VLSI Implementation of a Soft Bit-Flipping Decoder for PG-LDPC Codes

김종홍, 조준호, 성원용
서울대학교 전기공학부

초록

Low-density parity-check codes are known to show higher error correcting performance than conventional algebraic codes. However, the VLSI implementation of the codes has been considered very difficult especially when the row or column weight of them is high. In this paper, a projective-geometry (PG) LDPC code is implemented in VLSI employing the proposed soft bit flipping (SBF) algorithm. In addition to the processing unit sharing, the pipelining technique is employed to increase the decoding throughput. With the (1057, 813) PG-LDPC code, the implemented 4-bit SBF decoder consumes only a small area of 2.5mm^2, while providing the throughput of 6.5Gbps and good error performance close to the floating-point sum-product algorithm (SPA) by 0.6dB at the frame error rate (FER) of 10^{-4}.

1. 서론

Low-density parity-check (LDPC) 부호는 신뢰도 확산(belief-propagation) 기반의 반복 복호(iterative decoding) 알고리즘을 사용하며, 섀논의 한계(Shannon-limit)에 가까운 성능을 보이는 것으로 알려져 있다[1-2]. 가장 대표적인 알고리즘으로는 연판정(soft-decision) 기반의 가장 좋은 오류성능을 보이는 SPA[3]와 경판정(hard-decision)기반의 구현이 용이한 bit-flipping (BF)[1]알고리즘이 있다. 본 제안되는 SBF 알고리즘[4]은 기존의 SPA 와 BF 알고리즘의 장점을 취하여 고안된 알고리즘으로 내부 연결선의 복잡도와 계산 량을 줄이기 위해 BF 알고리즘의 기본 구조를 따르며, 에러 성능을 개선하기 위해 SPA 에서와 같이 신뢰도(reliability)를 이용한다. BF 와 같이 각 노드의 메시지는 단 하나의 값이기 때문에, 행과 열의 가중치(row/column weight)가 증가하여도 SBF 알고리즘의 연산 복잡도는 SPA 에서 보다 매우 천천히 증가한다. 따라서 좋은 에러 성능을 보이지만 가중치(weight)가 높은 PG-LDPC 부호[5]와 같은 것이 SBF 알고리즘을 사용하여 효율적으로 구현될 수 있다.

본 논문에서는 순환 PG-LDPC 부호를 이용하여 설계된 SBF 복호기를 설명한다. 하드웨어 면적을 줄이고자 두 노드 연산 유닛의 공통된 하드웨어 유닛을 공유하여, 하나의 노드 연산 유닛(shared node processing unit, SNPU)으로 대체하는 방법을 사용하였다. 처리량(throughput)을 높이고자 병렬화 된 복호기를 설계하였으며, 파이프라인 기법을 사용하였다. (1057,813) PG-LDPC 부호가 구현에 이용 되었다.

2. Soft bit-flipping 알고리즘

$M \times N$ 패리티 검사행렬(parity-check matrix) $\mathrm{H} = [h_{m,n}]$으로 정의된 정규(regular) *(N,K)* LDPC 부호를 이용하자. $B(m) = \{n | h_{m,n} = 1\}$를 m 번째 체크(check)노드와 연결된 변수(variable)노드의 집합으로, $\mathrm{A}(n) = \{m | h_{m,n} = 1\}$ 를 n 번째 변수노드와 연결된 체크노드의 집합으로 정의한다. 부호어(codeward)는 BPSK 변조되어 AWGN 채널을 거쳐 전송된다. 경판정 벡터와 양자화된 변수노드를 각각 $\mathbf{z} = (\mathrm{z}_1, \mathrm{z}_2, \cdots, \mathrm{z}_\mathrm{N})$, $\mathbf{y} = (\mathrm{y}_1, \mathrm{y}_2, \cdots, \mathrm{y}_\mathrm{N})$으로 정의한다. 이때 m 번째 체크노드의 가중치는 아래와 같이 계산된다.

$$w_m = \sum_{i \in B(m)} |y_i|, \qquad m \in [1, M] \qquad \text{(식 3.1)}$$

만일 각 노드가 q 비트로 양자화 된다면 변수노드와 체크노드에 대해 각각 $\delta_1^v < \delta_2^v < \cdots < \delta_{2^q-1}^v$과 $\delta_1^c < \delta_2^c < \cdots < \delta_{2^q-1}^c$의 문턱 값(thresholds)이 존재하며, 양자화된 각 노드는 ± 0을 제외한 $[-2^{\mathrm{q}-1} \sim -1]$과 $[1 \sim 2^{\mathrm{q}-1}]$ 의 값을 가진다. 반전강도(flipping strength)는 세 개의 반전 문턱 값 $(\delta_1^f < \delta_2^f < \delta_3^f)$에 의해 $[-2 \sim +1]$의 값의 범위로 결정된다. SBF 알고리즘의 순서는 아래와 같다.

