



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2011-0025433
(43) 공개일자 2011년03월10일

(51) Int. Cl.

H03M 13/41 (2006.01)

(21) 출원번호 10-2009-0083501

(22) 출원일자 2009년09월04일

심사청구일자 2009년09월04일

(71) 출원인

인하대학교 산학협력단

인천 남구 용현동 253 인하대학교

(72) 발명자

정덕진

인천시 남구 학익동 하나APT 104동 1108호

송재연

인천시 남구 용현3동 210-4 영동하이츠 빌라 4동 210호

(뒷면에 계속)

(74) 대리인

이원희

전체 청구항 수 : 총 9 항

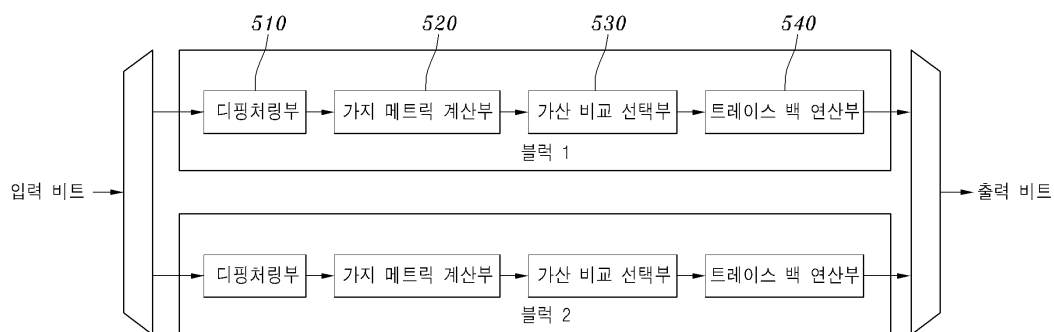
(54) 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더 및 이를 이용한 디코딩 방법

(57) 요약

본 발명은 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더에 관한 것으로, 해결하고자 하는 기술적 과제는 비터비 디코더의 성능 향상을 위해 슬라이딩 방법을 이용하는 동시에 Radix 4 Fast ACS 구조를 채용하여 병렬화 함으로써 슬라이딩 방법이 가지는 문제점인 오버랩 사이즈를 감소시켜 높은 작업처리량을 가지는 비터비 복호기를 제공하는데 있다.

이를 위해 본 발명에 따른 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더는 2이상의 비터비 디코딩 블록으로 구성되고, 상기 비터비 디코딩 블록은, 수신단에서 각 코딩 레이트에 따라 데이터가 재구성되도록 평치렁된 데이터를 복원하는 디평치렁부, 채널로부터 수신받은 데이터를 양자화하는 가지 메트릭 계산부와, 상기 가지 메트릭 계산부로부터 인가되는 새로운 가지 메트릭 값과 누적된 경로 메트릭 값을 각 천이마다 가산한 후 각 상태별로 비교하여 최적의 경로 메트릭 값을 갖는 생존 메트릭 값과 생존 경로를 출력하는 가산 비교 선택부와, 트리 구조의 비교기를 구성하여 최소 경로 메트릭 값을 찾아내고, 이를 통해 상기 최소 경로 메트릭이 존재하는 상태를 찾아내며, 상기 최소 경로 메트릭이 존재하는 상태에서부터 실제 출력 값을 생성하는 트레이스백 연산부를 포함하는 것을 특징으로 하는 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더를 개시한다.

대 표 도 - 도5



(72) 발명자

왕규열

인천시 남동구 구월1동 동아아파트 2동 410호

김병수

인천시 연수구 연수 1동 동남아파트 2동 410호

이상설

인천시 남구 용현3동 210-4 영동하이츠 빌라 4동
210호

특허청구의 범위

청구항 1

슬라이딩 방법을 기반으로 하는 비터비 디코더에 있어서,

상기 비터비 디코더는 2이상의 비터비 디코딩 블록으로 구성되고,

상기 비터비 디코딩 블록은,

수신단에서 각 코딩 레이트에 따라 데이터가 재구성되도록 평처링된 데이터를 복원하는 디평처링부;

채널로부터 수신받은 데이터를 양자화하는 가지 메트릭 계산부;

상기 가지 메트릭 계산부로부터 인가되는 새로운 가지 메트릭 값과 누적된 경로 메트릭 값을 각 천이마다 가산한 후 각 상태별로 비교하여 최적의 경로 메트릭 값을 갖는 생존 메트릭 값과 생존 경로를 출력하는 가산 비교 선택부;

트리 구조의 비교기를 구성하여 최소 경로 메트릭 값을 찾아내고, 이를 통해 상기 최소 경로 메트릭이 존재하는 상태를 찾아내며, 상기 최소 경로 메트릭이 존재하는 상태에서부터 실제 출력 값을 생성하는 트레이스 백 연산부;

를 포함하는 것을 특징으로 하는 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더.

청구항 2

청구항 1에 있어서,

상기 가지 메트릭 계산부는 개별 코드 워드와 입력 데이터의 해밍 거리를 계산함으로써 가지 메트릭을 계산하는 것을 특징으로 하는 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더.

청구항 3

청구항 2에 있어서,

상기 가지 메트릭 계산부는 두 개단의 가지 메트릭을 생성하는 두 블록의 구조를 가지며, 6개의 코드 워드 비트를 3비트씩 두 개의 단으로 나누고, 중복되는 각각 8개의 코드 워드를 각 단별로 8개씩 해밍거리를 계산하고 이를 조합하여 256개의 가지 메트릭을 계산하는 것을 특징으로 하는 슬라이딩 방법을 이용하는 UWB용 고속 비터비 디코더.

청구항 4

청구항 1에 있어서,

상기 가산 비교 선택부는 한 번에 두 단의 데이터를 처리하는 구조인 것을 특징으로 하는 슬라이딩 방법을 이용하는 UWB용 고속 비터비 디코더.

청구항 5

청구항 4에 있어서,

상기 가산 비교 선택부는 Radix 4 ACS 구조를 가지는 것을 특징으로 하는 슬라이딩 방법을 이용하는 UWB용 고속 비터비 디코더.

청구항 6

청구항 1에 있어서,

상기 트레이스 백 연산부는 한 번에 두 단의 경로값을 처리하여 한 클럭에 두 개의 비트를 출력하는 것을 특징으로 하는 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더.

청구항 7

청구항 6에 있어서,

상기 트레이스 백 연산부는 상기 트리 구조의 비교기에 의해 선택된 메트릭 경로 값을 비교 전의 메트릭 경로값에 대체함으로써 첫 번째 단의 레지스터와 비교기만을 반복적으로 사용하여 최소 메트릭 경로 상태를 찾기 위한 선택 정보를 생성하는 것을 특징으로 하는 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더.

