# 저전력 및 면적 효율적인 터보 복호기를 위한 정규화 유닛 설계

정회원 문 재 우*, 김 식*, 황 선 영*

# Design of the Normalization Unit for a Low-Power and Area-Efficient Turbo Decoders

**Je-Woo Moon*, Sik Kim*, Sun-Young Hwang*** ***Regular Members***

요 약

본 논문은 블록 MAP 터보 복호기의 상태 메트릭 계산 유닛을 위한 정규화 방식을 제안한다. 제안된 방식은 정규화를 위해 격자단(trellis stage)에서 하나의 상태 메트릭 값을 모든 상태 메트릭에 빼주고 쉬프트 시키는 구조를 갖게 되어, 상태 메트릭 계산이 많은 블록 MAP 복호기의 격자단에서 하나의 상태 메트릭을 줄여 파워 소모와 메모리 요구량을 줄일 수 있다.

시뮬레이션 결과 제안된 정규화 구조를 적용한 터보 복호기는 기존의 블록 MAP 터보 복호기에 비해 동적 파워 소모는 17.9%까지 감소하고 면적은 6.6%까지 감소함을 확인하였다.

Key Words : 저전력, 터보 복호기, 정규화

ABSTRACT

This paper proposes a novel normalization scheme in the state metric calculation unit for the Block-wise MAP Turbo decoder. The proposed scheme subtracts one of four metrics from the state metrics in a trellis stage and shifts, if necessary, those metrics for normalization. The proposed architecture can reduce power consumption and memory requirement by reducing the number of the state metrics by one in a trellis stage in the Block-wise MAP decoder which requires an intensive state metric calculations.

Simulation results show that dynamic power has been reduced by 17.9% and area has been reduced by 6.6% in the Turbo decoder employing the proposed normalization scheme, when compared to the conventional Block-wise MAP Turbo decoders.

## I. 서 론

터보 코드는 1993년 Berrou 등에 의해 처음 제안되었으며[1], 비교적 간단한 구조를 가지면서도 Shannon의 한계[2]에 근접하는 매우 우수한 오류정정 성능을 제공한다. 터보 부호를 이용한 복호기는 반복 복호를 위한 회귀 구조, 인터리빙이나 디인터리빙을 통한 매우 긴 복호 지연시간과 많은 메모리 접근을 요구하는 특성으로 인하여 실시간 처리가 필요 없는 장거리 우주 통신 분야쪽으로 응용이 제한되어 왔으나, VLSI 설계 기술의 발전과 더불어 실시간 전송을 위한 연구가 활발히 진행되고 있다.

*서강대학교 전자공학과 CAD & Computer System 연구실 (hwang@ccs.sogang.ac.kr)

논문번호 : 030050-0203, 접수일자 : 2003년 2월 3일

※본 연구는 한국과학재단 목적기초연구 (R01-2001-000-00321-0) 지원으로 수행되었습니다.

최근 차세대 이동통신 IMT-2000의 채널 부호화 방식의 저속 음성 통신용으로 길쌈 부호가 채택되고 32Kbps 이상의 고속 데이터 전송에 터보 코드가 채택 되면서 실시간 처리 외에 면적과 전력 소모측면에서 경쟁력 있는 디코더 설계를 위한 연구가 진행 중이다. 터보 부호기는 프레임 단위의 입력과 인터리버에 의해 재배열된 입력이 각각 구성 부호기(Constituent Code)를 통과해 출력된 패리티 비트를 발생하며 입력 정보 비트 열과 발생된 패리티 비트를 이용하여 부호화를 수행한다. 이때 인터리버는 채널의 연집 오류를 분산시켜 오류정정 성능을 높이는 역할을 할 뿐만 아니라 구성 부호기들간의 독립적인 패리티를 발생시키게 함으로써 반복 복호를 통한 성능 향상을 가능하게 한다. 터보 복호기는 연판정(Soft Decision) 정보를 사용하는 내부 복호기, 인터리버/디인터리버, 경판정(Hard Decision) 블록으로 구성된다. 내부 복호기에 사용되는 복호 알고리듬으로 MAP(Maximum A Posteriori) 알고리듬과 SOVA(Soft Output Viterbi Algorithm) 방식이 사용되고 있다. 일반적으로 MAP 복호기는 SOVA 복호기에 비해 4배가량의 복잡도를 가지는 반면 2배의 좋은 성능을 갖는다고 알려져 있다[3]. MAP 알고리듬을 적용한 복호기는 연판정 값을 입력으로 사용하는 SISO(Soft Input Soft Output) 방식을 사용하며 연판정 출력 값으로부터 부가정보(Extrinsic Information)를 계산하여 다른 복호기의 사전 확률 값(APP : A Priori Probability)으로 사용한다. 터보 코드에 사용된 MAP 알고리듬은 반복 복호를 통해 계속적으로 APP값을 갱신시킴으로써 오류정정 성능을 높일 수 있지만 복호지연 시간 및 복호 연산량을 고려하여 적절한 반복 복호 횟수를 결정하여야 한다[4].

Log-MAP 알고리듬은 MAP 복호기의 구현 단계에서 연산 복잡도를 낮추기 위해 곱셈 연산을 덧셈 연산으로 변환 처리하도록 제안된 알고리듬으로 여전히 인터리버 크기에 비례하는 상태 메모리와 가지 메트릭(Branch Metric) 저장용 메모리가 요구되어 메모리 사용량을 효율적으로 줄일 수 있는 블록 MAP 알고리듬이 제안되었다. 블록 MAP 알고리듬은 MAP 복호 알고리듬에 비해 적은 역방향 연산(Backward Processing)을 수행하기 때문에 MAP 알고리듬을 적용한 복호기와 비슷한 성능을 내기 위해 역방향 메트릭(Backward Metric)의 초기값을 구하는 트레이닝 과정을 수행한다[5][6]. 일반적으로 하드웨어의 활용도를 높이기 위해 트레이닝 크기와 블록 크기는 같게 한다.

