ITU-T J.83 ANNEX B의 Parity Checksum Generator를 위한 병렬 처리 구조

준회원 이 종 엽*, 정회원 홍 언 표*, 하 동 수*, 임 회 정**

Parallel Processing Architecture for Parity Checksum Generator Complying with ITU-T J.83 ANNEX B

Jong-yeop Lee* *Associate Member*, Eon-pyo Hong*, Dong-soo Har*, Hoi-Jeong Lim** *Regular Members*

요 약

이 논문은 ITU-T Recommendation J.83 Annex B 에서 패킷 동기화와 에러 검출을 위해 사용된 패리티 체크섬 생성기의 병렬 구조를 제안한다. 제안된 병렬 처리 구조는 기존의 직렬 처리 구조에서 일어나는 병목현상을 제거하여 패리티 체크섬을 생성하는데 필요한 처리 시간을 상당히 줄여준다. 실험 결과는 제안된 병렬 처리 구조가 16%의 면적증가로 처리 속도를 83.1%나 줄일 수 있다는 것을 보여준다.

Key Words : Parity Checksum, Parallel Syndrome, ITU-T J.83 Annex B, Checksum Generator, Syndrome Generation.

ABSTRACT

This paper proposes a parallel architecture of a Parity Checksum Generator adopted for packet synchronization and error detection in the ITU-T Recommendation J.83 Annex B. The proposed parallel processing architecture removes a performance bottleneck occurred in a conventional serial processing architecture, leading to significant decrease in processing time for generating a Parity Checksum. The implementation results show that the proposed parallel processing architecture reduces the processing time by 83.1% at the expense of 16% area increase.

Ⅰ. 서 론

ITU-T J.83 Annex B는 동축 케이블을 통해 디지털 비디오와 오디오 신호의 전송에 대한 정보를 제공해주는 북미에서 널리 사용되고 있는 표준이다[1].

전송 시스템은 그림 1에 도시한 바와 같이 MPEG 프레이밍 블록, 순방향 오류 정정 부호기 블록 그리고 직교 진폭 변조 변조기 블록으로 구성되어 있다[1,2]. MPEG 프레이밍 블록은 MPEG 프레이머와 피포 버퍼로 구성되어 있고 입력 MPEG 데이터의 싱크워드 바이트를 패킷 동기화와 에러 검출을 위해 패리티 체크섬으로 교체한다. 순방향 오류 정정 부호기 블록은 에러에 강한 전송을 위해 Reed-Solomon 부호기(RSE), 인터리버, 랜덤화로 구성되어 있다. 직교 진폭 변조 변조기 블록은 트렐리스 부호화 변조(TCM)과 Root Raised Cosine(RRC) 필터로 구성되어 있다. 각 클럭 사이클마다 처리되는 비트 수는 MPEG 프레이밍 블록에서 1비트, 순방향 오류 정정 부호기 블록에서 7비트, 직교 진폭 변조 변조기 블록에서 64 직교 진폭 변조 방식은 6비트 256 직교

※ 본 연구는 전남대학교의 지원으로 수행되었습니다.
* 광주과학기술원 정보기전공학부, 통신시스템연구실, ** 전남대학교 치의학연구소(hjlim@chonnam.ac.kr)
논문번호 : KICS2009-03-098, 접수일자 : 2009년 3월 9일, 최종논문접수일자 : 2009년 5월 14일

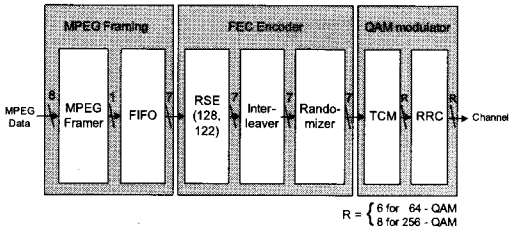


그림 1. 전송 시스템의 블록 다이어그램
Fig. 1. Basic block diagram of MPEG-2 TS transmission systems

진폭 변조 방식은 8비트이다. MPEG 프레이밍 블록은 1비트를 처리하므로 7비트를 처리하는 순방향 오류 정정 부호기 블록과 6 또는 8 비트를 처리하는 직교 진폭 변조 변조기와 데이터 처리 속도 차이로 인해 병목 현상이 발생한다. 이 병목 현상을 피하기 위해 MPEG 프레이밍 블록에 빠른 클럭을 고려할 수 있지만 이것은 높은 파워 소모가 발생할 뿐만 아니라 제한된 클럭 속도를 가지는 Field Programmable Gate Arrays(FGPA)에 적용하기에는 한계가 있다. 느린 처리 속도는 MPEG 프레이밍 블록에 있는 직렬 처리 패리티 체크섬 생성으로부터 발생하므로 병렬 구조로 기존의 직렬 구조로부터 발생하는 병목 현상을 제거할 수 있다[3]~[5].

