논문 2006-43SD-7-7

하드웨어 구현에 적합한 효율적인 LDPC 코덱의 설계

(Design of an Efficient LDPC Codec for Hardware Implementation)

이 찬 호*, 박 재 근**

(Chanho Lee and Jaegeun Park)

요 약

Low Density Parity Check (LDPC) code는 최근 그 우수한 성능으로 인하여 4세대 무선 이동 통신용 채널 코딩으로 주목받고 있고 유럽의 고화질 위성방송 규격으로 채택되었다. 그러나 기존의 연구들이 제안한 parity check matrix (H-matrix)는 실제로 하드웨어로 구현함에 있어서 인코더 혹은 디코더에 제약을 가지고 있다. 이러한 문제점을 해결하고자 본 논문에서는 인코더와 디코더 양쪽 모두 효율적으로 하드웨어로 구현이 가능한 hybrid H-matrix 구조를 제안한다. Hybrid H-matrix는 semi-random 방식과 partly parallel 방식을 결합하여 하드웨어로 구현시 partly parallel 방식이 가지는 디코더의 복잡도가 감소되는 장점을 유지하면서 인코더 또한 semi-random 방식을 사용하여 복잡도가 감소된다. 제안한 구조를 사용하여 LDPC 인코더와 디코더를 설계하고 합성하여 기존의 결과와 비교하였다.

Abstract

Low-density parity-check (LDPC) codes are recently emerged due to its excellent performance. However, the parity check (H) matrices of the previous works are not adequate for hardware implementation of encoders or decoders. This paper proposes a hybrid parity check matrix which is efficient in hardware implementation of both decoders and encoders. The hybrid H-matrices are constructed so that both the semi-random technique and the partly parallel structure can be applied to design encoders and decoders. Using the proposed methods, the implementation of encoders can become practical while keeping the hardware complexity of the partly parallel decoder structures. An encoder and a decoder are designed using Verilog-HDL and compared with the previous results.

Keywords: Low density parity check (LDPC), semi-random, Hybrid H-matrix, partly parallel structure

I. 서 론

최근의 무선 이동 통신 서비스는 다양한 멀티미디어 서비스를 제공하기 위해 고속 전송이 가능하도록 발전하고 있다. IMT-2000으로 불리는 3세대 무선 이동 통신은 데이터 전송 속도를 고속 이동시 144kbps, 보행시 384kbps, 정지 시 2Mbps까지 제공하는 것을 목적으로 하였다[1]. 하지만 이러한 조건도 사용자들이 요구하는 유선 통신 서비스와 동일한 품질의 무선 이동 통신 서비스와 고속, 실시간에서 비실시간까지 다양한 서비스를 지원하기에는 한계가 있다고 판단되었다. 그에 따라 4세대 이동 통신에 대한 개발이 진행되고 있다. 4세대 이동 통신과 3세대 이동 통신의 차이점은 여러 가지 있겠으나, 가장 뚜렷한 차이점은 고속 이동시 100Mbps, 저속 이동 및 정지 시 155Mbps~1Gbps까지 제공한다는 것에 있다. 따라서 매우 높은 전송 속도를 가지는 무선 전송 시스템이 필요하다.

더불어 무선 이동 통신의 경우 path loss, shadowing, fading 등의 다양한 원인 때문에 생기는 잡음에 대응하기 위한 고성능의 채널 코딩 기법들의 개발은 더욱 중요하다. 채널 코딩은 데이터 전송에 있어서, 채널에서 발생되는 잡음에 의한 오류를 수신측이 검출 혹은 정정할 수 있도록 원래의 데이터에 추가로 리던던시(redundancy)를 덧붙이는 방법이다. 이때 신호 파워 증

* 정회원, 숭실대학교 정보통신전자공학부
(School of Electronic Engr., Soongsil University)
** 학생회원, 숭실대학교 전자공학과
(Dept. of Electrinic Engr., Soongsil University)
※ 본 연구는 숭실대학교 교내연구비 지원으로 이루어졌음
접수일자: 2006년3월21일, 수정완료일: 2006년7월3일

가 없이 비트 에러율 (BER)을 줄일 수 있다. 이러한 코딩 기술은 GSTN(General Switched Telephone networks)과 같은 유한 파워 채널의 전송에 유용하다.