본 논문에서는 Log-MAP 알고리듬에서 데이터의 복원이 상태 메트릭의 절대값이 아닌 메트릭 간의 상대적 차이에 의해 결정된다는 성질[7]을 이용하여 모든 상태 메트릭에서 하나의 상태 메트릭을 빼주는 방식을 하드웨어에 적용해 상태 메트릭의 연산량과 면적을 줄이는 복호기 구조를 제안한다. 제 2절에서는 Log-MAP 복호기의 구조에 대해 설명하고, 제 3절에서는 제안된 복호기의 구조를 제시한다. 제 4절에서는 실험 결과를, 제 5절에서는 결론과 추후 과제를 제시한다.

## II. Log-MAP 복호기의 구현

MAP 알고리듬을 이용한 터보 복호기는 많은 메모리 사용과 복호지연, 구현의 복잡도가 큰 문제점이었지만 MAP 알고리듬에서 사용되는 곱셈연산을 덧셈연산으로 바꾸어 주는 로그 차원의 계산을 해줌으로써 계산의 복잡도를 크게 개선시킨 알고리듬이 Log-MAP 알고리듬이다[5][8]. Log-MAP 알고리듬은 식 (2-1)과 같은 자코비안 로그 정리를 이용한다.

$$\begin{aligned} MAX^*(\delta_1,\delta_2) &= \log(e^{\delta_1}+e^{\delta_2}) \\ &= \max(\delta_1,\delta_2)+\log(1+e^{-|\delta_2-\delta_1|}) \\ &= \max(\delta_1,\delta_2)+f_c(|\delta_1-\delta_2|) \end{aligned} \quad \text{식 (2-1)}$$

1. 가지 메트릭 계산 유닛 구조

$\delta_k^i(m)$은 현재 상태에서 입력의 확률로 가지 메트릭(branch metric)이라고 정의하며 식 (2-2)와 같

이 나타낸다.

$$\bar{\delta}_k^{i,m} = \frac{1}{2}\left(L_a(d_k)u_k^i + L_c x_k u_k^i + L_c y_k v_k^i(m)\right)$$

식 (2-2)

시간 $k$ 에서 수신된 데이터 비트열은 가지 메트릭 $(\delta_k^i(m))$을 발생시킨다. $u_k^i$는 현재 상태와 상관없는 조직적인(systematic) 정보 비트이고 $v_k^i(m)$은 $S_k = m$ 상태에서 $d_k = i$ 일때의 정상적인 패리티 비트이다[9]. $u_k^i$, $v_k^i(m)$은 부호화 방식에 의해 결정되며, 그림1은 $u_k^i$, $v_k^i(m)$의 모든 경우에 따라 가지 메트릭을 구하는 구조를 보이고 있다[10]. 여기서 $L_a(d_k)$는 사전 확률 값이며, $L_c X_k$와 $L_c Y_k$는 백색 가우시안 잡음(AWGN) 채널을 거쳐 수신된 신호이다. $L_c X_k$는 실질적인 정보를 가지고 있는 정보 신호이고, $L_c Y_k$는 부수적인(redundancy) 패리티 비트를 나타낸다. BM0는 Zero를 쉬프트 해서 얻은 메트릭으로 $u_k^i$, $v_k^i(m)$이 각각 0일때의 가지 메트릭이며 쉬프트를 수행해도 항상 0인 값을 유지한다[10].

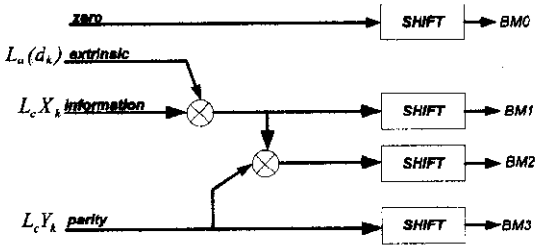



그림 1. 가지 메트릭 계산 유닛 구조

2. 상태 메트릭 계산 유닛 구조

상태 메트릭은 식 (2-1)의 $MAX^*(\delta_1, \delta_2)$ 연산을 통해 구해진다. $MAX^*(\delta_1, \delta_2)$의 계산을 위한 하드웨어 구조는 그림 2와 같다. $MAX^*(\delta_1, \delta_2)$의 입력 변수 $\delta_1$과 $\delta_2$는 가지 메트릭 $BM$과 상태 메트릭 $SM$의 합으로 얻어지며 구해진 두 변수중 큰 값을 구하는 $Max(\delta_1, \delta_2)$ 연산과 보정항인 $f_c(|\delta_1 - \delta_2|)$을 계산하기 위한 LUT(Look Up Table)과 최대값과 보정항을 더하는 덧셈기로 이루어져 있다.

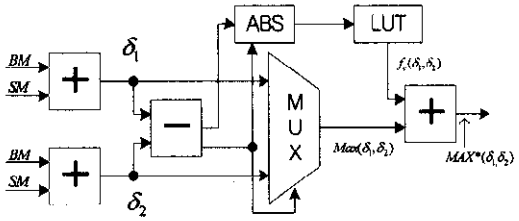



그림 2. 상태 메트릭 계산 유닛 구조.

