논문 2004-41SD-9-6

이중 정지 기준을 사용한 저 전력 터보 디코더 설계 기술

(Low Power Turbo Decoder Design Techniques Using Two Stopping Criteria)

임 호 영*, 강 원 경*, 신 현 철**, 김 경 호***

(Hoyeong Lim, Wonkyung Kang, Hyunchul Shin, and Kyungho Kim)

요 약

최근 3세대 이동통신의 오류정정부호의 표준으로 채택된 터보 코드는 샤논의 한계에 가까운 성능을 보이며, 반복적인 디코딩 과정의 특성상 이동통신 시스템에서 전력 소모가 많은 블록 중 하나이다. 따라서 이동통신 기기의 전력 소모를 최소화하기 위한 노력으로 터보 디코더의 전력 소모를 줄이는 방법들에 대한 연구가 진행되어왔다. 본 논문에서는 디코딩 가능 임계값과 불가능 임계값 등 두개의 정지 기준을 적용함으로써 기존의 반복 디코딩 정지 기준 알고리즘을 개선하여, 오류정정 성능과 전력 소모면에서 기존의 방법보다 효율적인 새로운 터보 디코더 기술을 개발하였다. 실험 결과, 제안한 방법은 기존의 대표적인 방법[3]에 비하여, 전체 12500회 실험 중 잘못된 오류정정 횟수는 평균적으로 89% 감소시키고 반복 디코딩 횟수는 29% 감소시킬 수 있었다.

Abstract

Turbo codes, whose performance in bit error rate is close to the Shannon limit, have been adopted as a part of standard for the third-generation high-speed wireless data services[6]. Iterative Turbo decoding results in decoding delay and high power consumption. As wireless communication systems can only use limited power supply, low power design techniques are essential for mobile device implementation. This paper proposes new effective criteria for stopping the iteration process in turbo decoding to reduce power consumption. By setting two stopping criteria, decodable threshold and undecodable threshold, we can effectively reduce the number of decoding iterations with only negligible error-correcting performance degradation. Simulation results show that the number of unsuccessful error-correction can be reduced by 89% and the number of decoding iterations can be reduced by 29% on the average among 12500 simulations when compared with those of an existing typical method[3].

Keywords: Low power, turbo decoder, iterative decoding, wireless design

I. 서 론

최근에 활발하게 연구되고 있는 차세대 멀티미디어 이동 통신에서는 개인을 위한 이동 전화, 무선 호출, 데이터 통신, 위성 통신 등 다양한 시스템을 통합하여 언제, 어디서나, 누구와도 통신이 가능하도록 국제 로밍(roaming)을 제공하고자 한다. 또한 음성, 데이터, 영상 등의 다중 정보 서비스를 통합 제공하는 것을 목표로 하고 있다. 이와 같이 이동 통신에서 멀티미디어 통신 서비스를 지원하기 위해서는 고속 전송과 다양한 종류의 데이터 전송이 요구되므로 채널 및 데이터의 종류에 따라 서로 다른 부호화 기법을 사용하여 시스템의 효율을 높이는 것이 필수적이다. 디지털 이동 통신 시스템에서는 음성, 영상, 데이터 등의 정보를 전송할 때 잡음, 간섭, 그리고 페이딩 등으로 인해 불가피하게 오류가 발생하여 정보의 손실이 생길 수 있다. 이러한 오류를 적절히 극복하여 시스템의 신뢰도를 높이기 위해서는 오류제어기법 (error-control techniques)을 도입하는 것

* 학생회원, ** 정회원, 한양대학교 전자컴퓨터공학부
(School of Electronics & Computer Engineering Hanyang University)
*** 정회원, 삼성전자 정보통신총괄 통신연구소
(Modem Team, Telecom R&D, TN Network Samsung Electronics)
접수일자: 2004년1월19일, 수정완료일: 2004년8월30일

이 필수적이다. 채널의 성격에 따라 오류제어기법은 여러 가지 형태로 변형될 수 있으나, 가장 기본적인 방법은 오류 정정 부호 (error-correcting code)를 사용하는 것이다. 이러한 부호들을 체계적으로 연구하는 부호이론은 지난 수십 년 간 눈부신 발전을 해왔다.

최근에 관심이 고조되고 있는 이동통신시스템을 위한 고신뢰 채널 부호기술에는 길쌈부호 (convolutional code)를 사용하는 연접부호 (concatenated code) 중에서 반복 디코딩 기법을 이용하여 샤논의 한계 (Shannon limit)에 근접하는 성능을 갖는 터보 코드와 기존의 비터비-리드솔로몬 (Viterbi-Reed Solomon)의 연접된 길쌈부호가 대표적이다. 특히 1993년 Berrou 등이 발표한 터보 코드[1]는 부호 이론의 역사상 중요한 발전이었다. 터보 디코더는 현재 3세대 이동통신의 표준이며, 이동통신의 특성상 저 소모 전력을 위한 터보 디코더에 대한 연구가 진행되고 있다.

본 연구에서는 두 개의 정지 기준 임계값을 사용한 새로운 저 전력 터보 디코더를 제안 하였으며 모의실험을 통하여 제안한 방법의 성능이 기존의 단일 정지 기준방법[3]에 비하여 크게 우수함을 보였다. 새로운 방법의 장점은 신호대 잡음비가 우수 할 때에는 디코딩 가능 임계값이 불필요한 디코딩 반복횟수를 줄여주며, 신호대 잡음비가 나쁠 때에는 디코딩 불가능 임계값이 디코딩 반복횟수를 줄여서 소모 전력을 감소시킨다는 것이다. Ⅱ장에서는 터보 인코더 및 디코더의 구조와 MAP 디코딩 알고리즘을 설명한다. Ⅲ장에서 기존의 저 전력 터보 디코딩 기술을 소개한다. Ⅳ장에서는 새로운 저 전력 터보 디코더 설계 알고리즘을 제시하였다. Ⅴ장에서는 실험결과를 보여준다. 끝으로, Ⅵ장에서 결론을 맺는다.

Ⅱ. 터보 코딩

3세대 이동통신 표준[4]에 따른 터보 인코더와 디코더를 간략히 살펴본다.

