논문 2008-45TC-7-1

WiBro 시스템을 위한 고속 LDPC 인코더 설계

(Fast Multi-Rate LDPC Encoder Architecture for WiBro System)

김 정 기*, 발라카난*, 이 문 호**

(Jeong Ki Kim, Balakannan S.P, and Moon Ho Lee)

요 약

Low Density Parity Check codes(LDPC)는 최근 우수한 성능으로 통신 분야에서 채널 코딩의 중요한 블록으로 주목받고 있다. 그리하여 Wibro를 포함한 여러 표준에서 LDPC 부호를 채널 코딩으로 채택하고 있다. 이러한 LDPC 부호의 Encoder를 구현하는데 있어서의 약점은 기존의 이진 Matrix Vector Multiplier가 throughput의 감소의 원인이 되는 clock cycle이 많다는 것이다. 본 논문은 표준에서 사용되는 H 행렬이 Circulant Permutation Matrix(CPM)로 정의되어 있다는 점을 이용하여 인코더의 구현에 있어서 기존의 Matrix Vector Multiplier 대신에 cyclic shift register와 exclusive-OR을 사용하는 설계구조를 제안한다. 또한, 제안한 구조를 이용하여 WiBro에 포함되는 다양한 부호율에 적용가능한 인코더를 설계하였다. 제안된 WiBro LDPC의 인코더는 기존보다 적은 clock cycle을 가지므로 높은 throughput에 도달한다.

Abstract

Low Density Parity Check codes(LDPC) are recently focused on communication systems due to its good performance. The standard of WiBro has also included LDPC codes as a channel coding. The weak point of implementation for LDPC encoder is that conventional binary Matrix Vector Multiplier has many clock cycles which limit throughput. In this paper, we propose semi-parallel architecture by using cyclic shift registers and exclusive-OR without conventional Matrix Vector Multipliers over the standard parity check matrices with Circulant Permutation Matrices(CPM). Furthermore, multi-rate encoder is designed by using proposed architecture. Our encoder with multi-rate for IEEE 802.16e LDPC has lower clock cycles and higher throughput.

Keywords : LDPC encoder, Wibro, clock cycles, semi-parallel architecture, Circulant Permutation Matrices

Ⅰ. 서 론

LDPC 부호의 encoding에 대한 복잡도 문제는 채널 코딩 분야에서 많은 연구주제로 주목 받아왔다. 그리고 그 우수한 성능 때문에 최근 들어 여러 표준들이(WiFi, WiMax, DVB-S2) LDPC 부호를 채택하고 있다. 특히 이러한 표준들에서 채택된 LDPC의 부호화 방식은 저밀도 H 행렬을 이용하여 부호를 생성하는 Richardson과 Urbanke가 제안한 encoding 방법을 사용하고 있다.[2] 표준의 encoding 방식은 Circulant Permutation Matrices(CPM)[1]를 기반으로 확장요소 z-factor에 따라 여러 부호길이로 확장이 가능하도록 정의되어있다. 기본적으로 LDPC 인코더의 하드웨어 구현은 XOR연산을 사용하는 Matrix Vector Multiplyer(MVM), Back-Substitution(BS), Vector Adder(VA)로 구성된다. 본 논문에서는 와이브로의 LDPC가 상당히 고속처리를 요구하기 때문에 이에 적합한 인코더를 설계하기 위해서 표준에서 사용되는 LDPC의 H행렬이 CPM으로 구성되는 특징을 이용하여 효율적인 설계 구조를 갖도록 IEEE 802.16e의 LDPC 인코더를 하드웨어로 구현하며, CPM에 따른 반-병렬구조의 이점을 제안하고 기존의 구현사례[3]와 비교한다.

* 학생회원, ** 정회원, 전북대학교 전자정보공학부
(Chonbuk National University)

※ 이 연구에 참여한 연구자는 2단계 BK21사업의 지원비를 받았음.
접수일자: 2008년1월31일, 수정완료일: 2008년7월21일

Ⅱ. Background

1. 부호화 알고리즘

WiFi, WiMax 등에 표준으로 채택된 효율적 부호 알고리즘은 Richardson과 Urbanke에 의해 제안되었는데, 즉 생성행렬(G matrix)을 사용하지 않고 H matrix의 submatrices을 재배열함으로써 encoding을 하는 방법이다. Richardson과 Urbanke의 H행렬은 다음과 같이 6개의 submatrices로 나누어진다.