3. LLR 계산 유닛 구조

3개 이상의 항을 가지고 있는 $MAX^*$ 연산자는 선형 로그 MAP 근사법(linearly approximated Log-MAP algorithm)을 이용해 식 (2-3)과 같이 확장할 수가 있다[11].

$$MAX^*(\delta_0, \delta_1, \delta_2, \delta_3)$$
$$= MAX^*(MAX^*(\delta_0, \delta_1), MAX^*(\delta_2, \delta_3))$$

식 (2-3)

Log-MAP 알고리듬은 두항의 최대값과 보정항의 합으로 구현되지만 LLR 계산에 있어 $MAX^*$ 연산자를 계산의 복잡성을 피하기 위해 두항의 최대값만을 사용하는 MAX-Log-MAP을 사용하여 계산하는 경우가 많다. 그림 3은 4개의 상태를 갖는 경우 MAX-Log-MAP을 이용해 LLR값을 구하는 연산기 구조를 보인다. $Adder(\alpha_{s0}, \beta^0{}_{s0})$는 첫 번째 상태의 순방향 메트릭 $\alpha_{s0}^0$와 0인 데이터가 다음 상태에서 현재 상태(첫 번째 상태)로 올 확률을 나타내는 $\beta_{s0}^0$의 합을 의미한다.

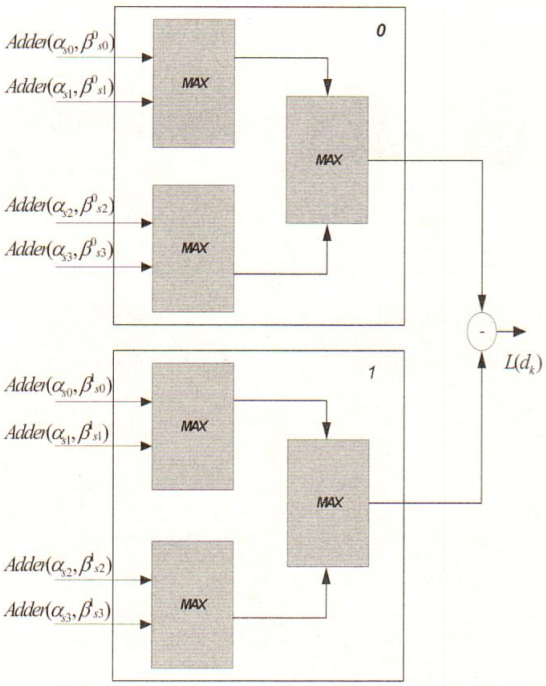



그림 3. LLR 계산 유닛의 구조.

4. 블록 MAP 복호 알고리듬

그림 4와 같이 블록 MAP 알고리듬은 N개의 정보열로 이루어진 입력 프레임을 길이가 L인 M개의 부블록(sub block)으로 나누어 부블록 단위로 복호를 하는 방식이다. 이 경우 MAP 복호와 비슷한 성능을 내기 위해 신뢰할 수 있는 역방향 메트릭 확률 값이 필요하기 때문에 이런 역방향의 초기값을 구하는 과정인 트레이닝 과정(training process)이 필요하다. 블록 0 에서 블록 $M-1$방향으로 계산이 진행되는 순방향 메트릭은 계속 연속적으로 수행을 하므로 이전 블록의 최종 메트릭 값을 현재 계산 블록의 초기값으로 사용하면 된다. 하지만 역방향의 경우 순서가 반대로 진행되기 때문에 초기값을 구하는 트레이닝 과정이 요구되어 3개의 상태 메트릭 프로세스가 필요하게 되며 계산량과 연산기의 증가를 초래하게 된다. 이러한 하드웨어의 낭비를 막기 위해 3단의 파이프라인 적용하여 3개의 프로세스를 하나의 프로세서로 수행함으로써 하드웨어의 활용도를 높인 구조가 제안되었다[3][6][12].

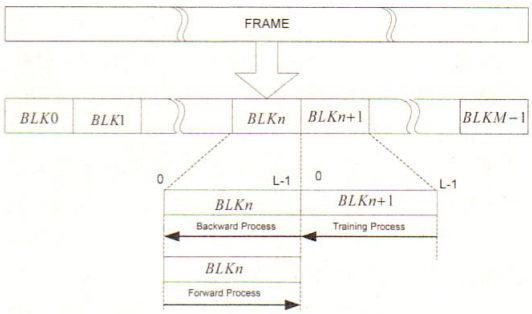



그림 4. 블록 MAP 복호 알고리듬.

## Ⅲ. 제안된 터보 복호기 구조

터보 코드 하드웨어 구현을 위해 일정한 성능내에서 메트릭이 갖는 최소 영역에 대한 연구가 진행되었다[13][14]. 하드웨어 구현시 복잡도를 줄이고 메모리의 요구량을 줄일 수 있도록 적합한 신호 단어 길이를 결정한 후, 전체적인 시스템 설계를 수행하며 정규화 과정은 고정된 신호 단어 길이를 갖는 터보 코드의 메트릭 연산 수행시 발생할 수 있는 오버 플로우를 방지하는 역할을 한다[7][15].

