논문 99-24-11A-11 한국통신학회논문지 '99-11 Vol.24 No.11A

직접 매핑 기법을 이용한 MPEG-2 TS-to-PS 변환 알고리듬

신화선*, 정회원 김용한*, 김제우**, 최병호**, 송병철**, 용석진**,정광모**, 동용배**

MPEG-2 TS-to-PS Conversion Algorithm Using Direct Mapping Method

Hwa Seon Shin*, Yong Han Kim*, Je Woo Kim**, Byeongho Choi**, Byoungchol Song**, Sukjin Yong**, Kwangmo Jung**, Yong Bae Dhong** ***Regular Members***

요 약

본 논문은 MPEG-2 TS(transport stream)를 PS(program stream)로 변환하는 효율적인 알고리듬을 제안한다. 기존 방식은 TS를 역다중화(demultiplex)한 뒤 PS로 재다중화(remultiplex)하는 과정으로 구성된다. 이 과정에서 PLL(phase locked loop)을 사용하여 시스템 클록을 복원하여 시간 정보를 재구성하는 과정이 필요하다. 본 논문에서는, 직접 매핑(direct mapping) 기법을 통한 효율적인 TS-to-PS 변환 알고리듬을 제안한다. 제안 방식에서는 PLL이 필요하지 않으며 시스템 클록을 복원하지 않는다. TS로부터 변환에 필요한 최소한의 정보만을 분리한 후 별도 처리하고, 나머지 시간 정보를 포함하여 두 스트림 간의 동일 부분을 수정없이 복사하는 방식이다. 기존 방식과 동일한 기능을 하면서도 PLL을 필요로 하지 않기 때문에 TS-to-PS 변환기의 제작 비용과 시간을 크게 절감시킨다. 제안된 알고리듬은 디지털 저장 매체의 TS 입력부 또는 S/W적인 TS-to-PS 변환에 활용될 수 있다.

Abstract

This paper describes an efficient algorithm to convert an MPEG-2 Transport Stream (TS) into Program Stream (PS). Conventional MPEG-2 TS-to-PS conversion algorithms are composed of two logical parts; de-multiplexing of TS and re-multiplexing into PS. In the process, it is necessary to recover system clock and resample timing information. In this paper we propose an efficient TS-to-PS conversion algorithm based on direct mapping. It eliminates the need of phase locked loop(PLL) for recovering system clock. It extracts out of a TS minimum information required to build PS headers and copies without modification the remainder of input TS data directly into the output PS file. The converted PS provides the same audio-visual presentations as the original TS through typical MPEG-2 H/W decoders. As a result, it reduces costs of H/W and computational complexity. Also, due to its simplicity, development period will be greatly reduced. It is very useful for applications such as digital storage media(DSM), which accepts a TS as an input, and software-based TS-to-PS conversion.

I. 서 론

MPEG-2 시스템 표준은 두 가지 다중화 방식을 명시하고 있다[1]. 하나는 PS(program stream)이고 다른 하나는 TS(transport stream)이다. 각각 다른 애플리케이션을 위해 최적화되어 있다[2]. PS는 종전의 MPEG-1 시스템 스트림[3]과 유사한 저장용 스트림으로 전송 오류 발생 가능성이 적은 채널, 보통 DSM(digital storage media)에 사용된다. 반면 TS는

* 서울시립대학교 전자전기공학부(yhkim@uoscc.uos.ac.kr), **전자부품연구원

논문번호 : 99355-0818, 접수일자 : 1999년 8월 18일

전송 오류 발생 가능성이 비교적 큰 채널, 보통 방송 및 전송 용도로 사용된다. 이러한 특징에 따라 PS는 가변 길이의 긴 패킷을 갖는 반면 TS는 짧고 고정된 길이의 패킷으로 구성된다. 디지털 방송 시대가 본격화됨에 따라 디지털TV 신호의 부호화 표준으로 채택된 MPEG-2는 그 중요성이 날로 커지고 있다[4,5]. 이에 따라 대부분의 디지털 방송용 장비와 방송용 매체에서는 TS 규격이 사용되고 있다. 반면 DVD(digital versatile disk) 같은 DSM에는 PS가 사용되고 있다[6]. 따라서 디지털TV 방송을 수신하여 이를 DVD나 DVD-RAM(digital versatile disk - random access memory)과 같은 저장 매체에 저장하려면 TS를 PS로 변환하는 것이 필요하다. 향후 디지털TV 방송 서비스가 확산되고 AV 디지털 저장 매체의 사용이 활성화되면, 여러 제품에 TS에서부터 PS로 변환하는 기능이 필수적으로 필요하다[7]. 본 논문에서는 TS로부터 PS를 생성하는 장치를 "TS-to-PS 변환기"라 부르기로 한다.

현재까지 알려진 TS-to-PS 변환기의 대부분은 TS를 역다중화(demultiplex)한 뒤 PS로 재다중화(remultiplex)하는 방식이다. 역다중화 과정은 프로그램 클록 참조치(PCR; program clock reference)와 PLL(phase locked loop)[8-10]을 통해 시스템 클록(system clock)를 복원하여 ES(elementary stream)의 동기(synchronization)를 맞추는 과정과, PES(packetized elementary stream)를 ES로 복원하는 과정을 포함한다. 재다중화 과정은 시스템 클록을 사용하여 동기가 가능하도록 ES의 타임 스탬프(time stamp)를 기록하여 PES를 생성하고 시스템 클록의 타임 스탬프, 즉 시스템 클록 참조치(SCR; system clock reference)를 기록하는 과정을 포함한다. 그림 1에 이러한 TS-to-PS 변환기의 블록도를 보였다.

역다중화한 뒤 재다중화 과정을 거치는 방식의 장점은 시스템 클록을 복원하여 재다중화하므로 변환 후에도 ES 간의 동기를 표준에서 정의한 대로 보장하는 것이다. ES의 동기는 시스템 클록과 PES 패킷(packet) 상의 타임 스탬프가 일치할 때 이루어지므로, 이러한 방식은 표준에 충실한 방식이다. 그러나 DSM 저장 등에 사용될 TS-to-PS 변환기는 역다중화기(demultiplexor)가 불필요함에도 불구하고 이를 구현해야 할 뿐만 아니라 이 과정에서 TS 역다중화기에 필수적인 PLL[11]까지 구현해야 하는 단점이 있다. TS 역다중화기를 이미 갖고 있는 장비의 경우, TS-to-PS 변환기는 그 자체의 기능보다 역다중화기에 첨가된 부가 기능의 성격이 강하다. 이러한 이유로 TS 역다중화기를 개발하지 않는 산업 현장에서는 필요할 때마다 MPEG-2 시스템 표준에 대한 연구 과정을 거쳐 미봉책으로 구현하는 것이 현실이다. 더구나 역다중화와 재다중화 과정을 거치지 않고도 TS-to-PS 변환의 가능한 지, 또 이 경우 시간 정보는 어떻게 처리하여야 하는지 등에 대한 공개된 해결책이 현재로서는 없기 때문에, 연구 과정 뿐만 아니라 구현 상의 시행 착오로 인한 시간적, 기술적 중복과 낭비를 초래하고 있다.

