MAP 기반 터보코드의 FPGA 설계

종신회원 서 영 호*

FPGA Design of Turbo Code based on MAP

Young-Ho Seo* *Lifelong Member*

요 약

본 논문에서는 높은 에러정정 효율을 보이는 터보코드 알고리즘을 FPGA H/W(hardware) 자원 내에 효율적으로 구현하였다. 본 논문은 구속장의 크기가 3, 1/3 인코더, 2048 사이즈의 랜덤 인터리버에 기반한 터보코드 알고리즘을 사용한다. 제안된 H/W는 델타를 이용하여 알파와 베타를 연산하는 MAP 블록과 각 값들을 저장하는 버퍼 및 람다의 계산을 위한 곱셈기와 람다를 저장하는 버퍼로 구성된다. 제안된 알고리즘과 하드웨어 구조는 C++ 언어를 이용하여 검증하였고, VHDL을 이용하여 하드웨어 구현한 후 FPGA에 적용하여 무선통신 환경에서 성능에 대한 유효성을 보였다. 구현된 H/W는 VERTEX4 XC4VFX12-12-SF363의 FPGA를 타겟으로 하였고 최대 131.533MHz (7.603ns)에서 안정적으로 동작할 수 있었다.

Key Words : Turbo code, Ecc, FPGA, H/W, Communication

ABSTRACT

In this paper, we efficiently implemented turbo code algorithm in FPGA H/W(hardware) resource. The used turbo code algorithm has the characteristics; the size of constraint is 3, encoder type is 1/3, the size of random interleaver is 2048. The proposed H/W consists of MAP block for calculating alpha and delta using delta value, storing buffer for each value, multiplier for calculating lamda, and lamda buffer. The proposed algorithm and H/W architecture was verified by C++ language and was designed by VHDL. Finally the designed H/W was programmed into FPGA and tested in wireless communication environment for field availability. The target FPGA of the implemented H/W is VERTEX4 XC4VFX12-12-SF363 and it is stably operated in 131.533MHz clock frequency (7.603ns).

Ⅰ. 서 론

통신시스템의 발달과 채널의 다양화 및 고속화는 정보 전송의 신뢰성있는 채널 코딩 방식을 더욱 요구한다. 통신 채널에서 잡음이나, 페이딩, 간섭 등에 의한 정보 손실이 발생하게 되고, 이로 인해 발생하는 오류를 정정해 주기 위해 오류 정정 부호의 사용이 필요하다[1].

오류 정정 부호(Error Correcting Code) 기술이란 디지털 통신 시스템에서 채널 상의 여러 요인에 의하여 발생하는 오류를 효율적으로 정정하기 위하여 사용되는 방식으로써 일반적으로 채널부호(Channel Coding)하고 한다[1]. 오류 정정 부호 기술은 정보 비트에 임의의 잉여 비트를 덧붙여 부호어를 만들고, 수신단에서는 송신단에서 부가된 잉여 비트를 이용하여 수신된 부호어로부터 오류가 없는 원래의 정보 비트를 복원하여 시스템의 데이터 전송 성능을 높이는 방식이다.

오류 정정 부호 방식은 잉여 비트를 첨가하는 방식에 따라 블록 부호방식(Block Coding)과 트리 부

※본 연구는 2007년도 한성대학교 교내연구비 지원과제임.
* 한성대학교 정보통신공학과 (www.hansung.ac.kr/design, yhseo@hansung.ac.kr)
논문번호 : KICS2006-10-141, 접수일자 : 2006년 10월 18일, 최종논문접수일자 : 2007년 3월 2일

호 방식(Tree 혹은 Trellis Coding)으로 구별할 수 있다[1]. 블록 부호 방식은 정보 비트에 대수적인 연산을 적용하여 잉여 비트를 첨가하고 수신단에서는 역연산에 의하여 원래의 정보를 복원하는 방식으로서 1950년에 Hamming에 의하여 최초로 소개되었다. 트리 부호는 1955년에 Elias에 의하여 소개되었고 1960년대 말에 Viterbi 알고리즘이 개발되어 오늘날까지 복호화 방식으로 널리 이용되고 있다. 트리 부호는 대수적인 연산이 부/복호 과정에 개입되지 않기 때문에 복호 성능을 구하기가 어렵지만 연판정(Soft Decision) 복호를 쉽게 수행할 수 있어 약 2dB 정도의 이득을 얻을 수 있다는 특징이 있다[3][4].