청구항 8

슬라이딩 방법을 기반으로 하는 비터비 디코딩 방법에 있어서,

채널을 통과한 입력 스트림을 블록 사이즈로 분할하는 제 1단계;

상기 분할된 각 디코딩 블록을 통하여 개별적으로 디코딩을 수행하는 제 2단계;

상기 제 2단계를 통하여 복호된 결과 중 오버랩에 대한 결과를 제외한 부분을 이용하여 출력 스트림을 연산하는 제 3단계를 포함하는 것을 특징으로 하는 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코딩 방법.

청구항 9

청구항 8에 있어서,

상기 제 2단계는 여러 단의 ACS 연산을 한 번에 처리하는 Radix 4 ACS 구조를 사용하여 디코딩을 수행하는 것을 특징으로 하는 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코딩 방법.

명세서

발명의 상세한 설명

기술 분야

[0001] 본 발명은 비터비 디코더에 관한 것으로서 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더에 관한 것이다.

배경 기술

[0002] 통신 기술과 컴퓨터의 발달과 함께 개인 휴대 통신 시장이 급격하게 발전하고 있다. 이와 함께 빠른 속도로 데이터를 전송하기 위한 통신 기술에 대한 요구도 커지고 있다. Ultra Wide Band(UWB) 기술은 넓은 주파수 대역을 사용하며, 고속의 통신이 가능하도록 하기 위한 기술을 말한다.

[0003] 한편, 직교 주파수 다중 분할(Orthogonal Frequency Division Multiplexing, OFDM) 방식은 다중경로 페이딩 및 협대역 간섭에서 우수한 성능을 가지기 때문에 통신 시스템에서 매우 유용한 기술로 부각되고 있다. 다중 대역 직교 주파수 다중 분할(Multi Band-Orthogonal Frequency Division Multiplexing, MB-OFDM) 방식은 이러한 장점에 여러 개의 주파수 밴드를 사용하는 장점을 부가한 방식으로 IEEE 802.15 Task 3a 통신 시스템 표준안 중 하나로 제안되었다. 이는 근거리에서 최고 480 Mbps의 매우 높은 데이터 전송률을 필요로 하는 시스템이다. 또한, 채널환경에 따라 다양한 통신 속도를 제공하며, 통신 속도에 따라 펀처링(Puncturing) 방식을 이용하여 다양한 코드 레이트(Code Rate)를 사용한다. MF-OFDM 시스템은 구축장의 길이가 7, 코드 레이트가 1/3dls 길쌈 코

드(Convolutional Code)를 채용하고 있다. 본 발명에서는 이러한 구조에 적합한 비터비 복호기 알고리즘을 제안한다.

[0004] 아날로그 통신방식과는 달리 디지털 통신방식에서는 채널 상에서 발생한 오류를 정정하는 방법으로 전진 에러 수정(FEC : Forward Error Correction) 방식을 사용한다. 이러한 방법 중 하나인 길쌈 코딩은 전송대역폭을 넓히는 단점을 가지지만 오류정정 효율이 우수하여 많은 통신 시스템에서 채택되어 사용되고 있다. 길쌈 부호기(Convolutional Encoder)의 특징으로 나타내는 것은 크게 3가지로 나누어 볼 수 있다. 첫 번째는 코딩 레이트로서 하나의 입력비트에 대해 몇 개의 출력 비트를 생성하는지를 나타낸다. 코드 레이트가 작을 수록 복호 가능성도 커지지만 그만큼 큰 대역폭을 필요로 하기 때문에 통신 시스템마다 원하는 비트 에러 레이트(bit error rate, BER)에 맞는 코드 레이트를 사용한다. 두 번째는 구속장의 길이이다. 구속장은 부호기(Encoder)의 출력에 영향을 미치는 입력비트의 수를 나타낸다. 구속장의 길이가 커질수록 하드웨어의 복잡도는 증가하지만 오류정정 능력이 증가하기 때문에 높은 오류정정능력을 필요로 하는 시스템에서는 구속장의 길이가 긴 길쌈 코드를 사용한다. 세 번째는 코드 벡터(Code Vector)이다. 이는 입력비트들에 의해 출력이 생성되는 구조를 나타내며, 이에 따라 레지스터의 연결형태가 결정된다. 각 구속장의 길이에 따라 복호 성능을 최대화 하는 코드 벡터가 연구되어 사용되고 있다. 도 1에는 MB-OFDM 길쌈 코드의 일반적인 구조를 도시하였다.

[0005] 한편, 비터비 알고리즘(Viterbi Algorithm)은 길쌈 코드를 복호화 하는 알고리즘으로 비터비에 의해 1967년 제안되어 현재까지 다양한 방식으로 연구 개발되고 있다. 이는 기본적으로 모든 데이터열의 발생확률이 동일할 경우 오류확률을 최소화 하는 복호기는 하기의 수학식 1에 의해 표현되는 조건 확률을 비교하여 최대가 되는 것을 선택하는 것이다.

수학식 1

$$P(Z | U^{(m)})$$

[0007] 여기서 Z는 수신 데이터이고 $U^{(m)}$ 은 가능한 송신 열 중 하나이다. 즉, 하기의 수학식 2로 표현된다.

수학식 2

$$P(Z | U^{(m)}) = \max P(Z | U^{(m)}) - all U^{(m)}$$

[0009] 가장 유사한 경로(path)를 선택하는 방법을 이용하기 때문에, 이러한 비터비 알고리즘(Viterbi Algorithm)은 Maximum likelihood Method라고 불리기도 한다.

발명의 내용

해결 하고자하는 과제

[0010] 본 발명은 비터비 디코더의 성능 향상을 위해 슬라이딩 방법을 이용하는 동시에 Radix 4 Fast ACS 구조를 채용하여 병렬화 함으로써 슬라이딩 방법이 가지는 문제점인 오버랩 사이즈를 감소시켜 높은 작업처리량을 가지는 비터비 복호기를 제공하는데 그 목적이 있다.

과제 해결수단

[0011] 상기한 바와 같은 목적을 달성하기 위해 본 발명에 따른 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더는 2

이상의 비터비 디코딩 블록으로 구성되고, 상기 비터비 디코딩 블록은, 수신단에서 각 코딩 레이트에 따라 데이터가 재구성되도록 평처링된 데이터를 복원하는 디평처링부와, 채널로부터 수신받은 데이터를 양자화하는 가지 메트릭 계산부와, 상기 가지 메트릭 계산부로부터 인가되는 새로운 가지 메트릭 값과 누적된 경로 메트릭 값을 각 천이마다 가산한 후 각 상태별로 비교하여 최적의 경로 메트릭 값을 갖는 생존 메트릭 값과 생존 경로를 출력하는 가산 비교 선택부와, 트리 구조의 비교기를 구성하여 최소 경로 메트릭 값을 찾아내고, 이를 통해 상기 최소 경로 메트릭이 존재하는 상태를 찾아내며, 상기 최소 경로 메트릭이 존재하는 상태에서부터 실제 출력 값을 생성하는 트레이스 백 연산부를 포함한다.