본 논문에서는 사실상 TS 역다중화와 PS 재다중화를 거치지 않고 변환에 필요한 데이터 필드들만을 직접 매핑함으로써, 변환에 소요되는 계산량을 최소화한 방식을 제안한다. 이 방식은 역다중화 및 재다중화 과정을 통한 방식과 같이 ES의 동기를 그대로 유지하면서도 PLL과 타임 스탬퍼(time stamper), 그리고 역패킷화기(depacketizer)와 패킷화

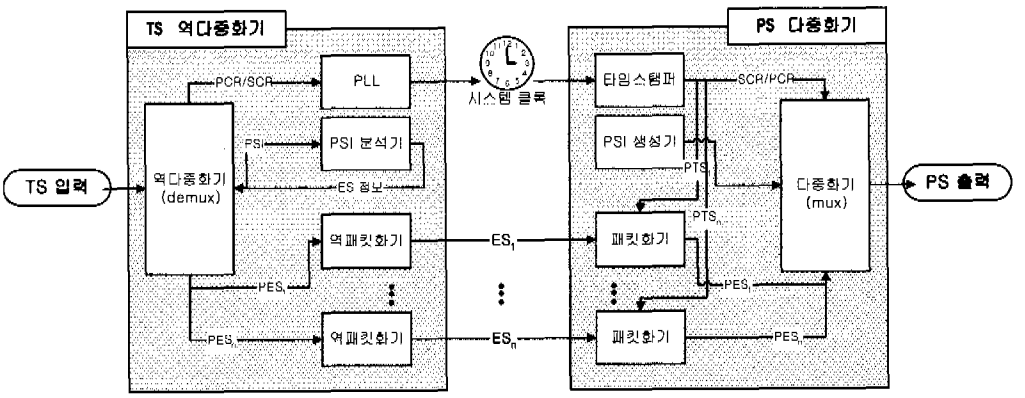


그림 1. 역다중화기와 다중화기를 갖는 기존 TS-to-PS 변환기

기(packetizer)를 구현할 필요가 없다. 따라서 하드웨어 제작 비용을 절감할 뿐만 아니라 소프트웨어로도 구현이 가능하다. 특히 TS-to-PS 변환기를 주 목적으로 사용하는 플랫폼에서는 비용을 크게 절감할 뿐만 아니라 개발 기간까지 크게 단축시킨다.

다음 장에서 직접 매핑을 사용한 방식의 구조와 알고리듬에 대해 설명한다. 우선 변환에 관련된 두 시스템 디코더 간의 관계에 대해 설명한 후, TS 헤더의 분석, PS 헤더의 구성, 그리고 PES 패킷의 직접 매핑 방식 등을 기술한다. III장에서는 알고리듬 구현에 대해 설명하고 실험 결과를 보인다. IV장에서 결론과 향후 연구 방향을 제시한다.

Ⅱ. TS-to-PS 변환 알고리듬

본 장에서는 직접 매핑을 이용한 TS-to-PS 변환기의 전체적인 구조와 알고리듬을 설명한다. 하위 모듈에 대한 동작과 이론은 이후 절에서 상세히 설명한다. 본 논문에서는 이미 디스크램블링(descrambling)된 TS 신호를 TS-to-PS 변환기의 입력으로 가정한다. 스크램블링(scrambling)은 플랫폼마다 서로 다른 방식에 의해 구현될 수 있다.

전체 변환 과정을 그림 2에 순서도로 나타내었다. 그림에서 그림자가 있는 블록은 하위 블록이 있음을 나타낸다. TS-to-PS 변환 알고리듬은 크게 TS 헤더 분석, PS 헤더 구성, 그리고 PES 패킷 매핑 과정의 연속으로 이루어진다. PSI 정보 분석은 항상 처음에 수행되어야 하며 분석 과정이 자주 반복되지도 않는다[12,13]. 우선 TS 헤더 분석을 통해 시간 정보의 유무를 검사한다. 시간 정보가 존재하면 PS 헤더를 구성한다. 이때 시간 정보는 PS 헤더로 그대로 매핑된다. 유료부하(payload)의 데이터가 PSI인지 PES패킷인지 검사하여 PSI이면 PSI 분석기를 수행하고, PES 패킷이면 PS 패킷의 PES로 매핑을 수행한다. PSI 분석 모듈은 일반적인 PSI 분석 방식을 따르며 프로그램 선택 기능과 선택한 프로그램을 구성하는 ES의 PID(packet ID), 종류(type) 그리고 개수(number) 정보 등을 제공한다[12-14]. TS 입력이 끝날 때까지 이러한 과정을 반복하고 입력이 끝나면 첫번째 PS 헤더의 시스템 헤더를 갱신하고 PS 종료 코드로 PS 파일을 닫는다.

이 변환기의 특징은 입력되는 TS 데이터를 바로 PS로 매핑하여 변환하는 것이다. PS 헤더 구성 과정에서 시간 정보를 그대로 복사하므로 PLL이나 타임 스탬퍼가 필요없다. PES 패킷 매핑 과정에서 PES 패킷을 바로 매핑하므로 역패킷화기나 패킷화

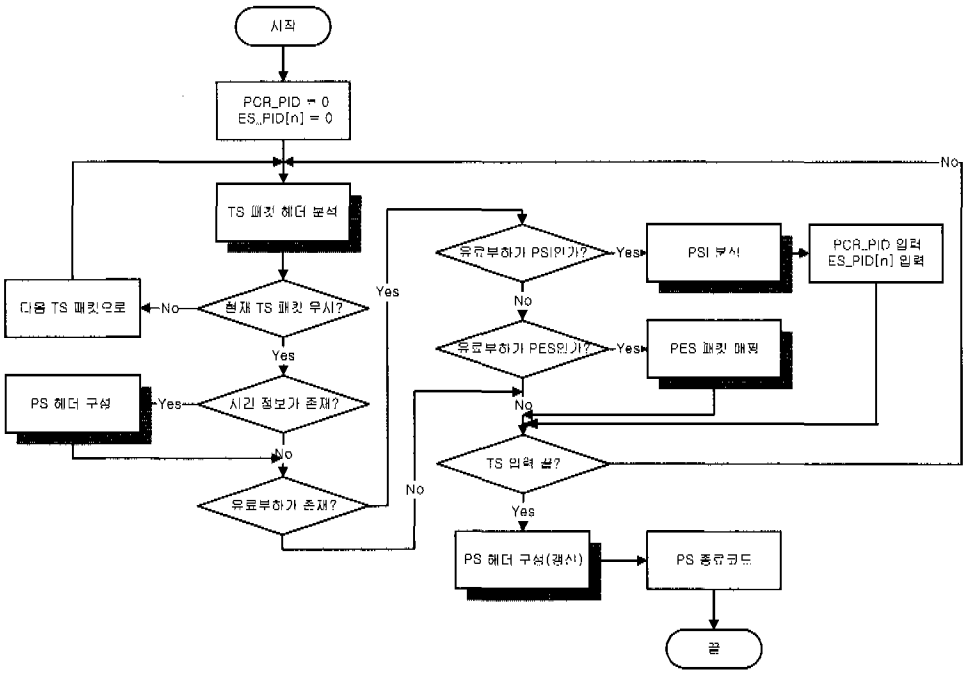


그림 2. 직접 매핑 기법을 이용한 TS-to-PS 변환 알고리듬

기가 필요없다. 그럼에도 불구하고 시스템 클록의 정보와 ES의 동기를 그대로 유지하면서 변환을 수행한다.

위에서 설명한 방식으로 알고리듬이 동작하려면 다음과 같은 일들이 검증되어야 한다.

(1) STD(system target decoder)의 일치: 서로 다른 구조를 갖는 TS와 PS 간의 변환이 이루어지기 위해서는 디코더 모델이 일치해야 한다. 다음과 같은 조건들을 만족시켜야 한다.

a) 시간 정보의 분해능(resolution)이 일치해야 한다.

b) 버퍼 모델이 일치해야 한다.

c) 시간 정보를 복사해도 전송률과 ES의 동기에 영향이 없어야 한다.

(2) TS 헤더 분석: TS 헤더 분석을 통해 변환 알고리듬에 필요한 정보를 얻을 수 있어야 한다. 다음과 같은 조건들을 만족시켜야 한다.

a) 시간 정보의 유무를 판단할 수 있어야 한다.

b) 유료부하의 유무의 판단하고, 만약 존재한다면 그 종류를 구분할 수 있어야 한다.

c) PES 패킷의 시작점을 알 수 있어야 한다.

d) PES 패킷 조각의 순서를 알 수 있어야 한다.