터보부호는 길쌈부호를 병렬로 연접하여 만들어진 부호라고 할 수 있는데 1993년 프랑스의 Berrou에 의하여 처음 소개되었고 Shannon의 극한 값에 도전하는 부호화 방식으로 집중적으로 연구되기 시작하였다[2][3][4]. 터보 코드가 소개되었을 때에는 부호기에 내재하는 인터리버(Interleaver)와 복호기(Decoder)에서 수행되는 반복 복호에 기인하는 성능에 있어서 엄청난 지연 시간이 요구되었다. 따라서 매우 빠른 전송율과 긴 지연을 감수해야 하므로 장거리 우주 통신을 제외한 다른 응용 분야에는 적용하기가 힘들었다. 하지만 메모리 크기의 증가와 프로세싱 시간의 단축으로 지연 문제를 해결함으로써 터보 코드를 좀 더 광범위한 범위에 사용할 수 있게 되었다[5][6][7].

터보 코드의 복호화는 MAP(Maximum a Posteriori) 복호기 혹은 SOVA(Soft-Output Viterbi Algorithm) 복호기를 이용한 반복 복호를 통하여 원래의 정보를 복원하게 된다. MAP 알고리즘은 SOVA 알고리즘에 비해 복잡성은 크나, 성능면에서 우수성을 가진다[8]. 이러한 이유로 복잡성은 크나, BER 성능이 우수한 MAP 알고리즘이 사용되고 있다[9]. MAP 알고리즘은 Bahl 등에 의해서 1974년 처음으로 제안되었으며, 잡음이 섞인 신호로부터 APP를 계산하는 알고리즘이다. MAP 알고리즘은 모든 경로에 대한 확률 값들을 계산하여 정보비트에 대한 연판정 데이터 값을 출력하게 된다[9]~[12]. 또한 터보코드의 복잡성으로 인한 속도향상을 위해서 H/W형태의 연구도 많이 이루어져 왔다[13].

본 논문에서는 MAP를 기반으로 하여 터보 코더를 하드웨어로 구현하였고 동작을 검증하였다. 2장에서는 MAP를 기반으로 하는 터보코드 알고리즘에 대해서 설명하였고, 3장에서는 H/W 구현을 위한 알고리즘의 분석 및 재해석과 이를 바탕으로 한 H/W 구조에 대해서 나타내었다. 4장에서는 H/W 구현을 위한 S/W 구현결과를 보인 후 H/W 구현 결과를 나타내었고 마지막으로 5장에서 결론을 내었다.

Ⅱ. 터보코드 알고리즘

2.1 부호화 알고리즘

터보코드의 부호기는 두 개의 길쌈 부호기와 한 개의 인터리버를 연결한 형태로서 부호화는 하나의 Systematic 정보와 두개의 Recursive Systematic 길쌈부호기(Convolutional Encoder)의 출력을 통하여 부호화 되며 길쌈 부호기 사이에 인터리버를 두어 서로 다른 입력 정보를 포함 할 수 있도록 한다.

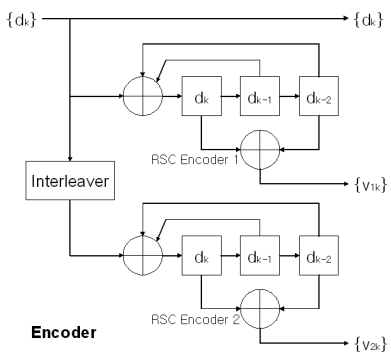


그림 1. 터보 인코더의 구조.