LDPC(Low Density Parity Check) code는 block code의 일종으로 1962년에 Gallager에 의해 처음 제안되었다[2]. 하지만, 그 당시의 기술력으로는 구현이 불가능한 복호의 복잡도로 인해서 실용화 되지 못하고 한동안 잊혀져 오고 있었다. 그러나 1995년에 Mackay와 Neal은 이를 재발견하였고, Gallager의 방식이 간단한 확률적 복호 법에 의해서 성능이 매우 우수함을 보였다[3]. 최근에는 백색 잡음(additive white Gaussian noise: AWGN) 채널에서 Shannon의 채널 용량에 불과 0.0045dB 떨어진 임계 치를 갖는 LDPC 코드가 Chung, et al에 의해 발견되었다[4]. 이러한 LDPC code는 LDPC code가 재발견되기 이전에 가장 우수한 채널 코딩 기법이었던 Turbo Code와 비교했을 때 작은 최소 거리(minimum distance)를 가지기 때문에 오류 마루(error floor) 현상이 거의 나타나지 않아 훨씬 좋은 bit error rate(BER)를 가진다는 장점이 있다. 또한 합곱(sum-product) 알고리즘을 기반으로 한 반복 복호 과정을 완전히 병렬로 처리할 수 있기 때문에 복호 속도가 빠르다는 장점도 있다[5]. 이러한 이유로 고속과 고성능을 요구하는 4세대 이동 통신용 채널 코딩으로 LDPC code가 주목받고 있다. 현재 유럽 방식 고화질 위성방송용 규격인 DVB-S2 규격은 LDPC code와 BCH code를 결합한 연접부호를 적용하고 있다[6].

그러나 인코더의 복잡도가 여전히 높다는 것이 하드웨어로 구현할 경우 가장 큰 문제점이다. 이러한 문제점을 해결하기 위한 방법으로 lower triangular matrix를 사용하는 방법[7]과 semi-random matrix[8]를 사용하는 방법이 제안되었다.

기존의 인코딩 과정은 일반적으로 H-matrix가 non-systematic한 구조를 가지고 있기 때문에 이를 systematic한 구조로 바꾸는 과정이 필요하다[8]. 이러한 과정을 위해 가우스 소거법을 이용하게 되며 이는 사용되는 메모리와 연산량을 증가시킨다. Semi-random 방식을 이용할 경우 인코딩 과정에 가우스 소거법을 이용하지 않기 때문에 인코딩 과정의 복잡도를 줄일 수 있다. 동시에 작은 크기의 메모리만을 사용하고도 linear time 인코딩을 할 수 있다.

디코더를 fully parallel 방식으로 구현할 경우 디코딩 속도가 매우 높다는 장점이 있으나 하드웨어로 구현할 경우 차지하는 면적이 굉장히 크다는 단점이 있다[9]. Belief propagation (BP) 디코딩 알고리즘[10]을 이용하여 fully parallel 방식으로 하드웨어를 구현할 경우 각각의 check node들과 variable node들 간의 메시지 전송이 완전히 병렬로 처리되므로 각 node들은 자신만의 프로세서를 가지고 있어야한다. 이는 node 수가 많아질수록 프로세서의 수도 늘어나며 node들 간의 연결도 복잡해진다는 것을 나타낸다. 따라서 디코더의 복잡도를 줄이기 위해서는 check node와 variable node의 수를 줄여야만 한다. Partly parallel 방식은 이러한 점을 개선하기 위하여 시분할 다중화 방식(time-division multiplex mode)[11]을 이용하여 부분 병렬 처리를 적용한 방식이다. Partly parallel 방식을 이용할 경우 디코딩 throughput과 하드웨어 복잡도간의 trade-off가 필요하다.

Partly parallel 방식을 사용하여 디코더를 구현할 경우 하드웨어 복잡도가 줄어들기는 하지만 H-matrix가 여전히 non-systematic한 구조를 가지고 있기 때문에 인코더의 복잡도가 여전히 크다는 문제점을 가지고 있다.

본 논문에서는 semi-random 방식과 partly parallel 방식을 결합하여 인코더와 디코더 모두 효율적인 구현이 가능한 hybrid H-matrix 구조를 제안한다. Hybrid H-matrix 구조는 semi-random 방식의 장점과 partly parallel 방식의 장점을 모두 가지고 있다. Semi-random 방식을 적용하여 인코더의 문제점을 해결하였으며, partly parallel 방식을 적용하여 디코더의 문제점을 제거하였다. 제안한 hybrid H-matrix를 이용하여 실용 가능한 인코더와 디코더를 구현할 수 있고 DVB-S2 규격의 LDPC 인코더와 디코더 구조에도 적합하다. Hybrid H-matrix를 이용한 인코더와 디코더를 구현하기 위하여 32 x 32 크기의 base matrix를 생성하고, 이를 이용하여 고성능의 LDPC 인코더와 디코더를 구현하였다.