- [0012] 상기 가지 메트릭 계산부는 8레벨 양자화 방식을 이용하여 데이터를 양자화할 수 있으며, 개별 코드 워드와 입력 데이터의 해밍 거리를 계산함으로써 가지 메트릭을 계산할 수 있다.
- [0013] 상기 가지 메트릭 계산부는 두 개단의 가지 메트릭을 생성하는 두 블록의 구조를 가지며, 6개의 코드 워드 비트를 3비트씩 두 개의 단으로 나누고, 중복되는 각각 8개의 코드 워드를 각 단별로 8개씩 해밍거리를 계산하고 이를 조합하여 256개의 가지 메트릭을 계산할 수 있다.
- [0014] 상기 가산 비교 선택부는 한 번에 두 단의 데이터를 처리하는 구조이며, Radix 4 ACS 구조를 가진다.
- [0015] 상기 트레이스 백 연산부는 한 번에 두 단의 경로값을 처리하여 한 클록에 두 개의 비트를 출력할 수 있으며, 트리 구조의 비교기에 의해 선택된 메트릭 경로 값을 비교 전의 메트릭 경로값에 대체함으로써 첫 번째 단의 레지스터와 비교기만을 반복적으로 사용하여 최소 메트릭 경로 상태를 찾기 위한 선택 정보를 생성할 수 있다.
- [0016] 또한, 상기한 바와 같은 목적을 달성하기 위해 본 발명에 따른 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코딩 방법은 채널을 통과한 입력 스트림을 블록 사이즈로 분할하는 제 1단계와, 상기 분할된 각 디코딩 블록을 통하여 개별적으로 디코딩을 수행하는 제 2단계와, 상기 제 2단계를 통하여 복호된 결과 중 오버랩에 대한 결과를 제외한 부분을 이용하여 출력 스트림을 연산하는 제 3단계를 포함하며, 상기 제 2단계는 여러 단의 ACS 연산을 한 번에 처리하는 Radix 4 ACS 구조를 사용하여 디코딩을 수행한다.

효 과

- [0017] 상기한 바와 같이 본 발명에 따른 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더는 비터비 디코더의 성능 향상을 위해 슬라이딩 방법을 이용하는 동시에 Radix 4 Fast ACS 구조를 채용하여 병렬화 함으로써 슬라이딩 방법이 가지는 문제점인 오버랩 사이즈를 감소시켜 높은 작업처리량을 가질 수 있는 효과가 있다.
- [0018] 또한, 기존의 슬라이딩 방법을 이용한 비터비 디코더 보다 적은 면적을 가지면서도 높은 데이터 처리율을 가질 수 있는 효과가 있으며, 병렬성 증가가 용이함에 따라 더 높은 데이터 처리율을 필요로 하는 시스템에 쉽게 적용되어 사용될 수 있는 효과가 있다.

발명의 실시를 위한 구체적인 내용

- [0019] 이하, 첨부된 도면을 참조하여 본 발명의 실시예를 상세히 설명한다. 우선, 도면들 중 동일한 구성요소 또는 부품들은 가능한 한 동일한 참조부호를 나타내고 있음에 유의해야 한다. 본 발명을 설명함에 있어서 관련된 공지 기능 혹은 구성에 대한 구체적인 설명은 본 발명의 요지를 모호하게 하지 않기 위해 생략한다.
- [0020] 다중 대역 직교 주파수 다중 분할(Multi Band-Orthogonal Frequency Division Multiplexing, MB-OFDM) 방식은 채널 환경에 따라 다양한 통신모드를 지원한다. 하기의 표 1은 각 통신방식에 따른 파라미터를 나타내고 있다.

표 1

Data Rate [Mbps]	Modulation	Coding Rate	FDS	TDS
53.3	QPSK	1/3	YES	YES
80	QPSK	1/2	YES	YES
106.7	QPSK	1/3	NO	YES
160	QPSK	1/2	NO	YES
200	QPSK	5/8	NO	YES
320	DCM	1/2	NO	NO
400	DCM	5/8	NO	NO
480	DCM	3/4	NO	NO

[0021]

[0022]

상기 표 1에서 가장 중점적으로 보아야 할 것은 데이터 레이트(Data Rate)로서, 각 모드에 따라 전송속도가 변화하며, 최고 480[Mbps]의 높은 데이터 레이트를 만족시키는 비터비 디코더를 필요로 하는데, 일반적인 비터비 디코더 구조에서 이러한 높은 데이터 레이트를 만족시키는 것은 어려운 것이 사실이다. 따라서 본 발명에서는 작업 처리량(Throughput)을 높이기 위해 여러 개의 디코딩 블록을 두어 데이터 스트림(Data Stream)을 나누고 개별적으로 디코딩을 수행하는 슬라이딩 방법(Sliding Method) 및 한 번에 여러 단의 데이터를 한꺼번에 처리하는 형식인 High Radix 구조를 채용하는 비터비 디코더를 제안한다.

[0023]

길쌈 코드의 경우 생성다항식에서 알 수 있듯이 이전값을 현재의 출력에 반영하기 때문에 일반적인 병렬화 방법을 적용할 경우 성능의 열화를 가져온다. 이러한 문제점을 해결하기 위한 구조가 슬라이딩 방법이다. 상기 슬라이딩 방법은 병렬화를 적용하기 위하여 채널로부터 수신된 데이터를 블록 단위로 분할하여 디코딩을 수행하는 방법에 이용한다. 한편, 단순한 블록 단위의 디코딩을 이용할 경우 비신뢰구간이 많이 포함된다. 이 경우 정확한 경로(Path)를 얻기 위해 구속장의 길이의 5배 정도의 상태전이 후 결과 값을 얻는 것이 바람직하다. 비신뢰구간은 이러한 범위에 포함되지 않는 구간으로 보통 블록 단위의 디코딩시 종단부분에 해당한다. 비신뢰구간이 많이 포함될 경우 이에 의해 비트에러율(Bit Error Rate, BER)이 열화된다. 이를 방지하기 위한 방법으로 비신뢰구간에 의해 디코딩 결과가 포함되지 않도록 데이터 분할 시 적당한 길이를 오버랩(Overlap)하면서 디코딩을 수행하는 방법이 사용될 수 있다. 도 2는 슬라이딩 방법을 나타낸다. 도 2에 도시된 바와 같이 채널을 통과한 입력 스트림(Input Stream)을 블록 사이즈로 분할하고, 개별적으로 디코딩을 수행한다. 또한 디코딩 결과 전체를 사용하는 것이 아니라, 오버랩에 대한 결과를 제외한 부분을 이용함으로써 BER의 감소를 최소화하는 방식을 사용한다.