이 논문에서 패리티 체크섬 생성기를 위한 8계층의 병렬 처리 구조가 제안되었다. 제안된 병렬 처리 구조는 16%의 면적 증가로 각 전송 패킷의 처리 속도를 83.1%나 감소시킨다.

Ⅱ장에서는 표준에 나와 있는 패리티 체크섬 생성을 위한 직렬 처리 구조를 소개하고 Ⅲ장에서는 제안된 병렬 처리 구조를 설명한다. Ⅳ장에서는 직렬 구조와 병렬 구조에 대한 분석과 합성 결과를 보여 주고 마지막으로 Ⅴ장에서는 결론을 맺는다.

Ⅱ. ITU-T J.83 Annex B에서 패리티 체크섬 생성기

패리티 체크섬은 1,496비트에 대해서 계산되고 즉시 MPEG-2 전송 패킷 내용(싱크 바이트는 제외) 앞에 위치한다[1,6]. 그림 2에 도시한 바와 같이 패리티 체크섬 생성기는 Linear Feedback Shift Register (LFSR) 블록, 신드롬 생성 블록, 유한임펄스응답 여과 블록 그리고 오프셋 덧셈 블록 등 4개의 블록으로 구성되어 있다.

부호화 과정에서 모든 메모리 요소들이 0의 값을 가지기 위해 LFSR 블록이 초기화되고 그 다음 1,496 비트의 MPEG-2 전송 패킷 패이로드가 LFSR 블록으로 이동된다. 1,496비트를 받은 후에 부호기 입력이 0으로 되고 그림 2에서 가장 위쪽에 위치하고 있는 스위치에 의해 초기화 및 이동과정이 수행된다. 신드롬 생성 블록은 8번의 추가적인 이동에 의해 8비트의 신드롬을 계산한다. 이 8비트의 결과는 유한임펄스응답 여과 블록으로 전해진다. 8비트의 신드롬이 전해지기 이전에 부호기 체크섬을 생성하기 위해 유한임펄스응답 여과 블록이 모두 0의 상태로 초기화된다. 보다 나은 자기상관 특성을 위해 오프셋 덧셈 블록에서 67(16진법)의 수의 오프셋을 이 체크섬에 더하고 유효한 코드 워드가 존재할 때 신드롬 디코딩 연산에서 47(16진법)의 수의 결과가 생산되도록 해준다[1]. 오프셋이 더해진 8비트의 패리티 체크섬의 최상위 비트가 먼저 전송되고 1,496 비트의 패이로드가 전송된다.

Ⅲ. 패리티 체크섬 생성을 위한 병렬 처리 구조

패리티 체크섬 생성기의 처리 속도는 기존의 직렬 구조를 병렬 구조로 바꿈으로써 증가될 수 있다. III장에서는 ITU-T J.83 Annex B 표준에서 패리티 체크섬 생성을 위해 병렬 처리 구조가 제안된다.

3.1 직렬 패리티 체크섬 생성기의 다항식 표현

직렬패리티 체크섬 생성기는 다항식으로 표현될 수 있고 가장 잘 알려진 방법이 이진 수열로 표현하는 방법이다[7]. 마지막 바이트가 모두 0인 MPEG-2 TSP의 1,504 비트 데이터를 $M=(m_{1503}, m_{1502}, ..., m_0)$라고 하자. 여기서 $m_i \in GF(2), i=0,...,1503 m_j = 0,...,7$이다. 8비트 패리티 체크섬 수열은 $P=(p_7, p_6, ..., p_0)$라고 하자. 여기서 $p_i \in GF(2), i=0,...,7$이다. 메시지와 패리티 체크섬 수열은 각각 메시지 다항식 $m(X)=m_{1503}X^{1503}+m_{1502}X^{1502}+\cdots+m_8X^8$과 패리티 다항식 $p(X)=p_7X^7+p_6X^6+\cdots+p_0$의 계수로 표현될 수 있다. 이와 비슷하게 전송기에 있는 LFSR 블록의 출력을 표현하는 다항식 $l(X)$는 다음과 같이 표현된다.