SISO 복호기에서 고정 신호 단어의 길이를 갖는 메트릭 값은 반복 복호 과정을 통해 계속 증가되기 때문에 오버 플로우가 발생하게 된다. 이를 피하기 위해서 로그 MAP 알고리듬에서 경판정 값이 메트릭의 절대적인 크기에 의해서가 아니라 상대적인 차에 의해서 이루어진다는 점을 이용해 모든 메트릭에 일정한 값을 빼주는 방식이 정규화 방식에 사용되었다[15][16][17][18]. 식 (3-1)은 최소값을 빼주는 방식을 이용한 정규화 방식을 나타내고 있다. 하지만 이런 뺄셈에 의한 정규화 방식은 부수적으로 일정한 값을 찾는 계산이 필요하며 많은 상태를 갖는 경우 계산의 복잡도를 증가시켜 전체 시스템의 속도를 저하시키는 결과를 초래한다.

$$\hat{\alpha}_k(s) = \alpha_k(s) - \min(\alpha_k(s')), \forall s$$

$$\hat{\beta}_k(s) = \beta_k(s) - \min(\beta_k(s')), \forall s \qquad \text{식 (3-1)}$$

이를 극복하기 위해 최근의 정규화 방식은 비터비 복호에 사용되는 모듈로 정규화 방식을 사용하고 있다. 비터비 복호에서 사용된 모듈로 정규화 방식

을 사용하기 위해서는 경로의 선택은 메트릭간의 차이에 의해 결정이 되고 경로 메트릭의 차이는 한정된다는 두가지 조건을 만족해야 한다[14]. SISO 복호기에서도 상태 메트릭 계산이나 경판정 출력을 내보내기 위한 계산 모두 메트릭간의 차이에 의해 결정이 되고 또한 모든 차이들이 제한되어 있음이 증명되어 모듈로 정규화 방식을 사용하게 되었지만, 일정한 값을 빼주는 방식에 비해 1~2비트의 추가적인 비트가 필요하다 [7][15][19].

본 절에서는 모듈로 정규화 방식에 비해 상태 메트릭의 고정된 비트열 길이를 줄일 수 있고, 최대값과 최소값을 빼주는 방식에 비해 계산 복잡도가 적은 정규화 방식의 구현을 제안한다.

1. 제안된 정규화 구조

MAP 복호기에서 상태 메트릭과 LLR 계산 유닛은 많은 연산량을 차지하고 있으며 블록 MAP 복호기에서는 트레이닝 연산을 추가하여 3개의 상태 메트릭 프로세스 연산을 수행하므로 상태 메트릭 블록과 LLR 연산 블럭의 연산량 감소나 메모리 접근 감소를 실현한다면 전력 소모면에서 큰 효과를 거둘 수 있다. 제안된 정규화 구조에서는 한 격자단에서 첫 번째단의 메트릭 값을 빼주는 방식의 정규화 방식을[20] 적용하였다.

$$\hat{\alpha}_k(s) = \alpha_k(s) - \alpha_k(s0), \forall s$$

$$\hat{\beta}_k(s) = \beta_k(s) - \beta_k(s0), \forall s \qquad \text{식(3-2)}$$

그림 5는 각 정규화 방식을 적용한 블록 MAP복호기내의 상태 메트릭의 범위를 보여주고 있다. 그림 5 (a)와 (b)는 각각 최소값과 평균값을 빼준 경우의 상태 메트릭 값의 분포를 보인다. 그림 5 (c)는 한 격자단에서 첫 번째 단의 상태 메트릭을 모든 상태 메트릭에 빼주는 방식은 같은 비트에서 평균값이나 최대값을 빼주는 경우보다 상태 메트릭의 값의 범위가 2배 이상 증가함을 보인다. 제안된 정규화 방식은 그림 5 (d)와 같다. 증가된 상태 메트릭의 값의 범위를 줄이기 위해 상태 메트릭 값이 일정값 이상일 경우 같은 격자단의 모든 상태 메트릭을 나누어주는 과정을 수행하며 이때 상태 메트릭 값의 분포는 그림 5 (c)에 비해 상태 메트릭의 값의 범위가 많이 줄어듦을 알 수 있다.

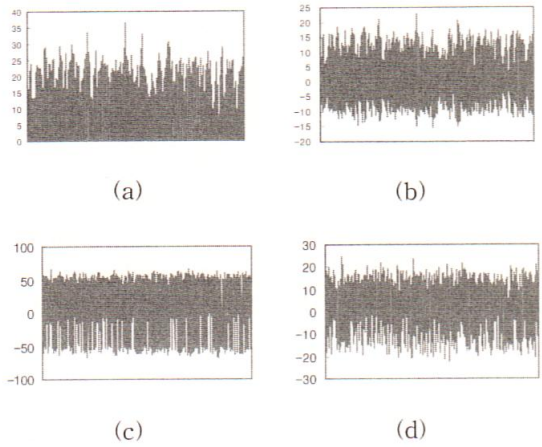



그림 5. 상태 메트릭 값의 범위. (a) 최소값을 빼는 정규화 방식의 경우[17], (b) 평균값을 빼는 정규화 방식의 경우, (c) 격자단에서 하나의 상태 메트릭값을 빼는 정규화 방식의 경우[20], (d) 제안된 방식의 경우.