그림 1에서 n는 코드워드의 길이를 m는 패리티 비트의 길이이다. Codeword를 $c=(S,p_1,p_2)$로 놓을 때, s는 information bits를 p_1과 p_2는 parity bits를 나타낸다. 이 때 $Hc^T=0$이므로 다음과 같은 방정식이 얻어진다.

$$As^T+Bp_1^T+Tp_2^T=0^T \quad (1)$$

$$(-ET^{-1}A+C)s^T+(-ET^{-1}B+D)p_1^T=0^T \quad (2)$$

여기서 $\phi=-ET^{-1}B+D$로 정의하면, 우리는 다음과 같은 방정식을 얻을 수 있다.

$$p_1^T=-\phi^{-1}(-ET^{-1}A+C)s^T \quad (3)$$

$$p_2^T=-T^{-1}(As^T+Bp_1^T) \quad (4)$$

표준에서 ϕ^{-1}는 행렬의 특성에 따라 $\phi^{-1}=I$이고, T^{-1}는 Back-substitution으로 구현된다.[2~3]

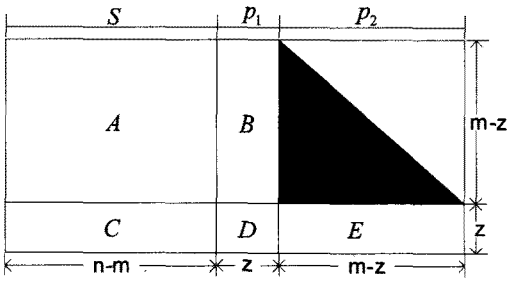


그림 1. 표준에서 사용되는 H 행렬
Fig. 1. The H matrix of standards.

2. 기존의 인코더 설계 구조

기존의 LDPC 인코더의 설계[3]는 각각 binary Matrix Vector Multiplier(MVM)와 Back-substitution(BS), Vector Adder(VA)로 구성된다. 각각 위의 구성요소들은 하나의 2-input exclusive-OR를 사용하여 매 클럭마다 계산을 하도록 되어있다. 기존의 MVM의 동작[3]을 설명하면, 먼저 matrix에 대한 1의 위치의 정보(그림 2에서의 data와 index)와 그 1의 위치가 행렬의 마지막인지의 여부를 표시하는 정보(그림 2에서 end)를 저장

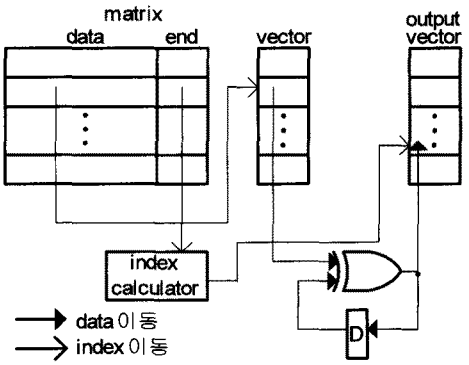


그림 2. 기존의 이진 행렬 벡터 곱셈기
Fig. 2. Conventional binary Matrix Vector Multiplier.

해놓았다가 곱셈을 계산해야하는 벡터를 입력받아 벡터의 비트순서에 따라 end신호가 있을 때까지 XOR연산을 수행하고 end신호가 있으면 출력값을 출력벡터의 index위치에 저장하여 출력벡터를 완성하는 방식이다.

그러나 그림 2와 같은 구조가 간단함에도 불구하고, 기존의 MVM은 클럭마다 행렬의 1의 위치를 갱신하는 것(index)과 행렬의 각 행의 끝의 위치를 저장하는 것이 필요하다. 즉, 기존의 MVM에서 클럭 사이클(clock cycle)은 그 행렬의 1의 개수에 비례하게 된다. 다시 말하면, 기존의 MVM은 throughput 감소의 원인이 되는 많은 클럭 사이클을 가지고 있다. Vector Adder 또한 매 클럭마다 한 개의 XOR을 사용하지만 VA의 클럭 사이클은 벡터의 길이에 비례하므로 VA의 클럭 사이클은 합당하다고 할 수 있다. 또한 만약 인코더의 설계를 위해 적은 클럭 사이클만 요구되는 완전 병렬 구조로 설계하는 경우에는 하드웨어의 크기가 상당히 커지게 된다. 오류정정기술의 특성상 에러없는 통신을 위해서 실제적으로 필요치 않는 리던던시(redundancy)를 추가하는 부분이기 때문에 크기에 대한 제약이 따르기 마련이므로 완전 병렬 구조는 고속 동작하지만 크기에 대한 요구를 충족하기에는 적합하지 않다. 본 논문에서는 적절한 하드웨어의 크기를 지니면서도 높은 처리율을 만족시키기 위해서 MVM과 BS의 사용 대신에 cyclic shift register와 XOR을 갖는 semi-parallel 구조로 이를 구성하여 클럭 사이클을 감소시킨다.