1. 터보 인코더

입력 비트 수를 N이라 하고 지원하는 부호율 (code rate)을 R이라 하자 (예, R=1/2 ,1/3, 1/4). 터보 인코딩에서는 N/R의 부호화 된 데이터의 출력 심볼 (output symbol)을 생성한 후, 그 다음 6/R의 꼬리 출력 심볼 (tail output symbol)을 생성한다. 즉, 모두 (N+6)/R의 출력 심볼을 생성한다. 터보 인코더의 기본구조는 그림

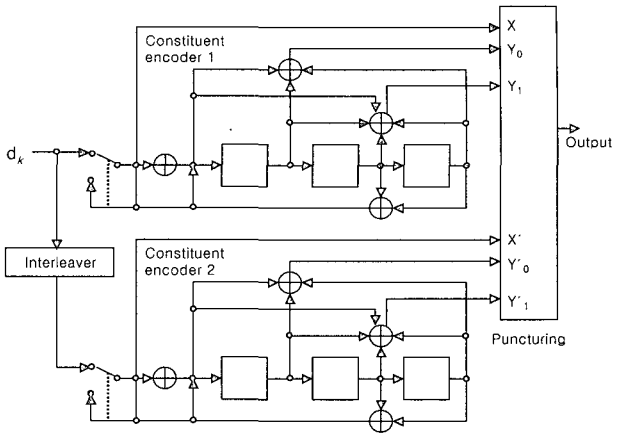


그림 1. 터보 인코더
Fig. 1. Turbo encoder architecture.

1과 같이 두 개의 귀환 길쌈 인코더 (recursive convolu-tional encoder)와 연접 오류 (burst error)를 제거하기 위한 인터리버로 구성되어 있다. 첫 번째 구성 인코더는 정보 비트 입력 순서 그대로 인코딩을 수행하지만, 두 번째 구성 인코더는 터보 인터리버에 의해 인터리빙된 순서대로 인코딩을 수행한다. 그리고 원하는 부호율 R의 출력 심볼을 얻기 위해서 두 개의 구성 인코더의 출력이 천공 (puncturing) 과정을 통과하여 출력된다. 인코딩 결과 입력비트 d_k에 대한 입력정보비트 X_k와 잉여비트 Y_k를 얻게 된다. 이러한 터보 인코더의 출력은 변조되어 채널을 통해 전송된 뒤 수신단에서 복조 (demodulation) 되어 터보 디코더의 입력으로 인가된다.

2. Maximum A-Posteriori (MAP) 알고리즘

터보 디코딩 알고리즘으로는 MAP 알고리즘, Log-MAP 알고리즘, Soft Output Viterbi Algorithm (SOVA)이 널리 사용되고 있다. MAP 알고리즘은 디코딩 성능은 좋으나 구연상의 복잡도가 높고, 전력 소모도 높다[6]. Log-MAP 알고리즘은 MAP알고리즘과 유사하나, 계산상의 복잡도를 낮추기 위해 MAP 알고리즘의 곱셈연산을 log를 사용하여 덧셈연산으로 계산하는 방법이다. SOVA는 계산상의 복잡도가 MAP알고리즘의 약 1/4이며, 구현이 용이하여 많이 사용되고 있다. 본 연구에서는 가장 성능이 좋으며, 전력소모도 높은 MAP 알고리즘을 사용하였다.

터보 디코더의 구조는 그림 2와 같이 두 개의 soft input soft output (SISO) MAP 디코더와 인터리버, 디인터리버 등으로 구성되어 있다. 그림 1에 나타냈듯이 인코더에서 시간 k일때 입력 d_k에 대한 출력 시퀀스 (output sequence)는 X_k와 Y_k로 이루어진다. 이 두 출

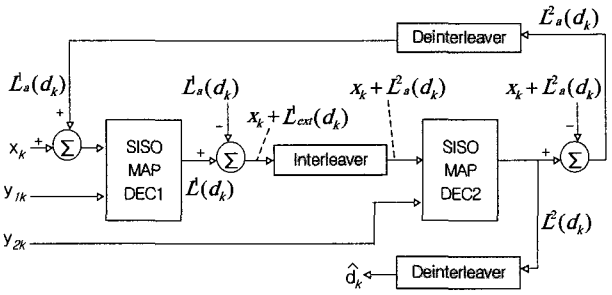


그림 2. 터보 디코더
Fig. 2. Turbo decoder architecture.

력 시퀀스가 Additive White Gaussian Noise (AWGN) 채널을 통과하고 이진 변조(binary modulation) 되었을 때, 그림 2의 디코더 DEC1과 DEC2의 수신심볼 R_k는 식 (2)와 같이 두 랜덤 변수 x_k와 y_k의 쌍으로 표현할 수 있다[1]. p_k와 q_k는 독립적인 잡음성분으로서 평균이 0이고 분산이 σ^2 인 가우시안 랜덤 변수이다. 결과적으로 채널을 통해 디코더로 수신되는 시퀀스는 식(1)의 R_1^N과 같다.

$$R_1^N = (R_1, \cdots R_k, \cdots, R_N) \tag{1}$$

$$R_k = (x_k, y_k) \tag{2}$$

$$x_k = (2d_k - 1) + p_k$$
$$y_k = (2Y_k - 1) + q_k \tag{3}$$

MAP 디코딩 알고리즘은 복잡도가 높은 알고리즘으로 구현상의 어려움이 있다[5]. 각 SISO 디코더 (DEC1과 DEC2)는 향상된 오류정정 인자인 외부추가정보 (extrinsic information) $L^i_{ext}(d_k)$ (i=1, 2)를 출력한다. 이 외부추가정보 $L^i_{ext}(d_k)$는 반복 디코딩 과정에서 다음 SISO 디코더의 사전정보 (a-priori information) $L^i_a(d_k)$ (i=1, 2)로 입력된다. 그리고 디코더의 마지막 입력은 채널 신뢰정보 $L_c(x_k)$이다. i번째 SISO 디코더의 출력은 입력 시퀀스가 1 또는 0으로 디코딩 될 가능성을 나타내는 연판정 (soft decision) 결과값 Log Likelihood Ratios (LLRs) $L_i(d_k)$이다. LLR은 식(4)와 같다[6].

$$L(d_k) = ln \frac{P(d_k = 1)}{P(d_k = 0)} \tag{4}$$

$L(d_k) \approx 0$ 인 것은 $P(d_k=1) \approx P(d_k=0) \approx 0.5$를 뜻하고 $L(d_k) \gg 0$ 인 것은 $P(d_k=1) \gg P(d_k=0)$으로 d_k=1로 디코딩 될 가능성이 높음을 뜻한다. 이와 같은 원리로 $L(d_k) \ll 0$ 일때는 d_k=0으로 디코딩 될 가능성이 높다.