$$l(X)=\sum_{k=0}^{1503} l_k X^k = Q(\frac{X^8 m(X)}{g(X)}) \qquad (1)$$

여기서 생성기 g(X)는 $g(X)=X^8+X^6+X^5+X+1$이고 $Q(\bullet)$는 다항식 나눗셈에서 얻어진 몫을

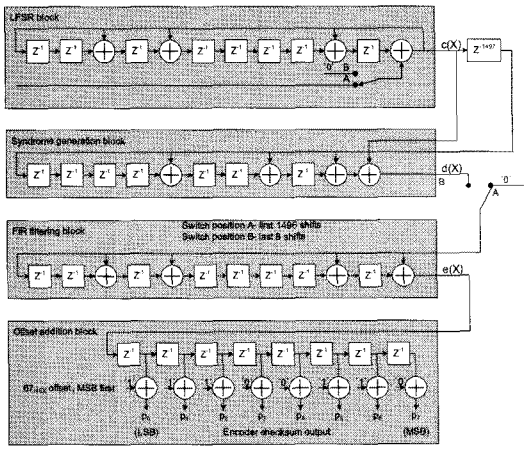


그림 2. 패리티 체크섬 생성을 위한 기존의 직렬 처리 구조
Fig. 2. Serial processing architecture of parity checksum generation in transmitter

의미한다[1]. 신드롬 생성 블록의 출력 다항식 s(X)는 다음과 같이 표현된다.

$$s(X)=\sum_{k=0}^{1503}s_kX^k = Q(\frac{l(X)[X^{1504}+X^7+X^6+X^4+1]}{X^{1504}}) \quad (2)$$

유한임펄스응답 여과 블록의 출력 다항식 f(X)도 다음과 같이 주어진다.

$$f(X)=\sum_{k=0}^{1503}f_kX^k = Q(\frac{[\sum_{i=0}^{7}s_iX^i][X^8+X^7+X^3+X^2+1]}{X^8}) \quad (3)$$

식 (3)에서 신드롬 생성 블록의 출력 수열의 오직 마지막 바이트만 사용된다. 마지막으로 오프셋 덧셈 블록의 출력 다항식 p(X)는 다항식 f(X)를 사용하여 다음과 같이 표현할 수 있다.

$$\begin{aligned}p(X)&=\sum_{k=0}^{1503}p_kX^k\\&=[f_7X^7+f_6X^6+\cdots+f_0]\\&\quad+[X^6+X^5+X^2+X+1]\end{aligned} \quad (4)$$

3.2 병렬 패리티 체크섬 생성기

8계층 병렬 처리 구조가 패리티 체크섬 생성기에 적합하다. 전송 시스템에서 256 직교 진폭 변조가 사용된다면 각 클럭 사이클 당 최대 처리 속도가 8비트이기 때문이다. 이 병렬 처리 구조는 각 클럭 사이클마다 8개의 출력 수열을 생성한다. 이런 병렬 처리 구조에서 다항식 (1), (2), (3), 그리고 (4) 의 계수 벡터는 GF(2)에서 8차원 벡터 스페이스로 다뤄질 수 있다. 다항식 p(X)의 8개의 계수는 8비트의 이진 계수 벡터 $P=[p_7,p_6,...,p_0]^T$를 형성한다. 계수 벡터 p의 벡터 형태의 표현은 식 (4) 로부터 얻을 수 있다.

$$p=f\oplus o \quad (5)$$

여기서 계수 벡터 $f=[f_7,f_6,...,f_0]^T$는 다항식 f(X)의 8개의 연속적인 계수에 해당하고 오프셋 덧셈 블록과 관련된 계수 벡터 o는 $[01100111]^T$이다. 심볼 $\oplus$는 GF(2)상에서 두 벡터의 덧셈 연산자이다. 식(5)에 있는 계수 벡터 f는 식 (3)에서 다항식 곱셈에 의해 얻어진다.