(3) PS 헤더 구성: 입력된 TS로부터 PS 헤더를 구성하는 데 필요한 정보를 얻을 수 있어야 한다. 다음과 같은 조건들을 만족시켜야 한다.

a) TS 헤더에서 바로 매핑 가능한 것과 그렇지 않은 것을 구분할 수 있어야 한다.

b) 바로 매핑할 수 없는 것은 입력된 TS로부터 만들어 낼 수 있어야 한다.

(4) PES 패킷 매핑: PES 패킷 조각들을 직접 매핑하기 위해서는 다음과 같은 조건들을 만족시켜야 한다.

a) 뒤바뀐 패킷 조각도 재조합하여 매핑할 수 있어야 한다.

b) 버퍼 크기를 결정하여 오버플로우(overflow)가 발생하지 않도록 할 수 있어야 한다.

1. T-STD와 P-STD의 일치

본 절에서는 TS와 PS의 STD 모델의 일치 여부를 확인한다. 직접 매핑을 통한 TS와 PS 간의 변환이 수행되기 위해서는 시스템 모델이 일치해야 한다. STD 모델은 크게 동기 모델과 버퍼 모델로 구성된다.

TS 안에는 여러 개의 프로그램을 포함할 수 있다[1,2,12]. 반면 PS는 하나의 프로그램만을 수용할 수 있으므로 TS와 PS의 디코더 모델은 원칙상 일치할 수 없다. 그러나 MPEG-2 시스템 표준은 TS 시스템 디코더 모델을 하나의 프로그램으로만 제한하였다[1]. 즉 PSI 정보를 통해 선택한 하나의 프로그램에 대해서만 디코더 모델을 정의한 것이다. 이로 인해 하나의 프로그램에 대해서만 디코더 모델을 정의한 PS와 유사성이 생기게 되고 이를 통해 TS와 PS 간 변환이 가능하다.

표준에서 제시한 TS와 PS의 시스템 모델을 인용하여 각각 그림 3와 그림 4에 제시하였다[1]. 두 시스템 모델은 유사한 모양을 갖고 있다. 제일 위쪽의 프리젠테이션(presentation) 버퍼(O_1)가 있는 곳은 비디오의 경우이고 그 아래는 오디오의 경우, 제일 아래쪽은 시스템 제어 부분이다. TS의 경우는 PES 패킷들이 조각들로 나뉘어 전송되기 때문에 이를 위해 PS와 달리 전송 버퍼(TB; transport buffer)를 필요로 한다. 전송 버퍼를 이용하여 수신된 PES 패킷을 조합하여 주 버퍼(B; main buffer)로 전송한다. 따라서 TS와 PS의 버퍼 모델은 주 버퍼 이후부터 일치한다. 또한 TS 측에 다중화를 위한 버퍼가 있는 것을 제외하고는 TS와 PS의 주 버퍼 크기 또한 일치한다[1]. TS와 PS의 시스템 클록 주파수(system_clock_frequency)는 27MHz로 일치한다. 이 주파수를 바탕으로 작동하는 시스템 클록의 시각을 각각 PCR 또는 SCR로 나타낸다. PCR과 SCR 모두 42비트 길이의 필드로 각각 PCR과 SCR 값을 300으로 나누어 몫과 나머지로 필드에 표기한다.

아래 식 (1)과 식 (2)는 TS의 비트율(bitrate)인 transport_rate와 PS의 비트율인 program_mux_rate 필드 값에 대한 계산식을 나타낸다.

$$R_{tr} = \frac{i'' - i'}{PCR(i'') - PCR(i')} \times f \qquad (1)$$

$$R_{pr} = \frac{i'' - i'}{SCR(i'') - SCR(i')} \times \frac{f}{50} \qquad (2)$$

여기서, f, R_{tr}, R_{pr}은 각각 system_clock_frequency, transport_rate, 그리고 program_mux_rate를 나타낸다. i'와 i''는 각각 PCR 필드의 마지막 바이트의 인덱스를 나타내고 $PCR(i')$, $PCR(i'')$는 그에 해당하는 PCR 값을 가리킨다. SCR의 경우도 마찬가지다. 두 식의 차이는 program_mux_rate가 50바이트 단위를 사용한다는 차이 밖에 없다.

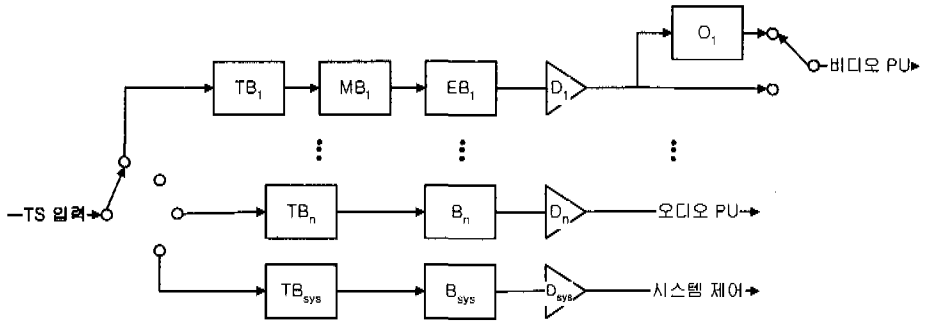


그림 3. TS의 STD 모델

program_mux_rate는 실제 PS의 팩 헤더(pack header)에 존재하는 필드이지만 transport_rate는 개념상 존재하는 값이다. PS는 SCR 필드 외에 program_mux_rate 필드를 두어 비트율을 표시하고 이를 이용해 ES 간의 동기를 맞추도록 한다[7].

TS의 시간 정보를 PS의 시간 정보로 변환하는 방식은 두 가지가 있다. 한 가지 방식은 시스템 클록을 27MHz로 가정한 후 PCR을 이용하여 비트율을 구한다. 이 비트율을 이용하여 바이트 당 소요되는 시간을 구하여 PS의 SCR 값을 구하는 방식이다. 이러한 방식은 TS와 PS의 비트율을 일치시킨다는 장점이 있다. 그러나 TS를 PS로 변환하는 과정에서 TS 내의 중복된 헤더들이 많이 사라지면서 바이트 수의 감소를 가져온다. 이로 인해 ES의 타임 스탬프에도 영향을 주게 되어 다시 타임 스탬프를 생성해야 되는 결과를 초래한다. 즉 비트율을 고정시킨 대신 SCR, PTS(presentation time stamp) 그리고 DTS(decoding time stamp)까지 재생성해야 하는 단점을 갖는다. 반면 PCR를 SCR로 바로 매핑하여 시간 정보를 고정하고 비트율을 변경하는 방식을 생각할 수 있다. PCR과 SCR은 같은 분해능을 가지므로 PCR 값을 SCR 필드로 복사하는 것은 문제가 되지 않는다. 변환 후 PS의 크기는 TS에 비해 짧아지므로 비트율은 떨어지게 된다. 따라서 비트율을 다시 계산해야 한다. 계산은 식 (2)를 사용한다. 이 때 f 값은 표준에 따라 27 MHz를 사용한다. 이 방식의 장점은 시스템 클록 정보가 재생성되지 않았으므로 PES에 기록된 ES의 타임 스탬프를 수정하지 않아도 된다는 것이다. 뿐만 아니라 PLL을 사용하는 PS 디코더에 대해서는 program_mux_rate 값을 계산하지 않아도 되는 장점도 있다. 3절에서 이에 대해 상세히 설명한다. 본 논문에서 제안하는 방식은 시간 정보를 그대로 복사하는 후자의 방식을 사용한다. 이 방식의 장점으로 PLL을 사용하지 않고서도 ES의 동기를 그대로 유지한 채 TS를 PS로 변환할 수 있으며 시스템 클록도 그대로 유지된다.