첫 번째 디코더의 출력은 식 (5)와 같이 나타낼 수 있고 채널 신뢰정보 $L_c(x_k)$는 가우시안 채널이라고 가정할 때 식 (6)와 같이 정의할 수 있다[1].

$$L^1(d_k) = L_c(x_k) + L^1_a(d_k) + L^1_{ext}(d_k) \tag{5}$$

$$L_c(x_k) = \frac{2}{\sigma^2} x_k \tag{6}$$

첫 번째 반복 디코딩 과정에서 첫 번째 디코더의 사전정보는 $L^1_a(d_k)$=0으로 초기화 한다. 두 번째 디코더의 출력은 식 (7)과 같이 나타낼 수 있다.

$$L^2(d_k) = L_c(x_k) + L^2_a(d_k) + L^2_{ext}(d_k) \tag{7}$$

이 때, 두 번째 디코더의 사전정보는 첫 번째 디코더의 외부추가정보와 같고 ($L^2_a(d_k)=L^1_{ext}(d_k)$) 다음 반복 디코딩 과정의 첫 번째 디코더의 사전정보는 이전 반복 디코딩 과정의 두 번째 디코더의 외부추가정보와 같다 ($L^1_a(d_k+1)=L^2_{ext}(d_k)$). 이러한 반복 디코딩 과정을 수행하면서 반복 디코딩이 끝났을 때 식 (8)의 $L(d_k|x_k)$의 함수를 거쳐서 0 또는 1의 결과가 디코딩 된다.

$$L(d_k | x_k) = ln \frac{P(d_k = 1 | R_1^N)}{P(d_k = 0 | R_1^N)} \tag{8}$$

두 SISO MAP 디코더의 출력은 디코딩 된 각각의 비트의 오류 확률의 수치를 나타내므로 디코딩 과정을 반복할수록 두 디코더 사이의 외부추가정보 교환 즉, 외부추가정보를 다른 디코더의 사전정보로 입력하는 과정을 통하여 향상된 오류정정 성능을 얻게 된다.

III. 기존의 저 전력 터보 디코딩 기술

터보 코드가 처음 발표되었을 때는 하드웨어 성능보다는 부호화 이론에서의 오류 정정 성능에 초점을 맞추었기 때문에 정해진 반복 디코딩 횟수만큼 디코딩 과정을 수행하였다. 그러나 휴대용 이동통신 기기의 발달과 보급확대로 인해 하드웨어의 지연시간 감소와 소모 전력 감소를 위한 연구가 요구되고 있다. 현재까지 진행된 연구에서는 터보 디코더의 SISO 구조의 특성을 이용하여 반복 디코딩 횟수를 조정함으로써 지연시간과 소모 전력을 감소시켰다[2,3].

SISO 디코더에서 출력된 정보 비트에 대한 LLR 값과 임계값을 비교하여 반복 디코딩 횟수를 결정하는 방법은 기본적으로 터보 디코더에서 사용할 수 있는 계산 방법이다. 기존의 연판정 정지 기준 방법은 임계값을

사용하여 디코딩을 중단하는 방법[3]과 반복 디코딩의 LLR 값과 전 단계의 반복 디코딩 LLR 값의 차이를 임계값과 비교하여 반복 디코딩 여부를 결정하는 방법[2]이 있다.

첫 번째 방법은 채널 환경에 따라 임계값을 적용하여 디코딩을 계속 반복할지 여부를 판단하는 판정 모듈(decision module)을 추가함으로써 유연성 있게 반복 디코딩을 수행하는 방법이다[3]. 그림 3은 일반적인 터보 디코더에 판정 모듈과 간단한 버퍼를 삽입하여 반복 디코딩 과정을 결정하는 구조를 나타낸다.

판정 모듈은 반복 디코딩 과정의 종료 시점을 결정한다. 2개의 MAP 디코더의 출력을 입력으로 받아 판정 모듈에 저장되어 있는 임계값과 비교한다. 두 MAP 디코더에서 출력된 정보 비트에 대한 LLR 값 중 하나라도 임계값을 초과한다면 반복 디코딩 과정을 끝내게 된다. 첫 번째 디코더에서 출력된 정보 비트에 대한 LLR 값이 임계값을 초과할 경우, 첫 번째 디코더의 출력값인 LLR 값은 경판정 (hard decision) 블록으로 인가되어 터보 디코더의 출력으로 나오게 된다. 두 번째 디코더에서 출력된 정보 비트에 대한 LLR 값이 임계값을 초과할 경우, 두 번째 디코더의 입력값이 인터리버를 통과하여 인터리빙 되었기 때문에 두 번째 디코더의 출력값은 경판정 블록으로 인가되기 전에 디인터리버(deinterleaver)를 통과해야 한다. 만약 두 MAP 디코더에서 출력된 정보 비트에 대한 LLR 값이 임계값보다 작을 경우, 다음 반복 디코딩 과정을 계속 수행한다.

이러한 반복 디코딩 횟수 조절 방법에 있어서 임계값을 정하는 것이 터보 디코더의 성능을 결정짓는다. 따라서 정확한 임계값을 결정하는 것이 무엇보다 중요하다. 임계값을 정하기 위해 오프라인 상에서의 LLR 값의 평균이나 정규화 된 분산, 그리고 디코더 출력의 BER을 파라미터로 정하고 모의실험을 여러 번 수행하여 임계값을 얻는다. 모의실험을 통해 각 채널 환경에 따른 LLR 값의 평균치를 임계값으로 정한다[3].

그림 3의 터보 디코더 구조에서 실험으로 정해진 임계값과 각 디코더의 출력 결과 얻어진 LLR 값을 비교하여 LLR 값이 임계값보다 크거나 같은 경우 경판정 모듈에서 LLR 값의 부호에 따라 데이터를 '0' 또는 '1'로 경판정하게 된다.