$$\begin{aligned}f(X)&=s_7X^7+(s_7+s_6)X^6\\&\quad+(s_6+s_5)X^5+(s_5+s_4)X^4\\&\quad+(s_4+s_3)X^3+(s_7+s_3+s_2)X^3\\&\quad+(s_7+s_6+s_2+s_1)X\\&\quad+(s_6+s_5+s_1+s_0)\end{aligned} \quad (6)$$

식 (6)으로부터 계수 벡터 f는 다음과 같이 표현될 수 있다.

$$f=E\otimes s \quad (7)$$

계수 벡터 s는 s(x)의 다항식 계수인 8개의 원소로 구성되어 있고 다항식 표현 f(X)로부터 행렬 E는 다음과 같이 얻을 수 있다.

$$E=\begin{bmatrix}10000000\\11000000\\01100000\\00110000\\00011000\\10001100\\11000110\\01100011\end{bmatrix}$$

여기서 심볼 $\otimes$은 GF(2)상에서 하나의 벡터와 행렬의 곱셈 또는 두 개의 행렬의 곱셈을 하는 연산자이다. 비슷하게 식 (7)에서 계수 벡터 s는 식 (2)를 통해 다음과 같이 표현될 수 있다.

$$s=F\oplus I_0\otimes l_{187} \quad (8)$$

k번째 계수 벡터는 $l_k(k=0 \text{or} 187)$는 $[l_{8\cdot(187-k)}+7\, l_{8\cdot(187-k)+6}\, l_{8\cdot(187-k)}]$이며 이것은 LFSR 블록의 출력 수열의 k번째 바이트와 관련이 있고 행렬 F는 다음과 같이 주어진다.

$$F=\begin{bmatrix}00000000\\10000000\\11000000\\01100000\\10110000\\01011000\\00101100\\00010110\end{bmatrix}$$

식 (8)에서 계수 벡터 s를 얻기 위해 LFSR 블록에서 출력 수열의 오직 첫 번째와 마지막 바이트만 필요하다. 계수 벡터 l_k를 얻기 위해 k번째 메시지 수열 $m_k=m_{8\cdot(187-k)+7}\, m_{8\cdot(187-k)+6}\cdots m_{8\cdot(187-k)}$라고 하고 식 (1)의 양쪽 면에 g(X)를 곱하면 다음과 같다.

$$g(X)l(X)=X^8m(X)+R(\frac{X^8m(X)}{g(X)}) \quad (9)$$

여기서 연산자 $R(\cdot)$는 나머지를 계산한다. m_k와 l_k사이의 관계는 식 (9)와 같이 주어지고 행렬로 표현될 수 있다.

생성기 다항식 함수 g(X)을 포함하는 행렬 G와 H는 다음과 같다.

$$\begin{bmatrix}G&0&0&0&0&0&0&0\\HG&0&0&0&0&0&0&0\\0&HG&0&0&0&0&0&0\\0&0&HG&0&0&0&0&0\\0&0&0&HG&0&0&0&0\\0&0&0&0&HG&0&0&0\\0&0&0&0&0&HG&0&0\\0&0&0&0&0&0&HG&0\\0&0&0&0&0&0&HG&G\end{bmatrix}\begin{bmatrix}l_0\\l_1\\l_2\\l_3\\l_4\\l_5\\l_6\\l_7\end{bmatrix}=\begin{bmatrix}m_0\\m_1\\m_2\\m_3\\m_4\\m_5\\m_6\\m_7\end{bmatrix} \quad (10)$$

$$G=\begin{bmatrix}10000000\\11000000\\01100000\\00110000\\00011000\\10001100\\11000110\\01100011\end{bmatrix},H=\begin{bmatrix}10110001\\01011000\\00101000\\00010100\\00001010\\00000101\\00000010\\00000001\end{bmatrix}$$

행렬 0는 모든 원소가 0의 값을 가지는 영 행렬이다. 벡터 m_{187}은 다항식 m(X)의 정의에 의해 제로 벡터이다. 벡터 l_0는 식 (10)에 있는 행렬의 첫 번째 열에 의해 다음과 같이 주어진다.

