2013년 11월 전자공학회 논문지 제 50 권 제 11 호
Journal of The Institute of Electronics Engineers of Korea Vol. 50, NO. 11, November 2013 http://dx.doi.org/10.5573/ieek.2013.50.11.050

논문 2013-50-11-7

고 처리율 병렬 터보 복호기 설계

(Design of a High Throughput Parallel Turbo Decoder)

이 원 호*, 박 희 민**, 임 종 석***

(Won-Ho Lee, Heemin Park, and Chong S. Rim©)

요 약

본 논문은 하나 이상의 다양한 길이의 패킷을 동시에 복호할 수 있는 고 처리율 병렬 터보 복호기의 설계를 보인다. 터보 복호기의 병렬 구조는 반복 복호로 인한 긴 디코딩 시간을 절감시키며, 입/출력의 이중 버퍼 구조 설계는 패킷들의 연속적인 복호를 가능하게 함으로써 복호기의 처리율을 향상시킨다. 병렬 터보 복호기는 가장 긴 길이의 패킷을 복호할 수 있도록 설계되기 때문에, 이보다 짧은 길이의 패킷의 복호 시에는 사용하지 않는 PE(Processing Element)가 존재한다. 본 논문의 아이디어는 이 유휴 PE들을 연속적으로 이어지는 다음 패킷의 복호에 즉시 이용함으로써, 복호기 내의 PE 사용 효율을 높이고 처리율을 향상시키는 데 있다. 이를 위하여 여러 패킷의 복호를 동시에 가능하게 하는 제어가 필요하며, 본 논문에서는 이러한 제어 방법을 기술한다. 제안한 방법을 적용하여, 32개의 PE를 사용하면서 최대 6144비트 길이의 패킷을 복호 할 수 있는 병렬 터보 복호기를 구현하였으며, 기존 터보 복호기와 비교하여 약 16% 의 면적 증가가 있었으나, 짧은 패킷의 경우 기존 복호기에 비해 최대 28배의 높은 처리율 향상 효과를 보였다.

Abstract

This paper provides a design of high-throughput parallel turbo decoder that is able to decode several packets of various length simultaneously. For high-speed communications, designing of Turbo decoder as parallel structures reduces the long decoding time caused by iterative turbo decode way. Also, by employing the double buffer structure for input and output packets improves the decoder throughput by enabling continuous decoding. Because parallel turbo decoder is designed to be able to decode the packet of the longest length, there exist idle PE's(Processing Element) in the case of decoding packets of short length. The main idea of this paper is to increase the utilization of PE's in parallel Turbo decoder and to improve the decoder throughput by using the idle PE's immediately for the subsequent packets decoding. For this, the control is necessary to enable the concurrent decoding of several short packets and we propose the method of this control. Applying the proposed method, we implemented Turbo Decoder with 32 PE's that can decode packets of 6144 bits maximum. Compared to the conventional Turbo decoder, although the area was increased about 16%, the decoder throughput was improved 28 times for short packets.

Keywords : Turbo Decoder, Parallel Turbo Decoder, Throughput.

Ⅰ. 서 론

현대의 통신 서비스는 대용량 데이터의 고속 전송을 요구하는 동시에 채널에서 발생하는 오류에 대한 정정 능력을 필요로 하는데 이는 디지털 통신 시스템에서 중요한 부분을 차지한다. 터보 코드[1]는 반복적인 복호화 알고리즘을 이용하여 섀넌의 이론적 한계에 근접하는

* 정회원, (주)AP위성통신 SoC팀
(SoC team, AP satellite communications Inc.)
** 정회원, 상명대학교 컴퓨터소프트웨어 공학과
(Dept. of Computer Software Engineering, Sangmyung University)
*** 평생회원, 서강대학교 컴퓨터공학과
(Dept. of CS&E, Sogang University)
© Corresponding Author(E-mail: csrim@sogang.ac.kr)
접수일자 : 2013년9월17일, 수정완료일 : 2013년10월30일

우수한 성능을 가진 데이터 전송용 채널코드로서 고속의 디지털 통신 시스템에서 많이 사용되고 있다.

터보 코드는 복호 과정의 반복회수가 많을수록 더 좋은 성능을 보이지만 터보 코드의 반복적인 복호방식으로 인하여 긴 복호 시간이 소요되기 때문에 복호시간을 단축하기 위하여 병렬 복호 방법[2][3]이 사용된다. 병렬 복호 방법은 입력된 패킷을 병렬 차수로 나누어, 나누어진 부분 패킷들을 동시에 병렬로 복호하는 방법이다. 또한 패킷들의 연속적인 복호를 위하여 이중버퍼구조[4][5]를 사용하며 이는 복호기의 처리율을 향상시킨다. 이중 버퍼구조는 입력, 복호, 출력 동작을 파이프라인으로 동작시키는 방법이다.

