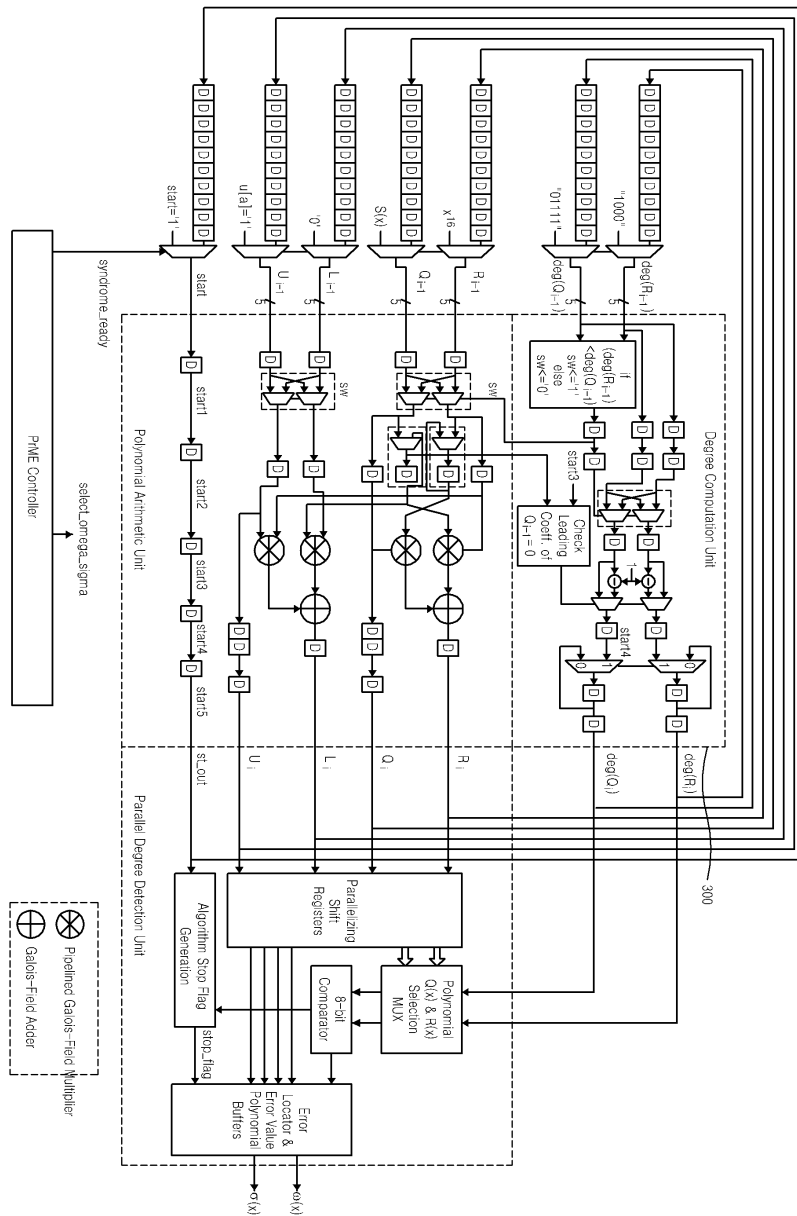
도면2a



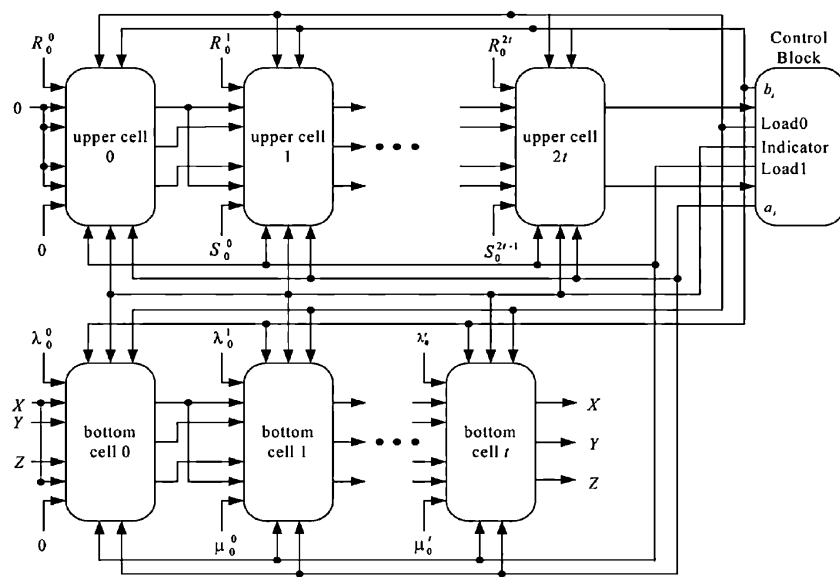
도면2b

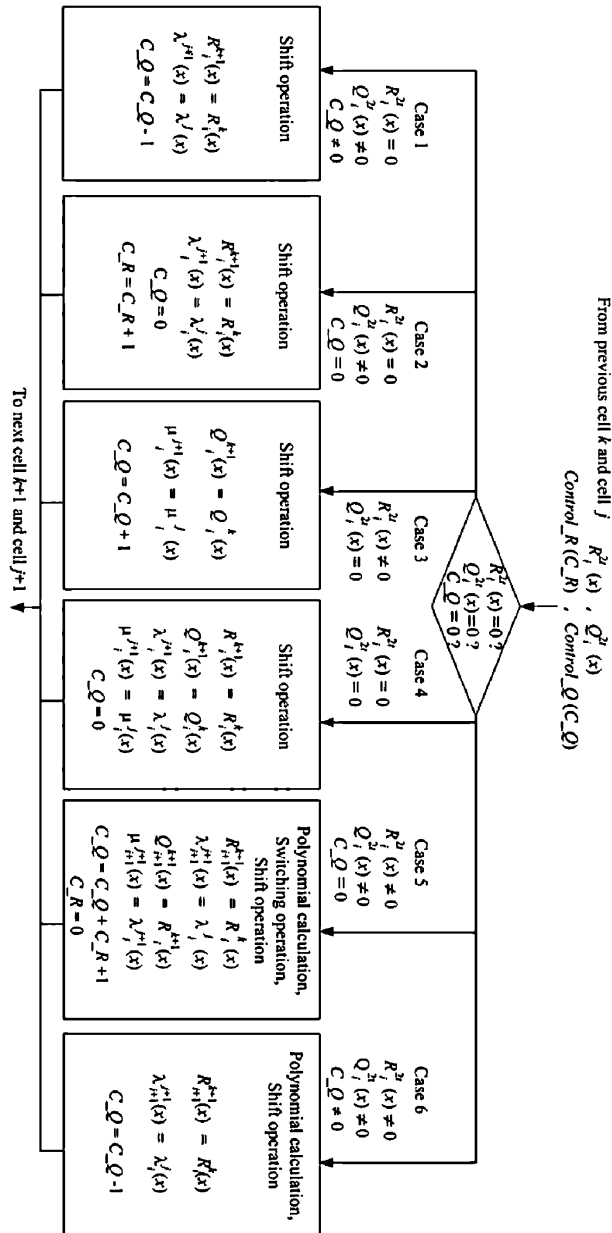