$$l_0=G^{-1}\otimes m_0 \quad (11)$$

행렬 G^{-1}는 행렬 G의 역함수이고 다음과 같이 표현된다.

$$G^{-1}=\begin{bmatrix}10000000\\11000000\\11100000\\11110000\\11111000\\01111100\\00111110\\00011111\end{bmatrix}$$

식 (10)에 있는 행렬 표현으로부터 m_1은 다음과 같이 주어진다.

$$(H\otimes l_0)\oplus(G\otimes l_1)=m_1 \quad (12)$$

식 (11)에 있는 $G^{-1}\otimes m_0$을 식 (12)에 있는 l_0에 대입하고 식 (12)의 양변에 $H\otimes G^{-1}\otimes m_0$을 더해주면 다음과 같다.

$$(G\otimes l_1)=m_1\oplus(H\otimes G^{-1}\otimes m_0) \quad (13)$$

G^{-1}를 식 (13)의 양변에 곱하면 다음과 같은 결과가 나온다.

$$l_1=G^{-1}\otimes m_1 \quad (14)$$

여기서 m_1은 와 $m_1=m_1\oplus(H\otimes G^{-1}\otimes m_0)$같이 주어진다. 일반적으로 식 (10)으로부터 계수 벡터 $l_j(j=0,...,187)$는 다음과 같이 주어진다.

$$l_j=G^{-1}\otimes m_j \quad (15)$$

여기서 m_j는 다음과 같다.

$$\begin{aligned}m_j&=m_j\oplus(H\otimes G^{-1}\otimes m_{j-1}),j=1,...,187\\m_0&=m_0\end{aligned} \quad (16)$$

식 (5)에 있는 계수 벡터 p를 구하기 위해 식 (7),(8),그리고 (15)에서 얻은 계수 벡터를 대입함으로써 계수 벡터 p는 다음과 같이 표현된다.

$$p=(E\otimes F\otimes G^{-1}\otimes m_0)\oplus(E\otimes G^{-1}\otimes m_{187}) \quad (17)$$

$E\otimes G^{-1}$가 단위 행렬이기 때문에 계수 벡터 p는 다음과 같이 간단하게 된다.

$$p=(E\otimes F\otimes G^{-1}\otimes m_0)\oplus m_{187} \qquad (18)$$

3.3 패리티 체크섬 생성을 위한 병렬 처리 구조

식 (16)과 (18)로부터 패리티 체크섬 생성을 위한 병렬 처리 구조가 그림 3과 같이 그려진다. 그림은 식 (16)의 반복적 연산에 해당하는 피드백 부분과 식 (18)의 연산과 관련된 피드포워드 부분으로 구성되어 있다. $T_{HG^{-1}}m_k$로 표현된 블록은 행렬 $T_{HG^{-1}}$와 입력 벡터 m_k의 행렬 곱셈을 수행한다. $T_{HG^{-1}}$는 다음과 같이 주어진다.

$$T_{HG^{-1}}=H\otimes G^{-1}=\begin{bmatrix}10001111\\11001000\\01100100\\10110010\\11011001\\01100011\\00111110\\00011111\end{bmatrix}$$

블록 Z^{-1}는 입력 벡터 데이터를 하나의 클럭 사이클만큼 지연시킨다. 피드포워드 부분에서 $T_{EFG^{-1}}$ m_0+o블록은 입력 벡터 m_0와 행렬 $T_{EFG^{-1}}$의 곱셈 연산을 수행하고 계수 벡터 o를 더한다. 행렬 $T_{EFG^{-1}}$는 다음과 같다.

$$T_{EFG^{-1}}=E\otimes F\otimes G^{-1}=\begin{bmatrix}00000000\\10000000\\11000000\\01100000\\10110000\\01011000\\00101100\\00010110\end{bmatrix}$$

블록 D는 블록 $T_{EFG^{-1}}m_0+o$의 출력 벡터를 저장한다.

부호화 과정에서 블록 Z^{-1}가 초기화되어 메모리 요소들이 모두 0의 값을 가진다. 결과적으로 식 (16)에 나와 있는 것처럼 벡터 m_0은 벡터 m_0과 첫 번째 클럭 사이클에서 동일하다. 피드백 부분은 다음 187클럭 사이클 동안 식 (16)에 나와 있는 반복적 연산을 수행한다. 188번째 클럭 사이클에서 생성된 m_{187}을 블록 D에 저장되어 있는 벡터 데이터에 더함으로써 패리티 체크섬이 생성된다.