2. TS 패킷 헤더 분석

본 절에서는 직접 매핑을 통한 TS-to-PS 변환 과정에 필요한 정보들을 TS 패킷 헤더로부터 얻을 수 있는 지 검증한다. 이 과정에서 시간 정보, 즉 PCR의 존재 유무와 유료부하의 유무 및 종류를 판단할 수 있어야 하며, PES 패킷을 재조합하는 데에 필요

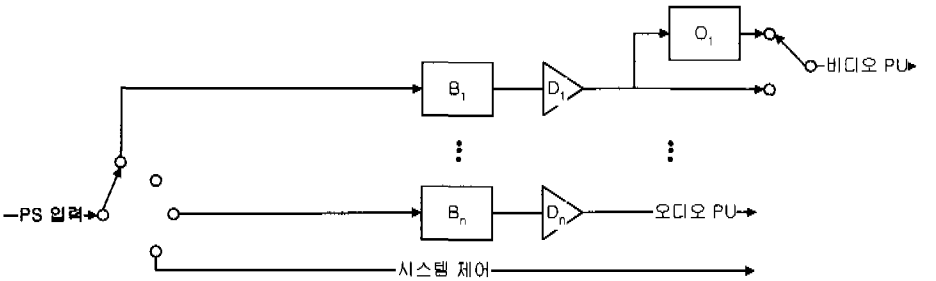


그림 4. PS의 STD 모델

한 시작점과 순서에 관한 정보를 얻어낼 수 있어야 한다.

TS 패킷 헤더 분석 과정을 그림 5에 보였다. TS 패킷 헤더 분석은 PSI 분석 모듈을 통해 얻어낸 PMT(program map table)정보를 통해 TS 패킷이 갖는 정보를 구별해 낸다[12-14]. 즉 PMT에서 현재의 PID 값을 참조하여 현재 유료부하에 있는 데이터가 PSI인지 PES 패킷인지 구별해 낸다. (PID가 0x1 FFF일 경우는 널(null) 패킷이고 이를 처리하는 과정이 그림 5에 있다.) 이를 위해 항상 PSI 분석 모듈은 가장 우선 수행되어야 한다. 시스템 클록을 복원할 수 있는 시간 정보인 PCR을 나타내는 PID가 PCR_PID이다. PCR 필드는 적응 필드(adaptation field) 내에 있으며 그림 5와 같이 PID가 PCR_PID이고 PCR_flag가 1일 때만 존재한다. PCR_PID는 다른 ES의 PID와 중복될 수 있다[2,13].

PES 패킷이 TS 패킷으로 다중화될 때 상대적으로 긴 PES 패킷이 여러 개로 나뉘어져 TS 패킷이 된다. 이때 처음 시작되는 패킷은 payload_unit_start_indicator 플래그(flag)가 1로 설정되고 나머지 패킷들은 0으로 설정된다. 새로운 PES 패킷은 새로운 TS 패킷에서만 시작할 수 있다[1]. 따라서 payload_unit_start_indicator가 1인 TS 패킷은 항상 PES 패킷 헤더를 갖고 있는 PES 패킷의 조각을 유료부하로 운반한다[2]. PSI 섹션(section)이 시작될 때도 payload_unit_start_indicator가 1로 설정되지만 방식은 PES 패킷 때와 조금 다르다[14].

TS 패킷이 생성될 때 TS 패킷 헤더의 continuity_counter 필드 값이 1씩 차례대로 증가한다. 이 필드는 4비트이므로 0부터 15까지의 값을 갖고 15 값을 지나면 다시 0으로 순환된다. 따라서 continuity_counter 값을 이용하여 전송받은 TS 패킷 순서를 재배열할 수 있으며 중복 패킷을 제거할 수 있다.

transport_error_indicator는 채널 측에서 알려주는 정보로서 이 플래그가 1이면 현재 TS 패킷의 내용이 올바르지 않다는 것을 알려준다. 이 정보를 이용하여 채널 오류가 있으면 현재 TS 패킷을 버리고 다음 TS 패킷을 처리한다. adaptation_field_control은 적응 필드와 유료부하의 존재 유무를 알려주는 필드이다. 적응 필드는 adaptation_field_control이 2 또는 3일 때만 존재하며 유료부하는 1 또는 3일 때만 존재한다. 따라서 adaptation_field_control이 0 또는 2일 때는 유료부하가 존재하지 않는다. 적응 필드는 PCR 정보가 존재하는 곳이기 때문에 중요한 의미를 갖는다.

3. PS 헤더 구성

본 절에서는 PS 헤더 구성을 위한 구문들의 매핑 가능 여부를 확인한다. TS로부터 바로 얻어낼 수 없는 정보를 입력된 TS로부터 만들어낼 수 있는

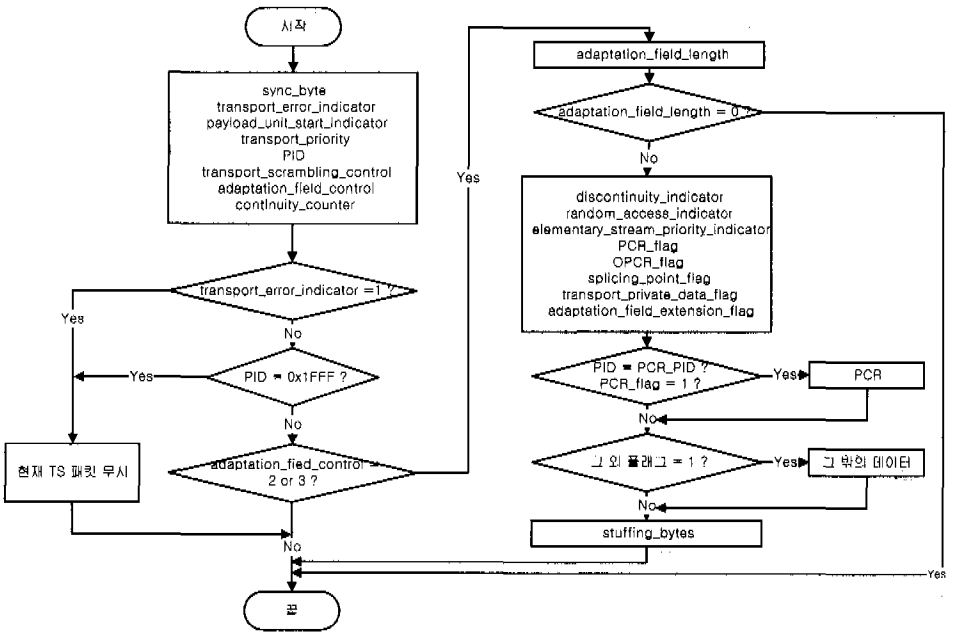


그림 5. TS 패킷 헤더 분석 과정

지에 대해서도 검증한다.

PS 헤더의 구성 과정을 그림 6에 나타내었다. PS 헤더 구성 절차는 일반적인 구성 과정 외에 별도의 갱신(update) 과정을 가진다. 이러한 갱신 과정은 시스템 헤더(system header)에 대해서만 수행된다. 시스템 헤더는 그림 6에서와 같이 반드시 PS 처음에 존재해야 한다[1]. 시스템 헤더에는 PS 전체에 대한 요약 정보가 포함되어 있으며 이를 이용해 PS 시스템 디코더가 자신이 수용할 수 있는 스트림인지 확인한다. 이러한 이유로 변환 중에는 전체 스트림에 대한 정보를 모두 알 수 없으므로 TS-to-PS 변환 과정 중에서 시스템 헤더만은 변환이 모두 끝난 후에 다시 수정되어야 한다. 그림 6에 보인 시스템 헤더의 구성은 첫번째 PS 헤더로 제한되어 있지만 실제 시스템 헤더는 여러 번 반복될 수 있다. 그러나 PS가 저장용 스트림이기 때문에 시스템 헤더를 반복하는 것은 특별한 경우를 제외하고는 불필요하다.