III. 제안한 LDPC 인코더 구조

이번 장에서는 표준의 패리치 체크 행렬(H marices)이 Circulant Permutation Matrices(CPM)로 구성되어 있기 때문에, 기존의 MVM을 cyclic shift registers와

multi-input XOR을 사용하여 LDPC 인코더를 구현할 수 있음을 보인다. 이는 효율적으로 throughput을 향상 시킬 것이다.

1. $ET^{-1}AS^T$의 계산을 위한 구조

그림 1에서, 행렬 T^{-1}과 E는 각각 H 행렬의 부행렬(submatrix)이다. 행렬 T^{-1}과 E는 다음과 같이 나타 낼 수 있다.

$$E=\begin{bmatrix} o & o & L & oL & o & \alpha^o \end{bmatrix}_{z\times(m-z)}$$

$$T^{-1}=\begin{bmatrix} \alpha^o & & & & \\ \alpha^o & \alpha^o & & & \\ M & o & o & & \\ M & o & o & \alpha^o & \\ \alpha^o & L & L & \alpha^o & \alpha^o \end{bmatrix}_{(m-z)\times(m-z)} \quad (5)$$

여기서 α^k는 단위 행렬로부터 오른쪽으로 k번 시프트된 $z\times z$ 순환행렬이다. 그리고 z 는 z-factor로 불리는 확장요소이다.[5] 0 과 α^o는 각각 크기가 $z\times z$ 인 영행렬과 단위행렬이다. 그러면, ET^{-1}는 $ET^{-1}=\begin{bmatrix}\alpha^o & L & \alpha^o & L & \alpha^o\end{bmatrix}_{z\times(m-z)}$이고, AS^T의 결과를 $AS^T=\begin{bmatrix}u_1 & u_2 & L & L & u_{(m-z)/z}\end{bmatrix}^T$라 하면, $ET^{-1}AS^T$는 다음과 같다.

$$\begin{aligned} ET^{-1}AS^T &= \begin{bmatrix}\alpha^o & L & \alpha^o & L & \alpha^o\end{bmatrix}\cdot\begin{bmatrix}u_1 & u_2 & L & L & u_{(m-z)/z}\end{bmatrix}^T \\ &= \begin{bmatrix}\alpha^o u_1^T \oplus \alpha^o u_2^T \oplus L \oplus \alpha^o u_{(m-z)/z}^T\end{bmatrix}_{z\times 1} \\ &= \begin{bmatrix}u_1^T \oplus u_2^T \oplus L \oplus u_{(m-z)/z}^T\end{bmatrix}_{z\times 1} \end{aligned} \quad (6)$$

여기서 T는 전치행렬의 표시이고, u는 크기가 z 인 벡터이다.

또한, $CS^T=[v]_{z\times 1}$이라 정의하면, p_1^T는 $p_1^T=ET^{-1}AS^T+CS^T=\left[u_1^T\oplus u_2^T\oplus L\oplus u_{(m-z)/z}^T\oplus v\right]_{z\times 1}$이다. 이는 $ET^{-1}AS^T$의 연산을 위한 하드웨어가 기존의 MVM대신에, 하나의 multi-input XOR로 대체될 수 있음을 의미한다. 그림 3에서 p_1^T의 연산을 위한 구조를 나타내었다.

2. Bp_1^T의 연산을 위한 구조

이번 절에서는 Bp_1^T의 설계 구조를 보인다. 표준의 H 행렬에서 그 부행렬인 B는 다음과 같이 많은 영행렬을 갖는다.

$$B=\begin{bmatrix}\alpha^{k_1} & O & L & O & \alpha^{k_2} & O & L & O\end{bmatrix}^T_{(m-z)\times z} \quad (7)$$

p_1^T를 $p_1^T=[w]_{z\times 1}$로 정의하면, Bp_1^T는 다음과 같이 표현될 수 있다.

$$Bp_1^T=\begin{bmatrix}\alpha^{k_1}\\ o\\ M\\ o\\ \alpha^{k_2}\\ o\\ M\\ o\end{bmatrix}[w]=\begin{bmatrix}\alpha^{k_1}w\\ ow\\ M\\ ow\\ \alpha^{k_2}w\\ ow\\ M\\ ow\end{bmatrix}=\begin{bmatrix}w^{k_1}\\ o_{z\times 1}\\ M\\ o_{z\times 1}\\ w^{k_2}\\ o_{z\times 1}\\ M\\ o_{z\times 1}\end{bmatrix}_{(m-z)\times 1} \quad (8)$$