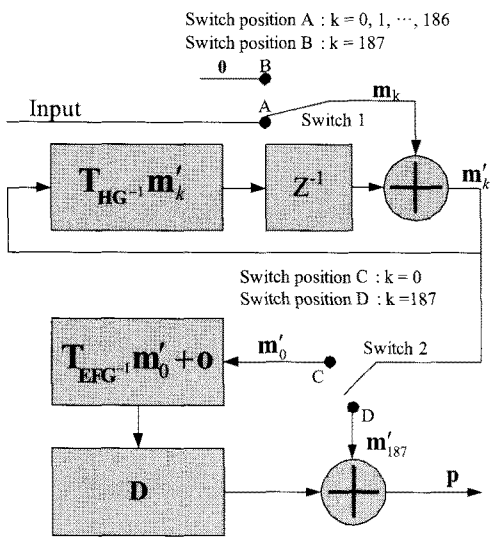


그림 3. 패리티 체크섬 생성을 위한 병렬 처리 구조
Fig. 3. Parallel processing architecture for parity checksum generation

Ⅳ. 실험 결과

패리티 체크섬 생성을 위해 제안된 병렬 처리 구조와 기존의 직렬 처리 구조를 Altera EP1S25F672C6 FPGA칩을 사용하여 구현하였다. 기능 검증은 Mentor Graphics Corporation의 ModelSim 소프트웨어에 의해 수행되었다.

표 1은 컴파일 결과를 하드웨어 복잡도, 처리 시간, 전력 소모 면에서 보여준다. 로직 셀의 수는 하드웨어 복잡도를 의미하고 최대 클럭 주파수의 역수인 최소 클럭 주기는 회로의 동작 속도를 나타낸다[10]. 처리 시간은 최소 클럭 주기와 TSP를 처리하는데 필요한 클럭의 수를 곱해서 계산된다. 전력 소모를 비교할 때 두 구조는 초당10.6K TPS 즉 16Mbps의 같은 처리 속도를 가진다고 가정하였고 이 처리 속도는 직렬 구조에서 클럭 주파수를 16MHz로, 병렬 구조에서 2MHz로 맞춤으로써 얻어진다.

패리티 체크섬 생성의 경우 제안된 병렬 처리 구조는 신드롬 생성 블록의 입력에 사용되는 1497개의 레지스터를 제거하기 때문에 직렬 처리 구조보다 더 적은 로직 셀을 필요로 한다. 실험 결과는 제안된 병렬 구조에서 패리티 체크섬 생성에 필요한 로직 셀의 수가97%나 줄어든다는 것을 보여준다.

표 1. 실험 결과
Table 1. Comparisions of implemenation results

Architecutures	Parity checksum generator	
	Serial [1]	Parallel
Logic cells	1554	50
Minimum clock period (nanosecond)	8.10	5.50
Processing time (nanosecond)	12182	1034
Power dissipation (mW)	8.12 @16MHz	0.02 @2MHz

제안된 병렬 처리 구조는 패리티 체크섬 생성과정에서 아주 적은 전력을 소모한다. 이것은 병렬 구조의 클럭 주파수를 직렬 구조의 1/8로 줄임으로써 가능하다. 또한 패리티 체크섬 생성에서 직렬 처리 구조에 비해 훨씬 더 적은 셀들이 사용되는 것이 전력 소모의 감소에 기여했다. 실험 결과는 패리티 체크섬 생성에서 병렬 처리 구조가 직렬 처리 구조에 비해 전력이 99%나 적게 소모된다는 것을 보여준다.

패리티 체크섬 생성을 위한 기존의 제안된 병렬 처리 구조는 직렬 처리 구조에 비해 더 긴 최소 클럭 주기를 가진다[8,9]. 하지만 본 논문에서 제안된 병렬 처리 구조는 더 짧은 최소 클럭 주기를 가질 수 있다.