병렬 터보 복호기는 패킷의 종류 및 채널 환경에 따라 다양한 길이의 패킷을 복호할 수 있어야하기 때문에 가장 길이가 긴 패킷을 처리할 수 있도록 설계된다. 병렬 터보 복호기는 길이가 긴 패킷을 복호할 경우에 병렬로 구성된 PE (Processing Element) 전체를 사용하여 패킷을 나누어 복호하기 때문에 PE의 사용률(utilization)이 높아지게 되고 결과적으로 비트 당 복호시간이 짧게 된다. 그러나 길이가 짧은 패킷의 복호에는 전체가 아닌 일부만의 PE를 사용하여 패킷을 복호하기 때문에 PE의 사용률이 줄어들게 되어 복호 시간면에서 병렬 복호 구조의 이득이 적다. 즉 기존의 터보 복호기는 짧은 길이의 패킷을 복호하는 경우, 비트 당 복호시간이 길어지기 때문에 낮은 처리율을 보인다[4].

본 논문에서는 기존의 병렬 터보 복호 방식에서 PE 사용률이 100% 보다 낮게 되었을 경우에 사용되지 않는 유휴 PE (Processing Element)를 다른 패킷의 복호에 사용하는 방법을 제안한다. 본 논문에서는 연속적으로 입력되는 패킷들을 병렬로 구성된 PE들에 순환적으로 할당될 수 있도록 제어하는 방법을 제안한다. 이렇게 PE에 할당된 패킷들은 동시간대에 각각 독립적으로 복호된다. 제안된 복호기는 PE 사용률 100%를 요구하는 입력 패킷에 대해서는 기존의 복호기와 동일하게 동작하지만, PE 사용률이 낮은 짧은 길이의 패킷들에 대해서는 그 길이가 짧을수록 PE 사용률이 높아지게 되어 높은 처리율 향상 효과를 보인다. 또한, 제안된 복호기는 기존의 이중 버퍼 병렬 복호기 구조를 그대로 사용하면서도 동작 방식을 간단하게 하여 제어의 복잡도를 줄이고, 면적을 크게 증가시키지 않으면서 PE의 사용률을 높인다.

본 논문의 구성은 다음과 같다. II 장에서는 이중 버퍼 구조를 가진 기존 병렬 터보 복호기의 구조와 동작방식을 보이고, III 장에서 본 논문에서 제안한 터보 복호기의 동작방식과 제어방법을 보인다. 그리고 IV 장에서 동일한 길이의 패킷의 연속적인 입력에 대한 처리율을 패킷의 길이 별로 측정한 결과와 임의의 길이의 패킷의 연속적인 입력에 대한 처리율을 측정한 결과를 보인다. V 장에서는 결론을 보인다.

II. 기존의 이중 버퍼구조의 병렬 터보 복호기의 구성 및 동작

그림 1은 4개의 PE를 사용하는 기존의 이중 버퍼 병렬 복호기의 전체 구성을 하나의 예로 보인다. 병렬 터보 복호기에서 복호 시 사용되는 PE의 개수는 패킷의 길이(K)에 따라 달라지며, 하나의 패킷 복호를 위하여 그림의 경우 PE1부터 PE4(1, 2, 또는 4)까지가 사용된다. 각 PE는 (K/사용되는 PE의 수)개의 데이터를 부분적으로 복호하며 이러한 병렬 복호를 위하여 모든 메모리는 병렬차수와 동일하게 4개로 구성된다. 그림과 같이 이중 버퍼 구조는 입력메모리와 출력메모리 그룹 두 개를 사용하여, 입력동작과 복호동작, 그리고 복호동작과 출력동작을 파이프라이닝으로 처리하여 패킷들이 연속적으로 복호될 수 있도록하기 위한 구조이다.

표 1에 다양한 크기의 패킷이 기존 병렬 복호기로 입력될 때 이의 동작을 보인다(각 PE는 160 샘플씩을 처

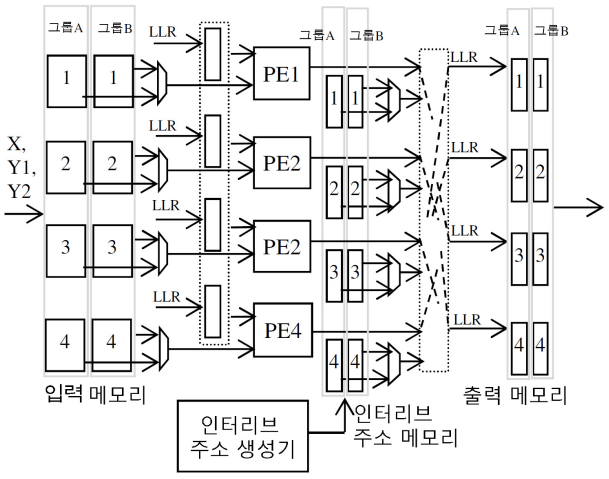

그림 1. 이중 버퍼 구조의 병렬 터보 복호기 구성
Fig. 1. Design of parallel Turbo decoder using double buffer structure.

표 1. 기존 터보복호기의 PE할당 및 그룹사용의 예
Table 1. PE allocation and group usage example of traditional Turbo decoder.

