HSS기반의 고속 LDPC 복호기 연구

정 지 원*

A Study on High Speed LDPC Decoder Based on HSS

Ji Won Jung*

요 약

본 논문에서는 DVB-S2 표준안에서 권고되고 있는 irregular LDPC 부호의 고속화 방안에 대한 연구를 하였다. LDPC 복호기에서는 많은 반복횟수와 많은 연산량이 복호 속도 저하의 원인이 되고 있으며, 성능 저하 없이 반복횟수와 연산량을 감소하기 위해서 HSS 기반의 LDPC 복호 구조를 제시하였다. 결과 반복횟수를 성능 저하 없이 절반으로 줄일 수 있으며, 이를 효율적인 설계방안을 제시하였다. 결과 600Mbps급의 throughput을 갖는 LDPC 복호기를 구현 가능케 하였다.

ABSTRACT

LDPC decoder architectures are generally classified into serial, parallel and partially parallel architectures. Conventional method of LDPC decoding in general give rise to a large number of computation operations, mass power consumption, and decoding delay. It is necessary to reduce the iteration numbers and computation operations without performance degradation. This paper studies Horizontal Shuffle Scheduling (HSS) algorithm. In the result, number of iteration is half than conventional algorithm without performance degradation. Finally, this paper present design methodology of high-speed LDPC decoder and confirmed its throughput is up to about 600Mbps.

Keywords : LDPC,HSS, Decorder, decoding delay

Ⅰ. 서 론

최신 위성 방송 및 무선 통신에서 적용되는 오류 정정 알고리즘으로 LDPC 채널 부호화 방식을 주로 적용하고 있다. ISDB-S나 DVB-S2 등의 표준을 적용 중인 일본이나 유럽, 미국 등지에서는 100 Mbps급 이상의 LDPC 복호기를 연구 중이며, 이를 동일한 성능에서의 고속화에 주안점을 두어 여러 논문에서 제안되고 있다. 본 논문에서는 여러 논문에서 제안되고 있는 알고리즘 보다 더욱 더 효율적인 알고리즘을 제안하여 고속화 시키고자 하며, 이에 따른 FPGA구현 결과를 제시하는 논문이다. 위와 같은 고속 LDPC 복호기를 구현하기 위해서는 알고리즘 측면과 구현 측면에서 여러가지 문제점이 있다. 알고리즘 측면에서는 첫째, LDPC 부호화 방식은 큰 블록 사이즈 및 많은 반복 횟수를 요구한다. DVB-S2에서 적용하는 LDPC의 경우 한 블록의 크기는 N=64800이고, 부호화율 1/2의 경우 60회의 반복횟수를 가진다. 이 때, 복호 속도를 높이기 위해서는 동일한 성능

* 해양대학교 (jwjung@hhu.ac.kr)
접수일자 : 2012년 7월 30일, 수정일자 : 2012년 8월 13일, 심사완료일자 : 2012년 9월 2일

을 유지하면서 반복 횟수를 줄일 수 있는 알고리즘이 필요하다. 둘째로는 복호 시 비트노드와 체크노드를 동시에 수행하는 알고리즘을 제시함으로써 고속화를 가능하게 한다. 기존의 LDPC 복호화 방식은 수신 데이터를 이용하여 체크노드의 값을 업데이트 한 후, 체크노드 값을 이용하여 비트노드의 값을 업데이트 함으로써 한번의 반복에 많은 시간이 걸리고, 복호 속도 저하의 원인이 된다.[1] 본 논문에서는 이를 위해 체크노드를 기반으로 하여 복호화 과정을 거치는 HSS 알고리즘에 대하여 연구하였다. 구현측면에서 복호 속도를 높이기 위해서는 데이터의 많은 병렬 처리가 필요하다. 이러한 병렬 처리에 의해 체크노드 업데이트(Check Node Update, CNU) 또는 비트노드 업데이트(Bit Node Update, BNU)의 연산 역시 병렬 처리가 가능하다.