2.2 MAP 방식의 복호화 알고리즘

MAP 알고리즘은 Bahl에 의해 제안된 잡음이 섞인 신호로부터 APP(A Posteriori Probability)를 계산하는 알고리즘이다. MAP 디코딩 알고리즘의 목적은 $R_1^N=(R_1,...,R_k,...,R_N)$이 수신된 후 수신된 심볼에 대해서 가장 가능성이 있는 정보 비트 d_k를 결정하는 것으로서 이는 LLR(Log Likelihood Ratio)로 정의되는 식 (1)과 같다.

$$L(d_k)=\log\left(\frac{P_r(d_k=1|R_1^N)}{P_r(d_k=0|R_1^N)}\right) \quad (1)$$

여기서 $P_r(d_k=i|R_1^N)$, i=0, 1은 정보 비트 d_k의 APP이며, 복호된 정보 비트 d_k의 APP는 식 (2)와 같이 정의된 결합 확률로부터 구한다. S_k 는 인코더의 레지스터 길이가 V일 때, $\sum_{i=0}^{V-1}2^iS_i$ 로 정의되는

시간 K에서의 인코더 상태이다. 식 (2)를 Bayes정리와 결합 확률을 적용하여 전개를 하면 식 (3) 및 (4)와 같다.

$$\lambda_k^i(m) = P_r(d_k = i, S_k = m \mid R_1^N) \tag{2}$$

$$\lambda_k^i(m) = \frac{P_r(d_k = i, S_k = m, R_1^k)\, P_r(R_{k+1}^N \mid d_k = i, S_k = m, R_1^k)}{P_r(R_1^N)} \tag{3}$$

$$\lambda_k^i(m) = \frac{\alpha_k^i(m)\beta_k^i(m)}{P_r(R_1^N)} \tag{4}$$

식 (3)과 (4)에서 $\alpha_k^i(m)$과 $\beta_k^i(m)$는 각각 $P_r(d_k = i, S_k = m, R_1^k)$ 와 $P_r(R_{k+1}^N \mid d_k = i, S_k = m)$ 에 해당한다. 식 (4)에서 $\alpha_k^i(m)$은 순방향 메트릭이라 하며, 정보 비트 i, 시간 K에서 상태 m부터 다음 상태 천이를 위한 상태 메트릭을 나타내고, 식 (5)와 같이 표현할 수 있다. 식 (5)에서 $\delta_i(R_k, m)$은 가지 메트릭 값으로 식 (6)과 같다.

$$\begin{aligned}\alpha_K^i(m) &= \frac{1}{2} Pr(R_k \mid d_k = i, S_k = m) \sum_{j=0}^{1} \alpha_{K-1}^j(S_b^j(m)) \\ &= \delta_i(R_k, m) \sum_{j=0}^{1} \alpha_{K-1}^j(S_b^j(m))\end{aligned} \tag{5}$$

$$\delta_i(R_k, m) = \Pr(R_k \mid d_k = i, S_k = m)/2 \tag{6}$$

$\beta_K^i(m)$은 역방향 상태 메트릭이라고 하며, $\alpha_k^i(m)$와 마찬가지로 모든 정보를 수신한 후, 식 (7)과 같이 $\beta_{K+1}(m)$부터 $\beta_K(m)$을 반복적으로 구할 수 있다.

$$\beta_k^i(m) = P_r(R_{k+1}, R_{k+2}^N \mid d_k = i, S_k = m) \tag{7}$$

여기서 K+2 이상의 수신 정보의 확률을 구하는 데 있어서 시간 K+1에서의 정보, 부호기 상태만을 알면 충분하므로 시간 K에서의 정보, 부호기 상태와 시간 K+1의 R_{K+1}은 고려할 필요가 없다. 따라서 식 (7)은 식 (8)과 같이 정리할 수 있다.

$$\beta_k^i(m) = \sum_{m'} \sum_{j=0}^{1} \beta_{k+1}^j(m')\, \gamma_{i,j}(R_{k+1}, m', m) \tag{8}$$

식 (8)에서 $\gamma_{i,j}(R_{k+1}, m', m)$ 부분만을 전개하고, 이를 Bayes 정리와 결합 확률을 적용한 후 정리하면 식 (9)로 귀결할 수 있다.

$$\beta_k^i(m) = \sum_{j=0}^{1} \beta_{k+1}^j(S_f^i(m))\delta_j(R_{k+1}, S_f^i(m)) \tag{9}$$

여기에서 $\beta_K^i(m)$을 그림 2와 같이 역방향 상태 메트릭 표현으로 나타낼 수 있다.