Ⅱ. Hybrid H-matrix 구조

1. Semi-random 방식

Semi-random 방식을 이용한 H-matrix의 구조는 H^d와 H^p 두 부분으로 분리된 형태로 구성된다. 여기서 H^d는 랜덤한 matrix 형태를 가지며, H^p는 deterministic한 형태를 가진다[8]. 그림 1과 같은 dual-diagonal한 형태의 정방형 matrix를 H^p로 사용함으로써 인코더 부분의 복잡도를 줄이게 된다.

생성된 H^d-matrix를 사용한 H-matrix는 $H=[H^d,H^p]$

$$H^p = \begin{bmatrix} 1 & 0 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 1 & 1 & 0 & \cdots & \cdots & 0 & 0 & 0 \\ 0 & 1 & 1 & \cdots & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \cdots & \cdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \cdots & \cdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 0 & 1 & 1 \end{bmatrix}$$

그림 1. Semi-random 방식에서의 deterministic한 형태의 matrix

Fig. 1. Matrix of deterministic form in semi- random technique.

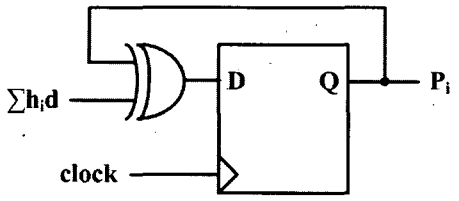

그림 2. 패리티 비트 생성 회로

Fig. 2. Parity bits generator.

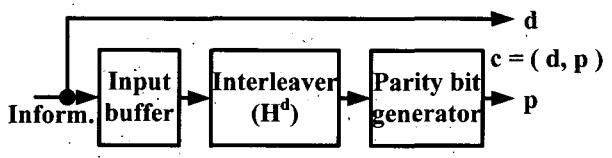

그림 3. Semi-random 방식을 이용한 인코더의 블록 다이어그램

Fig. 3. Encoder block diagram using semi-random technique.

의 구조를 가지며, 이때 code-word는 C=[d,p]^t(d:정보 비트, p:패리티 비트)의 형태로 만들어지며 패리티 비트는 다음과 같은 식을 이용하여 쉽게 계산할 수 있고[8] 그림 2와 같은 간단한 회로로 구현할 수 있다[12].

$$p_1 = \sum_j h_{1j}^d d_j$$
$$p_i = p_{i-1} + \sum_j h_{ij}^d d_j \quad (1)$$

그림 2와 같은 패리티 비트 생성 회로를 이용하여 그림 3과 같은 인코더를 구성할 수 있다. H^d-matrix의 구성에 따라 인터리버(interleaver) 부분이 달라지는 부분이며, 이 인터리버 부분의 변화에 따라 LDPC 코드의 성능이 결정된다.

2. Partly parallel 방식

Partly parallel 방식은 fully parallel 방식이 가지는 하드웨어 크기가 크다는 문제점을 해결한 방식이며 이

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} T_{1,1} & O & O & T_{1,4} & O \\ O & T_{2,2} & T_{2,3} & O & O \\ T_{3,1} & O & O & O & T_{3,5} \end{bmatrix}$$

Base Matrix (M_s x N_s) — Expanded Matrix (pM_s x pN_s)

그림 4. Partly parallel 구조를 위한 expanded matrix의 구성

Fig. 4. Expanded matrix for Partly parallel structure.

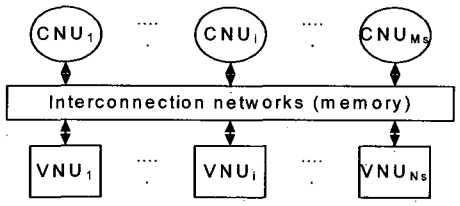

그림 5. Partly parallel 방식의 디코더 구조

Fig. 5. Decoder structure of Partly parallel method.

것은 Zhang 등이 처음 제안했다[13]. 이 방식은 우선 디코더의 구조를 구성한 후에 shift된 identity matrix를 이용하여 H-matrix를 만든다. 그러나 이러한 방식은 code rate의 변화에 민감하며 성능의 떨어지는 단점이 있다[11]. 이것을 개선한 방식이 matrix expansion 방식이다. 이것은 (M_s x N_s) 크기의 base matrix와 (p x p) 크기의 shift된 identity matrix로 구성된다.