[0024]

한편, 일반적으로 비터비 디코더의 성능개선이 어려운 부분은 가산 비교 선택(Add-Compare-Select, ACS) 구조이다. 상기 ACS 구조에는 피드백 루프(Feedback Loop)가 존재하기 때문이다. 이러한 ACS의 성능을 개선하기 위하여 여러 단의 ACS 연산을 한꺼번에 처리하는 High Radix 구조가 사용되는데, 여기서 Radix 4 ACS 구조는 한 번에 두 단의 데이터를 처리하는 구조로서 이를 사용할 경우 기본적인 Radix 2 ACS 구조보다 가산기(Adder) 하나에 해당하는 전파지연이 줄어들게 되어 전체적인 작업 처리량(Throughput)이 향상된다. 하지만 하드웨어 사이즈가 두 배 이상 증가하게 되어 보통 Radix 4 이상의 구조는 잘 사용되지 않는 실정이다. 도 3a 내지 도 3c는 다양한 Radix ACS 구조를 나타내는 도이다. 도 3a는 기본적인 형태인 Radix 2 구조를 나타내며, 크리티컬 패스(critical path)는 가산기(Adder) + 비교기(Comparator) + 댁스(Mux)에 해당한다. 도 3b는 Radix 4 구조로서, 전송지연은 가산기 + 2×비교기 + 2×댁스에 해당 하지만 한 번에 두 단의 데이터를 처리하기 때문에 가산기 하나에 대한 전송지연이 감소하게 된다. 상기와 같은 구조를 개선하기 위해 본 발명이 채용한 ACS 구조를 도 3c에 도시하였다. 도 3c에 도시된 ACS 구조는 Radix 4 Fast ACS 구조로서, 도 3c에 도시된 바와 같이 가산기 + 비교기 + 2×댁스의 전송지연을 가지며, 상기 도 3b에 도시된 구조와 달리 비교기 하나에 해당되는 전송지연이 감소된 것을 알 수 있다.

[0025]

한편, 슬라이딩 방법에서 오버랩 사이즈를 구속장 길이의 5배로 설정할 경우 보다 정확한 결과를 얻을 수 있다. 하지만 오버랩 사이즈의 증가는 추가적인 연산의 증가로 이어지며, 작업 처리량을 감소시키는 요인으로 작용한다. 이에 적당한 오버랩 사이즈를 확인하기 위해 각 채널 상황에 따른 성능변화를 확인하였다. 블록 사이즈가

작을수록 전체 블록의 수가 증가하며 이에 따라 오버랩되는 부분이 증가하게 된다. 오버랩의 영향을 확인하기 위해 블록 사이즈는 48로 설정하여 AWGN(Additive White Gaussian Noise) 채널을 통과시킨 후 디코딩하는 시뮬레이션을 수행하였다. 오버랩이 32일 경우 구축장의 길이의 약 5배에 해당하는 크기이며, 이는 비신뢰 구간이 거의 포함되지 않게 되리라는 것을 알 수 있다. 시뮬레이션 결과를 살펴보면 오버랩이 없을 경우 성능은 열화되지만, 오버랩 사이즈가 8이상일 경우 32인 경우와 거의 동일한 성능을 보임을 알 수 있다. 도 4에는 오버랩 사이즈에 따른 비트 에러율(Bit Error Rate, BER)을 도시한 그래프를 도시하였다.

[0026] 도 5는 본 발명에 따른 슬라이딩 방법을 이용한 MB-OFDM용 고속 비터비 디코더의 구성에 대한 개략적인 블록도이다. 본 발명에 따른 비터비 디코더는 슬라이딩 방법을 기반으로 하며, 도 5에 도시된 바와 같이 두 개의 비터비 디코딩 블록을 두어 개별적으로 디코딩을 수행한다. 또한, 시뮬레이션을 통해 오버랩이 8이상일 경우 성능 열화가 거의 없는 것을 확인하였으며, 기존의 슬라이딩 구조에 비해 디코딩을 수행해야할 데이터의 가외성(Redundancy)을 대폭 줄일 수 있었다.

[0027] 한편, 개별 블록의 비터비 디코딩 블록은 Radix 4 Fast ACS 구조를 사용하여 작업 처리량을 향상시키도록 하였으며, 이러한 구조에 적합하도록 가지 메트릭 계산부에서는 한 번에 두 단의 가지 메트릭(Branch Metric, BM) 값을 생성하도록 설계되었고, 트레이스 백(Trace Back) 구조 또한 한 번에 두 단의 경로(Path) 값을 처리하여 한 클럭(Clock)에 두 개의 비트를 출력하도록 설계되었다.

[0028] 이하, 디펄처링부(510)에 대해 설명한다. MB-OFDM UWB 표준에서는 코딩 레이트(Coding Rate) 1/3를 가지는 길쌈 인코더를 사용한다. 하지만 채널 상태에 따라 코딩 레이트를 달리하기 위해 펄처링(Puncturing) 방법을 이용한다. 이는 인코딩(Encoding)된 값을 모두 전송하는 것이 아니라 일부 데이터를 제외하여 전송하는 방식이다. 도 6에는 각 코딩 레이트에 따라 전송되는 데이터가 어떻게 변화하는지를 나타낸 코드 레이트 별 데이터 흐름도를, 도 7에는 디펄처링(DePuncturing) 구조를 도시한 개념도를 도시하였다. 한편, 수신단에서는 각 코딩 레이트에 따라 데이터가 재구성되도록 해야 하며, 상기 디펄처링부(510)에서는 펄처링된 데이터를 복원하는 디펄처링 과정을 수행한다. 도 7에 도시된 바와 같이 각 디펄처링 모드에 따라 전송되는 데이터의 위치가 틀리고, 각 데이터의 순서에 따라서도 데이터의 위치가 변환하게 된다.

[0029] 이하, 가지 메트릭 계산부(520)에 대해 설명한다. 상기 가지 메트릭 계산부(520)는 채널로부터 수신받은 데이터를 양자화(quantization)하는 과정을 수행한다. 한편, 채널로부터 수신받은 데이터를 양자화하는 과정은 경관정(Hard Decision) 및 연관정(Soft Decision)으로 나누어 볼 수 있다. 여기서 경관정 방식은 이진화 방식이기 때문에 정보의 손실이 발생하며, 연관정 방식을 사용할 경우 최대 2.2dB의 성능개선이 가능하다. 하지만 다양한 레벨의 양자화는 하드웨어 설계의 복잡도를 높이기 때문에 보통 8레벨의 양자화를 많이 사용하며, 경관정보보다 2dB의 성능개선을 얻을 수 있기 때문에 본 발명에서는 8레벨의 양자화 방식에 적합하도록 설계되었다.

[0030] 여기서, 가지 메트릭(BM)은 개별 코드 워드(Code Word)와 입력 데이터의 해밍 거리(Hamming Distance, HD)를 계산함으로써 얻을 수 있다. 코드 워드를 살펴보면 3비트의 정보이기 때문에 8가지 형태로 표현되며 동일한 코드 워드가 16개씩 존재한다는 것을 알 수 있다. 이러한 특징을 이용하면 동일한 코드 워드에 대해 모두 해밍 거리를 계산할 필요없이 8개에 대한 해밍 거리를 계산하고 이를 적절히 공유함으로써 가지 메트릭을 계산할 수 있다.