두 번째 연판정 정지 기준 알고리즘은 현재 반복 디코딩의 LLR 값과 전 단계의 반복 디코딩 LLR 값의 차이를 임계값과 비교하여 반복 디코딩 여부를 결정하는 것이다[2]. 즉, n 번째 반복 디코딩의 LLR 값에서 (n-1)

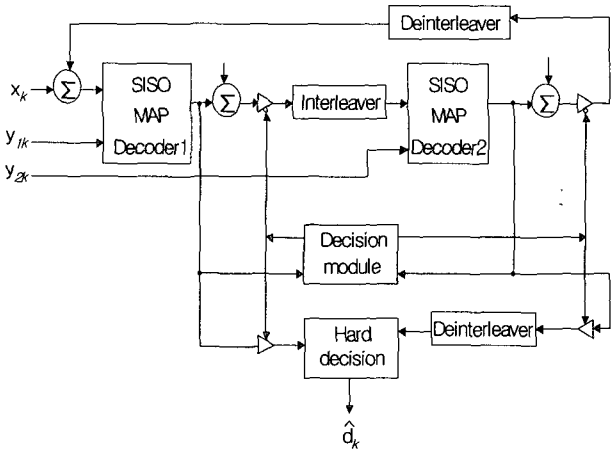


그림 3. 판정 모듈을 포함한 터보 디코더

Fig. 3. *Turbo decoder architecture including a decision module.*

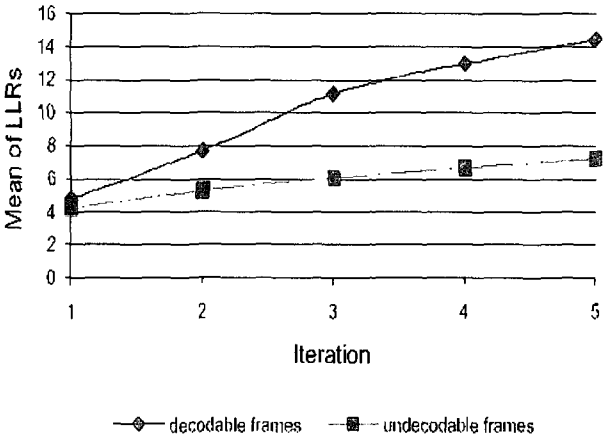


그림 4. SNR=-2.0dB일 때의 디코딩 가능 프레임과 불가능 프레임의 평균 LLR 값[2]

Fig. 4. *Mean of LLRs for decodable and undecodable frames at SNR=-2.0dB.*

번째 반복 디코딩의 LLR 값을 뺀 값을 정해진 임계값과 비교한다. 이 때, 그림 4에 나타냈듯이 디코딩이 가능한 데이터 프레임의 경우에는 반복 디코딩 과정을 수행할수록 LLR 값의 증가폭이 커지지만 디코딩이 불가능한 오류율이 높은 데이터 프레임의 경우에는 여러 번 디코딩 과정을 반복하더라도 LLR 값의 증가폭이 디코딩이 가능한 프레임에 비해 크게 작은 것을 실험을 통해 알 수 있다[2]. 따라서 식 (9)과 같이 현재 디코더에서 출력된 정보 비트에 대한 LLR 값이 임계값보다 작거나 또는 i 번째 반복 디코딩의 LLR 값에서 (i-1) 번째 반복 디코딩의 LLR 값을 뺀 값이 n 번째 반복 디코딩의 임계값 $th(i)$에서 (i-1) 번째 반복 디코딩의 임계값 $th(i-1)$을 뺀 값보다 작은 경우 디코딩이 불가능한 프레임으로 판정하여 반복 디코딩 과정을 끝마치게 된다.

$$|\dot{L}(d_k)| < th_1(i) \text{ or}$$
$$|\dot{L}(d_k)| - |\dot{L}(d_{k-1})| < th_1(i) - th_1(i-1) \quad (9)$$

그러나 디코딩 결과 얻은 LLR 값의 차이가 프레임에 따라 초기 반복 디코딩 과정에서 임계값보다 작은 경우에도 최대 반복 디코딩 횟수 내에 디코딩이 가능할 수도 있다. 만약 디코딩이 가능한 프레임의 i 번째 디코딩 결과와 $(i-1)$ 번째 디코딩 결과의 차이가 실험을 통해 정해진 평균적인 LLR 값의 임계값 기준보다 작다면 다음 반복 디코딩 과정을 통해 디코딩이 가능할지라도 현재 반복 디코딩 과정에서 디코딩을 종료하기 때문에 디코딩이 불가능한 프레임으로 잘못 판정할 가능성이 존재한다. 반복 디코딩 과정에서 디코딩이 불가능한 데이터 블록의 LLR 값의 합의 평균보다 디코딩이 가능한 데이터 블록의 LLR 값의 합의 평균이 더 작은 경우도 발생할 수 있기 때문에 식 (9)와 같이 반복 디코딩 횟수를 조절하는 것은 오류 정정 성능을 저하시킬 가능성이 있다.

Ⅳ. 새로운 반복 정지 방법

이동 통신에서는 채널 환경의 영향으로 데이터 손상이 심할 경우 하드웨어에서 정해 놓은 최대 반복 디코딩 횟수만큼 디코딩을 수행하더라도 디코딩이 불가능한 데이터 블록이 발생할 수 있다. 디코딩이 가능한지 여부만을 판단하는 기존의 단일 임계값을 이용한 연판정 정지 기준 알고리즘을 터보 디코더에 적용한다면 디코딩 불가능 데이터 블록은 디코딩이 되지 않더라도 최대 반복 디코딩 횟수만큼 디코딩을 수행해야한다. 따라서 채널 환경이 나쁠 경우 오류정정을 완벽하게 수행할 수 없음에도 불구하고 불필요하게 반복 디코딩 과정을 수행함으로써 불필요한 전력을 소모하게 된다.

기존 연판정 정지 기준 알고리즘의 이러한 단점을 개선하기 위하여 본 연구에서는 디코딩이 가능한 데이터 블록과 디코딩이 불가능한 데이터 블록에 대하여 각각 SISO 디코더의 동작 결과로 얻게 되는 LLR 값의 임계값을 설정하는 방법을 연구하였다. 또한 기존의 방법이 모든 반복 디코딩 과정에 대하여 하나의 임계값을 설정한데 비하여 매 반복 디코딩 과정마다 디코딩 가능 임계값과 디코딩 불가능 임계값을 설정함으로써 보다 최적화된 반복 디코딩 횟수를 찾아낼 수 있도록 하였다.