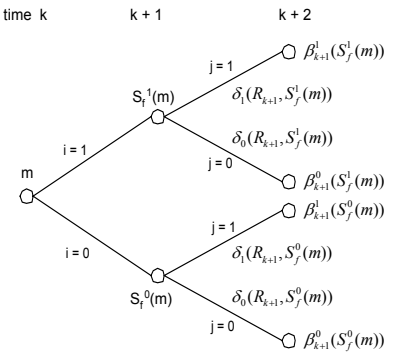


그림 2. 역방향 상태 메트릭.

터보코드는 반복 복호를 통해서 복호 성능을 향상시키는데 이는 베이스 정리에 근거한 LLR(Log-Likelihood Ratio)의 개념을 이용한다. 이러한 반복적인 MAP 복호기는 부호기와 마찬가지로 복호기와 복호기 사이에 인터리버를 두어 복원하려고 하는 정보들간의 상관관계를 최소화 한다. 이것은 인접한 정보들이 가진 에러의 영향을 최소한으로 감소시키기 위한 방법으로 독립적인 가우시안으로의 해석을 가능하게 한다. 터보코드에 사용되는 복호기의 출력은 연판정 데이터이므로 반복적인 복호가 가능하며 반복의 횟수가 증가할수록 성능이 향상된다. 표 1은 본 논문에서 구현한 S/W를 이용하여 반복복호를 수행했을 경우에 에러정정효율이 향상되는 정도를 실험한 결과이다.

표 1. 반복복호의 횟수에 따른 에러복구율

Error Number	The Number of Iteration			
	1	2	4	8
30	0	0	0	0
60	1	0	0	0
90	2	0	0	0
150	18	0	0	0
180	30	1	0	0
360	42	14	7	0
720	97	59	16	4

Ⅲ. 제안한 H/W 구조

3.1 H/W 기반의 복호화 알고리즘 분석

H/W 구현을 위해 터보 복호화 알고리즘을 그림 3에 해석 및 재정리 하였다. 부호화 데이터 중 일부를 가지고 LUT (Look-up Table) 방식의 델타(delta) 연산을 통해 서로 역방향으로 알파(alpha)와 베타(beta)를 구한다. 구해진 두 값을 이용하여 람다(lamda)를 계산한 후 나머지 부호화 데이터를 이용하여 세 가지 종류의 데이터에 대한 복호 정확도를 높인다. 또한 이를 반복적으로 수행함으로써 성능을 높이는 과정을 반복하고 최종적으로 값을 결정(Hard Decision)하여 복호를 완료한다.

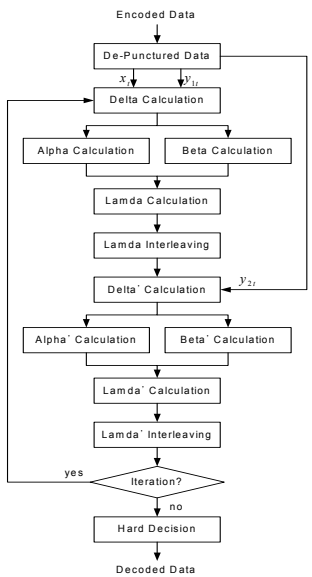


그림 3. 복호화 알고리즘

3.2 제안한 H/W 구조

제안한 H/W 구조를 그림 4에서 6에 나타내었다. 그림 4는 전체 터보코더의 구조를 나타내는데 입력된 부호화된 데이터는 MAP내에서 델타를 이용하여 알파와 베타가 계산되고 그 결과를 역순으로 대입하여 연산하기 위해 버퍼에 저장된 후 이 값들은 람다를 구하는데 이용된다.