	K	사용된 PE 개수	사용된 PE 인덱스	사용된 버퍼 그룹
1	320	2	1, 2	A
2	160	1	1	B
3	160	1	1	A
4	320	2	1, 2	B
5	640	4	1, 2, 3, 4	A
6	160	1	1	B
7	320	2	1, 2	A
8	320	2	1, 2	B

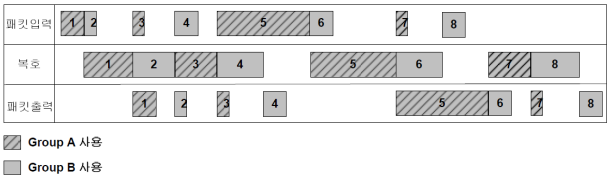

그림 2. 기존 터보 복호기의 동작 타이밍
Fig. 2. Timing diagram of traditional Turbo decoder.

리한다고 가정). 표와 같이 기존 복호기는 연속적으로 입력되는 패킷에 대하여 그룹을 번갈아 사용하며, 모든 패킷은 PE 1부터 할당되어 복호된다. 그림 2는 이에 대한 타이밍도를 보여 주며, 그림에서와 같이 입력동작과 복호동작이 파이프라이닝 되고, 복호동작과 출력 동작이 파이프라이닝 된다.

그러나 예에서와 같이 패킷 2의 입력이 완료되었고, PE3, PE4는 현재 유휴 상태임에도 불구하고 패킷2는 패킷1의 복호가 완료될 때까지 기다려야 하여 짧은 길이의 패킷에 대하여 PE 사용률이 현저히 떨어지게 된다. 패킷 3또한 패킷 2의 복호가 시작되지 않았기 때문에 입력이 지연 된다. 이어지는 모든 패킷들에 대하여도 마찬가지로 PE의 사용률이 떨어지고 모든 동작의 지연이 발생한다.

Ⅲ. 제안된 병렬 복호기의 동작 방식 및 제어

본 장에서는 Ⅱ장에서와 같이 PE 4개를 가지는 이중 버퍼 구조의 터보 복호기를 예로 들어 제안된 병렬 복호기의 동작 방식과 그 제어 방법을 보인다.

1. 동작 방식

제안된 병렬 복호기에 순차적으로 Ⅱ장의 예와 같은 8개의 패킷이 차례로 입력되면, 표 2와 같이 기존 복호기와는 다른 식으로 PE들과 버퍼 그룹을 사용한다.

표 2. 제안된 터보복호기에서의 PE할당 및 그룹사용의 예
Table 2. Examples of PE allocation and group usage in the proposed Turbo decoder.

	K	사용된 PE의 개수	사용된 PE의 인덱스	사용된 버퍼 그룹
1	320	2	1, 2	A
2	160	1	3	A
3	160	1	4	A
4	320	2	1, 2	B
5	640	4	1, 2, 3, 4	A
6	160	1	1	B
7	320	2	2, 3	B
8	320	2	1, 2	A

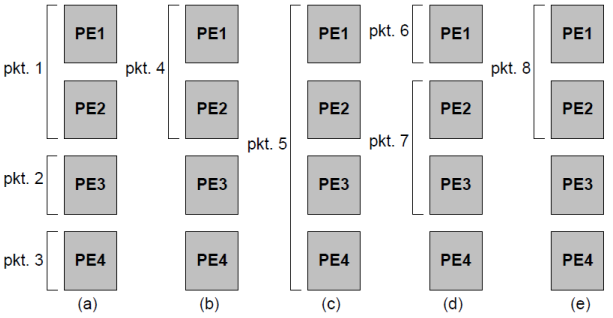

그림 3. 제안된 터보복호기에서의 PE할당 모습
Fig. 3. PE allocation of proposed Turbo decoder.

앞서 언급된 바와 같이 본 논문의 아이디어는 PE의 사용 효율을 높이기 위하여 이어지는 다음 패킷의 복호에 유휴 PE를 순차적으로 사용함에 있다. 그림 3은 이러한 아이디어를 반영하여 8개의 패킷을 PE에 할당하는 모습을 보여준다. 패킷1에 대해서 2개의 PE만을 사용하기 때문에 PE3, PE4는 유휴 PE가 된다. 패킷2와 패킷3은 1개의 PE만을 사용하기 때문에 제안한 복호기는 이를 PE3, PE4에 각각 할당하여, 패킷1, 2, 3이 동시에 복호될 수 있도록 한다. 패킷4는 남는 PE가 없기 때문에 패킷1의 복호가 완료되기를 기다렸다가 PE1, PE2에 할당된다. 이로써 패킷 2, 3, 4 또한 동시간대에 계속 복호를 진행할 수 있다. 그림 3의 (b)와 같이 패킷 2, 3의 복호가 끝나고 PE3, PE4가 사용되지 않는 패킷이 되어도 패킷5는 4개의 PE를 사용하여 복호되기 때문에 해당 패킷은 패킷4의 복호가 완료된 후 그림의 (c)와 같이 PE1~4에 할당된다. 패킷8의 경우 남아있는 PE가 그림의 (d)와 같이 PE4 한 개이기 때문에 PE1 , PE2에 할당된다. 표 3은 동시간대에 복호 될 수 있는 패킷들

표 3. 동시 복호 가능한 패킷리스트
Table 3. Concurrently decodable packet list.