[0031] 한편, Radix 2에서 Radix 4로 ACS 구조가 바뀔 경우 가지 메트릭 계산부 블록의 경우도 그 구조가 두 개 단의 가지 메트릭을 생성하는 형식으로 변경되어야 한다. 이러한 구조 또한 6개의 코드 워드 비트(Code Word Bit)를 3비트씩 두개 단으로 나누었을 경우 각각 8개의 동일한 코드 워드가 중복되는 것을 알 수 있다. 따라서 각 단별로 8개씩의 해밍 거리(HD)를 계산하고 이를 조합함으로써 256개의 가지 메트릭(BM)을 계산하는 것보다 간단하게 가지 메트릭(BM)을 구성할 수 있다. 상기 가지 메트릭 계산부(520)를 통한 가지 메트릭 생성 구조에 대해서는 도 8에 도시하였다.

[0032] 상기 가산 비교 선택부(530)는 상기 가지 메트릭 계산부로부터 인가되는 새로운 가지 메트릭(Branch Metric) 값과 누적된 경로 메트릭(Path Metric) 값을 각 천이마다 가산한 후 각 상태(State) 별로 비교하여 최적의 경로 메트릭 값을 갖는 생존 메트릭 값과 생존 경로를 출력한다. 본 발명의 가산 비교 선택부(Add-Compare-Selector, ACS)는 한 번에 두 단의 데이터를 처리하는 구조인 것을 특징으로 하는 슬라이딩 방법을 이용하며, Radix 4 ACS

구조를 가지는 것을 특징으로 하는 한다. Radix 4 ACS 구조에 대해서는 전술한 바 있으므로 여기서는 설명을 생략한다.

- [0033] 상기 트레이스 백 연산부(540)는 트리 구조의 비교기를 구성하여 최소 경로 메트릭(Minimum Path Metric, Min PM) 값을 찾아내고, 이를 통해 상기 최소 경로 메트릭이 존재하는 상태를 찾아내며, 상기 최소 경로 메트릭이 존재하는 상태에서부터 실제 출력 값을 생성한다.
- [0034] 비터비 디코더에서는 최대 우도(Maximum Likelihood) 방법을 이용한다. 이는 에러를 포함하더라도 가장 유사한 상태천이 경로(Path)를 찾아 이 경로에 대한 출력 값을 생성하는 것이다. 가장 유사한 상태천이를 찾는다는 것은 에러에 최소라는 의미이기도 하고, 각 경로(Path)를 거치면서 가지 메트릭(BM) 값이 누적된 경로 메트릭(Path Metric, PM)의 값이 최소하는 의미이기도 하다. 일반적인 길쌈 인코더에서는 입력 데이터 스트림의 끝단에 구속장 길이(k)와 동일한 길이의 '0'을 삽입함으로써, 마지막 K단의 상태천이를 거쳐 최소 경로 메트릭(Min PM) 값이 상태(state) 0에서 나타나도록 하고 있다. 한편, 본 발명에서는 슬라이딩 방법을 통해 데이터 스트림을 분할하여 사용하고 있기 때문에 영 채우기(Zero Padding) 방법을 통해 최소 경로 메트릭(Min PM) 값을 찾아낼 수 없다. 따라서 본 발명에서 사용하는 방식은 비교기(Comparator)를 사용하여 최소 경로 메트릭을 찾는 방법을 사용한다. 이러한 과정의 지연시간(Latency)을 줄이기 위해 트리(Tree) 구조로 비교기를 구성함으로써 6 클럭(Clock) 만에 최소 경로 메트릭(Min PM)을 찾을 수 있다. 보통 64개의 상태(State) 중 가장 작은 값을 찾기 위한 트리 구조를 사용할 경우 127개의 데이터를 저장할 레지스터(Register)와 63개의 비교기를 필요로 한다.
- [0035] 한편, 트레이스 백(Trace Back) 연산을 수행하기 위해 찾고자 하는 것은 최소 경로 메트릭(Min PM) 자체가 아니라, 최소 경로 메트릭(Min PM)이 존재하는 상태(State)이다. 또한, 각 상태(State)의 경로 메트릭(PM)값은 최소 경로 메트릭(Min PM)이 존재하는 상태를 찾기 위해서만 필요로 하기 때문에 이 값들이 계속 유지될 필요가 없다. 이러한 특징으로 이용할 경우, 각 비교기에 의해 선택된 경로 메트릭(PM) 값을 비교 전의 경로 메트릭(PM)에 대체함으로써, 첫 번째 단의 레지스터와 비교기만을 반복적으로 사용하여 최소 경로 메트릭(Min PM) 상태를 찾기 위한 선택 정보를 생성할 수 있다. 실제 최소 경로 메트릭(Min Path Metric)을 찾은 것은 각 단계의 선택 정보를 이용함으로써 찾을 수 있다. 각 단계별로 비교기의 비교 결과인 SA, SB, SC, SD, SE, SF가 저장되고, 현재 단계의 결과를 이용하여 이전 단계의 선택 신호를 선택하게 된다. 선택된 신호는 현재의 선택 신호와 합성(Concatenation)되어 다음단의 멀티플렉서(Multiplexer)로 입력되고 최종적으로 최소 경로 메트릭(Min PM) 상태(State)를 선택하게 된다. 도 9에는 최소 경로 메트릭 선택을 위한 모듈의 구성도를 도시하였다.
- [0036] 한편, 가산 비교 선택(Add-Compare-Select, ACS) 연산의 결과들은 보통 메모리에 저장되어 상기 ACS 연산이 모두 끝난 후에 최소 경로 메트릭 상태를 찾고, 그 상태에서부터 실제 출력 값을 생성하게 된다. 여기서 트레이스 백(Trace Back) 연산의 순서는 인코딩(Encoding) 순서와 반대로 진행된다. 왜냐하면, 디코딩(Decoding) 과정은 단순히 가지 메트릭(BM) 과정 만을 수행하는 것이 아니라 ML Path를 추적하여 가지 메트릭(BM)을 누적한 경로 메트릭(PM) 값을 이용하기 때문이다.
- [0037] 아울러, 본 발명에서 사용하는 ACS 구조는 Radix 4 구조이고, 이는 한 번에 두 단의 정보를 처리하기 때문에, 트레이스 백(Trace Back) 과정도 한 번에 두 개의 데이터를 출력하도록 설계된다. 또한, 초기에 최소 경로 메트릭 상태를 입력받고 ACS로부터 저장된 판정(Decision) 정보들을 이용하여 연산을 수행한다. 출력은 현재 상태정보로부터 하위 2비트를 출력하고, 다음 상태로 천이하기 위해 Decision RAM으로부터 두 단에 대한 선택 정보를 받아, 현재 상태에 해당되는 선택정보를 선택하여 쉬프트(Shift) 연산을 수행하게 된다.
- [0038] 이하에서는, 본 발명에 따른 비터비 디코더의 구현결과 및 검증 결과에 대해 설명한다. 본 발명에서는 알고리즘에 따라 Verilog HDL 코딩을 수행하였으며, 이를 검증하기 위하여 ModelSim 시뮬레이션을 수행하였다. 검증과정에 대해서는 도 10에 도시하였다. 먼저 C 코드 시뮬레이터를 각 단계별로 데이터를 확인할 수 있도록 설계하였으며, Verilog로 설계된 비터비 디코더 복호기의 입력으로는 C 코드 시뮬레이터에서 데이터를 생성하여 채널을 통과한 후 8-Level로 만들어진 코드 워드를 입력함으로써 각 단계별로 C 코드 시뮬레이터와 동일한 값을 출력하는지 확인하였다. 구현된 회로의 성능을 확인하기 위하여 Xilinx FPGA V4LX200에 적합하도록 합성하였다. 설계된 비터비 디코더는 전체 면적 중 11%(10,457)를 사용하여 약 16.5만 게이트가 해당하는 하드웨어 사이즈를 갖는다는 것을 알 수 있었다. 합성된 회로는 약 175 Mhz로 동작가능함을 확인하였다.
- [0039] 한편, 기존의 방식보다 높은 성능을 낼 수 있었던 요인은 크게 두 가지로 설명된다. 기존 방법은 Radix 2 방식

