論文2003-40TC-11-7

순서적 역방향 상태천이 제어에 의한 역추적 비터비 디코더

(Trace-Back Viterbi Decoder with Sequential State Transition Control)

鄭 且 根 *

(Cheong Cha-Keon)

요 약

본 논문에서는 역추적 비터비 디코더의 순서적 역방향 상태천이 제어에 의한 새로운 생존 메모리 제어와 복호기법을 제안한다. 비터비 알고리즘은 채널오류의 검출과 정정을 위한 부호기의 상태를 추정해서 복호하는 최우추정 복호기법이다. 이 알고리즘은 심볼간 간섭의 제거나 채널등화 등 디지털 통신의 광범위한 분야에 응용되고 있다. 반복연산의 과정을 내포하고 있는 비터비 디코더에서 처리속도의 향상과 함께 VLSI 칩 설계시 점유면적의 삭감을 통한 칩 사이즈의 축소 및 소비전력의 저감 등을 달성하기 위해서는 새로운 구조의 ACS 및 생존 메모리 제어에 관한 연구가 요구되고 있다. 이를 해결하기 위한 하나의 방안으로, 본 논문에서는 역추적 기법에 의한 복호과정에서 역방향 상태천이의 연속적인 제어에 의한 자동 복호 알고리즘을 제안한다. 제안방식은 기존의 방법에 비해 전체 메모리 사용양이 적을 뿐만아니라 구조가 간단하다. 또한, 메모리 억세스 제어를 위한 주변 회로구성이 필요없고, 메모리 억세스를 위한 대역폭을 줄일 수 있어 칩 설계시 area-efficiency가 높고 소비전력이 적어지는 특성이 있다. 시스톨릭 어레이 구조 형태를 갖는 병렬처리 구성과, 채널잡음을 포함한 수신 데이터로부터의 복호와 구체적인 응용 시스템에 적용한 결과를 제시한다.

Abstract

This paper presents a novel survivor memeory management and decoding techniques with sequential backward state transition control in the trace back Viterbi decoder. The Viterbi algorithm is an maximum likelihood decoding scheme to estimate the likelihood of encoder state for channel error detection and correction. This scheme is applied to a broad range of digital communication such as intersymbol interference removing and channel equalization. In order to achieve the area-efficiency VLSI chip design with high throughput in the Viterbi decoder in which recursive operation is implied, more research is required to obtain a simple systematic parallel ACS architecture and survivor memory management. As a method of solution to the problem, this paper addresses a progressive decoding algorithm with sequential backward state transition control in the trace back Viterbi decoder. Compared to the conventional trace back decoding techniques, the required total memory can be greatly reduced in the proposed method. Furthermore, the proposed method can be implemented with a simple pipelined structure with systolic array type architectue. The implementation of the peripheral logic circuit for the control of memory access is not required, and memory access bandwidth can be reduced. Therefore, the proposed method has characteristics of high area-efficiency and low power consumption with high throughput. Finally, the examples of decoding results for the received data with channel noise and application result are provided to evaluate the efficiency of the proposed method.

Keywords : 비터비 디코더, 생존 메모리, 역추적 기법, 역방향 상태천이, 시스톨릭 어레이 구조

* 正會員, 湖西大學校 電氣情報通信工學部

(Hoseo Univ., The School of Electrical Engineering)

接受日字:2003年7月23日, 수정완료일:2003年11月1日

Ⅰ. 서 론

통신기술의 획기적인 진전에 따라 데이터 전송속도가 고속화되어, 수십 Mbps 급 무선통신 시스템이 이미 실용화되어 있다[1]. 또한, 최근에는 100 Mbps 이상의 무선 통신 시스템 구현을 위한 연구가 활발히 진행되고 있으며[2], 무선 대역폭의 높은 사용밀도로 인해 인접 채널간의 신호간 간섭이나 멀티패스 페이딩 등의 영향이 증대하고 있어, 보다 강인한 채널오류의 검출과 정정에 대한 필요성으로 인해 관련된 많은 연구가 이루어져 왔다[1, 2].

길쌈부호기(convolutional coder)는 간단한 구조와 높은 부호이득을 갖는 가변 전송율을 구현할 수 있는 코더의 개발과 최적복호가 가능한 이론적인 알고리즘이 알려져 있어, 다양한 디지털 통신 시스템의 채널 부호화기로 사용되고 있다[6]. 비터비 알고리즘(Viterbi Algorithm : VA)은 길쌈부호기에 의한 채널부호의 최적 복호를 구현하기 위한 방법으로, 그 응용범위가 채널오류의 검출 및 정정뿐만 아니라 채널 등화 및 심볼간 간섭(Inter-Symbol Interference, ISI)의 제거, 저장매체의 재생 등으로 확대되고 있다.

VA는 채널로부터 잡음이 포함된 수신 데이터들의 모든 경로를 탐색한 후, 부호어와 유사성(likelihood)이 가장 높은 경로를 따라 상태를 선택하여 데이터를 복호하는 방법으로, 경로 탐색시 유사성이 적은 경로를 제거하고 최적의 유사성을 갖는 경로만을 탐색해서 수신 데이터로부터 부호기의 상태를 추정해서 검출하는 최우추정(Maximum Likelihood, ML) 복호 알고리즘이다. 따라서, VA에 의한 디코더는 최적의 유사성을 갖는 경로를 탐색하기 위해서 반복과정이 요구되고, 탐색에 요구되는 경로의 수에 따라 고속처리에 많은 제한이 동반된다[3]. 최근에 100Mbps 이상의 고속 통신을 위한 비터비 디코더를 구현하기 위한 방법으로, 멀티 스텝 Add-Compare-Select (ACS) 유니트의 파이프 라인 및 병렬 구조를 채택함으로써 140Mbps의 처리속도를 실현하고 있다[15, 16].

일반적으로 VA에 의한 복호 레이트는 반복 ACS의 연산과 최적 경로의 탐색에 요구되는 연속적인 생존 메모리(Survivor Memory : SM)의 제어에 의해 제한된다. 따라서, 소비전력을 감소시키고 시스템 통합 칩내에 비터비 디코더를 내장시키기 위해서는 ACS 유니트뿐만 아니라, 칩 내 점유공간의 면적을 줄일 수 있는 Area-Efficient 디코더 구현을 위한 ACS 및 SM의 제어에 관한 연구가 필요하다[9, 14, 17]. 현재 비터비 디코더에서 사용되는 SM 제어의 방법은 생존경로의 저장에 사용되는 메모리의 구성방법에 따라 레지스터 교환 (Register Exchange : RE) 방식과 역추적 (Trace-Back : TB) 방식으로 분류된다[5, 6].

RE 방식은 레지스터를 사용해서 각 상태의 생존경로에 관한 정보를 저장하고, 각 복호 사이클마다 저장된 전체 레지스터의 정보를 갱신해서 복호하는 방법으로 복호 지연시간, 메모리 사이즈, regularity 등의 파라미터들이 복호에 영향을 크게 미치는 시스템에 주로 사용되고 있다. 즉, RE 방식은 SM 제어 구조가 비교적 간단해서 짧은 구속장(contstraint length)를 갖는 코더의 복호에 보편적으로 사용된다. 그러나, 복호과정의 매 사이클마다 전체 레지스터의 값을 교환해야 하므로 광대역의 메모리 억세스 대역폭이 필요하게 되고, 이는 소비전력의 증가와 VLSI 구현시 칩 사이즈를 증가시키는 문제가 있다.