입력 패킷 순서	1, 2, 3, 4, 5, 6, 7, 8
동시에 복호 가능한 패킷	1, 2, 3
	2, 3, 4
	5
	6, 7
	8
출력 패킷 순서	1, 2, 3, 4, 5, 6, 7, 8

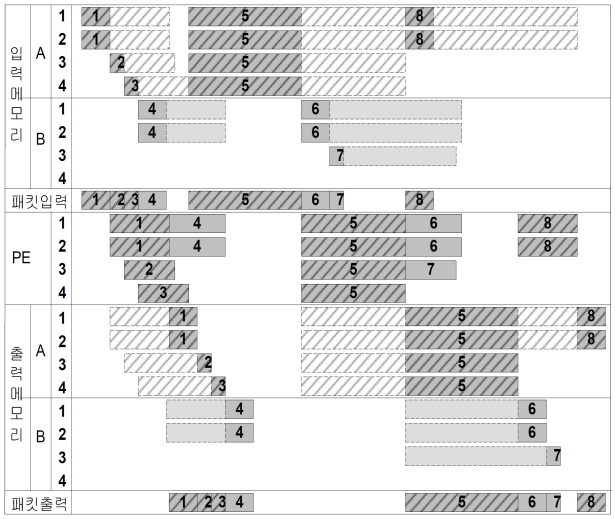

그림 4. 제안된 터보 복호기의 동작 타이밍
Fig. 4. Timing diagram of the proposed Turbo decoder.

의 목록을 보여 준다. 패킷 1, 2, 3이 동시간대에 복호될 수 있으며, 패킷 2, 3, 4 그리고 패킷 6, 7 또한 마찬가지이다. 단 패킷의 입력은 순차적이며, 출력 또한 마찬가지이다.

그림 4는 예제에 대한 이에 대한 타이밍도를 보여주며, 그림 3과 같은 PE할당을 위하여 입력 메모리 및 출력 메모리가 어떻게 사용되는지 입력, 복호, 출력의 시작은 어떻게 제어되는지에 대한 모든 내용을 포함한다.

그림에서와 같이 패킷 1은 그룹 A의 1, 2번 입력메모리에 저장된다. 기존의 복호기에서는 패킷2가 그룹B의 1번 입력 메모리에 저장되지만, 제안한 복호기는 패킷2를 PE3번을 이용하여 복호하고자하기 때문에 그룹A의 3번 입력메모리에 저장한다. 패킷3 또한 같은 방식으로 저장된다. 패킷2와 패킷3은 각각은 입력이 완료된 직후, 사용되지 않던 PE3과 PE4를 이용하여 복호되고, 그림과 같이 패킷 1, 2, 3은 동시간대에 복호된다. 패킷1이 출력되는 중에 패킷2의 복호가 완료되면, 패킷2의 출력의 시작은 패킷1의 출력의 완료를 기다려야 한다.

그림의 입력메모리에서 입력동작에 이용되는 입력메모리는 진한 색으로, 복호에 사용되는 입력메모리는 연한 색으로 표시하였다. 출력메모리 또한 마찬가지이다.

패킷4는 더 이상 사용되지 않는 PE가 없기 때문에 PE1, PE2를 이용하여 복호된다. 따라서 패킷4는 패킷3의 입력이 완료된 뒤, 그룹을 바꾸어 그룹B의 입력메모리 1, 2에 저장된다. 패킷4의 복호가 시작되기 위해서는 패킷1의 복호가 완료되어 PE1, PE2가 자유로워지기를 기다려야 한다.

패킷 4는 PE1, PE2를 사용하고 패킷5의 복호에 필요한 PE는 4개이기 때문에, 패킷5의 복호에 필요한 PE가 부족하므로, 패킷5는 그룹을 바꾸어 그룹 A의 입력메모리 1~4에 저장된다. 그림과 같이 패킷5 입력의 시작은 패킷1, 2, 3의 복호가 끝나서 입력 메모리 1~4가 자유로워 질 때까지 기다려야 한다. 패킷5의 복호는 패킷5의 입력 중에 모든 PE가 자유로워지기 때문에 입력의 완료와 동시에 시작된다.

패킷6, 7은 패킷5의 입력이 완료되면, 그룹 B의 입력메모리에 저장되고, 패킷5의 복호가 완료되기까지 기다렸다가 복호가 시작된다. 패킷6, 7의 출력은 패킷5의 출력 완료를 기다렸다가 시작되며, 패킷7이 패킷6보다 빨리 복호를 완료하였지만, 패킷6이 먼저 출력되도록 제어된다.