을 사용한다. 반해 본 발명에서는 Radix 4 Fast ACS 구조를 사용하여 기본 디코딩 블록의 복잡도는 증가한다. 그러나 기본 방법의 경우 병렬화 방법으로 기본 디코딩 유닛을 8개 사용함으로써 작업 처리량(Throughput)을 높이는 방식으로 접근한 반면, 본 발명에서는 2개의 디코딩 유닛을 사용하였기 때문에 하드웨어 사이즈의 감소를 이끌어 낼 수 있었다. 동작 속도면에서는 오버랩(Overlap) 사이즈를 줄임으로서 디코딩해야할 데이터의 가외성(Redundancy)을 줄임으로서 높은 작업 처리량(Throughput)을 얻을 수 있었다. 이렇게 구현된 본 발명에 따른 비터비 디코더는 MB-OFDM 표준안을 만족하며, 본 발명에 따른 비터비 디코더의 개선된 사항에 대해서는 하기의 표 2에 도시된 바와 같다.

표 2

	ISCAS 2007 ^[6]	Proposed	비교
Number of used Slices	22,208	10,457	47%
Number of used LUTs	33,719	19,842	58%
Critical Path[ns]	9.203	5.746	62.4%
Operating Frequency [Mhz]	108	174	161%
Throughput[Mbps]	432	561	130%

[0040]

[0041] 이상과 같이 본 발명에 따른 슬라이딩 방법을 이용한 UWB용 고속 비터비 디코더 예시한 도면을 참조로 하여 설명하였으나, 본 명세서에 개시된 실시예와 도면에 의해 본 발명이 한정되는 것은 아니며, 본 발명의 기술사상 범위내에서 당업자에 의해 다양한 변형이 이루어질 수 있음은 물론이다.

도면의 간단한 설명

[0042]

도 1은 MB-OFDM 길쌈 부호기의 구조를 나타내는 도이다.

[0043]

도 2는 슬라이딩 방법을 나타내는 개념도이다.

[0044]

도 3a 내지 도 3c는 다양한 Radix ACS 구조를 나타내는 도이다.

[0045]

도 4는 오버랩 사이즈에 따른 비트 에러율을 도시한 그래프이다.

[0046]

도 5는 본 발명에 따른 비터비 디코더를 도시한 블록도이다.

[0047]

도 6은 코드 레이트 별 데이터 흐름을 도시한 도이다.

[0048]

도 7은 디평처링 구조를 도시한 개념도이다.

[0049]

도 8은 가지 메트릭 생성 구조를 도시한 개념도이다.

[0050]

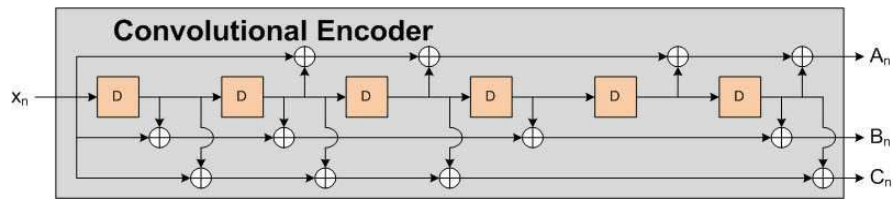
도 9는 최소 경로 메트릭 선택을 위한 모듈의 구성도이다.

[0051]

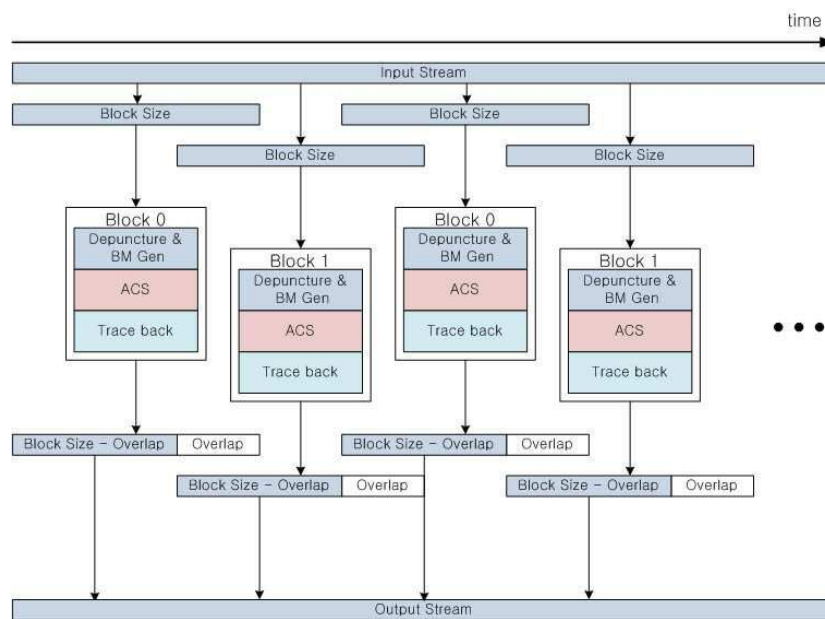
도 10은 본 발명에 따른 비터비 디코더의 검증 과정을 도시한 블록도이다.