TB 방식은 일정한 크기에 해당하는 메모리에 각 상태의 생존경로에 관한 정보를 저장한 후, 역추적의 방법으로 ML 경로를 탐색해서 복호하는 방법으로, 구속장이 비교적 크고, 높은 코딩 이득이 요구되는 시스템에 주로 사용된다. 또한, TB 방식은 온 칩 레지스터 대신에 RAM과 같은 외부 메모리에 의한 구현이 가능하며 전력소모가 RE 방식에 비해 적은 이점을 갖는다. 높은 throughput을 및 area-efficient 구조의 TB 디코더를 구현하기 위해 많은 연구가 진행되어 왔다. 즉, 경로 메모리에 저장된 데이터를 메모리 어드레싱의 포인터로 사용하는 방법이[8] 제안된 이래, O. Collins 등은[7] 멀티프로세서 비터비 디코더 구현을 위한 보다 자세한 방법을 기술하고 3개의 메모리 버퍼를 사용한 3-포인터 TB 방식에 의한 디코더를 구현했다. 일반화된 k-포인터 알고리즘[9, 10], 높은 throughput의 달성과 칩 내 점유 면적의 비율을 줄이기 위한 방법으로 RE에 의한 전처리와 TB에 의한 복호방법을 적용한 혼합기법[12], 기하학적 메모리 제어에 관한 해석[11] 등을 들 수 있다. 높은 throughput과 칩 구현의 효율을 향상시키기 위해 다양한 방법들이 강구되어 RE 방법보다 널리 사용되고 있으나, TB 방법은 그 특성상 사용되는 메모리 사이즈가 RE 방식에 비해 방대해지고, 생존경로에 관한 정보의 저장과 역추적 및 복호 등의 과정이 동시에 수행되어야 하므로, 메모리 억세스를 위한 제어가 복잡하게 될 뿐만

아니라 복호 지연시간이 길어지는 근본적인 제약을 동반하고 있다.

본 논문에서는 TB 방식에 의한 비터비 디코더의 문제점을 개선하고 칩 구현을 보다 용이하게 할 수 있도록 하기 위해, 기존의 기법과는 다른 새로운 TB 방식의 비터비 디코딩 기법을 제안한다. 제안 방법은 임의의 사이클 타임에서 시작된 각 상태에서 트렐리스도 상의 생존경로를 따라 역방향으로 상태천이를 추적하면, 일정한 시간 후에는 임의의 하나의 상태로 통합(merge)될 확률이 매우 높고, 역방향 상태천이에 관한 정보로부터 부호기의 정보를 복호할 수 있는 ML 추정 복호기의 기본 성질을 이용한 것이다. 즉, ACS에서 출력되는 생존경로에 관한 정보의 저장과 동시에 이 정보를 기반으로 각 상태의 역방향 천이를 순서적으로 변화되게 함으로서, 역추적과 동시에 데이터의 복호가 자동으로 수행되도록 한다. 제안 방식의 주요 특징을 요약하면 다음과 같다.

- 복호과정의 각 사이클 타임에서 최소 메트릭을 갖는 최적상태의 검출을 위한 비교기를 사용하지 않으므로, 회로가 간단하다.
- SM의 메모리 억세스 제어를 위한 주변 제어회로의 구성이 필요 없다.
- SM에 사용되는 메모리의 양이 기존 TB 방식보다 적게 사용된다.
- 생존경로에 관한 정보의 단순한 천이만으로 복호과정이 이루어지므로 메모리 억세스를 위한 대역 폭을 줄일 수 있어, 칩 설계에서 Area-Efficiency가 높고, 전력소모가 적다.

이하 본 논문에서는, 먼저 2절에서 VA의 기본동작에 관해 제안 방식을 기술하고 이해하는데 필요한 내용을 중심으로 간단히 언급한다. 3절에서는 TB 방식에 의한 복호과정과 기존 대표적인 TB 방식을 중심으로 특징을 간략히 기술하고, 제안 알고리즘의 동작원리 및 구성에 관해 상세히 설명한다. 다음으로 4절에서는 제안방식의 시스톨릭 어레이 구조의 사용에 의한 제안방법의 병렬 파이프라인 형태의 처리가 가능한 시스템 구현과 구체적인 예를 들어 동작과정의 결과를 제시하고, 실제 시스템에의 응용으로 IEEE 802.11a 고속 무선 랜에 적용한 VHDL 코딩 결과를 제공한다. 끝으로 본 논문의 간단한 결론을 5절에서 기술한다.

Ⅱ. 비터비 알고리즘

본 논문에서는 비터비 알고리즘에 의한 길쌈부호기의 복호과정을 3절에서 기술하는 제안 알고리즘을 이해하고 기술하는데 필요한 내용을 중심으로 간략히 기술한다. 보다 상세한 내용에 관해서는 참고문헌[3], [4], [5], [6] 등을 참조하기 바란다.

비터비 디코딩은 채널잡음이 포함된 수신 데이터로부터, 부호화기의 모든 상태에 대해 유사성이 가장 높은 상태천이를 추정해서 데이터를 복호하는 방법이다[3]. 이와 같은 VA에 의한 복호과정은 트렐리스도를 사용해서 간결하게 기술할 수 있다. 즉, VA는 주어진 트렐리스도의 각 상태에서 최소의 가중치를 갖는 경로를 탐색하고, 이를 송신 데이터와 유사성이 가장 높은 상태시퀀스에 해당하는 생존경로로 결정하는 것이다. 부호율 $R=\frac{\nu}{n}$ 인 길쌈부호기로부터 생성되는 부호어 $v_{j,k}$는 다음 식 (1)으로 발생된다.

$$v_{j,k}=\sum_{i=0}^{m} g_{i,j}\,u_{k-i}\ ,\ \text{단}\ \ j=0.1.\cdots,n-1 \quad (1)$$

여기서 $\{g_{i,j}\}$는 "0" 또는 "1"의 값을 갖는 차수 (degree) m인 생성다항식의 파라미터이고, u_k는 부호화기에 입력되는 정보비트이다. 부호기에서 출력되는 부호어는 생성다항식과 부호기에 사용되는 메모리 또는 레지스터의 수 m의 함수로 결정된다. 이 때, $K=m+1$을 구속장이라 한다. 부호기의 상태(state)는 m 레지스터 내에 저장된 값으로 정해진다. 따라서, 서로 다른 상태의 수 N은 $N=2^m$이고, 트렐리스도에서 임의의 상태에서 출력되고 통합되는 가지(branch)의 수는 2^{ν}이다.

구속장 $K=3(m=2)$, 부호율 $R=1/2$인 간단한 경우

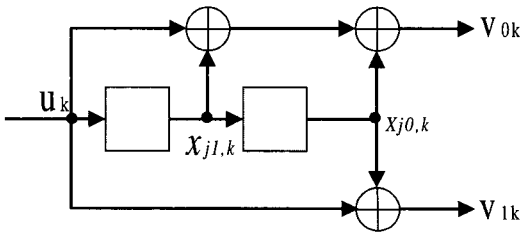


그림 1. (2,1/2) 길쌈부호기
Fig. 1. A (2,1/2) convolutional encoder.

의 (2,1/2) 길쌈부호기 구성의 예를 나타낸 것이 <그림 1>이다.