PS 헤더는 팩 헤더(pack header)와 시스템 헤더로 나누어 진다. 그림 6에서 pack_start_code, SCR, program_mux_rate를 제외하고는 모두 시스템 헤더이다. 팩 헤더 구성 과정에서는 TS 헤더 분석 모듈로부터 전달받은 PCR 값을 그대로 SCR로 매핑한다. SCR은 PS에서 시스템 클록을 복원할 수 있는 시간 정보이다. 제안된 방식은 이 과정을 통해 PLL을 구현하지 않고서도 시스템 클록의 복원과 ES의 동기가 가능하다. PLL이 차지하는 과정이 생략됨에 따라 변환 알고리듬은 더욱 간결해진다. program_mux_rate는 앞서 언급한 바와 같이 PLL이 없는 PS 시스템 디코더에서 시스템 클록에 관한 정보를 주기 위해서 존재한다. 즉 그러한 디코더는 SCR을 통해 시스템 클록을 복원할 수 있는 능력이 없으므로 program_mux_rate를 이용하여 ES 간의 동기를 맞춘다[13,14]. 제안된 방식에서는 계산 시간을 줄이기 위해 그림 6과 같이 사용될 디코더에 대한 옵션(option)을 추가하였다. 예를 들어 어떠한 장치의 내부에서만 사용할 목적으로 PS로 변환하는 것이고 그 장치가 PLL을 갖고 있다면 program_mux_rate 값은 0이 아닌 값으로도 충분하다. ([1]에서 program_mux_rate는 0이 될 수 없다고 명시되어 있다.) 이 내용은 이후 III장의 실험을 통해 검증되었다.

시스템 헤더는 앞서 언급한 바와 같이 PS 전체에 대한 정보를 포함한다. 따라서 실시간 변환 과정이 끝나기 전에는 모든 필드의 값을 정확히 알 수 없다. 처음 시스템 헤더를 구성할 때 바로 기록할 수 있는 필드는 정확히 기록하고 그렇지 못한 필드는 예측되는 값으로 채워 일단 공간을 만들어 둔다. 변환 과정이 모두 끝나면 그때까지 기록해 둔 전체 스트림 정보로 시스템 헤더의 필드를 갱신한다. 스

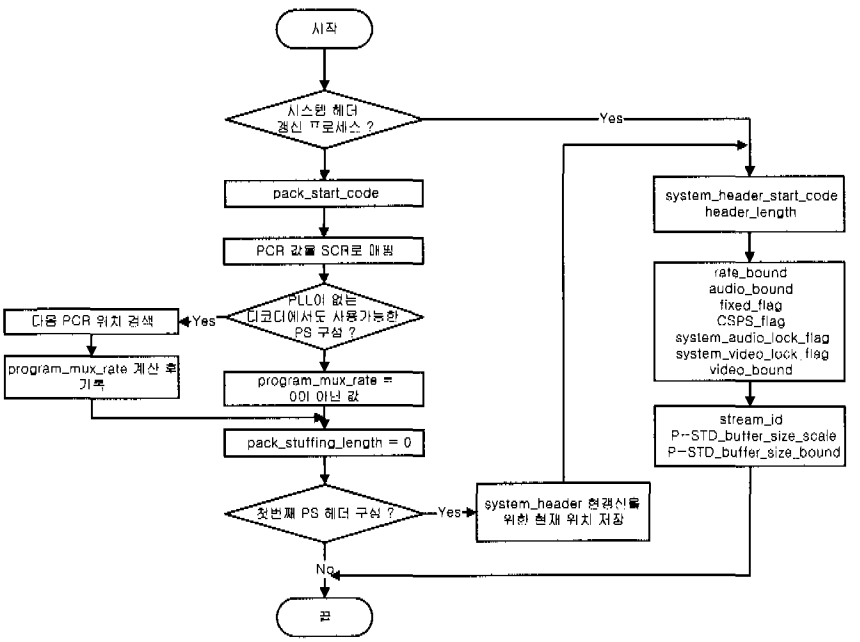


그림 6. PS 헤더 구성 절차

표 1. 시스템 헤더 필드 분류 및 필드 값

*종류	시스템 헤더 필드		필드 값	참고사항
A	audio_bound		오디오 ES의 개수	PSI의 PMT
	video_bound		비디오 ES의 개수	PSI의 PMT
	system_audio_lock_flag		1	
	system_video_lock_flag		1	
	CSPS_flag		0	
	private_rate_restriction_flag		0 또는 1	
	MPEG-1, -2 audio	stream_id	오디오의 stream_id	**0xc0 - 0xdf
		P-STD_buffer_bound_scale	0	
		P-STD_buffer_size_bound	23	
	MPEG-1 video	stream_id	비디오의 stream_id	**0xc0 - 0xdf
		P-STD_buffer_bound_scale	1	
		P-STD_buffer_size_bound	59	
B	MPEG-2 video	stream_id	비디오의 stream_id	**0xc0 - 0xdf
		P-STD_buffer_bound_scale	1	
		P-STD_buffer_size_bound	***(표 2) 참고	
C	rate_bound		비트율 중 최대값	program_mux_rate
	fixed_flag		비트율이 일정할 때 1	program_mux_rate

* A는 시스템 헤더 구성 시 바로 기록할 수 있는 필드를 나타낸다. B는 PSI의 PMT에 비디오 스트림 설명자(descriptor)가 있는 경우 바로 기록할 수 있는 필드를 나타낸다. 그 설명자가 없다면 비디오 PES 또는 ES 헤더를 분석해야 한다. C는 변환 종료 시 갱신해야 하는 시스템 헤더 필드를 나타낸다.

** [4]의 표 2-18 참고한다.

*** 표 2로부터 프로파일과 레벨을 통해 필드 값을 구한다.

트림에 대한 정보를 이미 알고 있는 플랫폼에서는 이러한 갱신 과정이 필요없으므로 그림 6의 갱신 루틴은 옵션 사항이다. 시스템 필드마다 특징을 분류하여 표 1에 제시하였다.

PSI 분석 모듈을 통해 ES의 개수 정보를 얻을 수 있다. 이 정보를 이용하여 audio_bound와 video_bound 필드를 채운다. 이 두 필드는 각각 동시에 재생할 수 있는 오디오와 비디오 ES 개수의 최대값을 의미한다. system_audio_lock_flag와 system_video_lock_flag는 샘플율 고정(sample rate locking) 상태를 지정하는 플래그로 TS의 경우는 이러한 상태의 스트림만을 허용한다[1]. 따라서 TS의 데이터를 변환하였기 때문에 이 두 플래그는 항상 1이다. CSPS_flag는 제한된 시스템 매개변수 PS(constrained system parameter program stream)을 지정하는 플래그이다. 이러한 PS는 일반적인 PS의 부분집합이므로 0으로 설정하여 일반적인 PS로 지정한다. private_rate_restirction_flag는 CSPS_flag가 1일 때만 의미를 가지므로 아무 값이든 상관없다. P-STD_buffer_size_bound는 PS의 STD의 주 버퍼 크기 중 최대값을 의미한다. P-STD_buffer_bound_scale은 가중치 요소(scale factor)로서 0이면 128 바이트 단위를, 1이면 1024 바이트 단위의 버퍼 크기를 의미한다. MPEG-1와 MPEG-2 오디오의 경우는 모두 P-STD_buffer_bound_scale은 0이고 P-STD_buffer_size_bound 값을 23으로 고정되어 있다. 따라서 이 값을 기록하면 된다. stream_id는 PES 패킷의 stream_id 값을 의미하며 [1]의 표 2-18에 정의되어 있다. 시스템 헤더의 stream_id에는 [1]의 표 2-18에서 정의된 값 외에 특별히 정의된 두 값(0xb8, 0xb9)이 있다. 각각 오디오 ES나 비디오 ES가 여러 개 있더라도 모두 한꺼번에 P-STD_buffer_size_bound를 지정하기 위해 사용되는 값이다. MPEG 비디오의 경우에 P-STD_buffer_size_bound는 비디오 ES의 프로파일(profile)과 레벨(level) 값에 의존한다. 그 계산식을 식 (3)

에 제시하였다.

$$B_v = \frac{VBV_{max}[p,l] + \frac{1}{750} \times R_{max}[p,l]}{(8 \times 1024)} \quad (3)$$

여기서, B_v는 비디오 P-STD_buffer_size_bound를 나타내고, p, l은 각각 프로파일과 레벨을 나타낸다. $VBV_{max}[p,l]$과 $R_{max}[p,l]$ 은 각각 [15]의 표 8-14와 8-13에 명시되어 있다. 이 값을 이용하여 계산해 놓은 값을 표 2에 제시하였다.