도면3



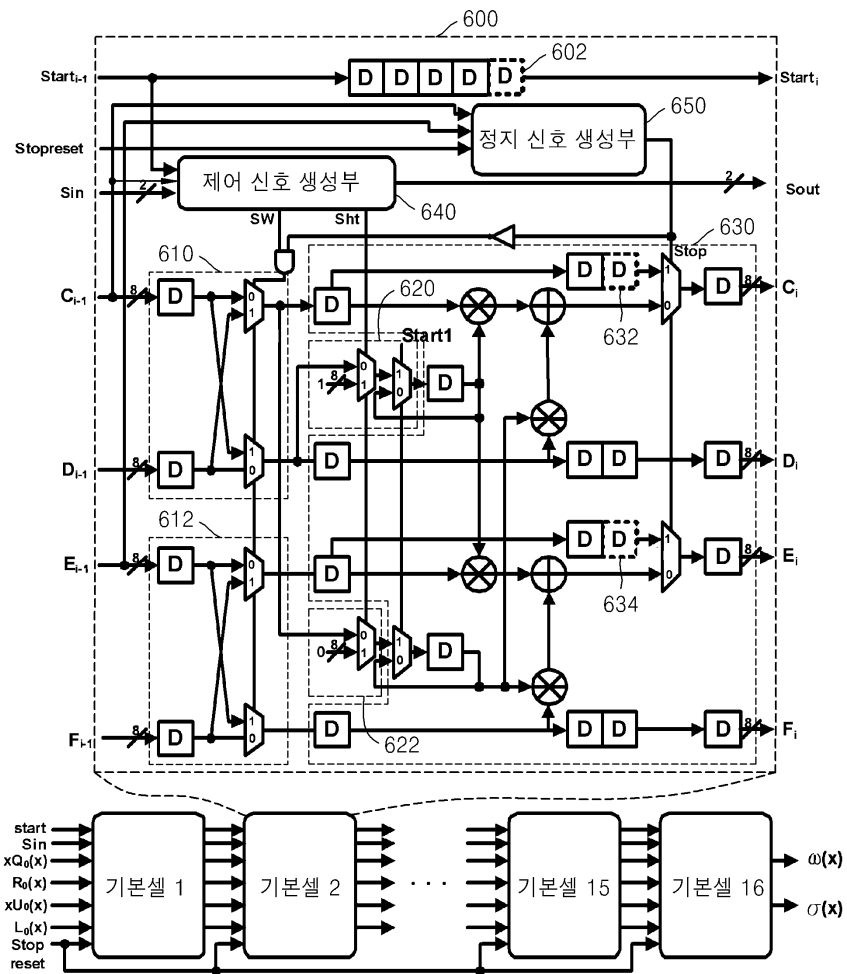
도면4



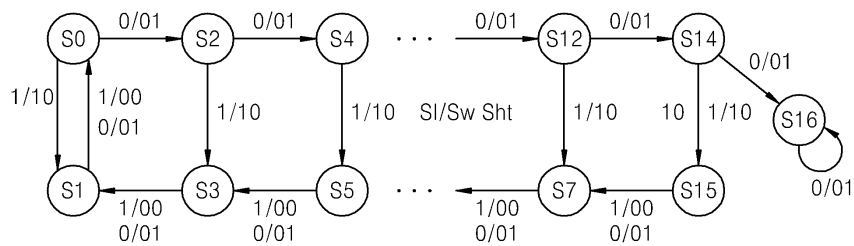


도면5

도면6



도면7



도면8

