부분병렬 알고리즘 기반의 LDPC 부호 구현 방안

정 지 원*

Design Methodology of LDPC Codes based on Partial Parallel Algorithm

Ji-Won Jung*

요 약

본 논문에서는 DVB-S2 표준안에서 권고되고 있는 irregular LDPC 부호의 다양한 부호화율에서 부호화 방식 및 복호화 방식에 대해 살펴보고 이에 대한 성능분석을 하였다. 또한 이의 구현에 있어서 효율적인 메모리 할당 및 이에 따른 구현 방법에 대해 연구하였다. LDPC 복호기를 구현하는 방안에는 직렬, 부분병렬, 완전병렬 방식이 있으며, 부분병렬방식이 하드웨어 복잡도와 복호속도를 절충하는 방안이다. 따라서 본 논문에서는 부분병렬 구조를 기반으로 하는 LDPC 복호기의 메모리 설계에서 효율적인 체크노드, 비트노드, LLR 메모리의 구조를 제안하고저 한다.

ABSTRACT

This paper makes an analysis of the encoding structure and the decoding algorithm proposed by the DVB-S2 specification. The methods of implementing the LDPC decoder are fully serial decoder, the partially parallel decoder and the fully parallel decoder. The partial parallel scheme is the efficient selection to achieve appropriate trade-offs between hardware complexity and decoding speed. Therefore, this paper proposed an efficient memory structure for check node update block, bit node update block, and LLR memory.

Keywords : LDPC, Partial parallel, DVB-S2, Check node, Bit node

Ⅰ. 서 론

무선통신 시스템은 무선채널의 특성으로 비트 오류가 발생하기 쉬우며 이를 정정하기 위해 사용되는 채널 부호는 무선통신 시스템에서 매우 중요한 기술요소이다. 위성통신 및 이동통신 등에서 사용되는 채널부호는 일반적으로 연판정이 가능한 길쌈부호와 연집 오류 특성에 강한 RS(Reed Solomon)부호를 결합한 연접부호(Concatenate Code)를 사용한다. 연접부호를 이용한 오류제어방식 또한 Shannon's limit에 다소 큰 격차를 보이고 있다. Shannon's limit에 근접하기 위한 최근의 부호화 방식으로는 1993년 Berrou등에 의해 제안되고 Eb/No 0.7dB, 부호율 1/2에서 비트오류확률

* 한국해양대학교 전파공학과 교수 (jwjung@hhu.ac.kr)
접수일자 : 2011년 11월 04일, 수정일자 : 2011년 11월 22일, 심사완료일자 : 2011년 12월 05일

10-5의 성능을 보이는 터보 부호, 1962년 Gallager에 의해 제안되어 1996년 Mackay에 의해 재정립된 LDPC (Low Density Parity Check) 부호등이 있다[1]. LDPC 부호는 터보 부호에 비해 정정되지 않는 오류들을 대부분 검출할 수 있고, 복호화의 복잡도가 낮을 뿐 아니라 좋은 거리 특성으로 오류마루 현상이 나타나지 않고, 완전 병렬 처리로 고속 처리가 가능한 장점이 있다. 반면에 부호화 부분의 높은 복잡도가 LDPC코드의 단점이었으나 최근에 삼각행렬 분해법, Linear-Congruence 방법 등 부호화 방법이 제안되고 있다.[2] 샤논의 채널 용량 한계에 근접한 LDPC 코드방식이 DVB-S2표준안 방식으로 채택되면서 복호기의 고속화 및 구현 방안에 대해 관심의 대상이 되어지고 있다.[3] DVB-S2에서는 HNS(Huge Network System)사에서 제안한 알고리즘을 표준안으로 채택하고 있으며, 큰 블록 사이즈(16200, 64800) 및 많은 반복 횟수를 요구하며, 대역폭 효율이 0.5bits/symbol에서 4.5bits/symbol을 지원하고 있으며, 10개의 서로 다른 부호율(1/4, 1/3, 1/2, 3/5, 2/3, 3/4 4/5, 8/9,9/10)과 네개의 다른 변조방식(QPSK, 8-PSK,16-APSK,32-APSK)를 지원한다. 따라서 본 논문에서는 HNS사에서 제시한 LDPC 각 부호화율에서 부호기 구성 방법 및 HNS사에서 제시한 복호 알고리즘을 분석하였으며, 분석을 토대로 성능분석을 하였다. 또한 이의 구현에 있어서 효율적인 메모리 할당 및 이에 따른 구현 방법에 대해 연구하였다. LDPC 복호기를 구현하는 방안에는 직렬, 부분병렬, 완전병렬 방식이 있으며, 부분병렬방식이 하드웨어 복잡도와 복호속도를 절충하는 방안이다. 따라서 본 논문에서는 부분병렬 구조를 기반으로 하는 LDPC 복호기의 메모리 설계에서 효율적인 체크노드, 비트노드, LLR 메모리의 구조를 제안하고저 한다.