II. 고속 복호 알고리즘

LDPC 부호는 '0'과 '1'만을 이용한 패리티 체크 메트릭스 H에 의해 만들어 진다.[5] H 메트릭스에서 '1'의 위치는 굉장히 sparse, 랜덤하게 분포되어 있다. 이러한 성질에 의해 LDPC 부호는 좋은 성능을 가지나[6], 하드웨어 구현에 있어서 문제가 되고, 큰 블록 사이즈와 많은 반복횟수로 인해 고속화가 힘들다. 본 논문에서는 이를 극복하기 위해 HSS 복호 방식에 대해 연구하였다.

2.1 HSS 방식

HSS 방식은 기존의 방식과는 달리 체크 노드 업데이트 연산을 하면서 비트 노드 업데이트 연산을 동시에 하는 것이 가능하다. HSS 복호 방식의 흐름도는 다음 그림 1과 같다 .LDPC 복호 과정은 반복에 의한 복호이다. 그러므로 적어도 각 노드들이 한번씩 업데이트가 되면 한번의 반복이 끝나는 것이다. 그리고 전체 복호 과정은 지정된 반복 횟수만큼 반복이 되거나 그렇지 않으면 충분히 신뢰성 있는 데이터를 구했을 때 끝나게 된다. 그 후, 비트 노드의 값을 이용하여 각 비트를 결정한다. HSS 방식을 이용한 LDPC 복호 과정 중 각 비트 노드의 값은 다음 식에 의해 구할 수 있다.

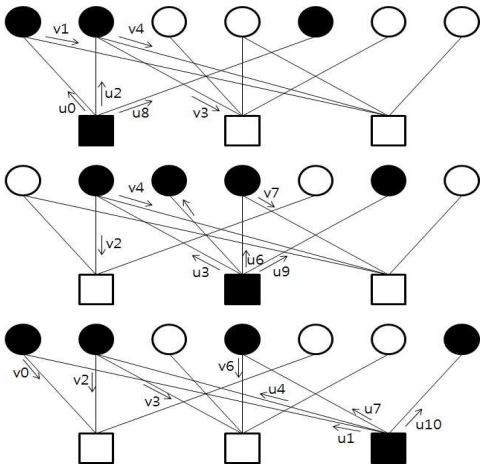

그림 1. HSS 복호 방식의 흐름도
Fig. 1. Data flow of HSS decoding

$$S_i = LLR_i + \sum_{i=1}^{d_v} u_i \tag{1}$$

여기서 S_i는 비트 노드의 최종 값을 나타내고, LLR은 수신 데이터를 나타낸다. i는 비트 노드이고, d_v는 비트 노드에 연결된 엣지의 수이다. 그리고 u_i는 체크 노드 업데이트를 통해 얻어진 각 엣지의 값이다. 위 그림 1를 통해 간단한 예를 들어 보면, 첫 번째 체크 노드 업데이트를 하기 위해 v_0, v_2, v_8 각각의 엣지 값을 가지고 체크 노드 업데이트를 한다. 이 때, 각 엣지의 값은 수신 데이터 LLR_0, LLR_1, LLR_4이고, 체크 노드 업데이트 방법은 다음 식으로 알 수 있다.

$$u_j = \bigoplus_{k=1, k\neq j}^{d_c} v_k \tag{2}$$

v는 비트 노드에서 체크 노드로 향하는 엣지를 나타내고, d_c는 체크 노드에 연결된 엣지의 수이다. 위 식에서 는 다음과 같이 구할 수 있다.

$$|v_i| \oplus |v_j| = \min(|v_i|, |v_j|) - offset \tag{3}$$

$$sign(|v_i| \oplus |v_j|) = sign(v_i) \times sign(v_j) \tag{4}$$

체크 노드 업데이트의 값을 이용하여, 각 비트 노드의 값은 $S_0 = LLR_0 + u_0$, $S_1 = LLR_1 + u_2$, $S_4 = LLR_4 + u_8$이 된다. 그 후, 두 번째 체크 노드 업데이트를 위해 v_3, v_5, v_6, v_8 을 가지고 온다. v는 v'를 이전 반복에서의 엣지 값이라 하면 다음 식에 의해 구해질 수 있다.