패킷6, 7은 PE1, 2, 3을 사용하고 남는 PE는 PE4 한 개인데 패킷8의 복호에 사용되는 PE는 2개이므로 패킷8은 그룹을 바꾸어 그룹 A의 입력 메모리 1, 2에 저장된다. 패킷8의 입력은 패킷5의 복호가 완료되어 입력메모리 1, 2가 자유로워 질 때까지 기다려야 한다. 패킷6의 복호가 완료되면 PE1, PE2는 자유로워지지만, 그룹 A의 출력 메모리 1, 2가 패킷5의 출력에 사용되고 있기 때문에 패킷8의 복호의 시작은 패킷5의 출력이 완료될 때까지 기다려야 한다.

그림의 패킷입력, 패킷출력은 복호기 외부에서 보이는 입, 출력 모습을 보여준다.

2. 동작 제어

본 절에서는 3.1 절의 동작 방식을 정리하여 이중버퍼 구조에서 그룹을 제어하는 방법과 입력, 복호, 출력의 스케쥴 방법을 보인다.

가. 그룹 제어

Ⅲ장 1절의 예제에서 살펴본 바와 같이 하나의 패킷에 대하여 PE들은 동일한 그룹의 입력, 출력메모리를 사용한다. 이러한 그룹의 제어 방법은 표 4에 보인다. 입력되는 패킷이 사용하는 PE가 남아 있는 PE 보다 작거나 같은 경우, 즉 PE가 여유가 있는 경우, 사용되지 않는 PE를 해당 패킷의 복호에 사용하기 위하여 그룹을 바꾸지 않는다.

입력되는 패킷이 사용하는 PE가 남아 있는 PE 보다 큰 경우는, 그룹을 바꾸어 해당 패킷의 복호를 위하여 PE1부터 사용될 수 있도록 한다. 만일 이와 같은 경우 해당 패킷을 두 개의 그룹에 나누어 할당하게 되면, 제어가 매우 복잡해지고 구현이 어려워진다. 또는 만일 해당 패킷을 하나의 그룹에 대기하였다가 할당한다면, 처리율의 저하를 가져올 수 있다.

본 논문에서 제안한 제어방식은 간단하게 제어기를 구현하면서 처리율을 높일 수 있는 장점이 있다.

표 4. n차 병렬 복호기에서의 그룹제어
Table 4. Group control of parallel turbo decoder degree n.

제어 1. n차 병렬복호기에서의 그룹 제어

```
초기 값
grp_idx = A; // A or B
pe_idx = 1; // 다음 패킷에 사용될 첫 PE 인덱스 (1~n)

입력되는 패킷에 대하여,
used_pe = PE 개수; // 입력 패킷이 사용하는 PE수
if (n - pe_idx + 1 < used_pe) begin // 여유 PE가 없으면
    change group; // A to B, B to A
    pe_idx = used_pe + 1;
end
else begin // 여유 PE가 있으면
    pe_idx = pe_idx + used_pe;
end
```

나. 입력 시작 제어

그룹 제어와 동시에 패킷이 입력될 위치 및 사용되는 PE가 결정되기 때문에 입력 제어기는 표 5와 같이 해당 위치의 입력 메모리가 사용가능하다면 입력을 시작하도록 제어한다.

그림 4의 패킷 5와 패킷 8과 같이 이전 패킷의 입력 후 바로 입력되지 못하고, 기다려야 하는 경우가 있다. 이는 이전에 입력된 패킷의 복호에 해당 입력 메모리가 사용되는 경우이다. 이러한 경우를 처리하기 위하여 입력 제어기는 입력 메모리의 상태를 모니터링 하여야 하고, 메모리의 상태 제어에 대하여는 뒤에서 다룬다.

표 5. 제안된 복호기의 입력 시작 제어
Table 5. Input start control of proposed Turbo decoder.

제어 2. 입력 시작 제어

```
입력될 패킷에 대하여,
if(해당 위치의 입력 메모리 상태가 IDLE 상태)
    입력 시작;
else
    wait;
```

다. 복호 시작 제어

표 6은 복호 시작의 제어 방법을 보인다. 패킷의 입력과 출력은 순차적이기 때문에 입력된 패킷의 순서대로 복호가 시작될 수 있도록 제어된다. 이를 위하여 패킷의 입력이 시작되면, 해당 패킷이 사용하는 그룹과 사용하는 PE 정보를 FIFO에 저장한다.

복호 제어기는 FIFO에서 가장 먼저 입력된 정보와 함께 입력 메모리, PE, 출력 메모리의 상태를 관찰하여, 표에서의 조건들을 만족하면 해당 패킷을 FIFO에서 꺼내고 복호를 시작한다. 예를 들어 그림 4의 패킷 5는 ②, ③ 조건을 먼저 만족하지만, ①조건을 기다렸다가 복호를 시작하는 경우이고, 패킷 4, 6, 7은 ①, ③ 조건을 먼저 만족하지만 ② 조건을 기다렸다가 복호를 시작하는 경우이다. 패킷 8은 ①, ② 조건을 먼저 만족하지

표 6. 제안된 터보 복호기의 복호 시작 제어
Table 6. Decode start control of proposed Turbo decoder.