그림 6은 정규화 기존의 정규화 방식을 적용한 경우와 제안된 정규화 방식을 적용한 경우의 터보 복호기의 유동 소수점 시뮬레이션과 고정 소수점 BER 성능 비교 시뮬레이션 결과이다. 유동 소수점 시뮬레이션 결과 격자단의 한 상태 메트릭 값을 빼주는 정규화 방식을 사용할 경우 반복 횟수가 증가할수록 BER 성능이 기존의 방식에 비해 우수함을 보인다. 모든 상태 메트릭에 평균값이나 최소값을 빼주는 기존의 방식은 일정한 성능 이상의 향상은 일어나지 않고 오히려 반복 횟수가 증가 할수록 성능 저하가 발생하고 있다. 하지만 격자단의 한 상태 메트릭 값을 같은 격자단의 모든 상태 메트릭에 빼주는 정규화 방식을 사용할 경우, 그림 5 (c)와 같이 상태 메트릭 값의 범위가 증가하게 되어 오버플로우의 발생 빈도를 줄이기 위한 추가적인 비트가 필요하게 된다. 제안된 정규화 방식은 같은 격자단(trellis stage)내에서 한 상태 메트릭이라도 일정값 이상일 경우 같은 격자단내의 모든 상태 메트릭을 나누어 주는 비교, 쉬프트 과정을 통해 상태 메트릭의 범위를 줄여 고정된 신호 단어 길이를 사용할 경우 발생하는 오버 플로우 빈도를 줄이도록 하

였으며 제안된 방식 역시 첫 번째 상태 메트릭을 빼준 경우와 같은 BER 성능을 보이고 있다. 제안된 방식의 고정 소수점 시뮬레이션의 성능 실험 결과 처리 신호 단어의 길이 고정과 쉬프트 연산 수행에 따른 BER 성능의 저하가 있지만 기존의 방식에 비해 성능 면에서 역시 우수함을 알 수 있다.

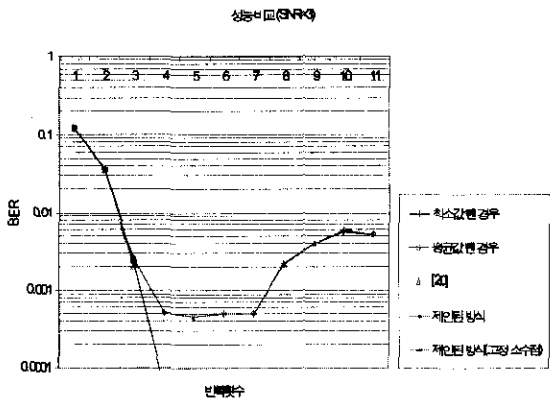



그림 6. 정규화 방식의 BER 성능 비교

그림 7은 제안된 정규화 유닛의 구조를 보인다. 첫 번째 상태의 상태 메트릭 $ST0$ 값을 모든 상태단의 상태 메트릭에 빼주는 방식을 사용한다. 이 경우 연산기의 복잡도는 줄어들게 되나 메트릭 값의 범위가 증가하게 되므로 기존의 정규화 과정이 적용된 고정 비트열의 길이를 가진 시스템에서 오버 플로우 발생이 증가하게 된다. 이를 방지하기 위해 추가적인 비트를 사용하지 않고 한 격자단의 모든 상태 메트릭의 사인 비트를 제외한 최상위 비트를 OR 연산으로 비교해 결과가 1일 경우 전체 상태 메트릭을 쉬프트 시키는 형태의 정규화 과정을 수행하도록 하였다. $S\hat{T}1$, $S\hat{T}2$, $S\hat{T}3$는 각 메트릭에 $ST0$ 값을 빼고 비교, 쉬프트 과정을 마친 메트릭 신호이다. 제안된 구조는 기존의 최소값이나 최대값 또는 평균값을 계산하는 블록을 제거할 수 있어 하드웨어 구현의 비용을 줄일 수 있고, 첫 번째의 상태 메트릭은 항상 0이라는 규칙이 있으므로 출력은 3개의 신호만 내보내도 되어 하나의 신호 라인을 제거할 수 있다. 계산에 필요한 신호의 감소는 상태 메트릭을 계산하는 블록과 LLR을 계산하는 블록의 구현 복잡도를 줄일 수 있고 상태 메트릭을 저장하는 메모리 사이즈도 줄일 수 있으며 비교 쉬프트 연산을 수행함으로써 오버 플로우 발생 빈도를 줄이기 위해 추가적으로 요구되는 비트를 줄일 수 있다.

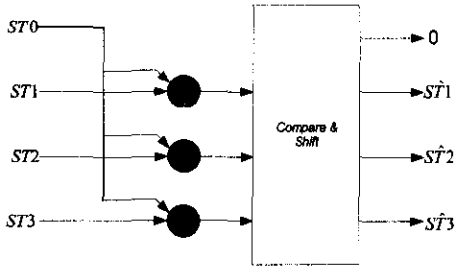



그림 7. 제안된 정규화 하드웨어 구조.

모듈로 정규화 방식은 최소값이나 최대값을 빼주는 방식에 비해 고속 동작이 가능 하지만, 최대값이나 최소값을 빼주는 방식에 비해 1~2 비트의 추가적인 비트가 필요하므로 메트릭을 계산하거나 저장할 때 많은 계산량과 메모리 크기가 요구되어 면적과 파워 소모면에서 비효율적인 단점이 있다 [15][19]. 제안된 구조는 최대값과 최소값을 찾는 과정을 생략하고 1~2 비트의 추가된 비트가 발생하더라도 비교 및 쉬프트 방식으로 요구되는 비트를 줄일 수 있으며 상태 메트릭 계산을 위한 메트릭의 수를 줄임으로써 연산량과 메모리 사용량 및 구현 복잡도를 크게 줄일 수 있다.