표 2. 프로파일과 레벨 값에 따른 P-STD_buffer_size_bound 값

	Profile				
Level	Simple	Main	SNR	Spatial	High
High		1,208			1,509
High-1440		906		906	1,208
Main	227	227	227		302
Low		59	59		

P-STD_buffer_bound_scale은 1이다. MPEG-1 비디오의 경우는 MP@LL(main profile at low level)의 값을 사용한다. MPEG-1 비디오와 MPEG-2 비디오의 구별은 PSI 분석 모듈에서 수행한다.

MPEG-2 비디오의 P-STD_buffer_size_bound 값을 찾기 위해서는 비디오 ES의 프로파일과 레벨 값을 알아야 한다. PSI 분석 시 PMT 내에 비디오 스트림의 설명자가 있는 경우 그 설명자의 profile_and_level_indication 필드 값으로부터 알아낼 수 있다. 그러나 설명자는 옵션 사항이기 때문에 PMT 내에 존재하지 않는 경우가 있다. 이 경우 이용할 수 있는 방법이 두 가지 있다. 하나는 PES 패킷 헤더 내에 있는 P-STD_buffer_scale과 P-STD_buffer_size 필드를 이용하는 방법이다. 그러나 이 필드 또한 TS 경우 옵션 사항이기 때문에 이 것 역시 존재하지 않는 경우가 있다. (PS의 경우는 각 ES의 첫번째 PES 패킷에는 이 것이 반드시 존재해야 한다.[11]) 다른 방법은 PES 패킷의 유료부하로 운반되는 비디오 ES 데이터를 분석하는 방법이다. 비디오 ES의 헤더로부터 프로파일과 레벨 값을 얻어와서 표 2로부터 P-STD_buffer_size_bound 값을 얻는 방법이다. 기존의 방식은 ES까지 모두 분리해 내기 때문에 문제가 되지 않는다. 제안된 방식은 가능한 한 PES 패킷을 직접 매핑하여 그 효율을 높이기 위한 것이다. PES로부터 ES를 만든 후 다시 패킷화하여 PES로 만드는 과정은 시스템 부하가 클 뿐 아니라 반드시 필요한 과정이 아니다. 그런데 비디오 ES의 헤더까지 읽어와야 하는 방식은 제안된 방식의 원칙을 위배하는 것으로 보인다. 그러나 처음 초기화 시 비디오 ES마다 한번만 헤더를 읽어오면 되므로 시스템에 큰 부하를 주지 않는다. 또한 영상의 크기가 정해져 있는 플랫폼에서는 이미 P-STD_buffer_size_bound가 정해져 있으므로 프로파일과 레벨을 찾는 과정은 필요없다. 따라서 제안된 방식은 여전히 변환의 효율성을 유지한다.

변환을 마친 후 갱신할 필요가 있는 필드는 rate_bound와 fixed_flag이다. rate_bound는 PS 전체의 program_mux_rate 중 가장 큰 값을 의미한다. fixed_flag는 program_mux_rate의 모든 값이 변하지 않고 동일할 경우 1로 설정된다. 즉 fixed_flag가 1이면 PS가 고정 비트율(constant bitrate; CBR)이라는 것을 의미하며, fixed_flag가 0이면 PS가 가변 비트율(variable bitrate; VBR)임을 나타낸다. 변환을 마친 후 두 필드 중 값이 변경될 필드가 있다면 시스템 헤더가 저장된 위치로 찾아가 새로운 값으로 변경한다.

4. PES 패킷 매핑

본 절에서는 직접 매핑을 사용하는 TS-to-PS 변환 기법 중 PES 패킷의 직접 매핑과 재조합 과정을 검증한다. 매핑 과정에 필요한 ES들의 최대 버퍼 크기 결정에 대해서도 살펴본다.

PES 패킷의 매핑 과정을 그림 7에 보였다. TS 패킷의 유료부하에서 분리된 PES 패킷은 실제로 PES 패킷의 한 조각이다. 따라서 이 조각들을 조합하여 완성된 PES 패킷을 복원하고 PS 파일로 직접 매핑한다. 제안된 알고리듬은 PES 패킷 조각을 재조합하지 않고 PS 파일로 바로 매핑할 수 있다. 채널(channel) 특성에 따라 TS 패킷의 순서가 뒤바뀌어 전송될 수 있다고 하면 PES 패킷의 조각도 뒤섞이게 되어 재조합 과정이 필요하게 된다. 올바른 순서 여부는 TS 헤더 분석 모듈에서 제공한 continuity_counter 값을 통해 확인할 수 있으며 그림 7에 이를 위한 재배열(reordering) 루틴이 존재한다. 이 때 중복된 TS 패킷은 제거한다. 마지막 PES 패킷이 완전히 수신되지 못한 경우는 이 불완전한 PES 패킷을 무시하였다. PES 패킷의 시작은 TS

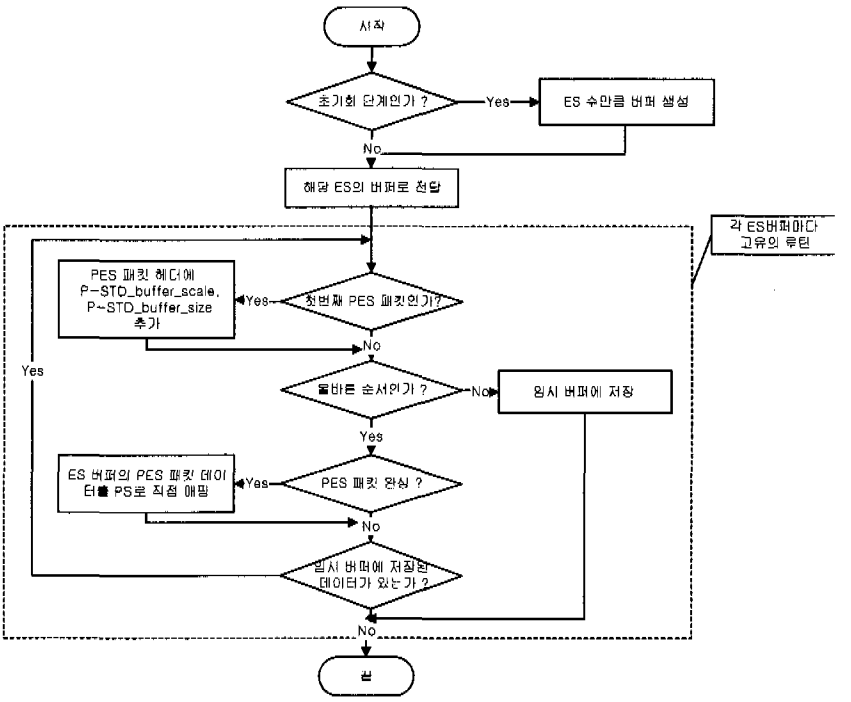


그림 7. PES 패킷 매핑 절차

헤더 분석 모듈에서 제공한 payload_unit_start_indicator가 1인 것으로 판단하며, PES 패킷의 완성은 다음 PES 패킷 조각이 payload_unit_start_indicator가 1일 때 ES 버퍼의 내용을 PS로 매핑함으로써 수행한다. ES들의 PES 패킷 조각을 뒤섞어서 TS 패킷을 만드는 TS 다중화기(multiplexor)가 있다면 이 경우에 대해서도 PES 패킷을 완성시킨 후 매핑하는 알고리듬으로 오류가 발생하지 않고 전달 수 있다. 앞 장에서 PS의 경우는 각 ES의 첫 번째 PES 패킷 헤더에 P-STD_buffer_scale과 P-STD_buffer_size 필드가 반드시 존재해야 한다고 언급한 바 있다. 이를 위해 각 ES의 첫번째 PES 패킷 헤더는 이 필드를 추가해야 하며 그 값은 표 2의 값을 이용한다. 그림 7에 이에 대한 루틴이 있다. PES_packet_length의 값을 증가시킬 필요가 있다면 증가시킨 후, PES 패킷 헤더를 수정하고 이 필드를 추가한 후 ES 버퍼에 저장한다.