제어 3. 복호 시작 제어

```
패킷의 입력이 시작되면,
push the packet information into FIFO // 사용하는 그룹과 PE
if(가장 먼저 입력된 패킷이 다음 조건을 만족
  ① 해당 위치의 입력 메모리 상태가 입력완료 상태
  ② 해당 위치의 PE가 IDLE 상태
  ③ 해당 위치의 출력 메모리 상태가 IDLE 상태
) begin
    복호 시작;
    pop the packet;
end
else
    wait;
```

만 ③ 조건을 기다렸다가 복호를 시작하는 경우이다.

라. 출력 시작 제어

표 7은 출력의 시작을 제어하는 방법을 보여준다. 패킷이 입력된 순서대로 패킷이 출력되어야하기 때문에 패킷이 입력되면, 복호 제어기와 마찬가지로 해당 패킷이 사용하는 그룹과 PE에 대한 정보를 FIFO에 넣는다. 이 때 사용되는 FIFO는 복호 시작 제어에서 사용된 FIFO와는 독립적인 FIFO이다. 출력 시작 제어기는 FIFO에 가장 먼저 입력된 패킷의 정보와 함께 출력 메모리의 상태를 모니터링 하여 표에서의 조건들을 만족하면 해당 패킷을 FIFO에서 꺼내고 출력을 시작한다.

그림 4의 패킷 2, 6, 7은 ①조건을 만족하지만 ②를 기다렸다가 출력을 시작하는 경우이다.

표 7. 제안된 터보 복호기의 출력 시작 제어
Table 7. Output start control of proposed Turbo decoder.

제어 4. 출력 시작 제어

```
패킷의 입력이 시작되면,
push the packet information into FIFO // 사용하는 그룹과 PE
if(가장 먼저 입력된 패킷이 다음 조건을 만족
  ① 해당 위치의 출력 메모리 상태가 출력 준비 상태(복호가 완료된 상태)
  ② 현재 출력중인 패킷이 없음
) begin
    출력 시작;
    pop the packet;
end
else
    wait;
```

마. 상태 제어기(FSM : Finite State Machine)

앞 절에서 보인 입력, 복호, 출력 시작의 제어를 위하여 입력, 복호, 출력 각각에 대한 상태 제어기가 필요하다. n차 이중 버퍼 병렬 터보 복호기에는 2n개의 입력 FSM과 출력 FSM, n개의 복호FSM이 필요하다.

그림 5는 각각의 FSM을 보여준다. 각 입력 메모리의 상태는 입력의 시작과 복호의 시작을 제어하는데 사용된다. 복호 시작 제어기는 입력메모리가 입력완료 상태인지, 입력 시작 제어기는 해당메모리가 IDLE 상태인지 모니터링하기 때문에 그림과 같이 입력 FSM은 4가지 상태를 가지도록 구성된다.

각 PE의 상태는 복호의 시작을 제어하는데 사용된

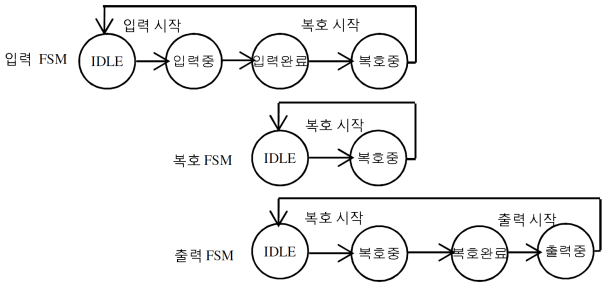

그림 5. 제안된 터보 복호기의 상태제어기
Fig. 5. Finite State Machine of proposed Turbo decoder.

다. 복호 시작 제어기는 PE가 IDLE인지 모니터링하기 때문에 그림과 같이 2가지 상태를 가지도록 구성된다.

각 출력 메모리의 상태는 복호의 시작과 출력의 시작을 제어하는데 사용된다. 출력 시작 제어기는 출력메모리가 출력 준비(복호 완료) 상태인지, 복호 시작 제어기는 해당 메모리가 IDLE상태인지 검사하기 때문에 그림과 같이 출력 FSM은 4가지 상태를 가지도록 구성된다.

각각의 FSM은 한 패킷에 대하여 동시에 상태를 전환하도록 제어되어야 한다. 즉 그림 4에서 패킷 1의 경우, 입력 FSM 1, 2는 동시에 상태를 전환하여야 하고, 복호 FSM 1, 2, 그리고 출력 FSM 1, 2 또한 마찬가지이다. 또한 한 패킷에 대하여 그림 5와 같이 복호 시작과 함께 세 가지 FSM 의 상태가 동시에 전환된다.

3. 인터리버(디-인터리버)

그림 1의 복호기 전체 구성에서 보는 바와 같이 인터리브(디-인터리브) 동작을 위해 인터리브주소(디-인터리브 주소)를 저장할 수 있는 메모리를 사용한다. 인터리브 주소의 저장은 패킷의 입력동작과 동시에 시작되어 입력의 종료와 동시에 완료되기 때문에 입력동작의 일부로 생각하면 된다.

n차 병렬 터보복호기에서 인터리브 주소 메모리는 n개로 구성되며, 각 메모리에는 어떤 메모리로 인터리브 될 것인가에 대한 메모리 인덱스(bank)와 해당 메모리에서 몇 번째 위치에 저장될 것인가에 대한 오프셋주소(offset)가 저장된다. n 개의 PE를 사용하는 패킷의 경우, 메모리 인덱스(bank)는 1~n 의 값을 가진다.