$$v_i = S_i - v'_i \tag{5}$$

두 번째 체크 노드의 업데이트가 끝나면 각 엣지의 값을 이용하여 다시, S_1, S_2, S_3, S_5 가 구해지고, 세 번째 체크 노드에 대한 업데이트를 하여 S_0, S_1 , S_3, S_7 역시 업데이트 되어 진다. 이와 같이 한 번의 반복이 끝나게 된다. 이러한 과정이 반복되면서 모든 반복이 끝나거나, 신뢰성 있는 데이터가 나올 때, 복호 과정은 끝나게 된다.

그림 2는 각 알고리즘에서 부호화율 1/2, N=64800 일 때, 반복횟수에 따른 성능이다.

HSS 방식의 경우 30회의 반복을 하였을 때, 기존의 방식(60회 반복)과 성능이 거의 일치함을 알 수 있다. 즉, HSS 방식을 적용하면 반복횟수 만으로 속도가 2배 상승하는 효과를 얻을 수 있다.

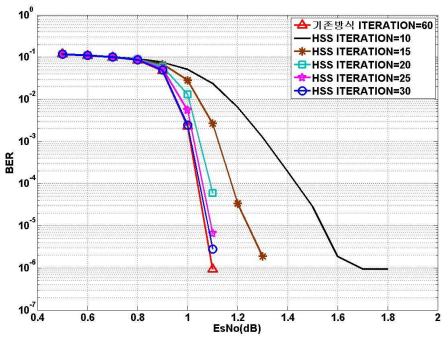

그림 2. HSS의 반복횟수에 따른 성능 비교
Fig. 2. Performance comparison by iteration number of HSS

Ⅲ. 고속 LDPC 복호기 설계방안

HSS 방식에 대한 복호기의 블록도는 그림 3과 같다.

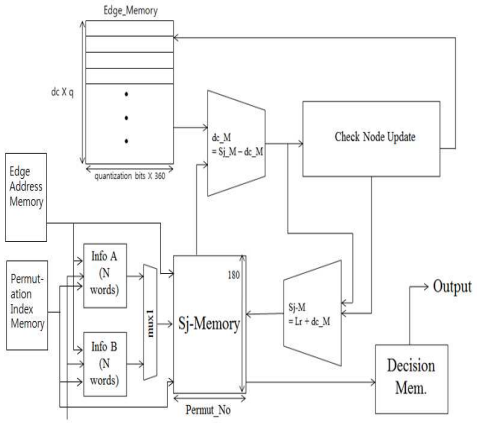

그림 3. HSS 기반의 LDPC 복호기의 구조
Fig. 3. Structure of LDPC decoder based on HSS

HSS 방식을 적용한 복호기의 구조를 보면 크게 메모리 부분과 CNU 부분으로 나눌 수 있다. 특히 수신 데이터를 저장하는 메모리와 Sj 메모리에서 데이터를 읽어 올 때는 H 매트릭스에 따라 랜덤하게 메모리 엑세스가 되어야 하기 때문에 이를 구현할 수 있는 여러 인덱스들이 필요하다. HSS 방식을 적용한 LDPC 복호기는 BNU 계산을 위한 블록을 따로 만들지 않는다. 그 이유는 CNU 블록에서 나온 출력 값들을 바로 Sj 메모리에 업데이트 시키기 때문에 BNU를 위한 연산만을 따로 하지 않기 때문이다.이러한 구조를 적용하여 성능과 속도 향상을 위한 HSS 기반의 FPGA 구현을 위한 고속 LDPC 복호기의 구조는 그림 4와 같다.