이 부호기의 생성다항식은 $G_0=111_2=7_8$, $G_1=101_2=5_8$ 이고, 서로 다른 상태 수는 $N=2^2=4$가 되고, 이들 각 상태 $s_{j,k}$ $(j=0,1,2,3)$는 다음으로 주어진다.

$$s_{j,k}=(x_{j1,k},\ x_{j0,k}) \tag{2}$$

<그림 1>에서 알 수 있는 바와 같이 $x_{j0,k}=x_{j1,k-1}$, $x_{j1,k}=u_{k-1}$의 관계가 성립한다. 각 상태에 입력되는 가지의 수는 $\nu=1$이므로 2개가 된다. 채널 잡음이 부가된 수신 데이터 열 $\mathbf{r}_k$ 로부터 부호기의 입력정보 u_k를 최적복호하는 VA는 송신 데이터 $\mathbf{v}_k$ 와의 유사성에 따라 다음과 같이 계산된 메트릭값이 최소인 경로를 선택하는 것이다.

$$\lambda_k^{(i,j)}=\| \mathbf{r}_k - \mathbf{v}_k \|_{(i,j)} \tag{3}$$

$\lambda_k^{(i,j)}$는 사이클 타임 $k-1$의 i번째 상태 $s_{i,k-1}$에서 사이클 타임 k의 j번째 상태 $s_{j,k}$로 천이되는 경우의 가지 메트릭(branch metric 또는 transition metric)으로, 이는 상태 $s_{j,k}$에 입력되는 i번째 가지의 메트릭에 해당된다. 비터비 디코더에서 가지 메트릭 $\lambda_k^{(i,j)}$의 연산을 수행하는 부분을 BMU(Branch Metric Unit)라고 하고, 실제 계산에는 유클리드(Euclidean) 거리나 해밍(Hamming) 거리 등이 연판정(soft decision) 또는 경판정(hard decision)의 적용에 따라 구분되어 사용된다[4, 5].

또한, 임의의 복호 사이클 시간에서 특정한 상태에 이르기까지의 최소 생존경로를 따라 누적된 가지 메트릭을 경로 메트릭(path metric) 또는 상태 메트릭(state metric)이라 한다. 사이클 타임 k의 j번째 경로 메트릭을 $\Gamma_{j,k}$이라 하고, <그림 2>에 나타낸 것처럼 사이클 타임 $k-1$의 t번째 상태와 i번째 상태로부터 천이되는 경우 $\Gamma_{j,k}$는

$$\Gamma_{j,k}=\min\{\Gamma_{t,k-1}+\lambda_k^{(t,j)},\ \Gamma_{i,k-1}+\lambda_k^{(i,j)}\},$$
$$\text{단}\ (t,j=0,1,2,3) \tag{4}$$

와 같이 사이클 타임 $k-1$에서의 경로 메트릭과 이 상태에 이르게 하는 가지 메트릭의 합이 최소가 되는 경로 및 가지 메트릭의 합으로 주어진다. 매 사이클 시간마다 각 상태에 도달하는 2^ν개의 경로 중에서 최소의 경로 메트릭이 되게 하는 경로를 특히 생존경로(survivor path)라 한다. 이와 같은 생존경로를 식별하는 정보를 결정비트(decision bit)라 한다. 즉, 사이클 타임 k에서의 j번째 생존경로에 대한 결정비트를 d_k^j라 하면, 이는 경로의 가지 메트릭이 사이클 타임 $k-1$의 상태에서 사이클 타임 k의 상태로 천이됨에 따라 부호기의 레지스터에서 빠져나가는 입력정보 $x_{j0,k-1}$에 해당된다. 이상의 경로 메트릭의 연산과 생존경로의 결정 및 이에 관한 결정정보를 출력하는 부분을 ACSU(Add-Compare-Select Unit)라 한다.

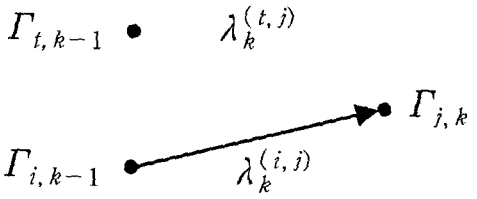


그림 2. 사이클 타임 $k-1$에서 k에 이르는 가지 메트릭과 경로 메트릭

Fig. 2. A branch metric and path metric at cycle time from $k-1$ to k.

VA에 의한 복호과정은 이 결정비트를 별도의 메모리에 저장하고 역추적하거나 매 사이클 스텝마다 메모리의 내용을 교환함으로서 최소 메트릭의 생존경로를 구하는 것으로 디코더내 SMU(Survivor Memory Unit) 부분에서 수행된다. <그림 3>은 이상의 과정으로 복호를 수행하는 비터비 디코더의 전체 구성을 나타낸 것이다.

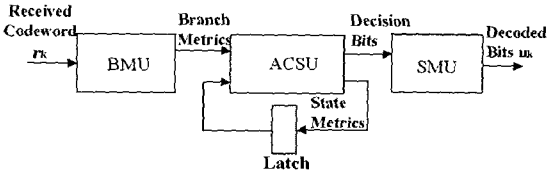


그림 3. 비터비 디코더의 전체 구성

Fig. 3. The overall structure of Viterbi decoder.

Ⅲ. 역방향 상태천이의 순서적 제어에 의한 역추적 디코더

1. 복호 깊이와 역추적 디코더

ACSU에서 출력되는 각 상태의 생존경로에 관한 결

정비트 d_k^s를 저장하는데 필요한 메모리를 생존 메모리(survivor memory)라 한다[3]. 구속장 $K=m+1$의 길쌈 부호기에서, 생존 메모리의 사이즈는 ACS가 수행되는 사이클마다 $2^{K-1}=2^m$의 단위로 증가하게 되고, 수행 사이클 횟수가 늘어남에 따라 필요한 메모리의 양은 폭발적으로 증가한다. 따라서, 이를 해결하기 위한 방법으로 적당한 크기의 수행사이클 단위마다 생존경로의 역추적을 수행하여, 최적경로를 결정하고 데이터를 복호하는 방법이 역추적에 의한 복호 기법이다[6]. 이와 같이 역추적에 필요한 메모리의 사이즈를 일정한 크기로 한정하는 것을 경로 메모리의 절삭(truncation)이라 하고[18], 메모리 사이즈를 한정시킴으로서 발생되는 에러를 메모리 절삭 오류(truncation error)라 한다[3]. 경로 역추적에 사용되는 메모리의 사이즈를 한정하지 않고 복호하는 최우추정 복호(Maximum Likelihood Decoding: MLD) 방법으로부터 발생할 수 있는 오류발생의 확률과 거의 동일한 복호오류 발생확률이 얻어지는 메모리 절삭 크기의 최소값을 복호 깊이(decoding depth) 또는 절삭길이(truncation length) $L_{\min}$이라 한다. $L_{\min}$의 값은 부호기의 구속장 K 값과 생성다항식에 의해 좌우된다. 일반적으로, $L_{\min} \simeq 4(K-1) \sim 5(K-1)$로 주어지지만[3, 19], 펑처링 등으로 보다 높은 부호율 R의 부호기인 경우 더 큰 $L_{\min}$의 값을 사용해야 된다.

역추적 기법에 의한 최적 생존경로의 결정은 $L_{\min}$이상으로 경로 메모리를 역추적하면, 역추적이 시작되는 시점의 모든 상태는 공통된 하나의 최적상태에 통합될 확률이 매우 높은 이론에 기반을 두고 있다. 즉, 모든 생존경로는 $L \geq L_{\min}$이상의 경로를 역추적하면, 모든 상태는 하나의 최적 상태에 통합될 확률이 매우 높다. 따라서, 임의의 사이클 타임 k의 모든 상태로부터 $L \geq L_{\min}$ 사이클 타임동안 역추적하면, 이들 상태는 ML의 의미에서 사이클 타임 $k-L$에서 최적 경로가 되는 하나의 상태에 통합되는 것이다[19].

일반적인 역추적에 의한 복호는 생존경로의 결정에 관한 정보 d_k^S를 기반으로 해서, 역순으로 생존경로의 갱신과 복호를 동시에 수행한다. 즉, 현재상태 s_k에서 그 전의 상태 s_{k-1}의 추정은

$$s_{k-1}=f(s_k, d_k^s) \tag{5}$$

으로 수행된다. 함수 $f(\cdot)$은 생존경로의 트렐리스 구조와 수신 비트열에 좌우된다. 이와 같은 역추적에 의한 복호과정은 연속적인 복호를 위해서는 다음의 3가지 과정이 하나의 사이클 타임에서 동시에 이루어져야 한다.