여기서 α^k는 $z\times z$ 크기의 단위행렬을 k번 시프트한

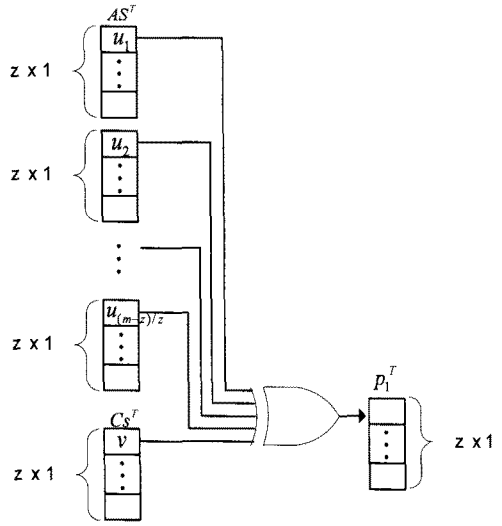


그림 3. p_1^T의 계산을 위한 구조

Fig. 3. Architecture for p_1^T.

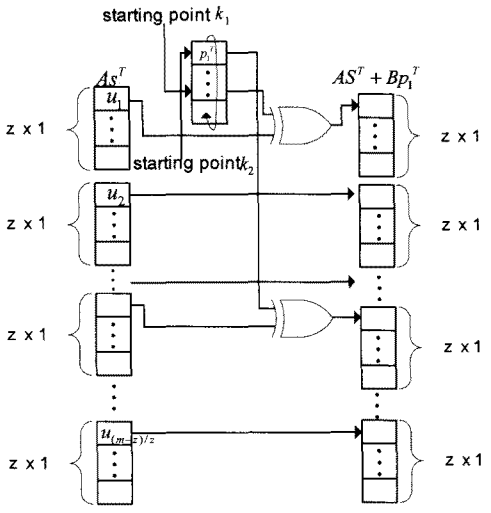


그림 4. Bp_1^T의 계산을 위한 구조

Fig. 4. Architecture for Bp_1^T.

행렬이다. 그러므로 $w^k = a^k w$는 벡터 w로부터 k번 오른쪽 시프트한 벡터이다. 위의 결과에 따라, Bp_1^T의 연산을 위한 하드웨어는 MVM의 사용 대신에 2개의 XOR와 2개의 다른 출발점(staring points)을 가진 cyclic shift register를 이용하여 설계 할 수 있다. 그림 4에서 Bp_1^T의 연산을 위한 제안한 구조를 보인다.

3. AS^T와 CS^T의 연산을 위한 구조

제안한 AS^T와 CS^T의 연산을 위한 하드웨어 구조를 설명하기 위해서 간단한 예제를 하나 보이면, IEEE 802.16e의 부호율 0.5의 행렬 A의 첫 번째 행행렬은 다음과 같다.

$$A_1 = \left[0\ \alpha^{94}\ \alpha^{73}\ 0\ L\ 0\ \alpha^{55}\ \alpha^{83}\ 0\ 0\right]_{Z\times 12Z} \quad (9)$$

여기서 A_1는 행렬 A의 첫 번째 블록 행행렬이다. 행렬 A 또한 많은 영행렬을 가지고 있으므로 $A_1 S^T$는 다음과 같이 계산된다.

$$\begin{aligned} &A_1 S^T \\ &= \left[os_1^T \oplus \alpha^{94} s_2^T \oplus \alpha^{73} s_3^T \oplus L \oplus \alpha^{55} s_9^T \oplus \alpha^{83} s_{10}^T \oplus L \oplus os_{12}^T\right]_{z\times 1} \\ &= \left[s_2^{94} \oplus s_3^{73} \oplus s_9^{55} \oplus s_{10}^{83}\right]_{Z\times 1}^T \end{aligned} \quad (10)$$

여기서 s는 다음과 같은 크기가 $1\times z$인 S의 부행렬(submatrix)이다. 메시지 비트인 S는 다음과 같다.

$$S = \left[s_1\ \ s_2\ \ L\ \ s_{(n-m)/z}\right]_{1\times(n-m)} \quad (11)$$

s^k는 벡터 s로부터 k번 오른쪽 시프트된 벡터로 정의된다. 식 (10)을 일반화하기 위해서 행렬 A와 C를 식 (12)과 같이 $z\times z$의 블록 행렬들의 형태로 나타낼 수 있다.

$$\begin{bmatrix} A \\ C \end{bmatrix} = \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,(n-m)/z} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,(n-m)/z} \\ \vdots & \vdots & \vdots & \vdots \\ A_{(m-z)/z,1} & A_{(m-z)/z,2} & \cdots & A_{(m-z)/z,(n-m)/z} \\ C_{1,1} & C_{1,2} & \cdots & C_{1,(n-m)/z} \end{bmatrix} \quad (12)$$