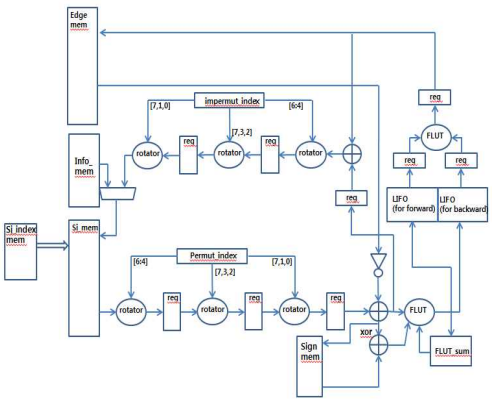

그림 4. 구현을 위한 HSS 기반의 LDPC 복호기의 구조
Fig. 4. Structure of LDPC decoder based on HSS for Implementation

그림에서 Info_mem은 수신데이터 메모리로 한 주소 당 360개의 수신 데이터를 저장한다. 첫 번째 반복 시 Info_mem에서 Sj_mem으로 처음 데이터가 입력 되어지고, 이 때는 Sj_index를 사용하지 않는다. Sj 메모리에 모든 데이터가 저장된 후 Sj_index 메모리에 저장되어 있는 8비트의 index 데이터가 출력이 되고, 이 index 데이터는 Sj 메모리의 주소 값이 되어 Sj 데이터가 출력된다. 이렇게 출력된 Sj 데이터는 rotator로 입력되어 진다. Rotator에 의해 우 순환 쉬프트 되어진 360개의 데이터는 CNU 되어진다. 또한, SC 알고리즘을 위하여 첫 번째 반복 때는 각 데이터의 부호 비트 360개를 부호 메모리에 저장한다. 첫 반복 때는 SC 알고리즘을 사용하지 않고, 원래의 데이터로만 CNU 연산을 실행한다. CNU 블록에서 연산되어 진 데이터는 각각의 엣지 값이므로 출력 순서대로 엣지 메모리에 저장이 된다. 또한, CNU 연산된 새로운 엣지 데이터와 CNU의 입력 데이터가 더하기 연산을 통하여 역 rotator로 입력되어 진다. 역 rotator는 데이터가 CNU 입력 전에 우 순환 엣지 되어진 것을 다시 원래의 데이터 순서대로 쉬프트 시키는 것이다. 이 때 역시, permutation index가 필요하다. 역 rotator의 permutation index는 rotator의 permutation index와 구조가 같다. 하지만 역 rotator는 우 순환 쉬프트 연산과는 반대로 좌 순환 쉬프트 연산을 필요로 한다. 역 rotator를 거쳐 원래의 데이터 순서대로 되돌려진 360개의 데이터는 Sj index에 따라 Sj 메모리에 입력되어 진다. 이렇게 첫 번째 반복이 끝나고, 두 번째 반복부터는 Info 메모리를 사용하지 않고, Sj 메모리의 데이터를 Sj index에 따라 출력시킨다. 이 360개의 데이터는 rotator를 거쳐 CNU 연산이 되어지는데, 두 번째 반복부터는 CNU 연산 블록에 입력되기 전에 엣지 메모리에서 이전 반복 시의 엣지 데이터를 출력시켜 Sj 데이터에서 뺀 값을 CNU 연산 블록으로 입력시킨다. 또한, 이 때의 부호 비트와 부호 메모리에 저장되어진 이전 반복 때의 부호 비트를 xor 연산을 시킨 뒤 Sj 데이터와 같이 입력시킨다. Sign 비트의 xor 연산은 SC 알고리즘을 적용시키기 위함이다. 이후의 과정은 첫 번째 반복과 동일하다.

제시된 복호기의 구조를 사용할 경우 한 번의 반복에 사용되는 클럭 수는 662클럭이다. 다음 표들은 복호기 parameter를 나타내었다. DVB-S2 LDPC는 블록 사이즈 N=64800이고, 데이터는 360개 씩 그룹으로 병렬 처리 되어진다. 그림 4에서의 데이터 쉬프트에 의한 지연은 3클럭이고, 부호화 율이 1/2 이므로 최대 반복 횟수는 30회이다. 수신데이터의 양자화 비트수는 5비트이나, 연산 시 데이터의 비트수는 6비트이다.

표 1 구현 파라메터
Table 1. Implementation Parameters.

Level of parallelism P	360
Clock frequency (Mhz)	200
Latency of rotation	3
Max iteration	30
Bit size of message	6
Bit size of intrinsic	5

표 2 소모 클럭과 복호 속도
Table 2. Required clocks and decoding throughput.