## 2. 제안된 정규화 구조를 적용한 터보 복호기

블록 MAP 복호를 수행하는데 있어 연속적인 데이터 처리를 위해 가지 메트릭 블록은 동시에 읽고 쓰는 기능이 수행되어야 하며, 각 상태의 가지 메트릭을 저장하기 위해 메모리 구조는 2포트 메모리의 4-뱅크 구조로 사용해야 하나 그림 2-6에서 보여지듯이 첫 번째 상태가 항상 0이므로 3-뱅크 구조로 구현해도 된다. 상태 메트릭의 경우도 첫 번째 상태의 메트릭 값이 모두 0이 되는 규칙이 있기 때문에, 3-뱅크 구조로 사용할 수 있어 메모리 요구

량을 획기적으로 줄일 수 있는 기존의 블록 MAP 구조보다 하나의 메모리 뱅크를 줄일 수 있다.

기존의 ACS 블록은 각각 두개의 상태 메트릭과 가지 메트릭을 이용해 연산을 수행하는 완전한 형태의 블록이 상태 수만큼 필요하지만 제안된 모델에서는 불필요한 연산을 최대한 줄이기 위해 3가지 유형으로 나눌 수 있다. 그림 8 (a)는 모든 메트릭이 0일 경우의 ACS 연산기이며 덧셈기 하나와 비교기가 생략된 구조이다. 그림 8 (b)는 각각 하나의 메트릭이 0일 경우의 ACS 블록이며 가지 메트릭과 상태 메트릭을 더하는 덧셈기 2개가 생략된 구조이다. 복호기는 그림 8에 제시된 두 가지 형태와 그림 2에 나타낸 기존의 ACS 블록을 모두 사용한다. 또한 LLR 계산 유닛에서도 모든 메트릭들을 고려하고 있는 기존의 구조와 비교 했을때 메트릭간의 매핑과정 복잡도를 줄이고 0이 나올 확률의 값을 계산할 경우 덧셈기 3개와 max 연산기 하나와 1이 나올 확률의 값을 계산할 경우 덧셈기 2개를 절약할 수 있다.

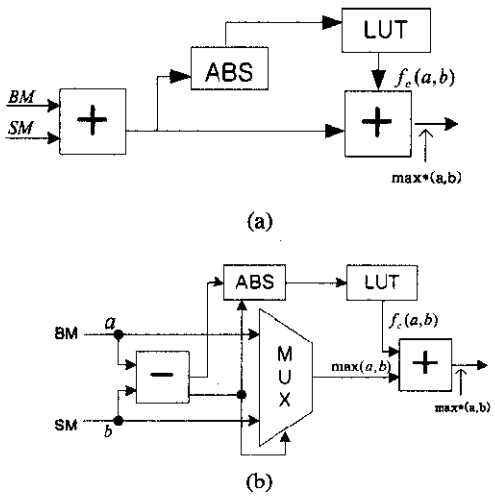



그림 8. 제안된 정규화 방식을 적용한 ACS 구조. (a)한 상태의 역방향 메트릭과 가지 메트릭이 모두 0인 경우, (b) 각 상태의 역방향 메트릭과 가지 메트릭중 하나가 0인 경우.

## Ⅳ. 실험 결과

본 장에서는 제안된 구조와 기존구조의 비교 실험을 위한 시뮬레이션 환경을 기술하고 파워 소모와 면적 감소량 그리고 메모리 요구량의 측정 결과를 제시한다.

### 1. 시뮬레이션 환경

터보 복호기의 시뮬레이션은 UNIX 환경의 C 언어를 사용하여 구현되어 있으며, VHDL로 구현된 모델은 Synopsys사의 Design Analyzer를 사용해 합성 과정을 수행 면적과 동적 파워(dynamic power)를 측정 했으며 하이닉스의 0.35 $\mu m$ 공정 라이브러리를 이용하였다. 기존의 방식은 한 블록의 크기를 256 정보 비트로 나눈 블록 MAP 복호 방식이며 최소값을 빼는 방식의 정규화 방식을 사용한 구조이다. 표 1은 시뮬레이션에 사용된 파라미터들을 보인다. 메모리의 면적과 파워는 UMC에서 제공하는 메모리 컴파일러를 이용해 측정 하였으며, 면적의 측정 단위는 2-input nand 크기를 1로 하는 게이트의 수를 의미한다.

표 1. 시뮬레이션 파라미터.

| 파라미터 | 값 |
|---|---|
| 구속장($K$) | 3 |
| 메모리 길이($V$) | 2 |
| 생성 다항식($g_1/g_2$) | 7/5 |
| 프레임 크기 ($N$) | 1024 |
| 인터리버 종류 | S Random |
| 블록 크기 | 256 |
| 부호율($R$) | 1/3 |
| 채널 변조 방식 | AWGN, BPSK |

### 2. 성능 분석

본 절에서는 제안된 정규화 방식이 적용된 블록 MAP 터보 복호기의 파워 소모 및 면적 감소를 중심으로 결과를 제시한다. 본 절에서는 기존 방식과 제안된 방식이 적용된 상태 메트릭과 LLR 계산 유닛의 면적과 파워 소모량 기존 방식과 제안된 방식이 적용된 전체 디코 복호기의 면적과 파워 소모를 비교한다.

2.1 연산 유닛 면적 및 파워 소모 감소량 비교

기존의 정규화 방식을 적용한 블록 MAP 터보 복호기의 상태 메트릭과 LLR 값을 계산하는 유닛이 전체 복호기 면적에서 차지하는 비율은 약 7% 이지만 블록 MAP 터보 복호기의 상태 메트릭 계산 유닛과 LLR 연산 유닛이 파워 소모에 차지하는 비율은 약 38%에 이른다. 제안된 정규화 방식을 적용한 블록 MAP 터보 복호기는 불필요한 연산들을 줄일 수 있으며 많은 파워 소모가 발생하는 LLR 계산 유닛과 상태 메트릭 유닛의 파워 소모를 효과적으로 줄일 수 있다. 표 2는 제안된 정규화 방식을 적용했을 때 상태 메트릭과 LLR값 계산 유닛의 면적과 파워 소모 감소량 비교를 보인다.