위의 행렬로부터 AS^T와 CS^T의 연산는 각각 $u_i^T = \sum_{j=1}^{(n-m)/z} A_{i,j} s_j^T$와 $v = \sum_{j=1}^{(n-m)/z} C_{1,j} s_j^T$의 식으로 일반화된다. 여기서 i와 j는 양의 정수 이다. 이 결과를 이용하여 AS^T와 CS^T의 연산은 반-병렬구조로 설계될 수 있다. 이는 인코더의 설계에서 기존의 MVM없이

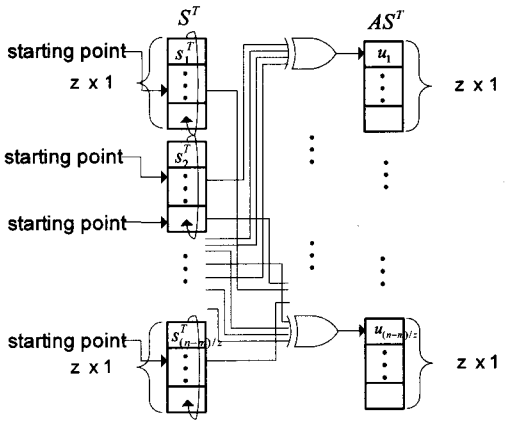


그림 5. AS^T의 계산을 위한 구조
Fig. 5. Architecture for AS^T.

몇 개의 multi-input XOR과 다른 출발점을 갖는 cyclic shift register로 구성될 수 있다. 그림 5에서와 같이 AS^T의 연산을 위한 하드웨어는 z 클럭 동안 업데이트되고 각각 $s_{1\times s}$의 크기마다 동시에 수행된다. 그림 5는 제안한 AS^T의 구조를 보여주고 있다.

참고문헌 [3]에서, 직렬 MVM은 많은 클럭을 소비할 뿐 만 아니라 행렬의 많은 1의 위치를 매번 갱신해야하는 단점이 있었다. 이에 비해 제안한 반-병렬구조가 더 효율적인 설계임은 명백하다. 또한 같은 H행렬에 기반하여 단지 다른 출발점을 갖도록 시프트 레지스터를 컨트롤 할 때, 제안한 구조는 코드워드의 길이에 대응하여 유연하게 적용될 수 있다.

4. Multi-port Memory

AS^T와 CS^T의 계산을 위한 구조와 같이 앞 절에서 제안한 반-병렬구조를 사용하기 위해서는 기존의 한두 개의 입출력 port를 갖는 메모리가 아닌 하나의 입력을 받아 저장하고 각기 다른 출발점을 지닌 시프트된 여러 개의 출력들이 필요하다. 때문에 효율적인 cyclic shift register를 구현하기 위해서 본 논문은 그림 6에서와 같

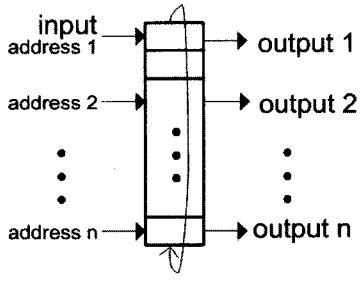


그림 6. Multi-port Memory
Fig. 6. Architecture for Multi-port Memory.

은 Mulit-port Memory(MPM)을 구성하였다. MPM은 하나의 입력을 저장하고 각기 다른 주소값들에 따라 서로 달리 시프트된 데이터들을 출력한다.

IV. LDPC 인코더의 구현

1. IEEE 802.16e LPDC 인코더

이번 장에서 와이브로 LDPC 인코더의 구현 결과를 보이고 기존의 것과 비교한다. 참고문헌 [3]의 인코더 구조가 각각 4개의 stage로 구성된 반면, 그림 7과 같이 제안한 인코더는 각각 3개의 stage로 구성되어 있다. 이는 제안한 인코더의 구조가 기존보다 클럭 사이클이 더 적은 것에 기인한 것이다. stage 1에서 인코딩될 입력 메시지 S는 z 클럭만큼 지연된다. 그 이유는 stage 2의 패리티 체크 비트의 계산을 위한 클럭 사이클이 z 클럭이기 때문이다. 또한 stage 1에서는 반-병렬 구조를 적용하기 위해서 입력 메시지를 버퍼에 저장한다. 그러므로 stage 1과 stage 2가 수행될 동안, z 클럭만큼 지연된 입력 메시지는 곧 바로 출력으로 나간다. stage 2에서는 각각 AS^T와 CS^T, $ET^{-1}AS^T+CS^T$를 제안한 병렬 구조를 이용하여 z 클럭 동안 계산한다. 그리고 stage 2에서의 연산이 끝나자마자 stage 3에서는 $T^{-1}\left(AS^T+Bp_1^T\right)$를 계산하면서 동시에 패리티 비트 p_1과 p_2를 출력시킨다.