Base matrix의 생성은 bit-filling 알고리즘을 이용하여 높은 girth를 가지도록 하였다[14]. 높은 girth를 가지는 base matrix는 cycle이 작을 때 나타나는 성능의 저하를 줄일 수 있다. 그림 4에서처럼 (M_s x N_s) 크기의 base matrix가 있다면 base matrix 안의 원소 '1'이 차지하는 위치에 랜덤한 수만큼 오른쪽으로 shift된 (p x p) 크기의 identity matrix(I)를 대치시키고, 원소 '0'이 차지하는 위치에는 (p x p) 크기의 영행렬을 대치시킴으로써 (pM_s x pN_s) 크기로 확장된 H-matrix를 얻을 수 있다. 그림 4에서 $T_{u,v}$는 identity matrix의 shift되는 정도를 나타낸다. 이러한 방식으로 base matrix를 확장시킴으로써 설계의 유연성이 증가된다. 이러한 구조의 H-matrix를 사용하여 디코더를 구현할 경우 모든 node가 프로세서를 가질 필요 없이 시분할 다중화 방식으로 공유할 수 있다[13].

Partly parallel 방식을 사용하여 디코더를 구현할 경우, 그림 5와 같이 CNU(Check Node processor Unit)과 VNU(Variable Node processor Unit)의 개수는 각각 base matrix의 열과 행의 개수와 같은 M_s개와 N_s개만을 필요로 한다[11]. H-matrix의 크기가 같을 경우, Fully parallel 방식의 디코더는 각각 pM_s개의 CNU와

pN_s개의 VNU를 필요로 하며 이것은 partly parallel 방식의 디코더보다 p배 많은 수이다. 하지만 fully parallel 방식의 디코더가 2 cycle만에 한번의 iteration이 가능한데 비하여 partly parallel 방식의 디코더는 2p cycle을 필요로 한다.

이러한 Partly parallel 방식의 디코더 구조는 디코더를 효율적으로 구현할 수 있도록 하지만 인코더 구조에 대해서는 여전히 복잡도가 크다는 문제점을 가지고 있다.

3. Hybrid H-matrix 구조

LDPC code의 인코더와 디코더의 하드웨어 구현에 대한 방법들을 알아보았으나 여전히 문제점이 남아있다. Semi-random 방식을 사용할 경우에는 인코더의 구현이 간단하지만 디코더의 구현이 굉장히 복잡하다. 반면 partly parallel 방식을 사용할 경우에는 디코더의 구현이 비교적 간단하지만 인코더의 구현에 문제가 있다.

이러한 하드웨어 구현의 문제점들을 해결하고자 본 논문에서는 hybrid H-matrix 구조의 LDPC code를 제안한다. Hybrid H-matrix 구조는 II.1, II.2에서 살펴본 semi-random 방식과 partly parallel 방식을 결합한 구조이다. 그림 6은 이러한 hybrid H-matrix의 구조를 보여주고 있다. Hybrid H-matrix는 $[H^d|H^p]$의 systematic 한 구조를 가지고 있으며, H^d는 partly parallel 방식을 따르는 random matrix이며, H^p는 semi-random 방식을

$$H^d = \begin{bmatrix} T_{1,1} & O & O & T_{1,4} & O \\ O & T_{2,2} & T_{2,3} & O & O \\ O & O & O & O & T_{3,5} \end{bmatrix} \qquad H^p = \begin{bmatrix} 1 & 0 & \cdots & \cdots & 0 & 0 \\ 1 & 1 & \cdots & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & 1 & 0 \\ 0 & 0 & \cdots & \cdots & 1 & 1 \end{bmatrix}$$

그림 6. Hybrid H-matrix의 구조
Fig. 6. Structure of hybrid H-matrix.

따르는 dual-diagonal matrix를 사용하고 있다. 이러한 구조로 인하여 인코더에서는 semi-random 방식의 장점을, 디코더에서는 partly parallel 방식의 장점을 살릴 수 있다. 따라서 hybrid H-matrix 구조를 사용할 경우 인코더와 디코더 양쪽 모두 실제로 사용 가능할 정도의 하드웨어로 구현이 가능하다.

4. 인코더 구조

제안한 hybrid H-matrix 구조를 사용한 인코더의 구조가 그림 7에 나와 있다. 메모리와 exclusive-OR 게이트, 멀티플렉서를 사용한 패리티 비트 생성 회로를 이용하여 비교적 간단하게 구현될 수 있음을 알 수 있다. 메모리 블록의 크기는 base matrix의 행의 수와 같다. 입력되는 정보 비트는 메모리에 저장되는 동시에 'd'포트로 전송되고, 정보 비트가 모두 전송된 후에 패리티 비트가 저장되어 있는 정보 비트들을 이용하여 한 비트씩 계산되어 'p'포트로 전송된다.

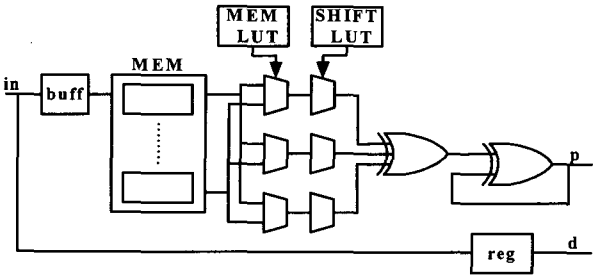

그림 7. Hybrid H-matrix를 이용한 인코더의 구조
Fig. 7. Encoder structure using hybrid H-matrix.