그림 5는 디코딩이 가능한 데이터 블록과 디코딩이 불가능한 데이터 블록의 LLR 값에 대한 임계값을 표시하였다. LLR 값은 디코딩의 결과가 '0' 또는 '1'이 될 가능성을 나타내므로 LLR의 절대값 (|LLR|)이 '1'에 가까울수록 디코더의 입력비트가 '0' 또는 '1'로 디코딩이 될 가능성이 높음을 뜻한다. 반면에 |LLR|이 '0'에 가까울수록 입력비트가 디코딩이 불가능할 가능성이 높다. 따라서 디코딩이 불가능한 데이터 블록이 디코딩이 가능한 데이터 블록에 비해 |LLR|이 작다. 또한 각 반복 디코딩 과정에서 얻어진 |LLR|의 합을 누적시킨다면 디코딩이 가능한 데이터 블록과 불가능한 데이터 블록의 |LLR|의 차이는 더욱 커지게 된다.

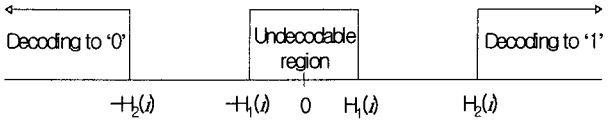


그림 5. 디코딩이 가능한 데이터 블록과 디코딩이 가능하지 않은 데이터 블록의 임계값 (H1: 디코딩 불가능 임계값, H2: 디코딩 가능 임계값)

Fig. 5. Decodable data threshold (H2) and undecodable data threshold (H1).

이때 디코딩 과정에서 디코딩 불가능 임계값 H_1과 디코딩 가능 임계값 H_2를 정하기 위해 각각 다른 기준을 적용해야 한다. 실험결과에 따르면 연접오류와 같은 요인으로 5회 미만의 반복 디코딩 초기 단계에는 디코딩을 성공한 데이터 블록과 성공하지 못한 데이터 블록의 |LLR|의 합의 차이가 크지 않기 때문에 디코딩 불가능 임계값 H_1을 초기 반복 디코딩 과정부터 설정하는 것은 오류정정 성능을 크게 떨어뜨릴 수 있다. 따라서 H_1은 많은 실험을 통해 디코딩을 성공한 데이터 블록과 성공하지 못한 데이터 블록의 |LLR|의 합의 차이가 K 배 이상 차이 나는 반복 디코딩 횟수부터 적용한다. 이때 K가 작을수록 오류정정 성능이 저하된다.

디코딩 가능 임계값 H_2는 디코딩을 성공한 데이터 블록들의 각 디코딩 결과 얻은 |LLR|의 합의 평균을 이용하여 매 반복 디코딩 과정에 적용한다. 또한 오류정정 성능의 저하를 줄이기 위해 cyclic redundancy checking (CRC)을 동시에 수행하여 디코딩 결과 |LLR|의 합이 H_2보다 크고 CRC 체크가 성공하면 반복 디코딩 과정을 정지한다. 결론적으로 최대 반복 디코딩 횟수를 *maxiter*, H_1이 적용되기 시작하는 반복 디코딩 횟수를 T_1, i 번째 반복 디코딩 과정의 j 번째 SISO 디코더의 임계값 $H_1^{(i,j)}$, $H_2^{(i,j)}$ 라고 했을 때 터보 디코더는 다음의 식 (10)와 (11)을 만족할 때 반복 디코딩 과정을 정지한다.

$$H_1^{(i,j)} \geq \sum_i |L(d_k)|, i \geq T_1 \quad (10)$$

$$H_2^{(i,j)} \leq \sum_i |L(d_k)| \quad (11)$$

and $CRC = TRUE$,
$0 \leq i \leq maxiter$, j=1, 2
for i-th iteration

j 번째 디코더의 사전정보를 $L^j_a(d_k)$, 현재 반복 디코딩 횟수를 i, j 번째 디코더 (j=1, 2)를 통과한 후 결과로 나온 |LLR|의 합을 $L_{sum}^{(i,j)}$,이라고 했을 때, 그림 5에 나타낸 두 개의 임계값을 이용하여 반복 디코딩 횟수를 정하는 알고리즘은 다음과 같다.

```
Algorithm : Stopping decision
Set the initial a-priori value L^1_a(d_k)=0;
Set the number of iterations i=0;
Set stop_decode=FALSE;

for (i=1; i≤maxiter; i++) {

 /* only apply H2 */
 if (i < T1) {
  for (j=1; j≤2; j++) {
     if ((L_sum^(i,j) ≥ H2^(i,j)) and
        (CRC == TRUE)) {
        stop_decode=TRUE;
        break;
     }
  }
 }

 /* apply both H1 and H2 */
 else if (T1 ≤ i ≤ maxiter) {
  for (j=1; j≤2; j++) {
     if((L_sum^(i,j) ≥ H2^(i,j)) and
       (CRC == TRUE)) {
        stop_decode=TRUE;
        break;
       }
     else if(L_sum^(i,j) ≤ H1^(i,j)) {
      /* undecodable data block */
        stop_decode=TRUE;
        break;
     }
  }
 }
 if (stop_decode==TRUE) {
    break;
 }
```

즉, H_2만을 적용시킬 T_1 이전 반복 디코딩 과정에서는 CRC 체크에 성공하고 |LLR|의 합 $L_{sum}^{(i,j)}$이 디코딩 가능 임계값 H_2보다 크거나 같으면 반복 디코딩 과정을 정지한다. T_1 이후 반복 디코딩 과정에서는 CRC 체크에 성공하고 |LLR|의 합 $L_{sum}^{(i,j)}$이 디코딩 가능 임계값 H_2보다 크거나 같으면 디코딩이 성공적으로 끝난 것으로 판단하여 반복 디코딩 과정을 정지하고, $L_{sum}^{(i,j)}$이 디코딩 불가능 임계값 H_1보다 작거나 같으면 디코딩이 불가능한 것으로 판단하고 반복 디코딩 과정을 정지한다.

V. 실험 결과

C 언어를 이용하여 터보 인코더와 디코더를 모델링하였다. 각 블록당 실험 횟수는 500회이며 25개의 데이터 블록에 대해 실험하였으며, 채널 환경은 AWGN으로 가정하였다. 인코더의 부호율은 1/2, 구속장 (constraint length)은 5, 인터리버 크기는 1024 비트로 하였으며, 최대 반복 디코딩 횟수는 10회로 하였다.