그림 5는 이들 중 핵심 블록인 MAP의 구조를 나타낸다. LUT에 미리 저장된 예측치들을 이용하여 델타를 고속으로 구하고 이들을 덧셈 및 곱셈 과정을 통해서 알파와 베타가 구해진다. 알파와 베타의 연산과 함께 이 값들의 분포를 분석하여 중간값을 구한 후 정규화시키고 최종 출력값으로 결정한다. 또한 그림 6은 중간 데이터를 임시적으로 저장하는 버퍼를 나타낸다.

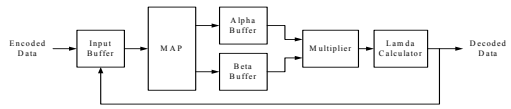


그림 4. 전체 구조

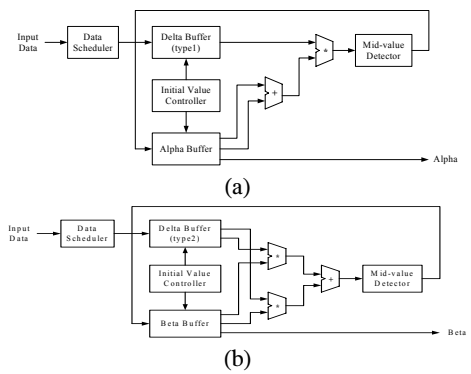


그림 5. MAP의 구조 (a) 알파 계산기 (b) 베타 계산기

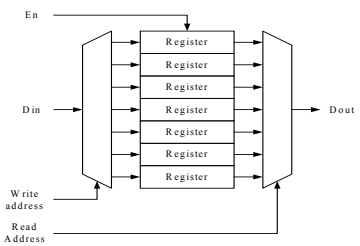


그림 6. 버퍼의 구조

Ⅳ. 실험 및 구현결과

4.1 실험결과

그림 7과 8은 본 논문에서 구현된 터보코드를 이용하여 영상에 대해서 복호 특성을 실험한 결과이다. 실험에 이용된 S/W는 C++을 이용하여 구현되었고, FPGA를 이용한 검증 시스템에서 운용을 위한 S/W로도 사용된다. 그림 7은 영상을 부호화시키고 가우시안 노이즈를 첨가한 후 다시 복호화한 결과를 보이고 있고 그림 8은 반복 복호에 따른 복호율을 가시적으로 나타내고 있다.

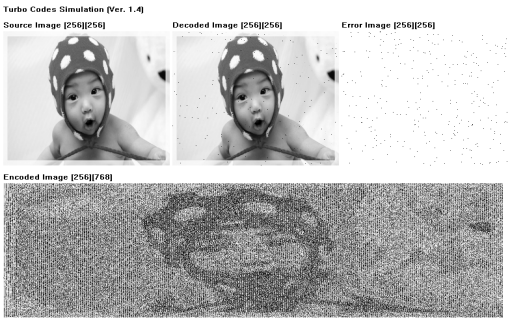


그림 7. S/W 시뮬레이션 예제.

(a)

(b)

그림 8. 반복 복호결과 (a) 1회 반복 (b) 4회 반복.

4.2 H/W 구현결과

구현된 H/W는 VERTEX4 XC4VFX12-12-SF363의 FPGA를 대상으로 하였고 최대 131.533MHz (7.603ns)의 동작 속도를 가진다. 표 2는 구현된 H/W의 자원 사용률을 나타낸다.

그림 9에서 11은 구현된 H/W의 RTL (Register Transfer Level) 합성도를 나타내었다. 그림 9는 전체 터보코드의 합성도를 나타내고 있고 그림 10은 그림 9에서 왼쪽에서 첫 번째 블록인 MAP 블록을 나타내고 있다. 또한 그림 11은 그림 10에서 왼편 위쪽 블록인 알파-베타 계산기를 나타내고 있다.