II. DVB-S2기반 LDPC 부호화 알고리즘

LDPC 코드의 parity check matrix는 sparse할지라도 일반적인 generator matrix는 encoding을 위하여 필요하다. 물론, 일반적인 linear code는 parity check matrix를 알고 있기에 generator matrix는 Gaussian elimination method를 사용하여 간단하게 유도될 수 있다. 그러나 그 결과 generator matrix는 더 이상 sparse하지 않다. 이 것은 저장용량과 encoding complexity 문제를 야기한다. 따라서 여기서는 parity check matrix를 다음의 형태로 적용시킨다. $H_{(N-K)\times N} = \lfloor A_{(N-K)\times K} B_{(N-K)\times(N-K)} \rfloor$. 여기서 N은 부호어 길이를 의미하며, K는 정보원 길이를 의미한다. B는 다음 그림 1의 형태이며 이는 Low Triangular parity check Matrix를 나타낸다.

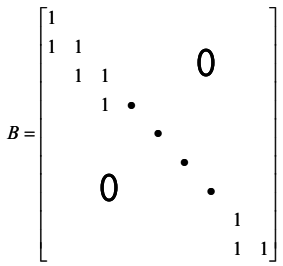


그림 1. 패리티 메트릭의 부 메트릭
Fig. 1 Sub matrix of Parity Check Matrix

Matrix A는 cycle-4를 피하고, cycle-6을 최소화 시키면서 sparse 하게 Matrix을 구성하며, 각각의 초기치도 random하게 분포시킨다. Matrix A는 cycle-4를 피하고, cycle-6을 최소화 시키면서 sparse 하게 Matrix을 구성하며, 각각의 초기치도 random하게 부포시킨다. A matrix의 초기치는 참고문헌 [3]에 주어졌다. LDPC부호기는 코드워드 크기 에 정보 블록 크기를 부호화한다. 여기서 N은 부호화어 크기이며, K는 정보원의 크기이다. M은 잉여비트의 크기이며 N-K개를 가진다. 부호화하는 절차는 아래와 같다.

1) 단계 1. 모든 parity bit를 초기화 시킨다.

$$p_0 = p_1 = p_2 = \ldots = p_{n_{ldpc}-k_{ldpc}-1} = 0$$

2) 단계 2. 모든 패리티 비트 주소를 더한다.

$$p_n = p_n + p_{n-1}(n = 1,2,\ldots)$$

$$p_{M\_Address} = p_{M\_Address} \oplus i_k (k = 1,2,\ldots\ldots K)$$

3) 단계 3. B submatrix인 Low Triangular parity check Matrix를 구하기 위해 마지막으로 다음과 같은 공식을 적용시킨다.

$$p_n = p_n + p_{n-1}(n = 1,2,...M-1)$$

단계 2에서 M_Address는 각 M개의 각 정보비트별 "1"의 위치를 저장해 놓은 어드레스를 나타내며, 이는 참고문헌 [1]에 나타내었다.