디코딩 실패율에 따른 |LLR|의 합의 평균을 그림 6에 나타냈다. 이 결과에 따르면 500회 모두 디코딩을 성공한 데이터 블록 (실패율 0%)의 |LLR|의 합의 평균과 500회 모두 디코딩을 실패한 데이터 블록 (실패율 100%)의 |LLR|의 합의 평균은 매 반복 디코딩 과정마다 차이가 있음을 알 수 있다. 4회 이상의 반복 디코딩 과정을 수행하고 난 결과를 보면 주어진 조건에서 500회 모두 디코딩이 가능한 데이터 블록 (0%)의 |LLR|의 합의 평균이 41.55로서, 42.6% 디코딩을 실패할 데이터 블록의 |LLR|의 합의 평균 26.54보다 약 2배 크고, 500회 모두 디코딩이 불가능한 데이터 블록의 |LLR|의 합의 평균 11.68보다는 약 4배가 크다. 따라서 본 연구에서는 디코딩 불가능 임계값 H_1을 적용할 반복 디코딩 시작 횟수 T_1을, 많은 모의실험을 통하여, '5'로 정하였다. 즉, 초기 4번의 반복 디코딩 과정까지는 디코딩 가능 임계값 H_2만을 적용시키고, 5번째 반복 디코딩 과정

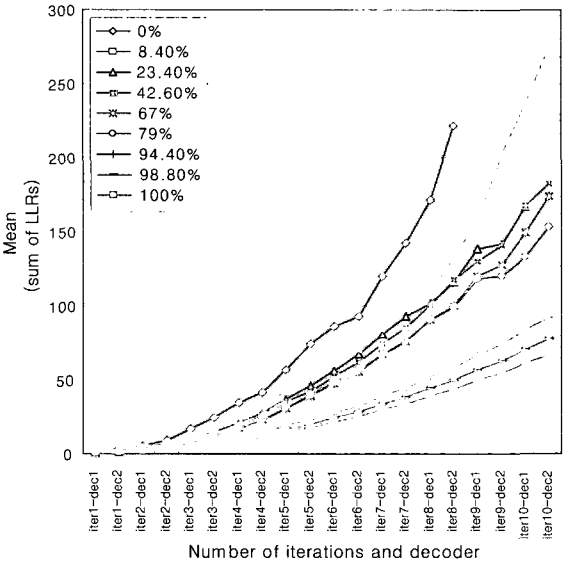


그림 6. 디코딩 실패백분율에 따른 반복 디코딩 횟수당 |LLR|의 합의 평균

Fig. 6. Mean of LLRs' sum for data blocks according to undecodable rate.

부터 H_1과 H_2를 함께 적용하였다. 그리고 실험을 통하여 디코딩이 가능한 데이터 블록들의 매 반복 디코딩 과정의 |LLR|의 합의 평균을 H_2로 설정하였다. 표 1과 표 2는 각각 매 반복 디코딩 과정의 디코딩 불가능 임계값 H_1과 디코딩 가능 임계값 H_2를 나타낸다. 디코딩 불가능 임계값 H_1은 디코딩 실패 백분율이 67%-100% 사이의 LLR값의 평균으로 정했고, 디코딩 가능 임계값 H_2는 SNR을 3dB-0dB까지 0.5dB간격으로 구간을 나누고, 각 구간에서 출력되는 디코더의 LLR값의 평균으로 정했다.

본 논문에서 제안한 디코딩 불가능 임계값과 디코딩 가능 임계값을 적용하여 반복 디코딩 횟수를 조절하는 방법의 효율성을 입증하기 위하여 기존의 [3]에서 제안한 방법과 비교 실험을 하였다.

우선 그림 7은 [3]의 방법으로 반복 디코딩을 정지한 결과 SNR에 따른 평균적인 반복 디코딩 횟수를 나타낸다. 터보 디코더 내에 두 개의 SISO 디코더가 있으므로 하나의 SISO 디코더를 통과했을 때 반복 디코딩 횟수는 0.5, 두 개 SISO 디코더를 모두 통과했을 때의 반복 디코딩 횟수를 1로 설정하였다. 예를 들어, 반복 디코딩 횟수가 4.5라는 것은 4번의 반복 디코딩 과정 (두 개의 SISO 디코더 통과) 후에 5번째 반복 디코딩 과정에서 첫 번째 SISO 디코더만을 통과한 후 반복 디코딩 과정이 끝났음을 뜻한다.

그림 8은 본 논문에서 제안한 방법으로 디코딩 불가

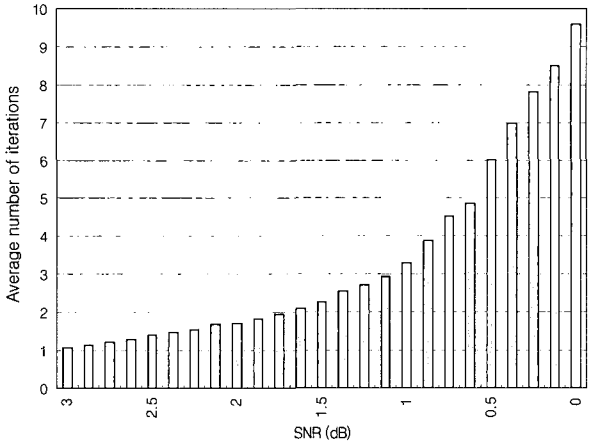


그림 7. [3] 방법의 결과, 평균 반복 디코딩 횟수

Fig. 7. Average number of iterations of [3].

표 1. 매 반복 디코딩 과정의 디코딩 불가능 임계값 H1(i,j)

Table 1. Undecodable data threshold H1(i i).

반복횟수 (i)	decoder#(j)	$H_1^{(i,j)}$	반복횟수 (i)	decoder#(j)	$H_1^{(i,j)}$
1	1	-	6	1	31.5
	2	-		2	36.6
2	1	-	7	1	43.7
	2	-		2	49.6
3	1	-	8	1	58.3
	2	-		2	64.6
4	1	-	9	1	75.3
	2	-		2	81.0
5	1	21.0	10	1	91.7
	2	26.0		2	102.9

능 임계값과 디코딩 가능 임계값을 모두 적용시켰을 때 SNR에 따른 평균적인 반복 디코딩 횟수를 나타낸다. 실험 결과 0.6dB~3.0dB에서는 [3]의 방법에 비하여 본 논문에서 제안한 두 개의 임계값을 적용한 방법이 약간 우수하였으나, SNR이 0.5dB 이하의 채널환경에서는 본 논문의 방법이 [3]에 비하여 상당히 적은 횟수의 반복 디코딩 과정을 수행한다는 것을 알 수 있다. 두 방법의 반복 디코딩 정지 기준의 실험 결과를 그림 9에 나타내었다.