- 역추적(TB) : 저장된 생존경로의 결정에 관한 정보를 읽어내서 역방향으로 상태간의 경로를 추정하는 과정
- 복호 : 역추적 과정과 비슷하며 추적된 경로에 대응하는 부호화된 비트를 최종으로 복호하는 과정
- 새로운 생존경로 결정에 관한 정보의 저장 : ACSU부에서 출력된 각 상태의 생존경로에 관한 결정 비트를 생존 메모리에 저장하는 과정

<그림 4>는 하나의 생존 메모리내에서 이상의 3가지 역추적에 의한 복호과정을 나타낸 것이다. 상태 벡터는 ACSU에서 출력되는 상태 수 $N=2^m$에 대응하는 결정 비트이다.

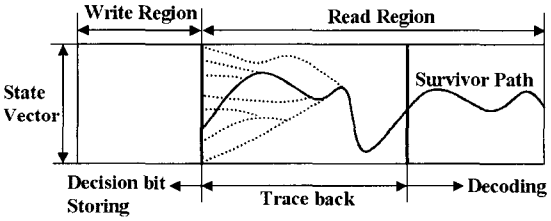


그림 4. 역추적 기법에 의한 복호과정
Fig. 4. Decoding procedure with trace back technique.

비터비 디코더의 VLSI 칩 구현에서 전력소모 및 칩 사이즈를 줄이기 위해서는, 최적 생존경로의 결정과 복호에 소요되는 메모리 사이즈를 최소화함과 동시에, ACSU 갱신 레이트에 대응할 수 있는 throughput을 얻을 수 있어야 한다. 이를 구현하기 위해 1-포인터 TB 구조와 P-포인터 TB 구조가 개발되어 있다[10]. 1-포인터 TB 구조에서는 <그림 4>와 같은 하나의 생존 메모리를 중심으로 결정비트의 저장, 역추적 및 디코딩을 동시에 수행한다. 따라서, 연속적인 복호를 가능케 하기 위해서는 상이한 주파수를 갖는 복수개의 클럭을 사용해야 된다. 또한, 필요한 생존 메모리의 사이즈 M은 클럭 사이클당 역추적 반복 횟수를 나타내는 역추적 반복율(Trace-back Recursion Rate: TRR)의 함수로서 다음과 같이 주어진다.

$$M=(1+\frac{2}{TRR-1})L, \text{ 단 } TRR>1 \tag{6}$$

따라서, 1-포인터 TB 기법은, 그 구조는 간단하지만, $TRR>1$ (보통 2 또는 3)이므로 2개 이상의 주파수가 다른 클럭을 사용해야 하므로 그 실용성이 제한된다.

P-포인터 TB 구조는 1-포인터 TB 구조의 문제점을 해결하기 위해 제안된 것으로 생존 메모리를 P개의 영역으로 나누어 최적 경로의 역추적과 복호를 P 개의 영역에서 동시에 수행하는 구조이다. 이 경우의 생존 메모리의 사이즈 M은 다음식으로 주어진다.

$$M=\frac{2P}{P-1}L,\ \text{단}\ \ P>1 \tag{7}$$

따라서, 역추적 포인터의 수 P 를 증가시키면 필요한 메모리의 사이즈는 감소되지만, 메모리 분할의 수가 증가하게 되어 메모리 제어를 위한 주변회로가 복잡하게 되고 전체적인 칩내 역추적 부분의 면적이 감소되지 않는 문제가 있다[12].

이상의 기존 역추적에 의한 복호는 레지스터 교환방식에 비해, 상태의 수가 많은 즉, 구속장 K 가 큰 경우 칩 사이즈 및 전력소모 등의 면에서 뛰어나지만, 생존 메모리 제어를 위한 별도의 회로가 필요하고, 복호 지연시간이 길어지는 문제가 있다. 따라서, 상태수가 큰 경우에도 역추적 및 복호, 생존 메모리의 제어를 보다 간단하게 처리함으로서 메모리 억세스에 소요되는 대역폭을 줄여 VLSI 칩 구현에 소요되는 면적을 적게하는 것이 필요하다.

2. 역방향 상태천이의 추적에 의한 역추적 디코더

ACSU로부터 출력되는 생존경로에 관한 결정비트 d_k^s 를 기반으로 역방향의 상태천이를 연속적으로 제어함으로서 간단한 구조의 역추적 복호를 구현할 수 있다. 이를 위해, 먼저 사이클 타임 k의 임의의 상태로부터 직전의 사이클 타임 $k-1$의 과거상태로 복귀하는데 필요한 매핑관계를 조사해 보자. <그림 1>의 길쌈부호기에서 4가지 상태의 트렐리스도에 의한 연속되는 사이클 타임에서의 상태간 천이의 접속관계를 나타낸 것이 <그림 5>이다. 사이클 타임 k의 각 상태 $s_{j,k}\,(j=0,1,2,3)$에 입력되는 가지는 상부가지(upper branch)와 하부가지(lower branch)의 2개로 구성된다. 이 때, 누적되는 경로 메트릭과 가지 메트릭의 합이 최소인 경로가 생존경로로 선정되고, 상부 또는 하부의 경로가 선정되는 여부에 따라 "0" 또는 "1"의 값인 결정비트를 출력한다. 한편, 사이클 타임 $k-1$의 각 상태 $s_{j,k-1}$는

$$s_{k-1}=(x_{j1,k-1}\ ,\ x_{j0,k-1}) \tag{8}$$

이고, 사이클 타임 k의 각 상태 $s_{j,k}$는

$$s_{j,k}=(x_{j1,k}\ ,\ x_{j0,k})=(u_k\ ,\ x_{j1,k-1}) \tag{9}$$

이므로 $x_{j0,k-1}$은 사이클 타임 k로 진행함 따라 길쌈부호기의 레지스터에서 천이되어 빠져나가는 정보이다. 이 정보는 트렐리스도에서 상부가지 및 하부가지에 관한 정보를 나타낸다. 즉, 상부가지의 경우에는 $x_{j0,k-1}=0$에 해당 (상태 (0,0) 및 (1,0))되고, 하부가지의 경우에는 $x_{j0,k-1}=1$에 해당 (상태 (0,1) 및 (1,1)) 된다. 따라서 사이클 타임 k, 상태 j의 생존경로를 나타내는 결정비트 d_k^j 는 부호기의 레지스터로부터 천이되어 빠져나가는 정보 $x_{j0,k-1}$과 동일한 값이 된다.

$$d_k^j=x_{j0,k-1} \tag{10}$$

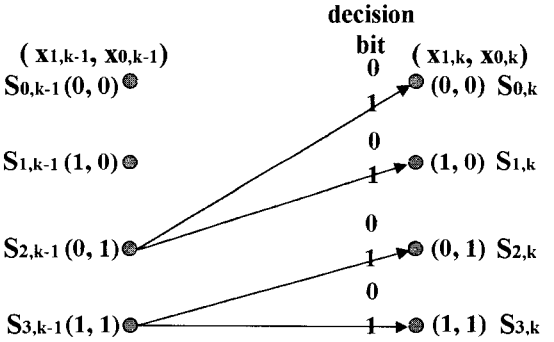


그림 5. 연속되는 사이클 타임에서의 트렐리스도에 의한 상태간 연결

Fig. 5. The trellis connection between states at consecutive 2 cycle times.