III. DVB-S2 기반 LDPC 복호 알고리즘

DVB-S2에 제시한 복호 알고리즘은 수신비트에다가 채널 추정 값을 구하는 초기화 과정, Check node 확률을 구하는 CNU(check Node Update), 비트 확률을 구하는 BNU(Bit Node Update)로 세 가지 단계로 나눌 수 있다.

1) 단계 1 : Initialization

$$u_n = -L_c \bullet r_n(L_c = \frac{2}{\sigma^2}) \quad (1)$$
$$n = 0,1,...,N-1, i = 1,2,...,\deg(bit\,node\,n)$$

rn은 n시점에서의 수신 심볼을 의미하며, Lc는 채널 추정치이다.

2) 단계 2 : Check Node Update(CNU)

다음 그림 2(b)는 Check node 확률을 구하는 CNU에서 하나의 체크 노드에서 올 수 있는 비트들의 확률을 구하는 그림이다. d_c개의 row weight를 가진다고 가정할 때, d_c개의 비트노드에서 k번째 체크 노드로 들어오는 비트들의 확률은 아래 식과 같다.

$$w_{k\to n1} = g(v_{n1\to k}, v_{n2\to k}, ..., v_{n_{i-1}\to k}, v_{n_{i+1}\to k}, ..., v_{n_{dc}\to k})$$

$$\begin{aligned} g(a,b) &= sign(a)\times sign(b)\times\{\min(|a|,|b|)\} + LUT_g(a,b) \\ &= sign(a)\times sign(b)\times\{\min(|a|,|b|) - LUT(|a-b|)\} \end{aligned}$$
$$LUT(|a-b|) = \begin{cases} 2 & |a-b| \le 2 \\ 1 & 3 \le |a-b| \le 6 \\ 0 & |a-b| \ge 7 \end{cases}$$

(2)

여기서

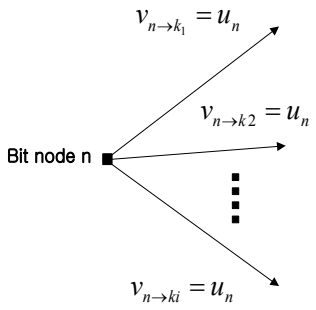
$LUT_g(a,b) = \log(1+e^{-|a+b|}) - \log(1+e^{-|a-b|})$

를 나타낸다.

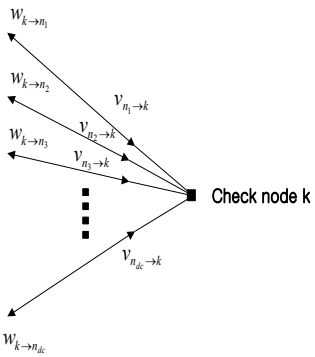
3) 단계 3 : Bit Node Update(BNU)

CNU에서 각 체크 노드로 d_v개의 연결된 비트들에 대한 업데이트가 이루어진 후, 각 컬럼에 해당하는 비트 노드의 확률을 구한다.

$$v_{n\to k_i} = u_n + \sum_{j\neq i} w_{k_j\to n} \quad (3)$$



(a)초기과정



(b)CNU과정

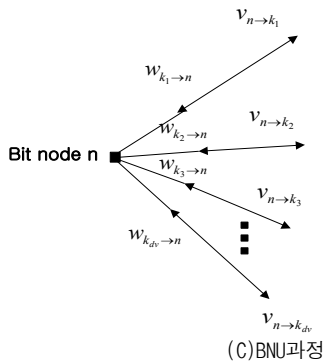


그림 2. LDPC 복호과정
Fig. 2 LDPC Decoding Process

그림 3은 parity accumulator address 기법의 부호기 구성법을 이용하여 N = 64800일 일 때 iteration을 40회 했을 때의 각 부호화율에서 Es/No에 따른 성능 곡선을 나타내었다.