이 병렬 처리 구조는 8비트 카운터에 의해 컨트롤되는 스위치에 필요한 간단한 로직 회로가 필요하다. 반면에 직렬 처리 구조는 11 비트의 카운터를 컨트롤하기 위해 상당한 사이즈의 조합 논리가 필요하다. 처리 시간은 직렬 처리 구조에서는 1504를, 병렬 처리 구조에서는 188을 최소 클럭 주기와 곱함으로써 얻어진다. 결과적으로 제안된 병렬 구조는 패리티 체크섬 생성에서 처리 시간이 92%나 줄어든다.

V. 결 론

디지털 비디오와 오디오 신호를 동축 케이블을 통해 전송할 때 생기는 병목 현상을 제거하기 위해 새로운 병렬 처리 구조가 제안되었다. 이 병렬 처리 구조는 ITU-T J.83 Annex B 표준을 따른다. 실험 결과는 제안된 병렬 처리 구조가 추가적인 하드웨어 없이 처리 시간을 상당히 줄여준다는 것을 보여준다.

감사의 글

This work was financially supported by Chonnam National University, 2005.

참 고 문 헌

[1] ITU-T, "Digital Multi-Programme Systems for Television Sound and Data Services for Cable Distribution," (ITU-T Recommendation J.83, 1997)

[2] Xilinx, "J.83 Annex B 변조기 v1.3," (Xilinx Logicore, 2004)

[3] G. Albertengo and R. Sisto, "Parallel CRC 생성," IEEE Micro, 10(5), pp.63-73, 1990.

[4] T. B. Pei and C. Zukowski, "High-Speed Parallel CRC Circuits in VLSI," IEEE Trans. on Communications, 40(4), pp.635-657, 1992.

[5] C. Chao and K. P. Keshab, "High-Speed Parallel CRC Implementation Based on Unfolding, Pipelining, and Retiming," IEEE Trans. on Circuits and Systems, 53(10), pp. 1017-1021, 2006.

[6] J. Yujen, "Erroneous MPEG packet synchronization in the MCNS/SCTE/ITU-T J.83 Annex B standard," IEEE Trans. on Consumer Electronics, 44(3), pp.963-968, 1998.

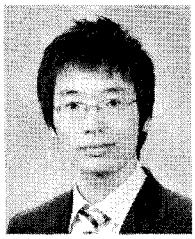
[7] S. Peter, "Error Control Coding from Theory to Practice (England: John Wiley and Sons Ltd. 2002).

[8] M. Sprachmann, "Automatic 생성 of Parallel CRC Circuits," IEEE Design and Test of Computer, Vol.18, Issue3, pp.108-114, May 2001.

[9] G. Campobello, G. Patane, and M. Russo, "Parallel CRC Realization," IEEE Trans. on computers, Vol.52, No.10, pp.1312-1319, Oct., 2003.

[10] Altera, "Stratix Device Family Data Sheet," Altera Inc., Jan., 2006.

이 종 엽 (Jong-yeop Lee) 준회원

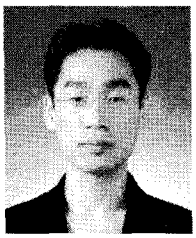


2008년 부산대학교 학사
2008년~현재 광주과학기술원 석사 과정
<관심 분야> 디지털 통신 시스템 이론 및 VLSI 구현

하 동 수 (Dongsoo Har) 정회원

1986년 서울대학교 학사
1988년 서울대학교 석사
1997년 Polytechnic Univ. 박사
2003년~현재 광주과학기술원 부교수
<관심분야> 멀티미디어 서비스 시스템을 위한 채널 용량 증가, 노이즈와 간섭하에서의 시스템의 성능 분석, 멀티미디어 시스템의 SoC 설계

홍 언 표 (Eon-pyo Hong) 정회원



2002년 순천대학교 학사
2005년 광주과학기술원 석사
2005년~현재 광주과학기술원 박사 과정
<관심분야> 디지털 통신 이론, 멀티미디어 통신과 VLSI 구현

임 회 정 (Hoi-Jeong Lim) 정회원

1988년 이화여자대학교 학사
1990년 이화여자대학교 석사
1994년 Columbia Univ. 석사
2001년 Columbia Univ. 박사
2005년~현재 전남대학교 조교수
<관심분야> 수치 해석 및 통계 분석, 코호트 연구에서의 통계분석, 표본수가 작을 경우의 통계기법, 상관관계가 있는 다변량 결과변수의 통계기법