5. 디코더 구조

그림 8은 hybrid H-matrix를 이용한 디코더 구조이다. 디코더의 복잡도를 줄이기 위하여 sum-product algorithm[3]대신 normalized UMP-BP (Uniformly Most Probable Belief Propagation) algorithm을 사용하였다[15]. 수신된 신호는 VNU (Variable Node Unit)와 CNU (Check Node Unit)를 통과하며 반복 복호 과정을 거친 후 Decision Unit에서 최종 판단을 하게 된다. 반복 복호는 미리 정해진 수만큼 하는 방법과 디코딩 상태에 따라 조절하는 방법이 있다.

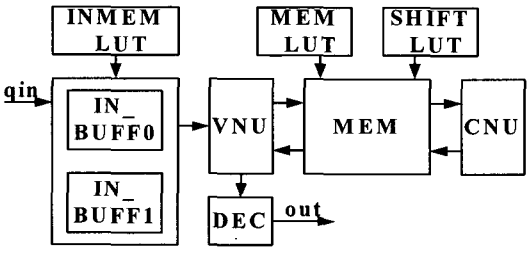

그림 8. Hybrid H-matrix를 이용한 디코더 구조
Fig. 8. Decoder structure using hybrid H-matrix.

III. 설계와 구현

제안한 Hybrid H-matrix 구조를 이용하여 인코더와 디코더를 구현할 경우 인코더와 디코더 양쪽 모두에 효

율적인 구현이 가능하다. 이를 위해 bit-filling 알고리즘을 이용하여 (32 x 32) 크기의 base matrix를 생성한 후 (256 x 256) 크기의 identity matrix를 이용하여 (8,192 x 8,192) 크기의 H-matrix를 만들었다. Base matrix 안의 '1'의 자리는 각각 랜덤한 크기로 오른쪽 shift된 identity matrix로 대치했으며, '0'의 자리는 영행렬로 대치했다. Base matrix가 6-cycle-free 특성을 가지기 위한 최소 크기는 (32 x 32)이다. 그림 9는 이러한 방식으로 생성한 (8 x 8) base matrix의 예이다. Base matrix 안의 숫자는 각각의 identity matrix가 오른쪽으로 shift된 정도를 나타낸다.

이렇게 생성한 Hd-matrix와 그림 1에서 보였던 방식의 (8,192 x 8,192) 크기의 H^p-matrix를 결합하여 H-matrix를 만든다. 최종적으로 H-matrix의 크기는 (16,384 x 8,192)이며, code rate는 1/2이다. Hybrid partly parallel 방식의 인코더를 0.35um CMOS standard cell library를 사용하여 합성한 결과는 표 1과 같다. 인코더 I은 직렬방식으로 처리하여 한 cycle에 한 bit씩 출력을 내보낸다. 면적은 33,900 gate counts이며 메모리는 8,608 bits가 사용되었다. 최대 동작 주파수는 99.5MHz이며 throughput 또한 99.5Mbps이다. 이러한 결과는 hybrid H-matrix를 사용하여 구현한 인코더의 복잡도가 낮은 것을 보여준다.

인코더 II는 throughput을 증가시키기 위해 하나의 identity matrix의 열 단위로 데이터를 처리하여 한 cycle에 256 bit씩 출력하도록 개선한 구조이다. 인코더 I에 비해 복잡한 멀티플렉서가 사라져 게이트 수는 17,970으로 감소한 반면, 병렬 데이터 처리를 위해 메모리 크기가 24,992 bit로 증가하였다. 또한 동작 주파수는 45.7MHz로 감소하였으나 한 cycle에 256 bit씩 출력하므로 throughput은 11,699 Mbps를 보인다. 따라서 다소 면적은 증가하지만 빠른 처리속도가 필요한 경우는 인코더 II 구조가 적합하다.

$$\begin{bmatrix} 0 & 41 & 35 & 62 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 33 & 0 & 0 & 44 & 0 \\ 22 & 0 & 46 & 0 & 0 & 18 & 0 & 0 \\ 16 & 0 & 0 & 0 & 9 & 0 & 0 & 49 \\ 0 & 49 & 0 & 0 & 0 & 59 & 0 & 41 \\ 0 & 0 & 0 & 43 & 51 & 38 & 0 & 0 \\ 0 & 0 & 27 & 0 & 0 & 0 & 60 & 7 \\ 0 & 12 & 0 & 0 & 62 & 0 & 25 & 0 \end{bmatrix}$$

그림 9. 구현을 위해 생성한 (8x8) base matrix의 예제

Fig. 9. Example of (8x8) base matrix for implementation.