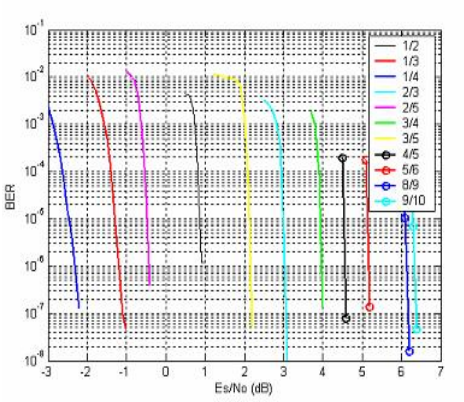


그림 3. 부호율에 따른 LDPC 부호의 성능 (반복횟수=40, N=64800)
Fig. 3 Performance of LDPC for Various Coding Rates(Iteration = 40, N=64800)

Ⅳ. 구현 방안

본 장에서는 R=1/2 부호화율에서 구현 방안을 제시하고자 한다. 구현 방안은 구현 보드가 FPGA 혹은 ASIC으로 하느냐에 따라 크게 나뉠 수 있으며, 구현 방법 상으로는 완전 직렬 구조, 그리고 부분 병렬 구조, 완전 병렬 구조로 나뉠 수 있다. FPGA 상에서 구현 하는 경우는 구현하는 칩의 종류에 따라, 즉 메모리 환경에 따라 메모리 엑세스 등이 바뀔 수 있다. 완전 직렬로 구현할 경우에는 속도에 문제가 있으며, 완전 병렬일 경우에는 칩의 용량을 초과하는 문제가 있으므로 부분 병렬 방식이 효과적인 방식이라 생각 할 수 있다.

4.1 부분병렬 기반의 부호기 구현 방안

Parity accumulator address 부호화 방법에서 마지막 단계는 $p_n = p_n \oplus p_{n-1} (n = 1,2,..)$ 로 계산된다. 기존의 방법은 이를 직렬로 계산을 수행하며, 부호율 1/2 일 경우 parity bit가 32400개이므로 32399 클럭이 필요하게 된다. 본 논문에서는 그림 4와 같이 parity 부분의 부호화 단계에서 부분 병렬 이용한 방법을 제안하는 바이다.예를 들어 부호율 1/2 일 경우 90개를 부분병렬로 부호화를 하게 되면 다음과 같다. 그림과 같이 Parity bit를 $(p_0,....p_{359}),(p_{360},...,p_{719}),...$ 로 90개의 블록으로 나눈다.

하나의 블록은 360개의 parity bit가 직렬로 $p_n = p_n \oplus p_{n-1}$ 같은 연산을 수행한다.

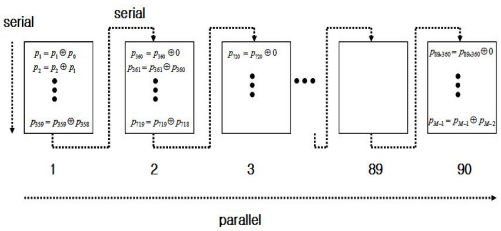


그림 4. 부분병렬방식의 부호화기 구현 방안
Fig. 4 Implementation of Encoder with a Partial Parallel Algorithm

이때 첫 번째 블록의 p_1은 p_0의 값을 알고 있으므로 계산이 가능하지만, 첫 번째 블록을 제외한 각 블록의 첫 번째 parity bit는 그 이전의 parity bit의 값을 알 수 없으므로 0으로 초기치를 정해 놓고 계산을 한다. 각 블록마다 계산이 다 수행된 후에 첫 번째 블록의 마지막 parity bit를 이용하여 그 다음 블록에 있는 360개의 값을 계

산 하게 된다. 마지막 parity bit가 0일 경우 두 번째 블록의 360개의 값은 그대로 유지되고 1일 경우 360개의 값은 반전을 시켜준다. 같은 방법으로 두 번째 블록의 마지막 값을 이용하여 세 번째 블록의 값을 유지 또는 반전 시켜 준다. 이와 같이 순차적으로 다음 블록의 값을 계산 하면 기존의 Parity accumulator address 부호화 방법에서 마지막 단계를 직렬로 계산 할 때 M-1개(1/2인 경우32399개) 클럭 이 필요 했다면 부분병렬을 이용한 기법은 450 클럭(= 360 클럭 + q(90) 클럭)이면 부호화 과정을 수행할 수 있게 된다. 수행 시간을 일반화 시키면 표 1과 같다.