생성된 메모리 인덱스는 인터리버 메모리 뒤에 위치한 먹스의 선택신호로 들어가서 복호 모듈들에서 출력되는 LLR(Log Likelihood Ratio) 데이터를 적절한 LLR 메모리로 보내어 주는 역할을 한다.

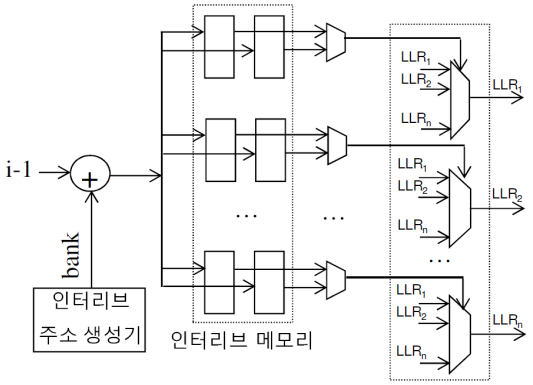

그림 6. 제안된 터보 복호기의 인터리버 구성
Fig. 6. Interleaver design of proposed Turbo decoder.

본 논문에서 제안한 것처럼 동시에 여러 패킷에 대하여 독립적인 인터리브/디-인터리브 동작을 위해서는 인터리브(디-인터리브) 주소를 생성하여 저장할 때, 메모리의 인덱스(bank)를 갱신하여 저장하여야 한다.

그림 6과 같이 생성되는 메모리 인덱스(bank)에 해당 패킷이 사용하는 첫 메모리의 인덱스(i-1)를 더하여 저장하고 오프셋주소(offset)는 생성되는 그대로 사용한다. PE 2개를 사용하는 패킷에 대하여 메모리 인덱스는 1, 2의 값을 가진다. 해당 패킷이 PE 5, 6 에 할당되었다면, 메모리 인덱스 1, 2에 첫 메모리 인덱스 4(=5-1)를 더하여 메모리 인덱스를 5, 6으로 저장하면, 해당 패킷은 5, 6 메모리 사이에서 인터리브(디-인터리브)된다.

Ⅳ. 실험 결과

본 논문에서 제안한 터보 복호기의 성능을 검증하기 위해 기존의 이중 버퍼를 이용한 병렬 터보 복호기와 제안한 터보 복호기를 VHDL을 이용해 구현하였다. 병렬 차수는 32로 하였으며 XILINX ISE를 이용하여 XILINX FPGA XCVLX200를 타겟 디바이스로 하여 합성하였다. 두 복호기 모두 동일한 PE를 사용하여 구현하였으며, 합성 결과 두 복호기는 80MHz에서 동작 가능하며, 표 8에서와 같이 제안된 복호기는 추가적인 제어기의 구성으로 인하여 타겟 디바이스의 슬라이스 사용면에서 기존 복호기보다 약 16.4%의 증가를 보인다.

두 복호기의 성능 비교를 위해 동일한 길이의 패킷의 연속적인 입력에 대하여, 그리고 임의의 길이의 패킷의 연속적인 입력에 대하여 두 가지 복호기의 처리율을 측정 및 비교하였다. 실험은 두 가지 복호기 모두 동일한

표 8. FPGA 사용 면적 비교
Table 8. FPGA Area Comparison.

	기존복호기	제안된 복호기	면적 증가율(%)
Slice	69,052	80,408	16.4

표 9. 동일한 길이의 패킷의 연속적인 입력에 대한 처리율
Table 9. Throughput comparison for continuous packets of same length.

K	PE	기존복호기		제안된 복호기	
		효율(%)	처리율 (Mbps)	효율(%)	처리율 (Mbps)
40	1	3.6	2.6	91.0	72.8
512	2	8.0	6.4	99.2	79.3
1024	4	16.1	12.9	99.6	79.6
2048	8	32.2	25.8	99.8	79.8
4096	16	64.3	51.5	99.9	79.9
6144	32	99.9	79.9	99.9	79.9

표 10. 임의의 길이의 패킷의 연속적인 입력에 대한 처리율
Table 10. Throughput comparison for continuous packets of various length.

패킷의 총비트수	기존복호기		제안된 복호기	
	효율(%)	처리율 (Mbps)	효율(%)	처리율 (Mbps)
1,933,120	39.9	31.9	83.3	66.6
1,821,752	37.4	29.9	81.6	65.3
1,865,088	38.4	30.7	82.3	65.8
1,866,080	38.5	30.8	84.0	67.2
1,946,224	39.9	31.9	83.6	66.9

조건으로 8회의 반복 복호를 수행하였다. 디코딩 효율을 이용해서 성능을 비교하였으며 효율 값은 처리하는 블록의 총 비트수를 첫 패킷의 출력 시작부터 마지막 패킷의 출력 종료에 소요되는 클록수로 나누어서 계산하였다. 표 9와 10에서 동작 주파수 80MHz 하에서의 효율과 처리율(Mbps)을 보인다.