1) 초기화: 반복 수 k 를 0 으로 초기화 하고, 수신된 메시지를 양자화 레벨(quantization level) δ_i^v $(i \in [1, 2^q - 1])$ 에 따라 q 비트로 양자화 하여 변수노드 y_n^k $(n \in [1, N])$에 저장한다.
2) 단계 1: m 번째 체크노드의 가중치 w_m^k $(m \in [1, M])$를 계산하고, δ_i^c $(i \in [1, 2^q - 1])$를 이용하여 양자화 한다. 또한 syndrome vector $\mathbf{s}^\mathrm{k} = \mathbf{z}^\mathrm{k}\mathbf{H}^\mathrm{T}$ 를 계산하여 만일 $\mathbf{s}^\mathrm{k}$이 영 벡터라면 $\mathbf{z}^\mathrm{k}$를 복호된 부호어 출력하고 복호를 종료한다.
3) 단계 2: 모든 변수노드에 대해 반전함수(flipping function)를 계산한다.

$$e_n^k = \sum_{m \in A(n)} (2s_m^k - 1)\, w_m^k - \alpha |y_n^k|$$

4) 단계 3: 각 변수노드를 아래와 같이 갱신한다.
   ①. 강 반전$\left(e_n^k \in (\delta_3^f, \infty)\right)$:
   $$y_n^{k+1} = \mathrm{sgn}(y_n^k - 2.5) \cdot \max(1, |y_n^k| - 2),$$
   ②. 약 반전$(e_n^k \in (\delta_2^f, \delta_3^f])$:
   $$y_n^{k+1} = \mathrm{sgn}(y_n^k - 1.5) \cdot \max(1, |y_n^k| - 1),$$
   ③. 유지 $(e_n^k \in (\delta_1^f, \delta_2^f])$: $y_n^{k+1} = \mathrm{sgn}(y_n^k) \cdot |y_n^k|$,
   ④. 강화 $(e_n^k \in (-\infty, \delta_1^f])$: $y_n^{k+1} = \mathrm{sgn}(y_n^k) \cdot \min(2^q, |y_n^k| + 1)$.

   만일 어떠한 변수노드에서도 비트 반전이 발생하지 않았다면 $\delta_3^f = \max_\mathrm{n} e_n^k, \delta_2^f = \delta_3^f - \beta$로 갱신한다.
5) 단계 4: 반복 수 k 를 1 증가 시킨다. 만일 이 값이 사전에 정의된 반복 수 K_{max} 보다 크면 복호를 종료(복호 실패) 하고, 그렇지 않다면 단계 1 로 이동한다.

SBF 기반의 복호는 심벌의 경판정 비트와 패리티검사를 사용할 뿐만 아니라 적은 비트의 정밀도를 신뢰도로 이용한다.

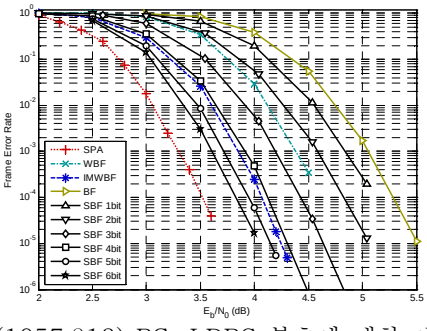

그림 1. (1057,813) PG-LDPC 부호에 대한 에러 성능

이 논문은 지식경제부 출연금으로 ETRI 와 시스템반도체산업진흥센터에서 수행한 IT SoC 핵심설계인력양성사업, 그리고 하이닉스반도체 주식회사의 지원을 받아 수행한 연구결과 입니다.

그림 1 은 (1057,813) PG-LDPC 부호에 대한 여러 알고리즘의 성능을 나타낸다. 그림에서 SBF 알고리즘의 성능은 고정소수점 연산의 결과이며, 나머지는 부동 소수점 연산의 결과이다. SPA 에 비해 적은 연산량을 가진 SBF 알고리즘이 이상적인 SPA 에 가까운 성능을 보인다.

3. Soft bit-flipping 복호기의 구조

a) 직렬구조(Serial Architecture)

그림 2 는 전체적인 복호기 구조를 나타낸 것으로, 크게 변수노드와 체크노드의 값을 저장하는데 쓰이는 N 개의 4-bit 쉬프트 레지스터 두 쌍과 하나의 변수노드 연산 유닛(variable node processing unit, VPU), 하나의 체크노드 연산 유닛(check node processing unit, CPU)으로 이루어져 있으며, 오버플로를 방지하기 위해 포화기(saturator)가 있다.