표 1. 두 알고리즘에 따른 요구되는 클럭수 비교
Fig.1 Comparison of Required Clocks for two Algorithms

Algorithms	Required clocks
Conventional Algorithm	M-1 clocks
Partial Parallel Algorithm	(q + 360) clocks

4.2 복호기 메모리 구조

그림 5는 q그룹으로 분할된 비트 노드와 체크 노드에 대한 Tanner 그래프이다.

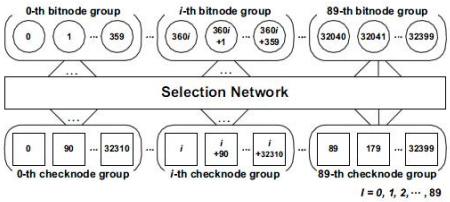


그림 5 q그룹으로 분할된 비트 노드와 체크 노드에 대한 Tanner 그래프
Fig. 5 Tanner graph of bit nodes and check nodes by group q

DVB-S2 규격의 LDPC 부호의 H 매트릭스 특성 상 360개의 노드를 하나의 그룹으로 묶어서 분할 한다. 위 그림은 부호화 율 1/2에서의 Tanner 그래프이고 비트 노드는 데이터의 순서대로 360개씩 그룹을 만들고, 체크 노드의 경우 q 만큼 떨어진 노드들로 360개씩 그룹을 만들 수 있다. 아래 그림 6은 복호기 FPGA 구현을 위한 전체 블록도 이다.

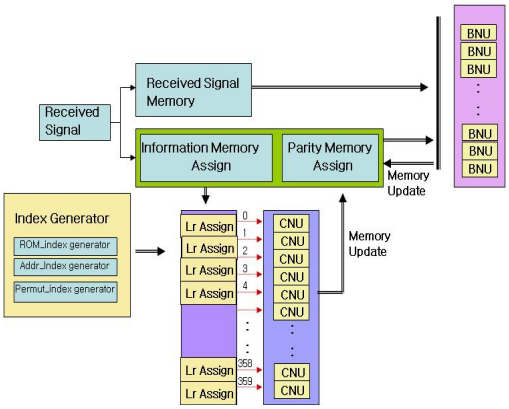


그림 6. FPGA 구현을 복호기 블록도
Fig. 6 Decoder Blockdiagram for FPGA Implementation

복호기 구조는 크게 수신신호와 채널 추정치를 곱하여 메모리에 할당하는 블록, CNU블록, 그리고 BNU블록으로 구성되어진다. 그림 6에서 수신된 심볼은 정보메모리(Information Memory Assign)와 패리티 메모리(Parity Memory Assign)로 각각 저장되며, 각각의 메모리 출력은 360개 단위로 CNU 계산을 하기 위해 dc개의 index들을 이용하여 할당한다(Lr Assign블록). CNU를 계산하기 위해서는 Address_Index, Rom_Index 그리고 Permut_Index가 필요하다 Rom_Index는 CNU를 계산하기 위해 같은 컬럼 어드레스가 어느 메모리에 있는지를 알려주며, 지시된 Rom_Index에서 . Address_Index는 90개의 블록 중 몇 번째 블록에 있는지를, Permut_Index는 그 블록에서 몇 번째부터 읽어야 하는지를 지시한다. 360개의 CNU를 동시에 계산하기 위해서 Lr Assign 블록에서 각 index 를 이용하여 저장한다. LLR 메모리 블럭 구조는 아래 그림7과 같다. 이는 HNS사에서 제시한 컬럼 별 주소와 동일하게 배치하였다[3]. LLR의 메모리 구조는 DVB-S2 표준에서 제시한 Column address register 구조와 동일하게 배열하며, 값 또한 순서대로 저장한다.