터보 코드는 블록 코딩이기 때문에 한 블록(프레임)을 모두 수신한 이후에 복호가 가능하다. 또, MAP 알고리즘은 복호 과정에서 알파와 베타라는 두 가지의 계산을 하게 되는데, 수신한 블록의 순방향과 역방향으로 복호하는 순서가 다르다. 그리하여 반드시 두 가지 정보 중 하나는 메모리에 저장한 후 다음 계산에 사용되어야 하고 파이프라인 특성과 성능을 고려한다면 둘 다 메모리에 저장한 후에 람다를 계산해야한다.

표 2. H/W 자원 사용률.

Item	Resource	Ratio
Slices	2014 / 5472	36%
Slice Flip Flops	3331 / 10944	30%
4 input LUTs	1352 / 10944	12%
bonded IOBs	42 / 240	17%
GCLKs	2 / 32	6%
DSP48s	4 / 32	12%

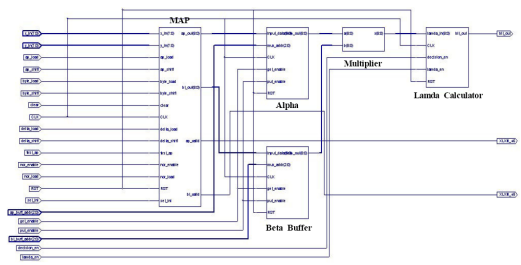


그림 9. 터보코드 전체 H/W의 RTL 합성도.

그림 10에서 MAP를 통해 계산된 알파와 베타값들은 정규화(normalizaiton) 과정을 거치고 곱해져서 람다값이 된다. 알파와 베타값을 정규화하는 과정은 각각의 연산값들에 대한 분포를 구하는 과정과 이들을 분석한 후 중간값을 찾는 과정, 그리고 중간값으로 각각의 값들을 나누는 과정으로 나누어진다. 따라서 이 과정은 상당한 H/W적인 복잡도와 자원을 요구한다.

알파, 베타 블록에서 나온 출력과 델타에서 나온 값이 서로 계산이 되어져서 정규화를 위한 정보를 제공한다. 이후에 알파, 베타 값이 다시 순환 데어 출력 될 때 이 정보가 사용되어 알파, 베타 값이 너무 크거나, 혹은 너무 작은 값이 되지 않도록 한다.

shift_a 모듈에서 한 바이트 크기의 정보 데이터와 패러티 데이터를 받아서 cal_delta 모듈로 한 비트씩 송신 해준다. cal_delta 모듈은 수신한 정보 데이터, 패러티 데이터의 비트에 따라 16개의 delta 값을 출력하고, 이와 동시에 alpha_buff, beta_buff 모듈은 delta 값과 계산하기 위한 정보를 출력한다.

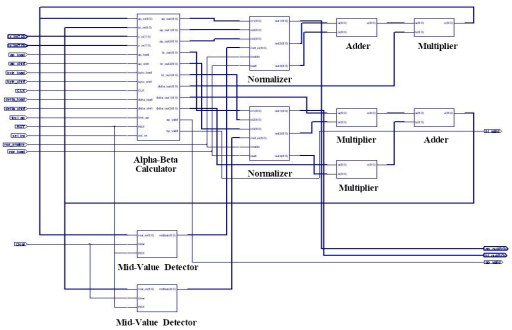


그림 10. MAP 블록의 합성도.

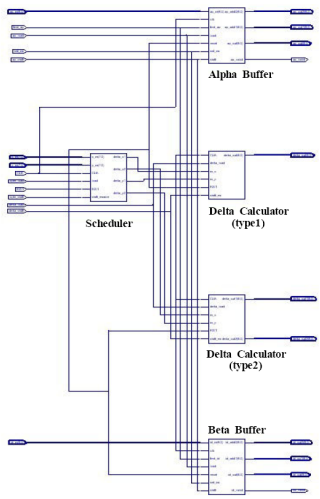


그림 11. 알파-베타 연산 블록.