Hybrid H-matrix 구조를 이용한 디코더의 구조는 partly parallel 방식을 이용한 디코더 구조와 유사하다. 그림 10에서 VNU 부분과 인터리버 부분은 H^p-matrix에 대한 연산을 위해 partly parallel 방식의 VNU와 인터리버를 변형한 것이다. 또한 CNU 부분과 디인터리버 부분도 hybrid H-matrix 구조에 맞게 약간 변형되었다. Hybrid partly parallel 방식으로 1개의 CNU와 VNU를 이용하여 직렬방식으로 디코딩을 하는 디코더 I의 경우 0.35um CMOS standard cell library를 사용하여 합성한 결과, 면적은 6,900 gate count이며 메모리는 337,600 bits가 사용되었다. 최대 동작 주파수는 98.1MHz이며 최대 throughput은 49Mbps이다. 디코딩 속도를 증가시키기 위해 32배의 CNU와 VNU를 이용하여 병렬 처리하는 디코더 II는 디코더 I과 동일한 구조를 갖고 CNU와 VNU만 32배로 증가하므로 디코더 I의 합성 결과에서 추정하여 보면 게이트 수가 31,400이고 메모리는 직렬 방식과 같이 337,600 bits, 동작 주파수는 98.1MHz, 최대 처리 속도는 1,569Mbps로 예상된다.

표 2는 제안한 LDPC 구조의 모체인 semi-random 방식과 partly parallel 방식에 대해 하드웨어 복잡도,

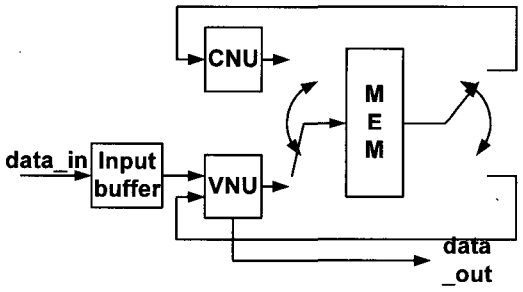

그림 10. 제안한 방식의 디코더 구조

Fig. 10. Decoder structure of proposed methods.

표 1. 0.35um CMOS standard cell library를 사용하여 합성한 결과

Table 1. Synthesis result using 0.35um CMOS standard cell library.

	Encoder I	Encoder II	Decoder I	Decoder II
Logic Gate-count	33,900	17,970	6,900	31,400*
Memory size [bits]	8,608	24,992	337,600	337,600
Operating frequency [MHz]	99.5	45.7	98.1	98.1*
Throughput [Mbps]	99.5	11,699	49	1,569*

*: 추정치

표 2. 세 가지 LDPC code 합성 방식의 비교
Table 2. Comparison with three synthesis mothods of LDPC codes.

(N = 16,384, p = 256, code rate = 1/2)

	Semi-random	Partly parallel	Hybrid H-matrix
HW complexity of encoder	N	>>N	N
CNU	8,192	32	32
VNU	16,384	64	64
Decoding latency (1 iteration)	2	2p	2p
memory in decoder	L+pN	L+N	L+N

N : Block length, p : identity matrix size

표 3. Hybrid H-matrix를 이용한 encoder II, decoder II와 Block-LDPC, LDPC-CC를 이용한 encoder와 decoder의 면적과 성능 추정치 비교
Table 3. Comparison with an area and a performance of encoder II, decoder II using hybrid H-matix and those of encoder, decoder using Block-LDPC, LDPC-CC.

	Gate-count (logic)	Memory size [bits]	Throughput [Mbps]	Operating frequency [MHz]
Hybrid H-matrix encoder II[(1)]	17,970	24,992	11,699	45.7
Block-LDPC encoder[(1)]	41,344	115,200	6.63	99.5
Hybrid H-matrix decoder II[(1)]	31,400	337,600	1,569	98.1
Block-LDPC decoder[(1)]	145,920	475,136	1,569	98.1
Hybrid H-matrix encoder II[(2)]	17,970	1,280	731.2	45.7
LDPC-CC encoder[(2)]	$0.0515mm^2$ 약 5천 GC[(4)]		430	430
Hybrid H-matrix decoder[(3)]	31,400	24,320	156.9 (I=10)[(5)]	98.1
	약 8-9만 GC[(4)]			
LDPC-CC decoder[(3)]	X	245,760	164 (I=10)[(5)]	164
	$10mm^2$ 약 40-50만 GC[(4)]			

(1) H-matrix 크기: 16,384 X 8,192
(2) H-matrix 크기: 1,024 X 512
(3) 1,024 X 512의 H-matrix 크기를 갖는 구조와 비슷한 BER 성능을 갖는 메모리 M=128인 구조
(4) GC : gate count (5) I : iteration 횟수

throughput, 메모리 크기의 관점에서 비교한 것이다. Semi-random 방식을 사용한 디코더는 fully parallel 디코딩 방식을 사용한 것으로 가정했다. 제안한 hybrid H-matrix 방식의 디코더는 partly parallel 방식과 비슷한 하드웨어 복잡도를 가지며, 인코더는 partly parallel 방식보다 낮은 하드웨어 복잡도를 보여준다.