도면

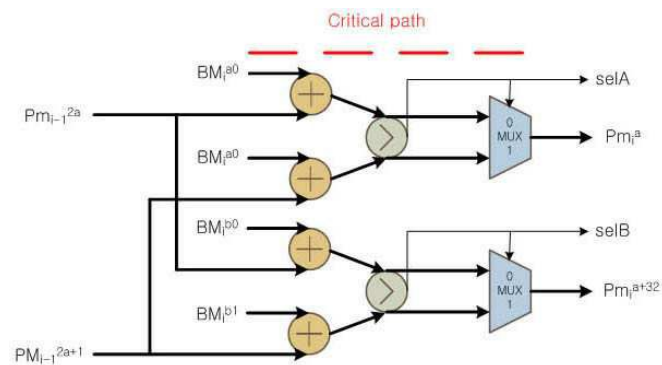
도면1



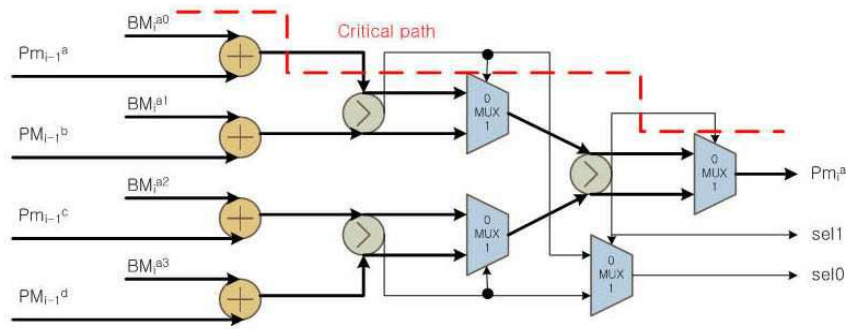
도면2



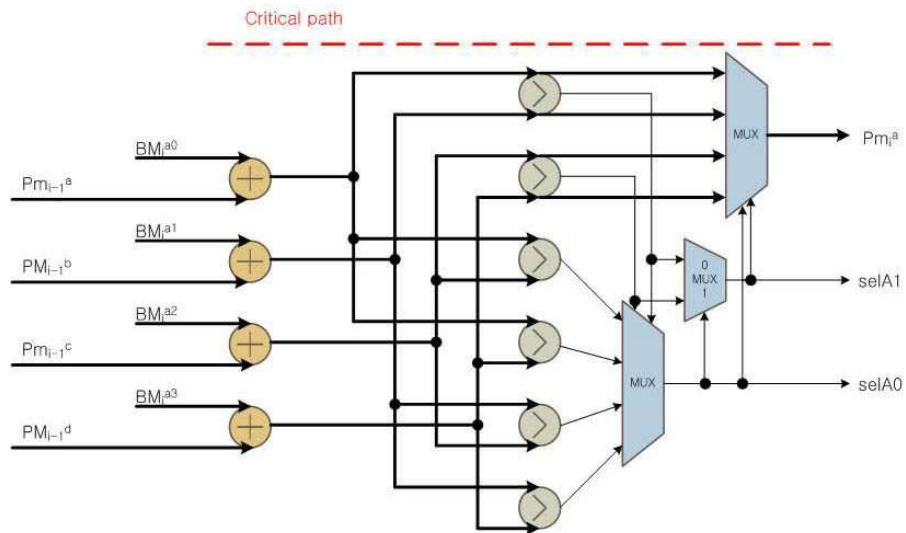
도면3a



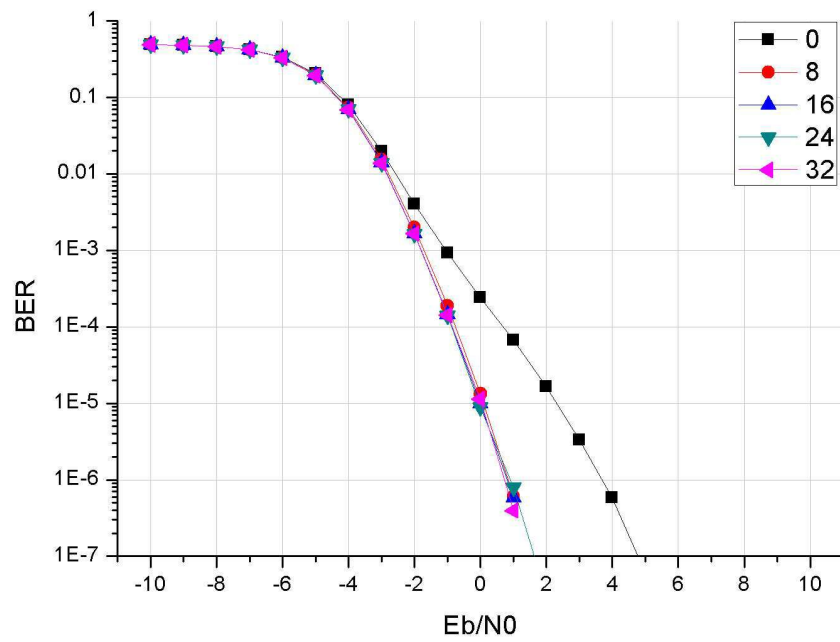
도면3b



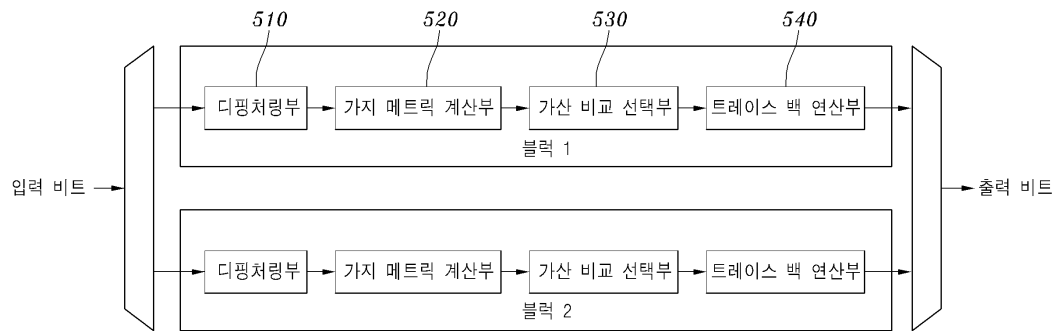