이상의 과정으로부터 생존 메모리에 저장된 결정비트를 기반으로 역추적에 의한 복호는 상태간의 역방향 변화를 연속적으로 추적함으로서 가능하다. 즉, 사이클 타임 k의 j 번째 상태 $s_{j,k}$에서 사이클 타임 $k-1$의 i 번째 상태 $s_{i,k-1}$로의 역방향 상태천이는 식 (9)와 (10)으로부터

$$s_{i,k-1}=(x_{j1,k-1}\ ,\ x_{j0,k-1})=(x_{j0,k}\ ,\ d_k^j) \tag{11}$$

의 관계가 되는 것을 알 수 있다. 이 과정을 복호깊이 $L\geq L_{\min}$ 이상 반복하게 되면 사이클 타임 $k-L$에서

하나의 상태로 통합되게 되고, 이 때 각 상태는 동일한 비트 정보를 갖는다.

<그림 6>은 본 논문에서 제안하는 역방향 상태천이의 연속제어에 의한 역추적 비터비 복호기의 구조를 블록도로 나타낸 것이다. 제안방법에서의 SMU의 구성은 결정비트를 저장하는 사이즈 $N\times M$의 생존 메모리와 $\frac{M+1}{2}\times\log_2 N$ 사이즈의 역방향 상태정보를 저장하는 상태 메모리로 구성된다. 이들 메모리는 ACU 처리 사이클 타임에 따라 메모리의 번지를 제어하는 별도의 제어회로를 요구하지 않는다. 따라서, 천이 레지스터나 어드레스를 단순 감소 또는 증가시키는 RAM과 같은 외부 메모리를 사용해서 구성할 수 있다.

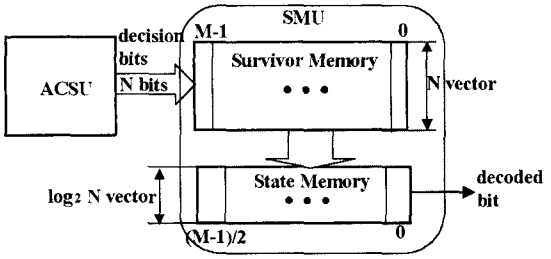


그림 6. 제안방법에 의한 역추적 디코더의 구성
Fig. 6. The architecture of the proposed trace back decoder.

제안방법에 의한 복호과정은 다음과 같다.

(1) 생존 메모리 M 및 상태 메모리 $L=\frac{M+1}{2}$ 의 초기화와 사이클 타임 $k=0$로 초기화

(2) 수신 부호어에 의한 가지 메트릭 및 경로 메트릭의 계산과 결정비트의 출력

(3) 결정비트를 생존 메모리에 사이즈 $k=\frac{M}{2}\geq L_{\min}$이 될 때까지 k를 증가시키면서 과정 (2)를 반복하면서 저장함. 이 때, 생존 메모리는 1 사이클 타임씩 우측으로 단순히 천이시킴.

(4) $k=L$이면,

- 생존 메모리의 천이 및 결정비트의 저장
- 각 상태값을 역방향으로의 추적을 개시함. 역방향 상태추적의 초기값은 다음으로 주어진다.

$$s_{j,k}=(x_{j0,k}\ ,\ d_k^j) \tag{12}$$

여기서 $x_{j0,k}$는 각 상태에 따라 자동으로 주어지는 값으로, 상태 (0,0)및 (1,0)는 $x_{j0,k}=0$, 상태 (0,1) 및 (1,1)에서는 $x_{j0,k}=1$의 값으로 저장된다.

- ACS가 처리되는 매 사이클 마다 각 상태의 역추적은

$$s_{j,k-L}=(x_{j0,k-L+1}\ ,\ q_{k-L}) \tag{13}$$

단 q_{k-L} 은 그 전의 상태값을 제어신호로 사용한 $N\times 1$ MUX에 의해 선정된 결정비트 d_{k-L+1}^j이다.

(5) $t=0$의 최종단에서, 상태 메모리에 저장된 각 상태값을 비교한다. 이 때, 상태 통합이 발생한 경우에는 상태 메모리의 모든 상태가 동일한 값을 가지게 되고, 상태 통합이 일어나지 않는 경우에는 복호 오류의 발생확률이 높은 것에 대응한다. 이 경우 다수결 결정(majority voting), 즉, 동일한 값을 갖는 다수의 상태값을 통합된 최종 상태로 판단하여 복호한다. 복호되어 출력되는 정보 $\widehat{u_k}$는 상태를 나타내는 비트 중 MSB $\widehat{u_k}=x_{1,k-M/2+1}$ 이다. 단, 첫 번째 출력되는 상태는 제외한다.

이상의 복호과정으로부터 수신 데이터는 생존 메모리의 사이즈에 대응하는 M 사이클 타임 이후부터 매 ACS 사이클 타임에 동기되어 복호된다. 제안기법에서 사용되는 전체 메모리의 크기는 생존 메모리 및 역방향의 상태변화를 저장하는 메모리의 합으로 주어진다.

$$\text{전체 메모리의 크기}: N\times M+\frac{M+1}{2}\times\log_2 N \tag{14}$$

표 1. TB 방법에 따른 전체 SMU부의 전체 메모리 사이즈
Table 1. The required total memory size within the SMU according to TB technique.

Number of states (N)	TB Method	Total Memory Size (bits)
4-states (L=10)	2-pointer TB	160
	3-pointer TB	120
	4-pointer TB	107
	Proposed TB	106
64-states (L=40)	2-pointer TB	10240
	3-pointer TB	7680
	4-pointer TB	6827
	Proposed TB	5430

또한, 복호 지연시간은 생존 메모리의 사이즈인 M 사이클 타임으로 고정된다.

식 (7)에서 기존의 2-포인터 역추적 복호의 경우 $M=4L$, 3-포인터 역추적의 경우 $M=3L$에 해당되는 생존 메모리가 필요하다. 이에 비해 제안방법에 의한 생존 메모리 사이즈 M의 값은 복호깊이 $L \geq L_{\min}$의 2배, 즉 $M=2L+1$의 값으로 고정시켜 사용해도 동일한 복호성능을 얻을 수 있다. 역추적에 의한 복호에서, 역추적 방법에 따라 소요되는 전체 메모리의 사이즈를 $N=4$, $L=10$과 $N=64, L=40$에 대한 비교결과를 <표 1>에 나타낸다.

제안방식에 의한 역추적 비터비 디코더의 특성은 다음과 같다.

- SMU에서 단일 클럭의 사용으로 전체 복호과정을 수행할 수 있다.
- 결정비트를 생존 메모리에 저장시키기 위한 별도의 제어회로가 필요없고, 단순한 메모리의 천이 동작만으로 충분하다.
- 저장된 결정비트를 읽어내기 위한 Read Pointer나 Decode Read를 위한 과정이 필요없다.
- 각 사이클 타임에서의 최소 경로 메트릭의 상태를 검출하기 위한 비교기의 사용이 필요없어 회로구성이 간단하다.
- SM에 사용되는 메모리의 양이 기존 TB 방식보다 적게 사용된다.
- 생존경로에 관한 정보의 단순한 천이만으로 복호과정이 이루어지므로 메모리 억세스를 위한 대역폭을 줄일 수 있어, 칩 설계에서 Area-Efficiency가 높고, 전력소모가 적다.