Number of group check M	90
Number of group variable N	180
Number of group edges E	630
Nb cycle check node	660 clock cycles
Total iteration	662 clock cycles
Total nit iteration	19860 clock cycles
Decoding throughput	653 M bits/s

표2에서는 전체 블록 사이즈가 64800이고, 360개의 데이터가 그룹을 이루므로 부호화율 1/2에서 체크노드의 그룹은 90개이고, 비트 노드의 그룹은 180개이다. 총 엣지의 수는 630개 이고, 체크노드 연산에 사용되어 지는 총 클럭 수는 660개이다. 한 번의 반복에 소모되는 클럭수는 662 클럭이고, 30회 반복이므로 총 클럭 수는 19860 클럭이 소모되었다. 이 때, 복호 throughput은 653Mbps이다.

IV. 결 론

본 연구에서는 첫째로 고속 복호 알고리즘 관점에서는 비트 노드 계산을 체크 노드 계산 후에 하지 않고 체크 노드 계산 중에 수행하는 HSS(Horizontal

Shuffle Scheduling) 방식을 연구하여 최적의 반복횟수를 제시하였다. HSS방식은 체크 노드 계산 시 동시에 비트 노드를 계산하기 때문에 별도의 비트 노드 계산을 할 필요가 없으므로 속도를 고속화 가능하며, 또한 반복 횟수를 감소시킬 수 있어 복호 throughput을 증가 시킬 수 있다. 시뮬레이션 결과 부호화율 1/2일 때, 약 50% 반복횟수를 동일한 성능에서 감소시킬 수 있음을 확인할 수 있었다. 또한 LDPC 복호기에서는 많은 반복횟수와 많은 연산량이 복호 속도 저하의 원인이 되고 있으며, 성능 저하 없이 반복횟수와 연산량을 감소하기 위해서 HSS 기반의 LDPC 복호 구조를 제시하였다. 결과 반복횟수를 성능 저하 없이 절반으로 줄일 수 있으며, 이를 효율적인 설계방안을 제시하였다. 결과 600Mbps 급의 throughput을 갖는 LDPC 복호기를 구현 가능케 하였다.

참 고 문 헌

[1] 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, vol. 2, pp. 1-6, Munich, Germany, March 2006.

[2] 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), pp. 157-162, October 2007.

[3] Xiao-Yu Hu, E. Eleftheriou, D.-M. Arnold, and A. Dholakia, “Efficient implementations of the sum-product algorithm for decoding LDPC codes.” In Global Telecommunications Conference, 2001. GLOBECOM'01. IEEE, vol. 2, pp. 1036-1036E, 2001.

[4] V. Savin. “Self-corrected min-sum decoding of LDPC codes.” ISIT IEEE(2008), pp. 146-150, 2008.

[5] R. G. Gallager, “Low-Density Parity-Check Codes,”IRE trans.information theory, vol. 8, pp..21-28, 1962.

[6] 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.

[7] Digital Video Broadcasting(DVB). “Second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications (DVB-S2).” European Standard (Telecommunications series) ETSI EN 302 307 V1.2.1(2009-08),2009.

[8] 김도석,정훈주,이용환 “ 직렬방식의 주변장치 통합 인터페이서 설계”,한국정보전자통신기술학회 4권 1호, pp.68-75, 2011

저자약력

정 지 원(Ji-Won Jung) **정회원**

1989년 2월: 성균관대학교 전자공학과(공학사)

1991년 2월: 성균관대학교 전자공학과(공학석사)

1995년 2월: 성균관대학교 정보공학과(공학박사)

1991년~1992년 : LG 정보통신 연구소 연구원

1995년~1996년 : 한국통신 위성통신연구실 선임연구원

1997년~1998년 : 한국전자통신연구원 초빙 연구원

1996년~현재 : 한국해양대학교 전파공학과 교수

2001년~2002년 : 캐나다 NSERC

<관심분야> 이동통신, 센서네트워크, 유비쿼터스 홈네트워크