제안된 인코더의 처리율을 계산하기 위해서는 각 stage의 클럭 사이클을 알아야 한다. 제안된 구조의 stage 1에서 클럭 사이클은 메시지의 길이이다. stage 2의 클럭 사이클은 반-병렬 구조에서 z의 크기 만큼이다. 여기서 z는 확장요소이다. stage 3의 클럭 사이클은 패리티 비트 p_1 과 p_2의 길이이다. 즉 stage 3의 클럭 사이클은 $z+(m-z)$이다. 제안된 인코더의 모든 클럭 사이클은 $e(X)$에 비의존적이기 때문에 클럭 사이클을 감소시킬 수 있고 처리율을 향상시킬 수 있다. $e(X)$는 임의의 이진 행렬 X가 포함하고 있는 1의 개수이다.

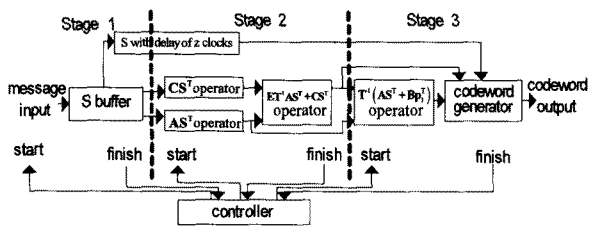


그림 7. 제안된 LDPC 인코더
Fig. 7. Proposed architecture of LDPC encoder.

표 1. Clock cycle 비교
Table 1. Comparison of clock cycles.

Reference [3]		proposed architecture	
stage	cycles	stage	cycles
1	$n-m$	1	$n-m$
2	$\max(e(A), e(C))$		
3	$e(T)+e(E)+(n-m)$ $+e(\phi)+e(B)+(m-z)$	2	z
4	$e(T)+2((n-m)+z+(m-z)$	3	$z+(m-z)$

표 2. IEEE 802.16e LDPC의 부호길이 1536인 경우, 다양한 부호율에 따른 Clock cycle의 비교
Table 2. Results of clock cycles with code length 1536 and different coderate.

name	stage	Coderate					
		1/2	2/3A	2/3B	3/4A	3/4B	5/6
reference [3]	1	768	1024	1024	1152	1152	1280
	2	2176	3392	3648	3840	3968	4544
	3	2432	2176	2176	2048	2048	1920
	4	3776	3520	3520	3392	3392	3264
proposed	1	768	1024	1024	1152	1152	1280
	2	64	64	64	64	64	64
	3	768	512	512	384	384	256

표 1에서 제안된 구조의 클럭 사이클과 기존의 것[3]을 비교하였다. 표 2는 표 1을 바탕으로 IEEE 802.16e LDPC 인코더의 각 stage마다 clock cycle을 계산하여 비교한 것이다. 표 1과 표 2에서와 같이 병렬 구조의 클럭 사이클은 기존의 직렬 구조의 것보다 더 적음을 확인 할 수 있다.

참고문헌 [3]으로부터 제안된 인코더의 성능(codeword thoughput)은 다음과 같은 식으로 유도할 수 있다.

$$\text{codeword throughput} = \frac{\text{blocklength} \times f_{\max}}{\max(\text{cycles in all stages})} \tag{13}$$

여기서 $f_{\max}$는 클럭 스피드의 최대치이고, block length는 출력될 코드워드의 길이이다. 그리고 식 (11)의 분모는 각 stage의 클럭 사이클 중에 가장 긴 시간이 요구되는 stage의 클럭 사이클을 선택하는 것으로 정의한다. 예를 들어 z-factor가 64이고, 부호율이 1/2인 부호길이 1536의 제안된 LDPC 인코더는 각 stage 별로 다음과 같은 클럭 사이클을 갖는다.

$$c(\text{stage1}) = 768$$

$$c(\text{stage2}) = 64$$

$$c(\text{stage3}) = 768$$

여기서 c(stage)는 괄호 안의 stage의 클럭 사이클이다. 제안된 구조의 검증을 위해서 Xilinx vertex-2 xc2v4000-6ff1152 device를 synthesis를 위해 사용하였고, 최대 주파수 208Mhz를 얻었다. 그러므로 식 (13)에 따라 코드워드 처리율은 416Mbps이다. 표 3에서 보여진 결과와 같이 이는 기존의 구현 사례[3]보다 상당히 빠른 것이다. 기존의 구현 사례와의 정확한 비교를 위해서, 구현 사례[3]과 같은 device를 사용하여 synthesis를 수행하였다. 구현 사례[3]은 Xilinx vertex-2 xc2v4000-6ff1152를 사용했었다. 표 3에서 각기 다른 코드워드 길이에 따른 결과를 비교하고 있다.