터보 코드는 블록 코딩이기 때문에 한 블록(프레임)을 모두 수신한 이후에 복호가 가능하다. 또, MAP 알고리즘은 복호 과정에서 알파와 베타라는 두가지의 계산을 하게 되는데, 수신한 블록의 순방향과 역방향으로 복호하는 순서가 다르다. 그리하여 필히 두 가지 정보 중 하나는 메모리에 저장한 후 다음 계산에 사용되어야 한다.

그림 11에 보이는 메모리 블록은 16개의 알파, 베타를 저장할 수 있는 셀이 8개인 형태이다. 설계에서 한 블록의 크기를 8비트를 설정하였기 때문에, 8개의 셀이 필요하다. 터보 코드는 한 블록의 크기가 커질수록 복호 성능이 좋아지는 특성을 가지고 있지만, 메모리 사이즈의 증가가 필요하고 복호 지연이 증가하여 이 특성들이 서로 상보적인 관계를 가지기 때문에 최적화된 성능을 내기 위한 여러 연구가 진행되고 있다.

4.3 시뮬레이션 결과

그림 12는 구현한 H/W의 시뮬레이션 결과 파형을 보이고 있는데 입력과 출력은 표 3에 나타내었다. Input 값에 대해서 부호화 과정을 통해 y1과 y2의 부가정보가 더해지고 복호과정을 거쳐서 Decoding 값이 정확히 검출되는 것을 확인할 수 있다. 그림 12에서 좌측 위에 입력값이 나타나있고, 우측 아래의 점선형태의 타원 내 신호가 복호된 결과인데 표 3의 결과값들과 동일함을 확인할 수 있다.

표 3. 부호화 및 복호화 예.

#	Item		Bit Sequence
1	Input		10110001
2	Encoding	y1	11010001
		y2	11110100
3	Decoding		10110001

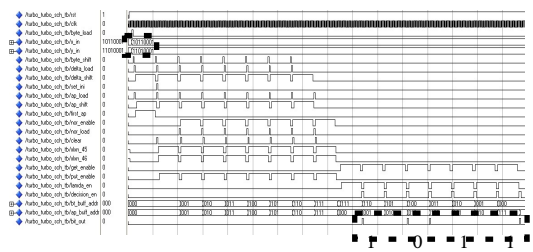


그림 12. 구현한 H/W의 시뮬레이션 결과.

0 0 0

1

4.4 검증

구현한 터보코더가 실제 통신환경에서 유효한지를 확인하기 위해서 상용 FPGA 보드를 이용하여 검증환경을 구성하고 실험하였다. 구현된 터보코드 외에 PC와의 통신을 위한 USART 회로, RF Transiver 및 Receiver와의 통신을 위한 인터페이스 블록, 터보 인코딩을 위한 인코더 등을 설계하고 그림 13과 같은 검증 시스템을 구현하였다. 실제 검증 시스템의 사진을 그림 14에 보였는데 두 개의 동일한 FPGA를 이용하고 있다. RF Transiver와 Receiver를 비롯하여 안테나는 상용 부품을 이용하여 조립하였다. 실제 보드를 이용하여 무선환경에서 실험한 결과, 구현된 터보코더의 에러정정능력을 확인할 수 있었다.

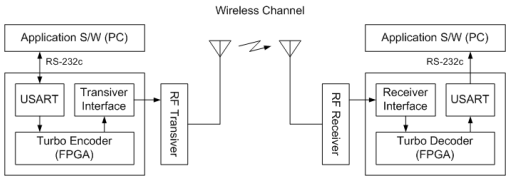
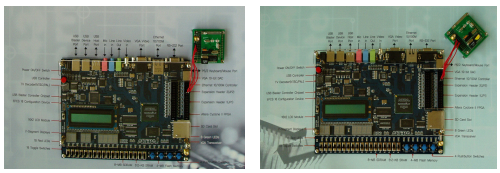


그림 13. 검증 시스템의 블록도



(a) (b)

그림 14. 상용 FPGA를 이용한 검증 시스템 (a) 터보 인코더 (b) 터보 디코더.