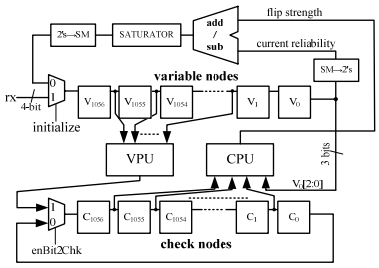

그림 2. 복호기 전체의 구조

VPU 와 CPU 가 각각 그림 3, 4 에 나타나 있다. 각 노드는 $[s_n : m_n]$로 나타낸다. VPU 의 패리티 출력은 모듈로-2 합의 결과이며, 이는 체크노드의 부호로도 쓰인다. 각 노드 값의 범위는 알고리즘에서는 ± 0 을 제외한 [-8,-1]과 [1,8]이지만, 4 비트로 구현할 경우 [-8,-1]과 [0,7]으로, 양수의 경우 1 이 적은데 이를 보상하기 위해 CPU 에 1 비트입력 덧셈기가 추가하여 양의 입력의 개수를 세도록 하였다. VPU 에서 변수노드의 절대값-신뢰도-의 합 연산이 이뤄지며, 체크노드에서 signed summation 이 이뤄지기 때문에 변수노드와 체크노드에 저장되는 메시지의 형태는 각각 부호-크기체계(sign-magnitude)와 2 의 보수이다. CPU 의 하단부는 문턱 값을 갱신하는데 쓰이는 모듈이다.

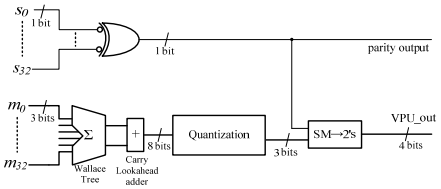

그림 3. Variable node processing unit

b) 병렬구조 (Parallel Architecture)

P 를 병렬도(parallel factor)라 할 때, 병렬 복호기는 p 개의 VPU 와 CPU 를 사용한다. 이 때 CPU 에서 쓰이는 문턱값은 p 개의 CPU 에서 공통으로 사용되기 때문에 각각의 CPU 는 임계치 적응(threshold adaptation) 모듈을 따로 따로 필요로 하지 않는다.

하드웨어 면적을 줄이기 위해 CPU 와 VPU 를 SNPU 로 대체하는 방법이 고안되었다. VPU 의 XOR 게이트의 출력은 CPU 의 1 비트 입력 덧셈기의 LSB(Least Significant Bit)로 대체 할 수 있으며, VPU 와 CPU 의 Wallace tree 는 서로 공유 된다. SNPU 를 사용한 면적감소는 병렬도가 증가할 수록 더 커지게 된다.

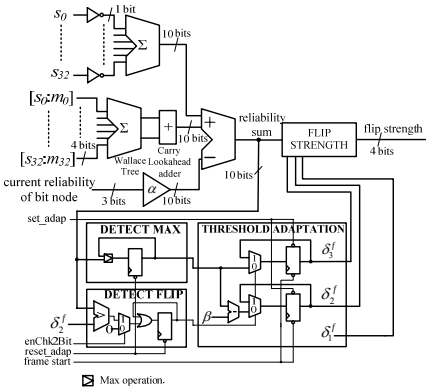

그림 4. Check node processing unit

클럭 주파수의 증가를 위해 SNPU 에 파이프라인 기법이 적용되었다. 파이프라인 단수는 2 로서, 파이프라인 레지스터는 임계경로(critical path)의 중간 지점인 Wallace tree 와 올림수 예측 가산기(carry look-ahead adder) 사이에 위치한다.

	Conventioal		SNPU		Pipelined SNPU	
	srial	p=64	srial	p=64	srial	p=64
Total area (μm2)	500,201	2,599,362	496,535	2,366,348	499,037	2,509,915
Max. frequency (MHz)	149.25	136.24	135.14	120.92	248.76	215.52
Max. throughput (Mbps)	74.63	4235.45	67.57	3759.16	124.32	6508.62

표 1. SBF 복호기의 구현 결과

4. 실험 결과

SBF 복호기의 구현결과가 표 1 에 나타나있다. SNPU 를 사용하여, 병렬도 64 에서 약 9%의 면적 감소 효과를 얻었으며, 파이프라인 기법을 적용한 결과, conventional serial architecture 와 비교하였을 때 5 배의 면적 증가로 87 배의 처리량이 증가 하였다.

5. 결론

본 논문에서는 (1057,813) PG-LDPC 부호에 대한 SBF 복호기를 구현하였다. 파이프라인 기법을 적용한 제안된 SNPU 구조의 복호기는 2.5mm^2 의 적은 면적을 차지하며 최대 6.5Gbps 의 처리량을 보였으며, 이것은 높은 가중치의 PG-LDPC 부호의 세계 최초의 구현 결과이다.

참고문헌

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

[2] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–432, Mar. 1999.

[3] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[4] J. Cho and W. Sung, "Soft Bit-Flipping Algorithm for Decoding of LDPC Codes," unpublished.

[5] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: A rediscovery and new results," *IEEE Trans. Inform. Theory*, vol. 47, no. 7, pp. 2711–2736, Nov. 2001.