표 3에서 제안된 IEEE 802.16e의 인코더는 2개의 합성 결과를 보여주고 있다. 첫 번째 결과는 cyclic shift register를 Xilinx IP core generator의 dual port rams으로 합성한 것이고, 이에 비해서 두 번째 결과는 chip에서 Block Rams을 줄이기 위해서 cyclic shift register를 앞에서 제안했던 Multi Port Memory(MPM)를 구성하여 합성한 결과이다. 그리고 표 3에서 edge는 행렬의 1의 개수를 의미한다. 제안한 인코더 구조는 비슷한 edge수를 가지는 기존의 구현 사례와 비교하여 slice는 40~60%, block rams는 70% 정도의 감소를 보이면서도 처리율은 향상된 것을 확인할 수 있다.

표 3. 기존 구현사례와 여러 부호길이에 따른 비교
Table 3. Comparison of different codeword length on Xilinx vertex-2 xc2v4000-6ff1152 for coderate 0.5.

Name	target	codeword length	edges	z	slices	block rams	Speed (MHz)	codeword throughput (Mbps)
reference [3]	general case	500	2418	2	562	12	161	100
proposed	IEEE 802.11n	648	2376	27	290	45	180	360
	IEEE 802.16e	768	2432	32	172	38	240	480
		768	2432	32	226	9	231	462
reference [3]	general case	1000	4859	2	682	13	152	96
proposed	IEEE 802.11n	1296	4752	54	352	45	168	336
	IEEE 802.16e	1536	4864	64	192	38	209	418
		1536	4864	64	399	9	208	416

표 4. 여러 instance에 따른 합성결과 및 성능
Table 4. Synthesis and performance results of different instances on one Xilinx vertex-4 xc4clx60-12ff668 device with codeword length 1536 bits and coderate 0.5 for IEEE 802.16e.

instance	slices	slice F/F	4 input LUTs	block rams	Speed (Mhz)	codeword throughput
1	541	219	963	9	250	500Mbps
3	1587	615	2806	27	247	1.48Gbps

또한, 위의 결과를 바탕으로 인코더의 병렬처리 혹은 파이프라인 구조를(instance) 이용하여 좀 더 빠른 처리율을 만족시킬 수 있다. 표 4에서, 제안된 인코더는 단지 3개의 instance를 사용하여 1.48Gbps에 도달하였다. 이는 부호율 1/2, 부호길이 1536일때 단지 한 개의 Xilinx vertex-4 xc4clx60-12ff668 device를 사용한 결과이다. 기존의 구현사례[3]가 16개의 instance와 부호길이 2000, 부호율 1/2일때 1.2Gbps에 도달한 것과 비교하여 제안한 구조가 더 효율적이라는 것을 쉽게 알 수 있다. 표 4은 여러 instance에 따른 synthesis 및 성능을 보여주고 있다.

2. Multi-Rate의 적용

지금까지, 제안한 반-병렬 구조의 LDPC 인코더가 고속 처리율을 가진다는 것을 보였다. 이러한 구조에 표준에서 정의된 여러 부호율을 지원하기 위해서 stage 2를 수행하기에 앞서, 각기 다른 부호율에 대응하는 A, C, B의 행렬에 속해있는 시프트의 출발점(starting points for cyclic shift)들을 ROM에 저장한 뒤 부호율에 관한 정보를 입력으로 받아 각기 다른 H 행렬로 입력 메시지를 부호화하도록 하였다. 제안한 구조는 단지 행렬의 시프트 출발점만을 ROM에 저장하면 되기 때문에 만약 기존의 구조에서 Multi-rate를 구성하려 할 때, 많은 행렬에 대한 정보를 저장했어야 하는 것보다 효율적으로 저장공간을 감소시킬 수 있다. 지원가능한 부호율은 IEEE 802.16e의 모든 부호율인 각각 1/2, 2/3A, 2/3B, 3/4A, 3/4B, 5/6이다. 그림 8은 다양한 부호율을 적용한 제안된 인코더 구조이다. device는 Xilinx vertex-4 xc4clx60-12ff668을 사용하였고, tool은 Xilinx ISE 9.1i를 사용하였으며 언어는 verilog-HDL로 기술하

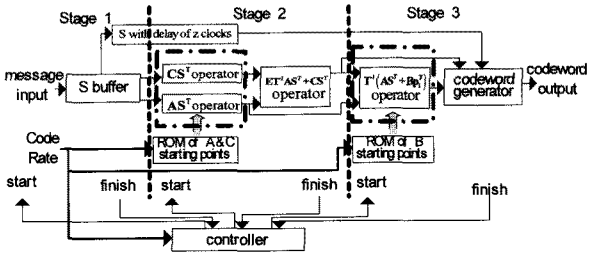


그림 8. 제안된 Multi-Rate 인코더
Fig. 8. Proposed architecture of Multi-Rate Encoder.