표 9와 같이 동일한 길이의 패킷의 연속적인 입력 시 기존 복호기는 패킷의 길이가 짧을수록 낮은 효율을 보이고 병렬차수가 32인 패킷에 대하여만 99%의 효율을 보인다. 복호기의 효율이 99%임은 거의 모든 순간에 모든 PE들이 사용되고 있음을 뜻하며 패킷들이 패킷간의 지연시간 없이 연속적으로 복호를 완료하고 출력됨을 의미한다. 기존의 복호기는 한 번에 하나의 패킷만을 복호할 수 있고, 병렬차수가 32인 패킷에 대해서만

99%의 효율을 보인다.

본 논문에서 제안한 복호기는 병렬차수가 32인 패킷의 경우, 기존의 복호기와 동일하게 동작하기 때문에 기존 복호기 결과와 같이 99%의 효율을 보인다. 또한 제안한 복호기는 병렬차수가 16이하인 패킷의 경우에도 대부분 99%의 효율을 보인다. 이는 하나의 패킷이 출력을 완료하기 전에 남는 PE에 할당된 다음 패킷의 복호가 완료되어 연속적으로 패킷이 출력됨을 의미한다.

표 10은 서로 다른 임의의 길이의 1000개의 패킷을 연속적으로 입력하였을 때의 처리율의 비교 결과를 보인다. 기존 복호기는 평균 40%이하의 효율과 32Mbps 이하의 처리율을 가지는 반면 제안된 복호기는 평균 80% 이상의 고 효율과 65Mbps 이상의 높은 처리율을 보인다. 임의의 길이의 패킷들의 연속적인 입력에 대하여 제안된 복호기는 기존 복호기의 처리율 대비 100% 이상의 처리율 향상을 보여준다.

V. 결 론

본 논문에서는 기존의 이중 버퍼 구조를 이용한 병렬 터보 복호기에 새로운 제어방법을 적용하여 동시에 하나 이상의 패킷을 복호할 수 있도록 하는 방법을 보인다. 기존의 복호기는 패킷의 길이가 짧을수록 낮은 처리율을 보이며, 제안한 제어기는 현재 패킷의 복호에 사용되지 않는 PE들을 이어지는 다음 패킷의 복호에 사용할 수 있도록 제어한다. 제안한 복호기는 새로운 제어기의 구성으로 인하여 기존의 복호기에 비하여 약 16%의 면적 증가를 보이지만, 동일한 길이의 패킷의 연속적인 입력에 대하여 기존 복호기의 처리율 대비 최대 28배의 높은 처리율을 보이며, 임의의 길이의 패킷의 연속적인 입력에 대하여서는 기존 복호기 대비 100% 이상의 처리율 향상을 보임을 확인하였다.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes,” *IEEE International Cpnference on Communication (ICC'93)*, vol. 2, pp. 1064-1070, May. 1993.

[2] J. Hsu, and C. Wang, “A parallel decoding scheme for turbo codes,” *IEEE International Symposium on Circuits and Systems*, Vol. 4. pp. 445-448, June 1998.

[3] Y. Zhang, K. K. Parhi, “Parallel Turbo Decoding,” *IEEE International Symposium on Circuits and Systems*, vol. 2, pp. 509-512, May 2004.

[4] B. Bougard et al., “A Scalable 8.7nJ/bit 75.6Mb/s Parallel Concatenated Convolutional (Turbo-) CODEC,” *IEEE ISSCC Dig. Tec. Papers*, 2003, pp. 152 - 153.

[5] R. Dobkin, M. Peleg, and R, Ginosar, “Parallel Interleaver Design and VLSI Architecture for Low-Latency Map Turbo Decoders,” *IEEE Trans. on VLSI Systems*, vol. 13, no. 4, pp. 427-438, Apr. 2005.

저 자 소 개

이 원 호(정회원)
2002년 서강대학교 컴퓨터학과 학사 졸업.
2008년 서강대학교 컴퓨터학과 석사 졸업.
2013년 현재 (주)AP위성통신 SoC팀 선임연구원.
<주관심분야 : 통신 시스템, 정보이론과 코딩, ASIC설계>

박 희 민(정회원)
1993년 서강대학교 전자계산학과 학사 졸업.
1995년 서강대학교 컴퓨터공학과 석사 졸업.
2006년 미국 UCLA 전자공학과 박사 졸업.
2013년～현재 상명대학교 컴퓨터소프트웨어 공학과 조교수.
<주관심분야: 웹기반정보시스템, 클라우드컴퓨팅, 사이버물리 시스템, 네트워크 임베디드 시스템>

임 종 석(평생회원)
대한전자공학회 논문지
제48권 SD편 제1호 참조.