Ⅳ. 시스톨릭 어레이 구조에 의한 구현

생존 메모리에 의한 역추적 복호기의 throughput을 높이기 위해서는, 병렬구조 ACSU의 사이클 타임과 동기화되고 같은 주파수의 사이클 타임으로 처리되어야 한다. 이를 효율적으로 구현하는 방법의 하나가 시스톨릭 어레이(systolic array) 구조이다[13, 15]. 시스톨릭 어레이 구조에 의한 디코더 구현의 이점은, 역추적 과정이 파이프 라인 형태의 레지스터 어레이 구조에서 연속적으로 처리되는 점이다. 즉, 각 반복과정에서 전체 역추적 과정이 완료될 때까지 기다려야 하는 기존의 역추적 기법과 다르게 시스톨릭 어레이 구조에서는 ACSU 사이클 타임 단위로 처리할 수 있다. 따라서, 시스톨릭 어레이 구조를 적용한 제안방식은 기존의 역추적 기법보다 복호 지연시간의 양을 줄일 수 있다[15].

<그림 7(a)>는 <그림 1>의 구속장 $K=3$인 길쌈부호기에 대해서 제안방식에 의해 역추적 및 복호과정을 수행하는 시스톨릭 어레이 구조의 단위 처리블록의 회로구성을 나타낸 것이다. 2-사이클 타임에 해당하는 생존 메모리와 역방향 상태천이의 변화를 저장하기 위한 1-사이클 타임의 상태 메모리로 구성된다. 따라서, 하나의 상태 메모리 변화는 2개의 생존 메모리의 천이에 대응하므로, 최종단에서 생존 메모리와 상태 메모리의 천이시간이 동기화 된다. 연속적인 역방향 상태천이의 추적은 식 (13)을 사용해서 각 사이클 타임마다 상태값을 제어신호로 사용하고, 단위 처리 블록에서 출력되는 결정비트를 입력 데이터로 사용하는 4×1 MUX에 의해 간단히 처리된다.

예로, 사이클 타임 k에서의 상태 메모리에 저장된 상태값이 $s_{0,k}=(0,1)$, $s_{1,k}=(0,0)$, $s_{2,k}=(1,0)$, $s_{3,k}=(1,0)$이고, 사이클 타임 $k-1$의 결정비트 d_{k-1}가 $d_{k-1}=(0,1,1,0)$ 이라고 하면, 상태 $s_{j,k-1}$은

$$s_{0,k-1}=(x_{00,k},\ d_k^1)=(1,1)$$
$$s_{1,k-1}=(x_{10,k},\ d_k^0)=(0,0)$$
$$s_{2,k-1}=(x_{20,k},\ d_k^2)=(0,1)$$
$$s_{3,k-1}=(x_{30,k},\ d_k^2)=(0,1)$$

의 값으로 자동 변환되어 저장된다. <그림 7(b)>는 <그림 7(a)>의 단위 처리블록의 회로구성을 블록도로 나타낸 것으로, 이를 생존 메모리 및 역방향 상태천이 제어(Survivor Memory and Backward State Transition Control : SM & BSTC) 블록이라 부르기로 한다.

<그림 8>은 <그림 7(b)>의 SM & BSTC 블록을 나열해서 시스톨릭 어레이 구조로 역추적 및 복호 과정의 구성을 나타낸 것이다. 입력단(가장 좌측의 SM & BSTC 블록)에서는 식 (12)에 따라 $k=L=\frac{M}{2} \geq L_{\min}$이 되는 사이클 타임에서 4가지 상태 (0,0), (0,1), (1,0), (1,1)에 대해, ACU의 출력인 4비트 d_k^j를 입력받아

$$s_{0,k} = (0, d_k^0),\ s_{0,k} = (1, d_k^1),$$
$$s_{0,k} = (0, d_k^2),\ s_{0,k} = (1, d_k^3)$$

의 값으로 상태 메모리에 초기값으로 저장된다. 최종 출력단의 다수결 결정 블록은 채널잡음이 많은 환경에서 낮은 SNR의 신호를 전송하는 경우, 생존 메모리 사이즈 M 동안 역추적을 해도 하나의 상태로 통합되지 않을 가능성을 보완하기 위한 것이다. 비록 발생확률은 매우 낮은 경우에도 모든 상태가 하나의 상태로 통합되지 않을 경우, 같은 상태값을 갖는 상태가 많은 것을 보다 정확한 상태라 판단해서 복호하기 위함이다.

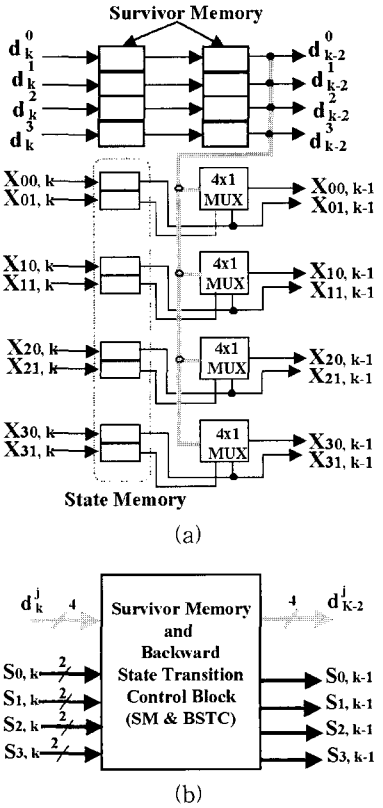


그림 7. 단위처리블록의 시스톨릭 어레이 구조의 회로 구성(a)과 블록도(b)

Fig. 7. The Circuit implementation (a) and block represenation (b) of processing unit with systolic array architecture.

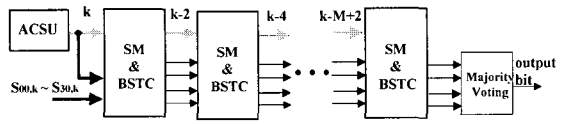


그림 8. 역방향 상태천이의 연속적인 제어와 복호를 위한 시스톨릭 어레이 구성

Fig. 8. The systolic array implementation for the control backward state transition and decoding.

제안기법에 의한 역추적 복호과정의 구체적인 동작을 검증하기 위해 <그림 1>의 부호기의 예를 들어 기술한다. 채널오류가 존재하는 경우에도 정상적인 복호가 이루어지는 것을 확인하기 위해, 부가 백색 채널 잡음이 존재하는 BSC(Binary Symmetric Channel)로 가정한다. <표 1>에 사이클 타임 $k=1$에서 $k=24$까지의 소스비트 u_k , 부호어 $\mathbf{v}_k=(v_{0k}, v_{1k})$ 그리고 채널로부터 수신한 데이터 $\mathbf{r}_k=(r_{0k}, r_{1k})$를 나타낸 것이다. BPSK 변조에 의한 3.5dB의 SNR을 갖는 것으로 가정했다.

표 2. 소스비트 및 채널잡음을 포함한 수신 데이터

Table 2. Source bit and received data including channel noise.

Cycle time k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Source bit	1	0	1	0	1	1	1	0	1	0	1	1	1	1	0	1	0	1	1	1	0	1	0	0
Code-Word v1	1	1	0	1	0	0	1	0	0	1	0	0	1	1	0	1	1	1	0	0	1	0	0	1	1
Code-Word v2	1	0	0	0	0	1	1	0	0	0	0	1	1	0	1	1	1	0	0	1	0	1	0	0	1
Received Code-Word r1	0.47	1.57	2.02	-0.75	1.57	2.31	2.15	1.04	-0.74	1.25	0.81	0.44	-0.38	1.59	1.41	-0.28	0.81	2.22	1.74	1.57	2.02	1.87	-0.47	
Received Code-Word r2	0.43	0.31	1.91	-2.33	0.36	-0.75	1.59	1.11	0.83	1.19	1.39	1.57	1.57	2.3	0.25	1.41	1.79	2.54	1.22	1.19	0.42	-0.7	1.45	1.25

비터비 복호에서 가지 메트릭의 계산에서 메트릭으로 다음으로 정의되는 AED (Absolute Euclidean Distance)를 사용했다.

$$AED=\sum_{i=0}^{1}|r_{ik}^{q}-C_{ik}|$$

여기서 $r_{ik}^{q}=Q[\alpha r_{ik}]$로 수신된 부호어에 저당한 가중치 α를 곱한 후 3비트 양자화된 값이고, $C_{ik}=\beta v_{ik}$로 주어지는 데이터이다. 본 논문에서 사용한 가중치 파라미터는 $\alpha=32$, $\beta=7$이고, 3비트 데드 존(dead zone)을 갖지 않는 양자화기를 사용했다. <표 3>은 이상의 방법으로 구한 각 사이클 타임에서의 생존경로의 가지 메트릭을 나타낸 것이다.