표 3에서 Block LDPC와 LDPC-CC의 인코더와 디코더를 제안한 구조와 비교한 결과를 보여주고 있다[16-19]. Block LDPC 구조를 이용하여 인코더의 복잡도와 속도를 예측하는 식에 base matrix의 행의 개수, 항등 행렬 크기 등을 적용하여 추정된 사이즈와 성능을 비교한 것이 표 3의 2, 3행에 해당된다[16]. Block LDPC의 인코더 면적은 가장 작은 값을 갖는 경우에 대해 추정한 값이나 제안한 인코더 II보다 면적이 크다[16]. 디코더의 경우에도 H-matrix의 크기를 16,384 x 8,192로 했을 때의 면적을 4, 5행에서 비교하였다[16]. 역시 동일한 성능에서 제안된 구조의 면적이 더 작음을 알 수 있다.

기존 LDPC 코드는 일정한 블록 길이를 요구하는 block-oriented code(BC)로 구현된 반면, 임의의 블록 길이가 가능한 convolutional code(CC) 방식도 제안되었다[17]. LDPC-CC에서 메모리 크기가 M=128 인 경우와 LDPC-BC의 블록 길이가 N=1,024 인 경우, 각각 iteration 횟수가 32일 때 BER이 비슷하다[18]. 따라서 N=1,024 로 맞추기 위해 Hybrid H-matrix에서 base matrix는 (32 x 32)로 하여 성능이 열화 되지 않게 하면서, 항등 행렬을 (16 x 16)로 조정하여 인코더 II와 디코더의 면적과 성능을 추정한 결과가 표 3의 6, 8행에 나타나 있다. 그리고 LDPC-CC 방식으로 0.18um CMOS technology를 이용하여 ASIC으로 구현된 인코더와 디코더의 성능과 면적이 표 3의 7행과 9행에 나타나 있다[19]. 제안된 구조의 throughput이 2배 정도 좋은 반면 면적은 4배 정도 큰 것을 볼 수 있다. 그러나 제안된 구조에서는 H-matrix의 크기가 커지면 면적은 그다지 증가하지 않으면서 throughput은 행렬 크기에 비례하여 증가한다. 따라서 현실적으로 행렬 크기가 N>10,000 이상으로 이용되는 것을 고려하면 제안된 구조가 더 우수함을 알 수 있다.

디코더의 경우 10번의 iteration 횟수를 가정하였다. 비슷한 throughput에 대해 hybrid H-matrix 방식의 디코더가 더 작은 면적을 갖는 것을 알 수 있다.

IV. 결 론

LDPC의 구현에서 Semi-random 방식을 사용할 경우 인코더의 효율적인 구현이 가능하지만 디코더의 하드웨어 복잡도가 매우 높다. 반면에 partly parallel 방식을

사용할 경우 디코더의 효율적인 구현이 가능하지만 인코더의 효율적인 구현이 어렵다. 기존에 제안된 구조들이 인코더와 디코더 어느 한쪽에 최적화된 반면 Hybrid H-matrix는 semi-random 방식과 partly parallel 방식의 장점을 조합하여 semi-random 방식과 비슷한 인코더의 구현이 가능하고 디코더 하드웨어 복잡도는 partly parallel 방식과 비슷하다. 이러한 특징을 가진 (16,384 x 8,192) 크기의 hybrid H-matrix를 생성하여 인코더와 디코더를 설계하였다. 합성 결과 인코더는 33,900 게이트와 8,069 bit의 메모리, 디코더는 6,900 게이트와 337,600 bit의 메모리로 구성되어 적절한 하드웨어 크기를 가짐을 보였다. 또한 면적을 크게 증가시키지 않고도 구조의 병렬성을 증가시켜 처리속도를 크게 향상시킬 수 있음을 확인하였다.

참 고 문 헌

[1] O'Brien, F. E., Jr., Guenther, R.D., "Global standardization of IMT-2000", *Emerging Technologies Symposium: Broadband, Wireless Internet Access, 2000 IEEE* ,10-11 April 2000.