표 2. 상태 메트릭과 LLR 계산 유닛의 면적과 파워 소모 감소량 비교.

| | 연산유닛 | 기존방식 [17] | 제안된 방식 | 감소량 |
|---|---|---|---|---|
| 면적* | 상태 메트릭 계산 유닛 | 8,763.3 | 7,230.7 | 17.4% |
| | LLR 계산 유닛 | 3,691.1 | 3,148.7 | 14.6% |
| 파워 | 상태 메트릭 계산 유닛 | 13.6 mW | 8.5 mW | 37.5% |
| | LLR 계산 유닛 | 19.1 mW | 12.1mW | 36.6% |

*Unit gate : 2-input NAND

2.2 터보 복호기 면적과 파워 소모 감소량 비교

표 3은 제안된 정규화 방식을 적용했을 때 블록 MAP 터보 복호기의 메모리 부분과 연산기 및 제어기부의 면적과 파워 소모 감소량을 비교하였다. 메모리의 면적은 전체 터보 복호기 면적의 약 90%를 차지하며, 전력 소모면에서 약 60%를 차지하고 있어 메모리 요구량의 감소는 터보 복호기 면적과 파워 소모 감소에 큰 비중을 차지한다. 또한 연산기 및 제어기부의 파워 소모는 전체 복호기의 40%를 차지하므로 연산기의 파워 소모 감소 또한 터보 복호기의 파워 소모 감소에 큰 비중을 차지한다. 본 논문의 기존 방식을 적용한 블록 MAP 터보 복호기는 10K byte의 2포트 RAM과 10K byte의 싱글 포트 RAM을 사용하고 있으며 제안된 방식을 적용한 터보 복호기는 10K byte의 2포트 RAM과 8K byte의 싱글 포트 RAM을 사용한다. 제안된 방식을 적용한 블록 MAP 터보 복호기는 상태 메트릭 메모리 요구량 감소와 연산 유닛의 면적 감소에 의해서 면적이 6.6%가 감소하였고 메모리 요구량 감소와 연산을 수행하는 상태 메트릭 수의 감소로 인한 연산량 감소로 파워 소모는 17.9%까지 감소하였다.

표 3. 블록MAP복호기의 면적 및 파워감소량 비교.

| | | 기존 방식[17] | 제안된 방식 | 감소량 |
|---|---|---|---|---|
| 연산기 및 제어기 | 면적* | 36,616.7 | 32,733.1 | 10.6% |
| | 파워 | 67.5mW | 48.9mW | 27.6% |
| 메모리 | 면적* | 321,500.0 | 301,553.6 | 6.2 % |
| | 파워 | 105.5mW | 93.1mW | 11.8% |
| 전체 | 면적* | 358,116.7 | 334,286.7 | 6.6% |
| | 파워 | 173.0mW | 142.0mW | 17.9% |

*Unit gate : 2-input NAND

## V. 결론 및 추후 과제

블록 MAP 터보 복호 연산기의 상태 메트릭과 LLR 연산 유닛은 전체 복호기 면적의 약 7%를 차지하지만 많은 연산량이 요구되어 파워 소모면에서 차지하는 비율은 38%에 이른다. 또한 메모리는 터보 복호기내의 면적의 약 90%를 차지하고 있으며 60% 정도의 파워 소모를 한다. 제안된 구조는 상태 메트릭 계산시 한 격자단의 모든 상태 메트릭에 첫 번째 상태 메트릭을 빼주는 방식을 사용하며 오버플로우의 발생을 상태 메트릭의 사인 비트를 제외한 최상위 비트들의 OR 연산을 통해 비교 쉬프트

연산을 수행하도록 하여 오버 플로우 발생 빈도를 줄이고, 하나의 상태 메트릭이 항상 0이라는 규칙을 이용해 상태 메트릭과 LLR값 연산기의 계산 복잡도를 줄이고 메모리 요구량을 줄여 터보 복호기의 면적과 파워 소모를 줄였다. 제안된 정규화 구조를 적용한 블록 MAP 터보 복호기는 기존의 복호기와 비교했을때 파워 소모는 17.9%의 감소를 보이며 면적은 6.6%가 감소됨을 확인하였다.

추후 과제로 반복 연산량에 따른 복호 지연과 연산량 증가로 인한 전력 소모를 줄이기 위해 반복 복호에 관한 연구가 필요하며, 정규화 방식과 반복 횟수의 상관관계에 의한 저전력 아키텍처 연구가 필요하다.

## 감사의 글

본 연구는 한국과학재난 목적기초연구 지원에 의해 수행되었읍니다. (R01-2001-000-00321-0)

## 참 고 문 헌

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-correcting Coding and Decoding: Turbo-Codes(1)," in Proc. ICC, Geneva, Switzerland, pp. 1064-1070, May 1993.

[2] C. Shannon, "A Mathematical Theory of Information," Bell System Technical Journal, Vol. 27, pp. 379-423, July 1948.

[3] S. Barabulescu and S. Piebrobon, "Turbo Codes : A Tutorial on a New Class of Powerful Error Correcting Coding Schems, Part2 : Decdoer Design and Performance," IEEE Journal of Electrical and Electronics Engineering, Vol. 19, No. 3, pp. 143-152, Sept. 1999.