터보 디코더가 무선 채널 환경에서 발생하는 오류를 정정하는 역할을 한다는 점에서 단순히 반복 디코딩 횟수를 조절하여 저 전력 터보 디코더를 구현하는 것은 오류정정 성능에 악영향을 미칠 수 있다. 따라서 [3]의 방법과 본 논문에서 제안한 이중 정지기준을 적용한 방법의 오류정정 성능을 비교하였다.

표 3은 각 데이터 블록 당 500회의 실험 횟수 중 디코딩이 불가능한 횟수가 4회~500회로 다양한 데이터 블록들에 대해 디코딩 불가능 임계값 H_1을 적용했을 때의 조기 반복 디코딩 정지 횟수와 백분율을 보여준다.

표 2. 매 반복 디코딩 과정의 디코딩 가능 임계값 H2(i,j)
Table 2. Decodable data threshold H2(i,j).

반복횟수 (i)	decoder#(j)	SNR (dB) 3.0~2.6	2.5~2.1	2.0~1.6	1.5~1.1	1.0~0.6	0.5~0.1
1	1	2.3	1.7	1.3	1.0	0.8	0.7
	2	7.8	5.6	3.8	2.7	1.9	1.5
2	1	21.2	15.7	10.5	6.8	4.5	3.4
	2	39.6	29.9	21.1	13.4	7.8	5.3
3	1	-	50.6	36.2	23.9	13.8	8.8
	2	-	-	46.8	34.5	20.7	12.6
4	1	-	-	74.0	50.9	29.3	18.4
	2	-	-	-	62.7	37.6	24.4
5	1	-	-	-	84.2	50.2	32.7
	2	-	-	-	104.7	60.4	41.2
6	1	-	-	-	135.0	72.4	50.8
	2	-	-	-	177.5	81.7	60.4
7	1	-	-	-	195.3	96.1	71.7
	2	-	-	-	237.3	111.8	82.5
8	1	-	-	-	250.3	127.5	96.6
	2	-	-	-	-	149.5	110.1
9	1	-	-	-	-	150.1	129.7
	2	-	-	-	-	182.3	135.6
10	1	-	-	-	-	190.6	162.6
	2	-	-	-	-	227.3	177.7

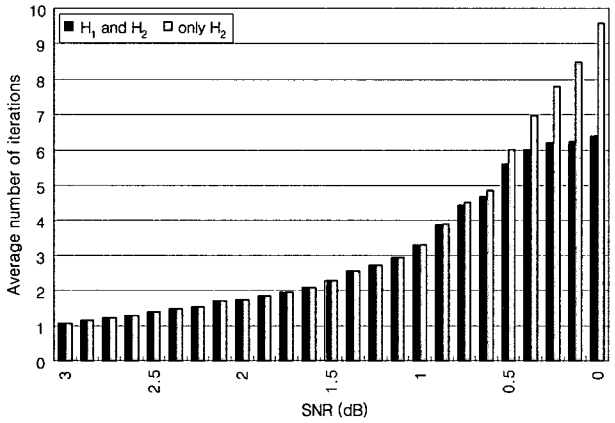


그림 8. 디코딩 불가능 임계값 H1과 디코딩 가능 임계값 H2을 모두 적용시킨 결과 평균 반복 디코딩 횟수
Fig. 8. Average number of iterations resulting in applying the undecodable and decodable data threshold H1 and H2.

기존의 방법과 같이 디코딩 불가능 임계값을 적용하지 않는다면 디코딩이 불가능한 데이터 블록에 대하여 최대 반복 디코딩 횟수만큼 반복 디코딩 과정을 수행하게 되지만, 디코딩 불가능 임계값을 반복 디코딩 정지 기준에 적용시키면, 디코딩이 불가능한 데이터 블록일수록 조기에 반복 디코딩을 정지할 수 있어서 반복횟수가 감소한다.

표 4 에서는 디코딩 실패 백분율이 서로 다른 데이터 블록들에 대해, 본 논문에서 제안한 방법과 [3]의 방법

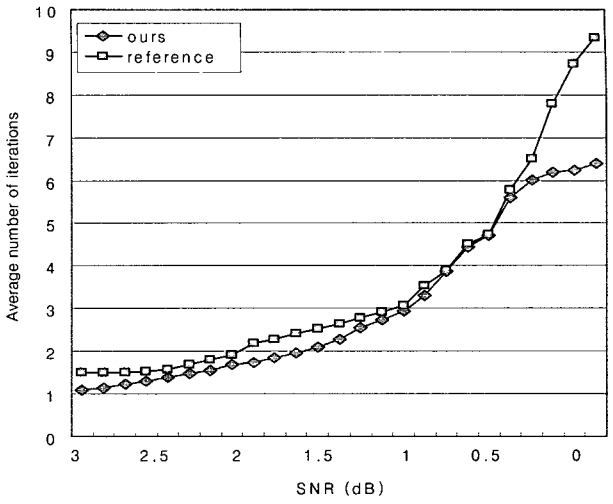


그림 9. 새로운 저 전력 터보 디코더 설계 방법과 [3]의 방법의 평균 반복 디코딩 횟수 결과 비교
Fig. 9. Comparison of average number of iterations with proposed turbo decoder design methods and [3].

표 3. H1을 적용한 결과
Table 3. Simulation results applying the undecodable data threshold H1.

전체 디코딩 불가능 횟수 (총 500회)	조기 반복 디코딩 정지 횟수	조기 반복 디코딩 정지 백분율 (%)
4	0	0
18	4	22
42	14	33
117	48	41
213	105	49
335	170	51
395	255	65
472	346	73
483	395	82
494	437	88
498	466	94
500	497	99

으로 반복 디코딩 과정을 정지할 경우의 잘못된 오류정정 횟수와 오류정정의 성능 저하, 그리고 평균 반복 디코딩 횟수를 표시하였다. 잘못된 오류정정 횟수란 임의의 데이터 블록에 대하여 임계값을 설정하지 않았을 경우 최대 반복 디코딩 횟수 내에 디코딩이 가능하지만 임계값을 적용함으로써 디코딩이 불가능하게 된 횟수를 의미한다. 표 4에서 디코딩 실패 백분율이 23.6%인 경우, [3]의 방법을 적용했을 경우 500회중 122회 오류정정을 잘못 수행했지만 본 논문에서 제안한 방법을 적용시켰을 경우 단 7회만 오류정정을 잘못 수행하였다. 결과적으로 전체 12500회 실험 중 본 논문에서 제안한

표 4. 오류정정 성능저하와 평균 반복 디코딩 횟수 (각 500회 실험결과)

Table 4. Error correction performance degradation and average number of iterations (simulation: 500 times).