[2] R. G. Gallager, "Low density parity check codes", *IRE Trans. Inform. Theory,* vol. IT-8, pp. 21-28, Jan. 1962.

[3] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes", *Electron. Lett.,* vol. 32, pp. 1645-1646, Aug. 1996.

[4] S.-Y. Chung, G. D. Forney Jr., T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045dB of the Shannon limit", *IEEE Commun. Lett.,* vol. 5, pp. 58-60, Feb. 2001.

[5] Tong Zhang, Z. Wang, and K. K. Parhi, "On finite precision implementation of low-density parity-check codes decoder", *in Proc. of 2001 IEEE Int. Symp. on Circuits and Systems (ISCAS),* vol. 4, pp. 202-205, Sydney, Australia, May 2001.

[6] DVB-S2 DRAFT ETSI EN 302 307 V1.1.1 (204-06), Digital Video Broadcasting-Satellite version 2, ETSI, 2004.

[7] T. J. Richardson, and R. Urbanke, "Efficient Encoding of Low-Density Parity-Check Codes", *IEEE Trans. Inform. Theory,* vol. 47, No. 2, pp. 638-656, Feb. 2001.

[8] Li Ping, W. K. Leung, and Nam Phamdo, "Low density parity check codes with semi-random parity check matrix", *IEE Electronics Lett.,* vol. 35, pp. 38-39, Jan. 1999.

[9] Tong Zhang, and Keshab K. Parhi, "A 54 Mbps (3,6)-regular FPGA LDPC decoder", *Signal Processing Systems, 2002. IEEE Workshop (SiPS).* pp. 16-18, San Diego, USA, Oct. 2002.

[10] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices", *IEEE Trans. Inform. Theory,* vol. 45, pp. 399-431, Mar. 1999.

[11] Hao Zhong and Tong Zhang, "Design of VLSI Implementation-Oriented LDPC codes", *IEEE Vehicular Technology Conference,* Orlando, USA, Oct. 2003.

[12] R. Echard, and Shih-Chun Chang, "The pi-rotation low-density parity check codes", *IEEE Global Telecom. Conf.,* vol. 2, pp. 980-984, San Antonio, USA, Nov. 2001.

[13] T. Zhang, and K. K. Parhi, "VLSI implementation-oriented (3,k)-regular low- density parity check codes", *IEEE Workshop, signal processing systems(SiPS),* pp. 25-36, Antwerp, Belgium, Sept. 2001.

[14] J. Campello, and D. S. Modha, "Extended bit-filling and ldpc code design", *IEEE Global Telecom. Conf., 2001,* San Antonio, USA, pp. 985-989, Nov. 2001.

[15] Jinghu Chen, Fossorier, M.P.C., "Near optimum universal belief propagation based decoding of low-desity parity check codes", *Communications, IEEE Transactions on,* Volume 50 Issue 3 Pages 406-414, March 2002.

[16] Hao Zhong, and Tong Zhang, "Block-LDPC: A Practical LDPC Coding System Design Approach", *IEEE Transaction on Circuit and Systems, Regular papers;* vol. 52, No. 4 pp. 766-775, April 2005.

[17] A. J. Felstrom and K. S. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrix," *IEEE Trans. Information Theory,* vol. 45, no. 6, September 1999.

[18] S. Bates and G. Block, "A memory-based architecture for FPGA implementations of low-density parity-check decoders", *in Proceedings of IEEE Symposium on Circuits and Systems (ISCAS), 2005.* pp. 336 - 339 Vol. 1, Kobe, Japan, 23-26 May 2005.

[19] S. Bates and G. Block, "A memory-based architecture for FPGA implementations of low-density parity-check decoders", in *Proceedings of IEEE Symposium on Circuits and Systems (ISCAS), 2005.* pp. 4513-4516 Vol. 5, Kobe, Japan, 23-26 May 2005.

저 자 소 개

이 찬 호(정회원)

1987년 서울대학교 전자공학과 학사졸업.

1989년 서울대학교 대학원 전자공학과 석사졸업.

1994년 University of California, Los Angeles 전자공학과 박사졸업.

1994년 8월~1995년 2월 삼성전자 반도체연구소 선임연구원.

1995년 3월~현재 숭실대학교 정보통신전자공학부 부교수.

<주관심분야 : SoC on-chip-network, 3D 그래픽 프로세서 설계, 채널코덱의 구현, SoC 설계방법론, H.264 codec 구현>

박 재 근(학생회원)

2005년 숭실대학교 전자공학과 학사졸업.

2005년~현재 숭실대학교 전자공학과 석사재학.

<주관심분야 : 채널 코덱의 VLSI 구현 >