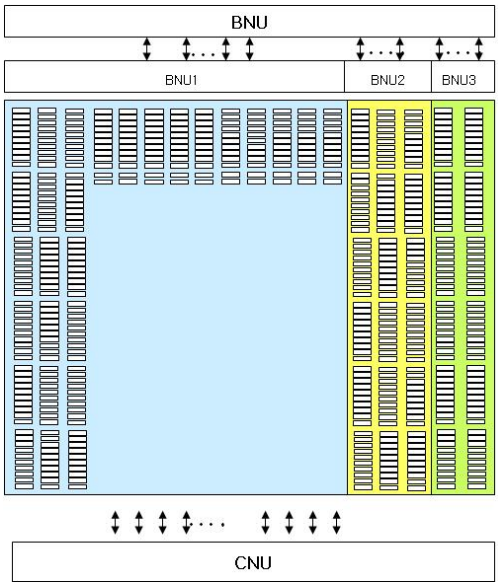


그림 7 메모리구조
Fig. 7 Memory Structure

위의 그림에서 세로의 개수는 총 0부터 17까지 18개의 메모리 bank로 구성된다. 하나의 블럭에는 360 개의 데이타로 되어 있으며, 총 60개의 블럭이 각각 메모리에 저장된다. 컬럼 주소는 0 부터 59가 두번 반복되어 있기 때문에, 첫번째 부분은 ADDRESS 번호 0 부터 12 까지 메모리에 할당하였고 두번째 부분은 13부터 15에 할당하였다. Parity 부분은 ADDRESS 번호 16부터 17에 할당하였다.

이의 구조는 HNS에서 제안한 Column Address 구조와 동일하며, 가로 블럭들은 서로 다른 메모리에 저장함을 원칙으로 한다. 초기값의 저장은 수신데이타를 360개씩 병렬로 각 메모리에 저장하면 된다. 세로 블럭들은 같은 메모리에 저장하며, 메모리의 개수는 maximum Column weight 13과 두번째 반복하는 column weight 수 3, 그리고 parity 부분의 column weight 수 2 동일한 총 18개의 bank를 사용한다. 하나의 메모리에 사용되는 총 데이터 개수는 21600(60*360) 개의 6비트로 구성되어 있다. CNU(Check Node Updat)의 계산 과정은 Parity 부분을 포함하는 메모리 18개를 부록에 제시한 Rom_index, Address_index, Permute index를 기초로 각 메모리에 있는 데이타를 병렬로 가져와서 계산한다. BNU의 계산 과정은 세부분이 동시에 병렬적으로 처리된다. 첫번째 분은 정보 부분에 관한 것이며, 이는 정보 비트 0 에서 21599 번째까지의 BNU 계산 과정이며, 메모리의 0 번부터 12번까지 해당되며, 두번째는 정보비트 21600 부터 43199까지이며, 이는 메모리 13번 부터 15번까지 해당된다. 마지막으로 패리티 부분은 메모리 16번 17번에 해당된다. 각각의 BNU는 서로 상이한 메모리에 저장되기 때문에 병렬로 가능하다.위의 그림에서 BNU1은 정보비트 0 부터 21599 까지의 BNU 계산블럭이며, BNU2는 정보비트 21600 부터 43199까지이며, 이는 메모리 13번 부터 15번까지 해당된다. 마지막으로 메모리 16번 17번에 해당되는 패리티 부분은 BNU3에서 계산한다 . 각각의 BNU는 서로 상이한 메모리에 저장되기 때문에 병렬로 가능하다.