표 5. Multi-rate 인코더의 synthesis 결과
Table 5. The results of synthesis for Mult-rate Encoder.

target	codeword length	slices	slice F/F	4 input LUTs	block rams	Speed (Mhz)
IEEE 802.16e	768	980	627	1613	7	234
	1536	1297	738	2435	7	219

표 6. 부호율에 따른 성능 비교
Table 6. Codeword throughput for various coderate

단위 : Mbps

name	target	codeword length	CodeRate 1/2	2/3	3/4	5/6
reference [3]	general case	2000	88	49.5	-	-
proposed	IEEE 802.16e	768	468	351	312	280
		1536	438	328	292	262

였다.

표 6의 비교대상[3]은 Multi-rate 인코더가 아니고 각기 다른 부호율에 맞도록 구현된 사례이다. 제안된 LDPC 인코더는 다양한 부호율의 적용함에도 불구하고 multi-rate 구조가 아닌 인코더와 비교해보아도 여전히 codeword throughput의 손실이 크게 없음을 알 수 있다. 뿐 만 아니라, ASIC으로 구성하는 경우 좀 더 빠른 throughput을 만족시킬 수 있을 것이다.

V. 결 론

본 논문에서는 IEEE 802.16e를 위한 LDPC 인코더를 구현하고 기존의 MVM대신에 cyclic shift register와 XOR을 이용하여 새로운 반-병렬 구조를 제안하였다. 그리고 그 throughput은 기존의 구현사례와 비교해 볼 때 상당히 빠른 성능을 보였다. 또한 여러 표준의 H 행렬이 CPM으로 구성되어 있기 때문에 제안된 LDPC 설계 방법은 와이브로 LDPC 인코더 뿐 만 아니라 여러 표준에 있어서도 사용할 수 있다.

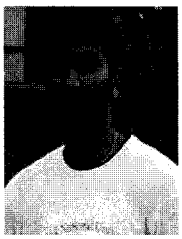
참 고 문 헌

[1] Marc. P.C. Fossorier, "Quasi-cyclic low density parity check codes", in Proc 2003 IEEE Int. Symp. Information Theory (ISIT 2003), p.150, Yokohama, Japan, Jun/Jul/2003.
[2] T.J. Richardson and R.L. Urbanke, "Efficient Encoding of Low Density Parity Check Codes" *IEEE Trans. IT*, vol.47, pp.638-856, Feb.2001.
[3] L.Dong-U, L.Wayne, W.Connie, J.Christopher, "A flexible hardware encoder for low density parity check codes", Proc. IEEE Symp. Field-Programmable Custom Computing, Napa. CA, USA, 2004.
[4] Zahid Khan, Tughrul Arslan, "Pipelined Implementation of a Real Time Programmable Encoder for Low Density Parity Check Code on a Reconfigurable Instruction Cell Architecture", Design, Automation & Test in Europe, Nice, France, 2007.
[5] IEEEP802.16/D12, Oct. 2005.
[6] IEEE Std 802.22-05/005r7 Sept. 2005.
[7] Jia-ning Su, Zhi Liu, Ke Lie, Bo Shen, Hao Min, "An efficient low complexity LDPC encoder based on LU factorization with pivoting", ASICON 2005. 6th International conference, vol.1, p107-110, Oct. 2005.
[8] T. Brack, M. Alles, F. Kienle, N. Wehn. 17th International Symposium on Personal, Indoor and Mobile Communications, Helsinki, Finland, September 2006.

저 자 소 개

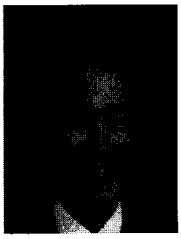
김 정 기(학생회원)

2007년 전북대학교 전자정보 공학부 학사 졸업
2008년 전북대학교 전자정보 공학부 석사과정
<주관심분야 : 채널코딩, LDPC, 통신, 반도체>



발라카난(학생회원)

2003년 인도 Bharathiar University, Coimbatore 석사 졸업
2006년 인도 Junior Project Assistant in IIT, Kharagpur
2008년 전북대학교 정보통신공학과 박사과정
<주관심분야 : 이동통신, 네크워크, 암호이론>



이 문 호(정회원)

1967년 전북대학교 전자공학과 학사
1984년 전남대학교 전기공학과 박사
1990년 동경대학교 정보통신 공학과 박사
1980년 10월~현재 전북대학교 전자정보공학부 교수
<주관심분야 : 이동통신, 정보이론, 암호이론>