V. 결 론

본 논문에서는 터보코드 알고리즘을 FPGA를 이용하여 H/W로 구현하였다. 제안된 H/W는 델타를 이용하여 알파와 베타를 연산하는 MAP 블록과 각 값들을 저장하는 버퍼 및 람다의 계산을 위한 곱셈기와 람다를 저장하는 버퍼로 구성된다. 제안된 알고리즘과 하드웨어 구조는 C++ 언어를 이용하여 S/W로 구현한 후 검증하였고, VHDL을 이용하여 H/W 설계하고 이를 실제 FPGA에 적용하여 무선통신 환경에서 성능을 검증하였다. 구현된 H/W는 VERTEX4 XC4VFX12-12-SF363의 FPGA를 타겟으로 하였고 최대 131.533MHz (7.603ns)에서 안정적으로 동작할 수 있었다.

구현된 터보코드는 S/W 및 H/W적으로 모두 검증하였고, 무선환경에서 실제 검증을 거쳤기 때문에 상용 IP로도 손색이 없고 다양한 무선응용 기기들에 적용되어 여러 기능블록들과 함께 에러를 정정하는 기능을 수행할 수 있을 것으로 사료된다.

참 고 문 헌

〔1〕 C. Shannon, "A Mathematical Theory of Information," Bell SYstem Technical Journal, Vol. 27, pp.379-423, July 1948.

〔2〕 S. Lin, Error Control Coding, Prentice-Hall, pp.3-9, 1983

〔3〕 C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error Correcting Coding and Decoding : Turbo Code", Proc. IEEE ICC, pp.1064-1070, May 1993.

〔4〕 P. Jason and H. Lajos, "Comparative Study of Turbo Decoding Techniques: An Overview," IEEE Trans. on Vehicular Technology, Vol. 49, No. 6, pp.2208-2238, Nov. 2000.

〔5〕 D. Divsalar and F. Pollara, "Turbo Codes for PCS Applications", Proc. of IEEE ICC'95, Seattle, Washington, pp.54-59, Jun. 1995.

〔6〕 L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", IEEE Trans. Inform. Theory, vol. IT-20, pp.284-287, Mar. 1975.

〔7〕 S. Barbulescu and S. Piebrobon, "Turbo Codes: A Tutorial on a New Class of Powerful Error Correcting Coding Schemes, Part 2: Decoder Design and Performance," IEEE Journal of Electrical and Electronics Engineering, Vol. 19, No. 3, pp.143-152, Sept. 1999.

〔8〕 L. Papke, P. Robertson, and E. Villebrun, "Improved decoding with the SOVA in a parallel concatenated (turbo-code) scheme", Proc. of ICC'96, Dallas, TX, USA, pp.102-106, Jun. 1996.

〔9〕 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.

〔10〕 G. Park, S. Yoon, I. Jin, and C. Kang, "A Bock-wise MAP Decoder Using a Probability Ratio for Branch Metric" in Proc. of VTC'99, Amsterdam, Netherlands, pp.1610-1614, Sept. 1999.

〔11〕 S. Donilar and D. Divsalar, "Weight Distributions for Turbo Codes Using Random and Non-random Interleaving," JPL TDA Progress Report 42-122, pp.56-65, 1995.

〔12〕 P. Robertson, E. Villebrun and P. Hoeher, "A Comparison of optimal and Sub-optimal

MAP Decoding Algorithms Operating in the Log Domain," in Proc. ICC' 95, pp.1009-1013, Sattle, June 1995.

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

서 영 호 (Young-Ho Seo) 종신회원

1999년 2월 : 광운대학교 전자재료공학과 졸업(공학사).
2001년 2월 : 광운대학교 일반대학원졸업(공학석사).
2000년 3월~2001년 12월 : 인티스닷컴(주) 연구원.
2004년 8월 : 광운대학교 일반대학원졸업(공학박사)
2003년 6월~2004년 6월 : 한국전기연구원 연구원
2004년 12월~2005년 8월 : 유한대학 연구교수
2005년 9월~현재 : 한성대학교 전임강사
<관심분야> 2D/3D 영상 및 비디오 처리, 디지털 홀로그램, SoC 설계, 워터마킹/암호화