디코딩 실패 백분율	잘못된 오류정정 횟수				평균 반복 디코딩 횟수	
	제안 방법	%	[3]	%	제안 방법	[3]
0%	0	0	36	7	3.301	3.525
0.8%	0	0	30	6	3.869	3.9
3.6%	1	0.2	71	14.2	4.45	4.5
12.6%	4	0.8	90	18	4.71	4.741
23.6%	7	1.4	122	24.4	5.6	5.775
42.6%	10	2	123	25	6.02	6.505
59%	7	1.4	104	20	6.2	7.793
67%	35	7	71	14.2	6.25	8.728
79%	10	2	51	10	6.4	9.348
92.4%	2	0.4	17	3.4	5.35	9.962
98%	1	0.2	4	0.8	5.01	9.992
99.6%	0	0	2	0.4	5.04	9.998
100%	0	0	0	0	5.04	10
합계	77 11%	15.4 11%	721 100%	143.4 100%	67.24 71%	94.77 100%

이중 정지 기준 알고리즘을 적용했을 때는 잘못된 오류정정횟수가 총 77회로서 721회인 [3]의 방법에 비해 89% 감소하였으며, 전력 소모의 척도가 되는 반복 디코딩 횟수는 평균적으로 29% 감소하였다.

이동통신 시스템에서는 오류 정정 성능이 중요하다. 오류 정정 성능 저하가 발생하지 않도록 하였을 경우에는 [3]보다 9%의 반복 디코딩 횟수가 감소하였다. 실제 이동통신 시스템에서는 채널 환경을 정확히 추정하는 것이 어렵다. 따라서 SNR을 모를 경우에는 [3]보다 9%의 반복 디코딩 회수가 감소하였다.

VI. 결 론

본 논문에서는 이동통신 시스템에서 많이 사용되는 터보 디코더의 전력 소모를 줄이기 위한 저 전력 터보 디코더 설계 기술에 대해 연구하였다. 기존의 반복 디코딩 정지 기준 알고리즘보다 개선된 두 개의 반복 디코딩 정지 임계값을 이용한 새로운 디코딩 방식을 개발하였다. 채널 환경이 우수하여 신호대 잡음비가 클 때에는 디코딩 가능 임계값이 조기에 경판정을 할 수 있도록 하며, 채널 환경이 나빠서 디코딩이 불가능할 가능성이 높은 경우에는 디코딩 불가능 임계값 기준에 의해 디코딩을 조기에 정지함으로써 불필요한 반복 디코딩 횟수를 효과적으로 감소시켜서 오류정정 성능은 기존의 방법에 비하여 우수하면서도 소모 전력을 감소시킬 수 있었다.

고정된 횟수의 반복 디코딩 방법에 비하여, [3]의 방법은 디코딩 반복 횟수를 55%-90% 감소시켰으며, 본 논문에서 제안한 방법은 [3]에 비하여 추가적으로 평균 29% 반복 횟수를 감소 시켰다.

참 고 문 헌

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, *"Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes"*, ICC'93, pp. 1064-1070, May 1993.

[2] F. Gilbert, A. Worm, and N. Wehn, *"Low Power Implementation of a Turbo-Decoder on Prog-rammable Architectures"*, Proc. of the ASP-DAC 2001, pp. 400-403, Feb. 2001.

[3] D. Kim, S. Kim, and S. Hwang, *"A Novel Turbo Decoder Architecture for Hand-Held Communi-cation Devices"*, IEEE Trans. on Consumer Electronics, vol. 48, no. 2, pp. 202-208, May 2002.

[4] L. Lee, A. Hammons, F. Sun, and M. Eroz, *"Application and Standardization of Turbo Codes in Third-Generation High-Speed Wire-less Data Services"*, IEEE Trans. on Vehicular Tech., vol. 49, no. 6, pp. 2198-2207, Nov. 2000.

[5] G. Masera, G. Piccinini, M. Roch, and M. Zamboni, *"VLSI Architectures for Turbo Codes"*, IEEE Trans. on VLSI Syst., vol. 7, no. 3, pp. 369-379, Sep. 1999.

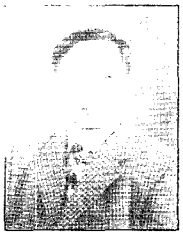
[6] J. Woodard and L. Hanzo, *"Comparative Study of Turbo Decoding Techniques: an Overview"*, IEEE Trans. on Vehicular Tech., vol. 49, no. 6, pp. 2208-2233, Nov. 2000.

저 자 소 개

임 호 영(학생회원)
2002년 한양대학교 전자컴퓨터공 학부 학사.
2004년 한양대학교 전기전자제어 계측공학과 석사.
<주관심분야: 저전력 설계, 버스 분할, 컴퓨터, 반도체>

강 원 경(학생회원)
2003년 한양대학교 전자컴퓨터공학부 학사.
2004년 한양대학교 전자전기제어 계측공학과 석사과정.
<주관심분야: 저전력 설계, 터보코드, MPEG4, 반도체>

신 현 철(정회원)
1978년 서울대학교 전자공학과 학사.
1980년 한국과학기술원 전기 및 전자공학 석사.
1983~1987년 U.C. Berkeley Ph.D
1983~1987년 Fulbright scholar ship
1987~1989년 MTS, AT&T Bell Lab's, Murray Hill N.J., USA
1989~현재 한양대학교 전자컴퓨터공학부 교수
1997~현재 IDEC 한양대학교 지역센터 센터장
<주관심분야: CAD&VLSI, 통신용 반도체 설계, 저전력 설계>



김 경 호(정회원)
1984년 연세대학교 전자공학과 학사
1987년 한국과학기술원 전기 및 전자공학 석사
1991년 한국과학기술원 전기 및 전자공학 박사
1983~현재 삼성전자 정보통신총괄 통신연구소 상무
1997년 Marquis's Who's Who 인명록 등재
1998년 IBC 인명록 등재
2003년 자랑스런 삼성인상 수상(모뎀개발)
<주관심분야: 저전력 설계, 통신 모뎀, IT SOC 등>