PES 패킷을 완성하여 매핑하는 방식은 앞서 살펴본 것처럼 오류 내성의 장점을 가지고 있다. 또한 DSM의 I/O 접근에 드는 시간적 비용이 클 경우 PES 패킷 조각을 자주 매핑하기보다 PES 패킷을 한번에 매핑하는 것이 더 효율적일 것이다[16]. 본 방식은 I/O 특성에 따라 모듈을 최적화함으로써 변환에 걸리는 시간을 크게 줄일 수 있다. PES 패킷을 복원하기 위해서는 각 ES마다 버퍼가 필요하다. 이 버퍼의 크기를 결정하여 오버플로우가 발생하지 않도록 하는 것이 중요하다. 최대 버퍼 크기는 2^{16}(65,556) 바이트이다. 이 값은 PES 패킷 헤더의 PES_packet_length 필드에 기인한다. 따라서 I/O 특성에 따라 버퍼 크기를 2^{16} 바이트의 배수로 잡아서 여러 PES 패킷을 모아서 한꺼번에 처리할 수도 있고 오버플로우는 절대 발생할 수 없다. 단 예외적으로 비디오 ES에 대해서만, PES_packet_length의 값을 0으로 하고 PES 패킷 데이터를 TS 패킷에 넣을 수 있다[1]. 이 경우 해당 ES 버퍼가 가득 차면 일단 직접 매핑을 수행하여 오버플로우를 예방한다.

Ⅲ. 실험 과정 및 결과

실험 과정은 다음과 같다. H/W 인코더[17,18]를 이용하여 제작된 TS를 제안된 알고리듬을 구현한 S/W를 통하여 PS로 변환한 후, H/W디코더[19,20]를 이용하여 변환된 PS를 확인하는 절차를 거쳤다. 직접 매핑을 통한 TS-to-PS 변환 방식은 PES 패킷을 수정하지 않으므로 비디오나 오디오 ES의 내용을 변화시키지 않는다. 따라서 변환 후에도 원본은 그

대로 보존된다.

표 3은 실험된 비트스트림의 명세를 나타내고 있다. 변환 후 비트스트림의 바이트 수가 3-6% 정도 감소하였다. 짧은 TS 패킷마다 존재하였던 헤더들이 PS로 변환되면서 크게 줄어들었기 때문이다. CPU 클록 350MHz의 PC 상에서 실제 상영 시간의 약 60%의 시간으로 변환되므로 실시간 구현 이상의 속도로 변환이 가능함을 보이고 있다. 구현된 I/O 엑세스(input/output access) 루틴은 MSSG (MPEG software simulation group)의 MPEG-2 비디오 코덱(codec) 소스의 getbits()와 putbits() 함수[21]를 통해 구현되었다. 이 루틴은 1바이트에서 4바이트 단위의 I/O 엑세스를 수행하므로 직접 매핑의 고속 I/O 루틴에는 적합하지 않다. 따라서 저장 미디어 특성에 따라 I/O 루틴을 최적화하면 더욱 빠른 속도의 변환이 가능하다[16]. 또한 방송 혹은 네트웍에서 DSM으로의 저장은 메모리에서 DSM으로의 저장이 대부분이므로 입력 측의 부하, 즉 실험에서 하드디스크 상의 TS를 읽어오는 데에 소요되는 접근 시간이 없어져서 더욱 빠른 매핑을 수행할 수 있다. 실험에서 사용한 디코더는 모두 PLL을 가지고 있고 두 디코더 모두 program_mux_rate 값과 무관하게 작동한다는 것을 확인하였다. 따라서 II장 3절에서 제시한 디코더에 대한 옵션은 필요에 따라 유용하게 활용될 수 있다. 또한, PES 패킷 매핑 모듈 내에서 불완전한 마지막 PES 패킷은 저장하지 않으므로 임의의 위치에서 종료된 TS를 변환하여 얻은 PS의 경우에도 올바르게 재생됨을 확인할 수 있었다.

Ⅳ. 결 론

본 논문에서는 직접 매핑 기법을 통한 MPEG-2 TS-to-PS 변환 알고리듬을 제안하였다. 본 논문에서 소개한 직접 매핑 방식은 기존의 역다중화와 재다중화 과정을 거치는 방식과 같이 시스템 클록과 ES의 동기를 그대로 유지하면서 변환을 수행한다. 기존의 방식과는 달리 PLL을 필요로 하지 않는다. PES 패킷을 분해하지 않고 그대로 매핑하므로 변환 시간을 단축할 뿐만 아니라 구현까지 단순화시킨다. 본 논문에서 제안한 방식은 기존의 방식처럼 역다중화기 기능에 부가적인 역할을 하기보다는 별도의 모듈로서 독립적인 기능을 수행할 수 있다. 디지털 TV와 DSM의 사용이 활성화되면 MPEG-2 TS와

표 3. 제안된 방식에 의한 TS-to-PS 변환 실험 결과

실험 스트림	시간 (초)	규격	비트율	변환 전 바이트 (패킷수)	변환 후 바이트 (패킷수)	변환시간 (초)	변환 시간 비율	압축률
A	5 (5)	704x480x29.976 (NTSC)	732 KBytes/s	4,346,910 (168)	4,441,124 (23,623)	3 (3)	60%	97%
		MPEG-1 Layer II, stereo, 32000	256 KBits/s					
B	18 (18)	704x480x29.976 (NTSC)	732 KBytes/s	14,907,648 (79,296)	14,593,014 (558)	11 (11)	61%	97%
		MPEG-1 Layer II, stereo, 32000	256 KBits/s					
C	1:06 (66)	704x480x29.976 (NTSC)	936 KBytes/s	55,278,956 (294,037)	52,036,614 (23,884)	38 (38)	58%	94%
		MPEG-1 Layer II, stereo, 32000	24 KBits/s					
D	1:37 (98)	704x480x29.976 (NTSC)	936 KBytes/s	94,290,084 (501,552)	88,813,966 (40,740)	59 (59)	60%	94%
		MPEG-1 Layer II, stereo, 32000	24 KBits/s					
E	4:04 (244)	704x480x29.976 (NTSC)	732 KBytes/s	196,904,432 (1,047,364)	192,756,730 (7,344)	2:40 (160)	65%	97%
		MPEG-1 Layer II, stereo, 32000	256 KBits/s					

PS의 방식이 서로 다름으로 인해 이러한 전문 모듈을 필요로 하는 제품의 수가 급증할 것이다. 본 논문은 이러한 제품의 개발 기간과 비용 절감에 도움이 될 것이다.