그림 7의 하나의 블록을 확대하여 그림 8을 예를 들어 설명하면, column address register는 54 부터 90씩 증가하여 360번째인 32094 까지 첫번째 블럭에 저장된다. 여기서 메모리 구조에서 54는 Address_Index 0번에 위치하며, H 메트릭의 가로 부분에서 0 번째에 위치하여 Rom_Index는 0 그리고 360개 중 첫 번째 위치하기 때문에 Permut_index는 0이 할당된다. 그리고 column 측면에서 54는 dc개 만큼 존재하기 때문에 두 번째 54는 메모리의 세로적인 측면인 Address_Index 1에 위치하며, 세로적인 측면인 Rom_Index는 83번째 Permut_Index는 360개 중 310번째 위치한다.

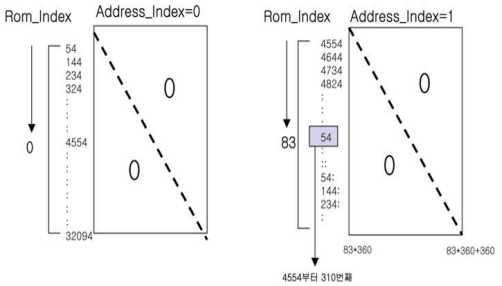


그림 8 CNU에서 데이터 엑세스 하는 예
Fig. 8 Example of Data Acess in CNU

하나의 블럭에는 360 개의 데이터로 되어 있으며, 총 90개의 블럭이 각각 메모리에 저장된다.

이의 구조는 HNS에서 제안한 Column Address 구조와 동일하며, 가로 블록 들은 서로 다른 메모리에 저장함을 원칙으로 한다. 초기값의 저장은 수신데이타를 360개씩 병렬로 각 메모리에 저장하면 된다. 세로 블록 들은 같은 메모리에 저장하며, 메모리의 개수는 Column weight의 수와 동일한 총 8개를 사용한다. 하나의 메모리에 사용되는 총 데이터 개수는 32400 개의 6비트로 구성되어 있다.

R=1/2 일 경우에 패리티 메트릭 주소에는 각각이 90씩 증가하여 하나의 블록에는 360개의 데이터가 저장되어 있다. 360개의 데이터를 각각의 블록으로 나눌 시에는 그림 9와 같이 나누어서 복호 과정을 진행한다. 부분 병렬은 360개의 CNU를 BNU를 가지고 있어야 하며, 이를 동시에 row weight의 개수 만큼 동시에 계산한다. 360개의 가지고 있는 하나의 블록의 CNU 계산이 끝나면 차례대로 90개의 블록 들을 직렬로 구현한다. 이는 CNU 계산 시 각 블록 인덱스와 각 블록에서 시작 위치에 대한 인덱스가 필요하며, BNU 계산 시에는 시작위치는 항상 각 블록의 시작점이므로 블록 인덱스만 필요하다. BNU 계산 시에 블록 인덱스를 어떻게 할당하느냐에 따라 필요 없을 수두 있으리라 사료된다.