[4] Y. Wu and J. Ebel, "A Simple Stopping Criterion for Turbo Decoding", IEEE Comm. Letter, Vol. 4, pp. 258-260, Aug. 2000.

[5] A. Viterbi, "An Intuitive Justification and a Simplified Implementation of the MAP Decoder for Convolutional Codes," IEEE Journal on Selected Areas in Communications, Vol. 16, No. 2, pp. 260-264, Feb. 1998.

[6] G. Park, S. Yoon, I. Jin, and C. Kang, "A Block-wise MAP Decoder Using a Probability Ratio for Branch Metric" in Proc. VTC, Amsterdam, Netherlands, pp. 1610-1614, Sept. 1999.

[7] G. Masera and G. Piccinini, "VLSI Architectures for Turbo Codes," in IEEE Trans. on VLSI Systems, VOL. 7, No 3, pp. 369-379, Sept. 1999.

[8] Z. Wang, H. Suzuki, and K. Parhi, "VLSI Implementation Issues of Turbo Decoder Design for Wireless Applications", in Proc. IEEE Workshop on Signal Processing Systems, Taipei, Taiwan, pp. 503-512, Oct. 1999.

[9] S. Hong and W. Stark, "Design and Implementation of a Low Complexity VLSI Turbo-Code Decoder Architecture for Low Energy Mobile Wireless Communication," Journal of VLSI Signal Processing Systems, Vol. 24, pp. 43-57, Feb. 2000.

[10] G. Lee and S. Park, "Turbo Decoder Design for IS-2000 System," in Proc. VTC, pp. 412-415, Sept. 2000.

[11] J. Cheng and T. Ottosson, "Linearly Approximated Log-MAP Algorithm for Turbo coding," in Proc. VTC, pp. 2252- 2256, May 2000.

[12] G. Park and S. Yoon, "An Implementation Method of a Turbo-code Decoder using a Block wise MAP Algorithm," in Proc. VTC, pp. 2956-2961, May 2000.

[13] Y. Wu and B. Woerner, "The Influence of

Quantization and Fixed Point Arithmetic upon the BER Performance of Turbo Codes, " in Proc. VTC, Houston, TX, pp. 1683-1687, May 1999.

[14] G. Jeong and D. Hsia, "Optimal Quantization for Soft-decision Turbo Decoder," in Proc. VTC., Amsterdam, Netherlands, pp. 1620-1624, Sept. 1999.

[15] Y. Wu and D. Brian, "Data Width Requirements in SISO Decoding with Modulo Normalization," IEEE Trans. Comm., Vol. 49, No. 11, pp. 1861-1868, Nov. 2001.

[16] Z. Wang and H. Suzuki, "Finite Wordlength Analysis and Adaptive Decoding for Turbo/MAP Decoder," in Journal of VLSI Signal Processing 29, 209-221, Nov. 2001.

[17] Z. Wang and H. Suzuki, "VLSI Implementation Issues of Turbo Decoder Design for Wireless Applications," in Proc. IEEE Workshop on Signal Processing Systems, pp. 503-512, 1999.

[18] J. Harrison, "Implementation of a 3GPP Turbo Decoder on a Programmable DSP Core," Communication Design Conference, San Jose, CA, Oct. 2001.

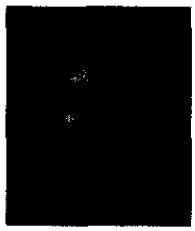
[19] A. Worm and H. Michel, "Advanced Implementation Issues of Turbo-Decoders," in Proc. 2nd International Symp. on Turbo Codes & Related Topics, pp. 351-354, Sept. 2000.

[20] M. Valenti and J. Sun, "UMTS Turbo Code and an Efficient Decoder Implementation Suitable for Software-Defined-Radios," International Journal of Wireless Information Networks, Vol. 8, No. 4, pp. 203-215, Oct. 2001.

황 선 영 (Sun-Young Hwang) 정회원

1976년 2월 : 서울 대학교 전자공학과 졸업
1978년 2월 : 한국 과학원 전기 및 전자공학과 공학석사 취득
1986년 10월 : 미국 Stanford 대학 전자공학 박사학위 취득
1976~1981 : 삼성 반도체 주식회사 연구원, 팀장
1986~1989 : Stanford 대학 Center for Integrated System 연구소 책임 연구원
Fairchild Semiconductor Palo Alto Reaserch Center 기술 자문
1989~1992 : 삼성전자(주) 반도체 기술 자문
1989년 3월~현재 : 서강대학교 전자공학과 교수
2002년 4월~현재 : 서강대학교 정보통신대학원장
<주관심분야> SoC 설계 및 framework 구성, CAD 시스템, Computer Architecture 및 DSP System Design 등

김 식 (Sik Kim) 정회원



1994년 2월 : 서강대학교 전자공학과 졸업
1996년 2월 : 서강대학교 전사공학과 석사학위 취득
1996년 3월~10월 : 현대전자(주) 연구원
2003년 8월 : 서강대학교 전자공학과 박사학위 취득
2003년 3월~현재 : 삼성전자 System LSI 사업부 CAE센터 책임연구원
<관심분야> 고속 저전력 digital 하드웨어 구조 설계, SoC 설계 및 CAD 시스템 등

문 제 우 (Je-Woo Moon) 정회원

2000년 2월 : 홍익대학교 전자공학과 졸업
2003년 2월 : 서강대학교 전자공학과 석사학위 취득
2003년 3월~현재 : (주) 팬택 중앙 연구소 연구원
<주관심분야> Channel Coding, SoC, OFDM, DMB