<그림 9>는 <표 2>와 <표 3>으로 주어진 소스 정

표 3. 각 상태에서의 생존경로에 대한 가지 메트릭

Table 3. The calculated branch metric to the survivor path at each state.

Cycle time k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
BM00	15	9	14	7	11	5	7	7	14	7	14	9	7	7	4	14	5	14	7	7	9	14	7	1
BM01		2	7	5	6	1	7	0	3	0	7	13	14	3	3	7	5	7	14	14	2	7	7	6
BM10	1	5	5	2	3	5	7	7	7	5	7	6	7	7	10	2	5	7	7	7	5	0	7	13
BM11	1	12	7	9	8	7	7	14	3	14	7	1	8	3	13	7	12	7	0	7	12	7	14	8

보와 가지 메트릭에 대해, 사이클 타임 $t=24$까지의 생존경로를 나타낸 트렐리스도이다. 이 그림에서 굵은 선으로 표시된 경로는 $t=24$의 각 상태에서 역추적에 의한 최적 경로이다. 복호깊이 $L=10$, $M=21$을 사용해서 제안방법에 의한 복호과정을 순서적으로 나타낸 것이 <그림 10>이다. 생존 메모리에 저장된 데이터는 트렐리스도에 따라 ACSU에서 출력된 각 상태의 경존경로를 나타내는 결정비트이고, 상태의 순서는 (0,0), (0,1), (1,0), (1,1)의 순이다. 사이클 타임 $t=10$까지 상태 메모리의 상태천이 정보는 저장되지 않고, <그림 10(a)>에서 나타낸 것처럼 사이클 타임 $t=11$에서 처음으로 상태천이의 정보가 식 (12)에 따라 저장된다. 사이클 타임 $t=12$에서, 결정정보 및 식 (13)으로 계산된 역방향 상태천이 정보가 각각 생존 메모리와 상태 메모리에 저장되는 형태를 나타낸 것이 <그림 10(b)>이다. <그림 10(c)>에서 <그림 10(g)>까지는 사이클 타임 동일한 과정으로 사이클 타임 $t=20$에서 $t=24$까지의 변화과정을 나타낸 것이다. $t=21$에서 부호기에서 시작된 초기상태로 통합되고, $t=22$에서 다음상태로의 천이 (1,0)가 모든 상태에서 동일한 값으로 주어지고, 이는 모든 상태가 하나로 통합된 것을 의미한다. 따라서, 복호된 정보 $\hat{u}_1=x_{j1,1}=1$이다. 이상의 과정을 반복하면, $t=23$에서 $\hat{u}_2=x_{j1,2}=0$, $t=24$에서 $\hat{u}_3=x_{j1,3}=1$으로 복호되고 이는 <표 2>의 소스비트와 일치하는 것을 알 수 있다.

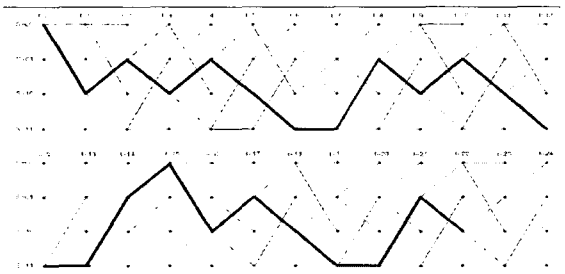
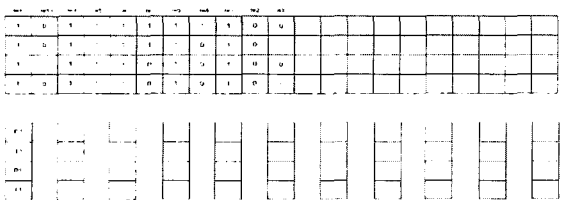
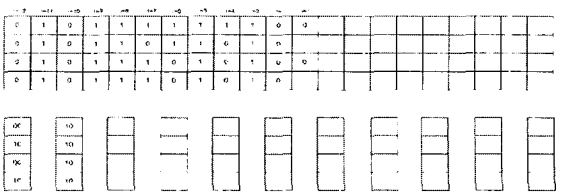


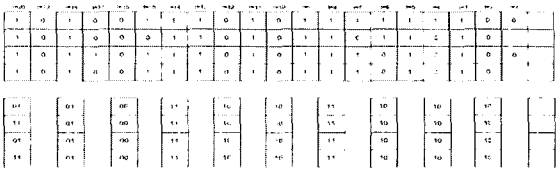
그림 9. 사이클 타임 $t=24$까지의 트렐리스도
Fig. 9. The trellis diagram at cycle time $t=24$.



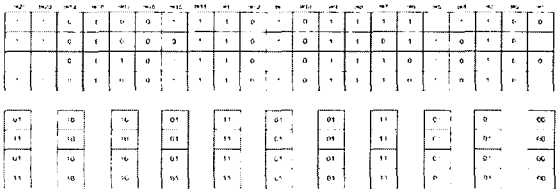
(a) $t=11$



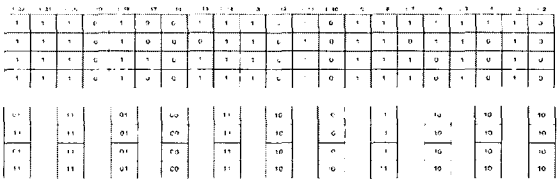
(b) $t=12$



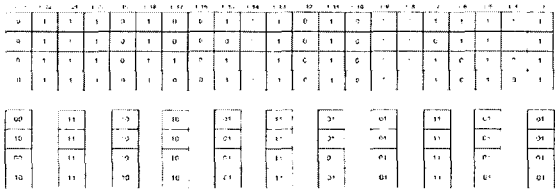
(c) $t=20$



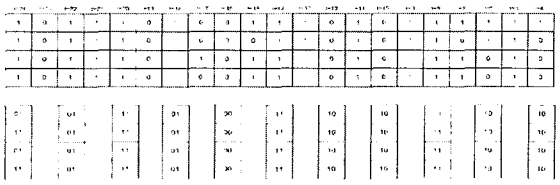
(d) $t=21$



(e) $t=22$



(f) $t=23$

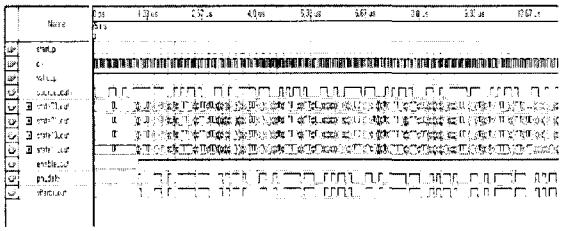


(g) $t=24$

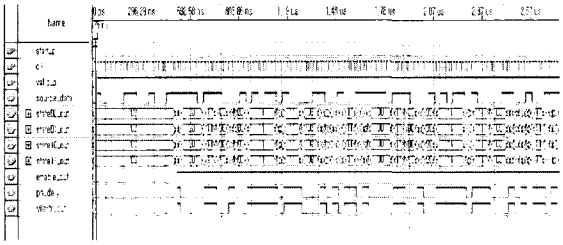
그림 10. 사이클 타임에 따른 생존 메모리 및 상태천이 메모리 제어와 복호과정
Fig. 10. Survivor memory and state transition memory control and decoding procedure according to cycle time proceeding.