MPEG-2 시스템 표준에는 여러 가지 서비스에 공통되는 많은 요구를 수용하기 위해 매우 다양한 기능이 포함되어 있다. 또한 서비스에 따라 달리 정의될 수 있는 부분은 표준에서 규정하지 않고 확장 필드를 사용하여 수용한다. 따라서 실제 서비스에 적용될 때는 표준을 확장 또는 제한하는 것이 대부분이다. 현재 지상파 혹은 위성방송 등으로 서비스 중인 미국의 디지털TV 표준인 ATSC나 유럽의 DVB 규격은 MPEG-2를 기반으로 하고 있으나 확장 부분과 제한 사항이 서로 다르다. 특히 PSI의 정보 중 사용자 정의 테이블(user private table)을 여러 가지 정의하여 사용 목적에 맞게 사용하고 있다[4,5]. 따라서 이후 연구는 이러한 규격과 한국의 디지털 방송 규격에 따라 변환 알고리듬을 수정 또는 보완하여 서비스 고유의 변환 프로그램을 생성할 필요가 있다. MPEG-2 TS-to-PS 변환과 함께 그 역변환인 PS-to-TS 변환도 필요하다. 예를 들어 디지털TV 내의 DSM 장치에 수신된 방송 프로그램을 PS로 녹화하였다가 그 내용을 다시 재생할 때, TS 디코더만 가지고 있는 디지털TV는 이러한 PS를 재생할 수 없다. 따라서 이러한 경우 그 역변환을 수행하여야 하며 이에 관한 연구도 필요하다.

참 고 문 헌

[1] ITU-T Rec. H.222.0 | ISO/IEC 13818-1, Information technology Generic coding of moving pictures and associated audio information: Systems, 1995.

[2] P. A. Sarginson, "MPEG-2: Overview of the system layer", BBC R&D Report, RD1996/2, 1996.

[3] ISO/IEC 11172-1, Information technology Coding of moving pictures and associated audio for digital storage media at up to about 1,5Mbit/s Part 1: Systems, 1993.

[4] http://www.atsc.org

[5] http://www.dvd.org

[6] http://www.dvdforum.org

[7] 전자부품연구원, *멀티미디어/DTV용 시스템 제어 및 Data Storage Interface 표준 규격 개발*, KETI-RD-1999011, 1999.3.

[8] R. E. Best, *Phase-Locked Loop 3/e*, McGraw-Hill, 1997.

[9] C. Tryfonas, A. Varma, "A Resampling Approach to Clock Recovery in MPEG-2 Systems Layer", Dep. Comp. Eng., UCSC, Tech. Rep. UCSC-CRL-98-4, 1998.

[10] C. Tryfonas, A. Varma, "Timestamping Schemes for MPEG-2 Systems Layer and Their Effect on Receiver Clock Recovery", Dep. Comp. Eng., UCSC, Tech. Rep. UCSC-CRL-98-2, 1998.

[11] 정제창, *최신 MPEG*, 교보문고, 서울, 275쪽, 1997.

[12] 유시룡, 장규환, 이병욱, 김종일, 정해묵, *MPEG 시스템*, 대영사, 서울, 48-60쪽, 1997.

[13] 전자부품연구원, *다매체 DTV용 Bit-stream 변환 S/W 기술 개발에 관한 연구*, pp. 29-39, KETI-RD-1999008, 1999.2.

[14] B. G. Haskell, A. Puri, A. N. Netravali, *Digital Video: An Introduction to MPEG-2*, International Thomson Publishing, pp.46-49, 1996.

[15] ITU-T Rec. H.262 | ISO/IEC 13818-2, Information technology Generic coding of moving pictures and associated audio information: Video, 1995.

[16] H. M. Deitel, *Operating Systems 2/e*, Addison Wesley, pp.185-284, 285-311, 359-385, 1990.

[17] http://www.vela.com/oem/products/aff.htm

[18] http://www.proh.com/techspec.html

[19] http://www.optibase.com/products/videoplex_plus.htm

[20] http://www.sigmadesigns.com/product_hw+.htm

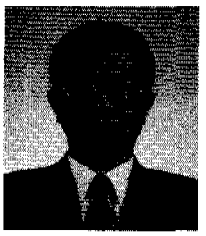
[21] http://www.mpeg.org/MPEG/MSSG

신 화 선(Hwa Seon Shin)

1998년 2월 : 서울시립대학교 제어계측공학과 졸업 (공학사)
1999년 11월 : 서울시립대학교 전자공학과 재학 (공학석사 과정)

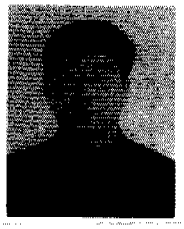
<주관심 분야> 영상 압축 부호화, 멀티미디어 통신

김 용 한(Yong Han Kim) 정회원



1982년 2월: 서울대학교 제어계측공학과 졸업 (공학사)
1984년 2월: 서울대학교 대학원 제어계측공학과 졸업 (공학석사)
1990년 12월: Rensselaer Polytechnic Institute 졸업 (공학박사)
1984년 3월~1996년 3월: 한국전자통신연구원
1991년 10월~1992년 9월: 일본 NTT 휴먼인터페이스연구소 객원연구원
1996년 3월~현재: 서울시립대학교 전자전기공학부 조교수
<주관심 분야> 영상 압축 부호화, 멀티미디어 통신

김 제 우(Je Woo Kim)

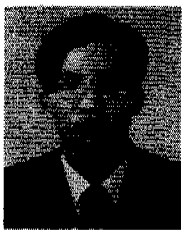


1997년 2월: 서울시립대학교 제어계측공학과 졸업 (공학사)
1999년 2월: 서울시립대학교 제어계측공학과 졸업 (공학석사)
1997.09 - 현재 전자부품연구원 연구원
<주관심 분야> 시스템 관련 알고리듬

최 병 호(Byeongho Choi)

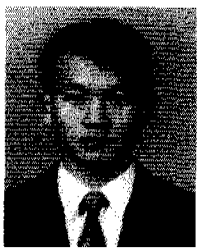
1991년 2월: 한양대학교 전자공학과 졸업 (공학사)
1993년 8월: 한양대학교 전자공학과 졸업 (공학석사)
1993년 7월~1997년 8월: LG전자 Video 연구소
1997년 9월~현재: 전자부품연구원 선임연구원
<주관심 분야> 유무선 통신 분야, 네트웍, 암호화 및 보안, RF Modulation, 마이콤 Programming, DSP 설계

송 병 철(Byoungchol Song)



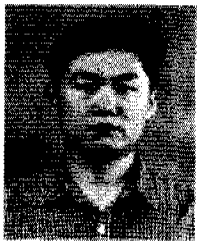
1994년 2월: 명지대학교 전자공학과 졸업(공학사)
1996년 2월: 명지대학교 전자공학과 졸업(공학석사)
1996년 3월~현재: 전자부품연구원 전임연구원
<주 관심 분야> 데이터 통신 시스템 및 시스템 IC 연구 및 프로토콜 연구

용 석 진(Sukjin Yong)



1994년 2월: 한양대학교 전자공학과 졸업(공학사)
1996년 2월: 한양대학교 전자공학과 졸업(공학석사)
1996년 2월~현재: 전자부품연구원 전임연구원
<주관심 분야> Network 관련 device driver 개발, Network programming, Network system 설계

정 광 모(Kwangmo Jung)



1990년 2월: 광운대학교 전자공학과 졸업 (공학사)
1990년~1994년: LG 정보통신 연구소
1994년~현재: 전자부품연구원 선임연구원

<주관심 분야> 광대역 무선통신 시스템(IMT-2000, WLL, 무선 LAN, 무선 ATM), ATM switch, 네트워크 시스템 (WAN 억세스 스위치, IP 스위치, Switched 라우터), ADSL 시스템 (DSLAM, ADSL 모뎀/라우터)

동 용 배(Yong Bae Dong)

1978년 2월: 고려대학교 전자공학과 졸업 (공학사)
1981년 2월: 한국과학기술원 전자공학과 졸업 (공학석사)
1991년: 웨스턴 오스트레일리아 대학교 컴퓨터공학과 졸업 (공학박사)
1981년~1987년 : 금오공과대학 전자공학과 조교수
1992년~현재: 전자부품연구원 수석연구원
<주관심 분야> 시스템 IC, 디지털 TV, 멀티미디어