도면3c



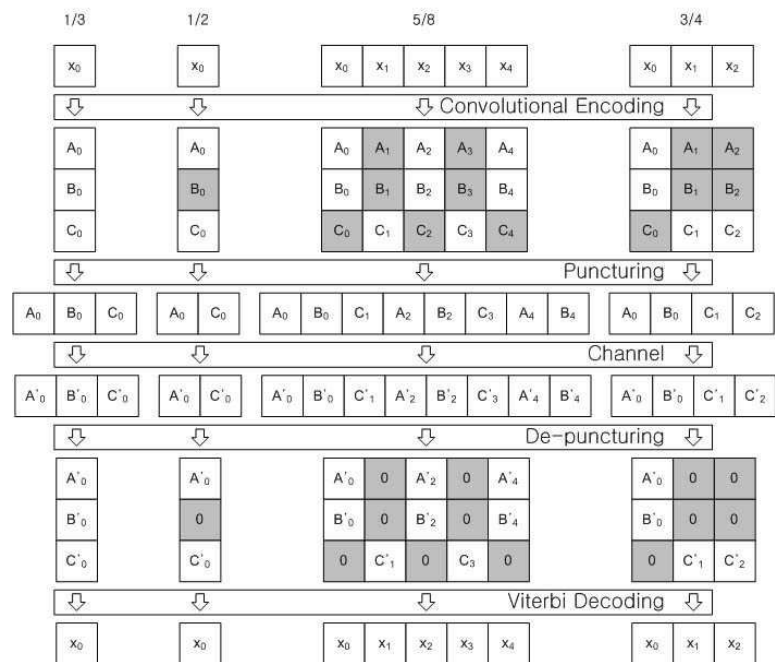
도면4



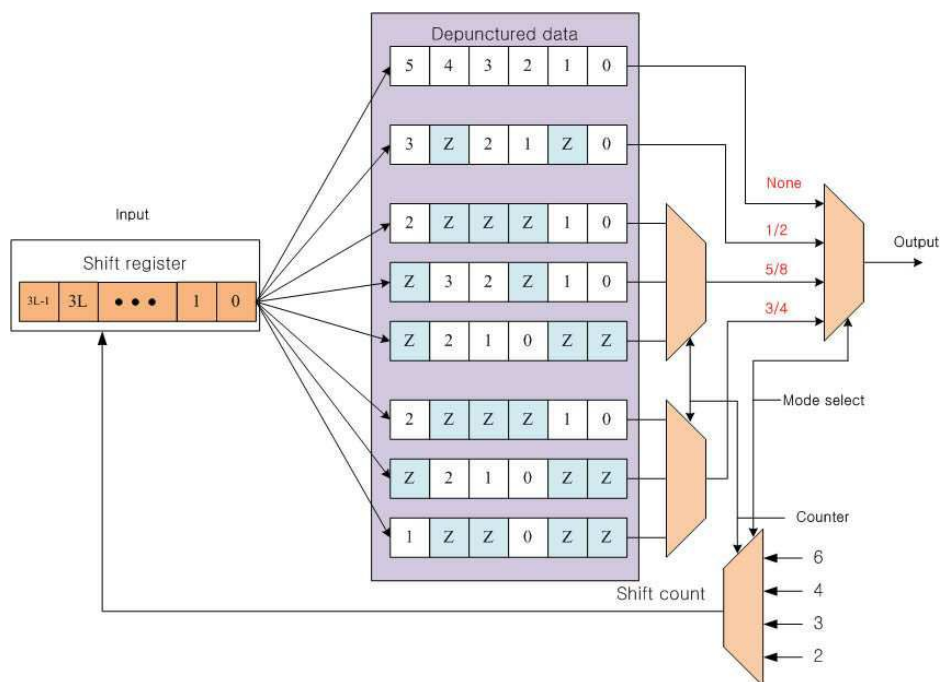
도면5



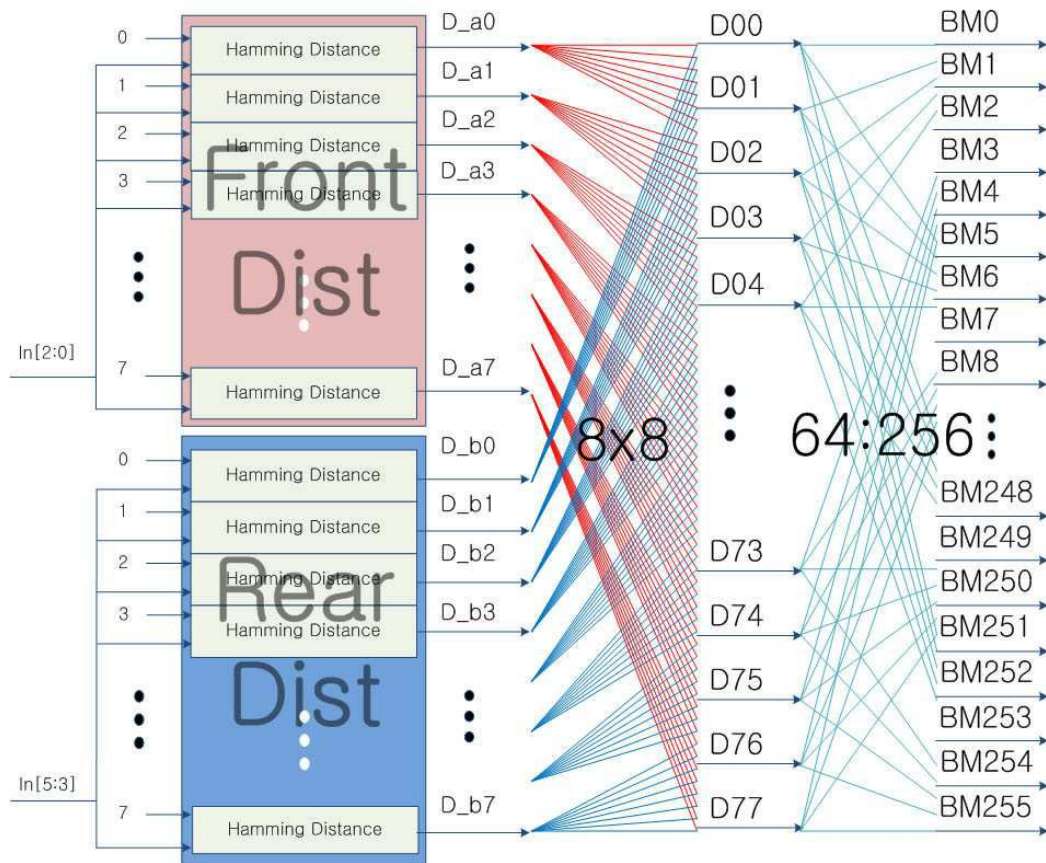
도면6



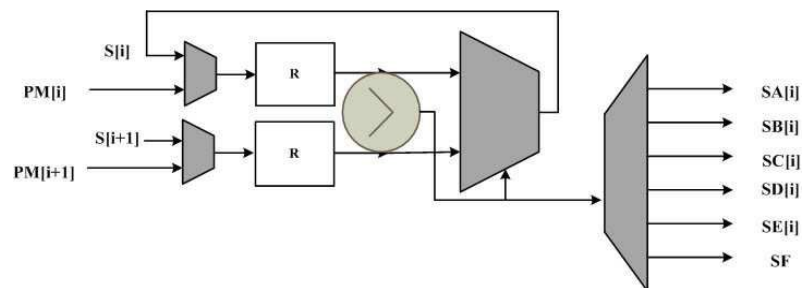
도면7



도면8



도면9



도면10