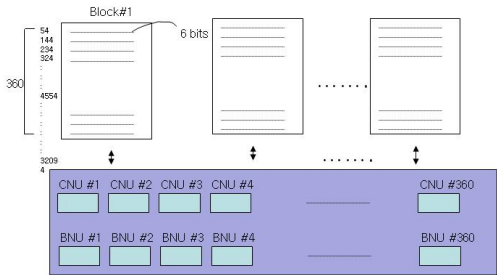


그림 9 부분 병렬 알고리즘 복호 구성도
Fig. 9 Decoder Structure of Partial Parallel Algorithm

V. 결 론

Ka 대역에서의 위성 방송 서비스시, 강우 감쇠, 비선형성등에 의한 신호 품질 저하는 매우 심각한 문제로 신호 손실을 효율적으로 보상하여 방송 서비스를 지속적으로 제공 할 수 있는 적응형 전송 기술 개발이 필요하다. 신뢰성이 낮은 광대역 위성방송 채널에서 다채널 및 고품질의 서비스를 제공하기 위해서는 채널 상태에 능동적으로 대처할 수 있는 적응형 오류 제어 방식을 사용해야 한다. 이를 위한 오류 정정 방식으로 실제적인 복호 알고리즘에서 샤논의 채널 용량 한계에 근접하여 관심의 대상이 되는 터보 코드와 LDPC코드방식이 관심의 대상이 되어지고 있고 이를 적응형 위성방송을 위한 설계기법 및 최적화 연구가 필요하다. 따라서 본 논문에서는 현재 DVB-S2 표준안 문서에 근거하여 부호율 R이 1/2, 2/3, 4/3, 5/6, 7/8, 8/9, 9/10 에 대해 블록사이즈 N = 64800 인 LDPC 복호화 방법에 대해 연구하였다. 아울러 R = 1/2 에 대해 H/W 구성 시 적합한 설계 기법을 제시하였다. 또한 향후 구현을 위해 FPGA 구현 및 ASIC 구현 시 구현할 수 있는 방안인 완전 병렬 과 부분 병렬 방식에 대해 부호기 및 복호기 구성 방법과 이에 대한 메모리 구조를 분석 하였다. 향후 본 논문에서 분석한 결과를 토대로 하여 FPGA 구현을 할 예정이다.

참 고 문 헌

[1] R. G. Gallager, "Low-Density Parity-Check Codes,"IRE trans. information theory,vol.8, PP.21-28, 1962.

[2] D. J. C. Mackay and R. M. Neal, "Near Shannon Limit Performance of Low-Density Parity-Check Codes," Electron. Letter, Vol.32, PP. 1645-1646,Aug.1996. [3] Draft ETSI EN 302 307, Digital Video Broadcasting(DVB); Second Generation framing structure, channel coding and modulation for Broadcasting Interactive Services, News Gathering and other broadband satellite applications, 2004.

[4] M. Sipser and D.A.Spielman,"Expander Codes, "IEEE Trans. Information Theory,vol.42, pp.1720-1722, Aug. 1996.

[5] J. Dielissen, A. Hekstra, and V. Berg. “Low cost LDPC decoder for DVB-S2.” In design, Automation and Test in Europe, 2006. DATE'06. Proceedings, volume 2, pages 1-6, Munich, Germany, March 2006.

[6] O. Eljamaly and P. Sweeney. “Alternative approximation of check node algorithm for DVB-S2 LDPC decoder.” In Second International Conference on Systems and Networks Communications (ICSNC 2007), pages 157-162, October 2007.

[7] M. Gomes, G. Falcao, V. Silva, V. Ferreira, A. Sengo, and M. Falcao. ”Flexible parallel architecture for DVB-S2 LDPC decoders.” In Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE, pages 3265-3269, Washington, USA, November 2007.

[8] C.Marchand, J.-B. Dore, L. Conde-Canencia, and E. Boutillon. “Conflict resolution for pipelined layered LDPC decoders.” ” In Signal Processing Systems, 2009. SiPS 2009. IEEE Workshop on. Pages 220-225, Tampere, Finlande, November 2009.

[9] 이규희, 김재권,윤상균 “ MCS레벨에 따른 적응형 수신기 ”, 한국정보전자통신기술학회 논문지 제 2권 1호, pp.59-64, 2009.

저자약력

정 지 원(Ji-Won Jung) **정회원**

1989년 2월 :성균관대학교 전자공학과(공학사)
1991년 2월 :성균관대학교 전자공학과(공학석사)
1995년 2월 :성균관대학교 정보공학과(공학박사)
1991년 1월~1992년 2월 : LG 정보통신연구소 연구원
1995년 9월~1996년 8월 : 한국통신 위성통신연구실 선임연구원
1997년 3월~1998년 12월 : 한국전자통신연구원 초빙 연구원
1996년 9월 ~ 현재 : 한국해양대학교 전파공학과 교수
2001년 8월 ~ 2002년 8월 : 캐나다 NSERC Fellowship (Communication Research Center 근무)

<관심분야> 위성통신, 이동통신, 변.복조기술, 채널코딩, FPGA 기술 등