제안방법에 의한 비터비 복호의 구체적인 동작을 검증하기 위해, IEEE 802.11a 고속 무선 랜 [2]의 채널 코

덱에 적용했다. IEEE 802.11a 고속 무선 랜에서는 구속장 $K=7$, 부호율 $\frac{1}{2}$과 펑처링에 의한 가변부호율 $\frac{2}{3}$과 $\frac{3}{4}$을 규격으로 정의하고 있다. <그림 11>은 이 코덱의 $(6,\frac{1}{2})$ 길쌈부호기에 적용한 VHDL 코딩 결과의 파형을 나타낸 것으로, <그림 11(a)>는 24Mbps, <그림 11(b)>는 54Mbps에 대한 결과이다. 이 결과로부터 일정시간 지연 후, 복호된 데이터 파형이 (viterbi_out) 부호기의 입력 데이터의 파형(source_data)과 잘 일치하고 있음으로 확인할 수 있고, 또한 복호시점에서의 각 상태의 값 (state00_out~state11_out) 이 모두 일치하고 있음을 볼 수 있다.



(a) 24Mbps



(b) 54 Mbps

그림 11. IEEE 802.11a 고속 무선 랜 채널 코덱에 대한 VHDL 코딩 결과

Fig. 11. The VHDL coding results to the IEEE 802.11a wireless LAN channel codec.

V. 결 론

본 논문에서는 비터비 알고리즘에 의한 복호과정에서 생존 메모리의 제어에 의한 새로운 역추적 복호 알고리즘을 제시했다. 각 상태의 생존경로를 나타내는 결정비트는 ACS 사이클 타임에 동기되어 단순 천이 동작만으로 생존 메모리에 저장되고, 동시에 최적 생존경로를 따라 역방향의 상태추적이 이루어지면서 최종단에서의 상태정보로부터 복호 데이터가 출력되는 것을 입증했다.

이와 같은 역방향 상태천이의 연속적인 제어에 의한 복호는 기존 역추적 방법에 비해, 전체 메모리 사용양이 적고 회로구성이 간단하다. 단일 클럭만으로 전체 복호과정이 이루어지고 메모리 제어를 위한 별도의 회로구성이 요구되지 않는 특성을 갖는다. 또한, 시스톨릭 어레이 구조에 의한 파이프라인 병렬구조에 의한 구성으로 고속연산이 가능해 통신 throughput을 향상시킬 수 있다. 부가 백색잡음을 포함한 채널의 수신 데이터로부터 제안방식에 의해 복호가 정확하게 수행되는 것을 확인했다. 구체적인 응용의 예로, IEEE 802.11a 고속 무선 랜 시스템에 적용해서 24Mbps 및 54Mbps에 대한 VHDL 코딩 결과를 제시했다.

참 고 문 헌

[1] D. J. Costello, Jr., H. Hegenaur, H. Imai, and S. B. Wicker, "Applications of error-control coding," IEEE Trans.Information Theory, vol. 44, no. 6, pp. 2531-2560, Oct. 1998.

[2] A. Sabanmaria and F. J. Lopes-Hernandez, Wireless LAN: Standards and application, Artech House, 2001.

[3] A. J. Viterbi and J. K. Omura, Princilples of digital communication and coding, McGraw-Hill, NY, 1979.

[4] G. G. Forney, "The Viterbi algorithm," Proceedings of the IEEE, vol. 61, no. 3, pp. 268-278, March 1973.

[5] H. Liou, "Implementing the Viterbi algorithm," IEEE Signal Processing Mag., pp. 42-52, Sept. 1995.

[6] H. Dawid, O. J. Joeressen, and H. Meyr, "Viterbi decoders: High performance algorithms and architectures," Digital Signal Processing for Multimedia Systems edited by K. Parhi and T. Nishitani, Marcel Dekker 1999.

[7] O. Collins and F. Pollara, "Memory management in trace back Viterbi decoders," TDA Prog. Rep. 42-99, JPL, Nov. 1989.

[8] C. M. Radar, "Memory management in a Viterbi algorithm," IEEE Trans. Commun., vol. 29, pp. 1399-1401, Sep. 1981.

[9] G. Feygin and P. G. Gulak, "Architectural

tradeoffs for survivor sequence memory management in Viterbi decoders," IEEE Trans. Commun., vol. 41, no. 3, pp. 425-429, March 1993.

[10] R. Cypher and C. B. Shung, "Generalized trace back techniques for survivor memory management in the Viterbio algorithm," In Proc. GLOBECOM., pp. 1318-1322, Dec. 1990.

[11] G. Fettweis, "Algebraic survivor memory management for Viterbi detectors," In Proc. Int. Conference on Commun. 92 (ICC'92), pp. 339-343, 1992.

[12] P. J. Black and T.H. Meng, "Hybrid survivor path architecture for Viterbi decoders," In Proc. Int. Conference on Acoustics, Speech, and Signal Processing 93 (ICASSP'93), pp. 433-436, Nov. 1993.

[13] T. K. Trung, M. Shih, I. S. Reed, and E. H. Satorius, "A VLSI design for a trace-back Viterbi decoder," IEEE Trans. Commun., vol. 40, no. 3, pp. 616-624, March 1992.

[14] C. Shung, H. Lin, R. C. Sypher, P. H. Siegel, and H. K. Thapar, "Area-efficient architecture for the Viterbi algorithm - Part I: Theory," IEEE Trans. Commun., vol. 41, no. 4, pp. 636-644, April 1993.

[15] G. Fettweis and H. Meyr, "High-rate Viterbi processor: A systolic array solution," IEEE J. Selected Areas Commun., vol. 8, no. 8, pp. 1520-1534, Oct. 1990.

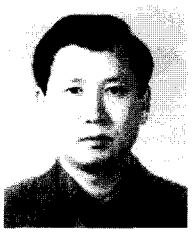
[16] P. J. Black and T. H. Meng, "A 140-Mb/s, 32-state, radix-4 Viterbi decoder," IEEE J. Solid-State Circuits, vol. 27, no. 12, pp. 1877-1885, Dec. 1992.

[17] J. Sparso, H. N. Jorgensen, E. Paaske, S. Pedersen, and T. R. Petersen, "An area-efficient topology for VLSI implementation of Viterbi decoders and other shuffle-exchange type structures," IEEE J. Solid-State Circuits, vol. 26, no. 2, pp. 90-97, Feb. 1991.

[18] R. J. McEliece and I. M. Onyszchuk, "Truncation effect in Viterbi decoding," In Proc. Proc. of the IEEE Confer. on Military Commun., (Boston, MA), pp. 29.3.1-29.3.3 Oct. 1989.

[19] I. M. Onyszchuk, "Truncation length for Viterbi decoding," IEEE Trans. Commun., vol. 39, no. 7, pp. 1023-1026, July 1991.

저 자 소 개



鄭 且 根(正會員)

1982년 2월 : 경북대학교 전자공학과 졸업. 1984년 2월 : 서울대학교 대학원 전기공학과 공학석사. 1993년 2월 : 일본 동경대학 전기공학과 공학박사. 1984년 1월~1997년 8월 : LG종합기술원 책임연구원. 1995년 4월~1996년 3월 : 일본 방송통신기구 초빙연구원. 2002년 1월~2002년 4월 : 동경대학 초빙교수. 1997년 9월~현재 : 호서대학교 정보제어공학과. <주관심분야 : 디지털 신호처리, 디지털 통신, 영상처리 및 부호화, Image Senser 등.>