(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2012년07월06일
(11) 등록번호 10-1154721
(24) 등록일자 2012년06월01일

(51) 국제특허분류(Int. Cl.)
H04N 5/93 (2006.01) *H04N 5/92* (2006.01)
H04N 7/32 (2006.01) *G11B 20/10* (2006.01)
(21) 출원번호 10-2006-7025927
(22) 출원일자(국제) 2005년06월03일
심사청구일자 2010년05월31일
(85) 번역문제출일자 2006년12월08일
(65) 공개번호 10-2007-0032677
(43) 공개일자 2007년03월22일
(86) 국제출원번호 PCT/JP2005/010627
(87) 국제공개번호 WO 2005/122566
국제공개일자 2005년12월22일
(30) 우선권주장
JP-P-2004-00174570 2004년06월11일 일본(JP)
(56) 선행기술조사문헌
JP2000312362 A
전체 청구항 수 : 총 4 항

(73) 특허권자
소니 컴퓨터 엔터테인먼트 인코포레이티드
일본국 도쿄도 미나토쿠 코난 1쵸메 7반 1고
소니 주식회사
일본국 도쿄도 미나토쿠 코난 1-7-1
(72) 발명자
후지나미, 야스시
일본 141-0001 도쿄도 시나가와꾸 기따시나가와 6쪼메 7-35 소니가부시끼 가이샤 내
다까하시, 구니아끼
일본 107-0062 도쿄도 미나또꾸 미나미 아오야마 2-6-21가부시키가이샤 소니 컴퓨터 엔터테인먼트 내
(74) 대리인
구영창, 이중희, 장수길

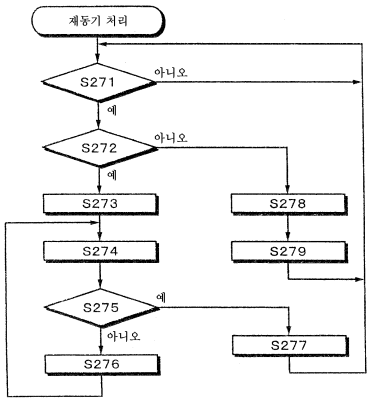
심사관 : 이병우

(54) 발명의 명칭 **데이터 처리 장치, 데이터 처리 방법 및 기록 매체**

(57) 요 약

비디오 데이터와, 그것과 동기해서 출력되어야 할 출력 데이터의 동기를 취한다. 비디오 데이터의 출력이, 그것과 동기해서 출력되어야 할 출력 데이터의 출력보다도 지연되고 있는 경우, 스텝 S273에서, 1개의 액세스 유닛의 처리를 스킵할 것이 지시된다. 또한, 스텝 S275에서, 처리의 스킵이 지시된 액세스 유닛의 비디오 데이터에 대한 au_ref_flag에 기초하여, 그 비디오 데이터가, 다른 비디오 데이터의 디코드에 참조되는지 여부가 판정되어, 처리의 스킵이 지시된 액세스 유닛의 비디오 데이터가, 다른 비디오 데이터의 디코드에 참조되지 않은 경우, 스텝 S277에서, 처리의 스킵이 지시된 액세스 유닛의 처리가 스킵된다. 본 발명은, 예를 들면, DVD를 이용한 게임 장치 등에 적용할 수 있다.

대 표 도 - 도39



특허청구의 범위

청구항 1

부호화 데이터를 처리하는 데이터 처리 장치로서,

상기 부호화 데이터는,

소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와,

상기 비디오 데이터와 동기해서 출력될 출력 데이터와,

상기 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 상기 부호화 비디오 데이터의 디코드에 이용되는 이용 정보

를 포함하고,

상기 이용 정보는, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 상기 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보를 포함하고,

상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다도 지연되고 있는지의 여부를 판정하는 제1 판정 수단과,

상기 제1 판정 수단에서, 상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 수단과,

상기 지시 수단에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 상기 참조 정보에 기초하여, 상기 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 수단과,

상기 제2 판정 수단에서, 상기 지시 수단에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 상기 지시 수단에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 수단

을 포함하는 것을 특징으로 하는 데이터 처리 장치.

청구항 2

제1항에 있어서,

상기 제1 판정 수단에서, 상기 출력 데이터의 출력이, 상기 비디오 데이터의 출력보다도 지연되고 있다고 판정된 경우에, 상기 비디오 데이터를 반복해 출력시키는 출력 제어 수단을 더 구비하는 것을 특징으로 하는 데이터 처리 장치.

청구항 3

부호화 데이터를 처리하는 데이터 처리 방법으로서,

상기 부호화 데이터는,

소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와,

상기 비디오 데이터와 동기해서 출력될 출력 데이터와,

상기 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 상기 부호화 비디오 데이터의 디코드에 이용되는 이용 정보

를 포함하고,

상기 이용 정보는, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 상기 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부

호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보를 포함하고,

상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다도 지연되고 있는지의 여부를 판정하는 제1 판정 스텝과,

상기 제1 판정 스텝에서, 상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 스텝과,

상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 상기 참조 정보에 기초하여, 상기 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 스텝과,

상기 제2 판정 스텝에서, 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 스텝

을 포함하는 것을 특징으로 하는 데이터 처리 방법.

청구항 4

삭제

청구항 5

부호화 데이터를 처리하는 데이터 처리를, 컴퓨터에 행하게 하는 프로그램이 기록되어 있는 프로그램 기록 매체로서,

상기 부호화 데이터는,

소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와,

상기 비디오 데이터와 동기해서 출력될 출력 데이터와,

상기 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 상기 부호화 비디오 데이터의 디코드에 이용되는 이용 정보

를 포함하고,

상기 이용 정보는, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 상기 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보를 포함하고,

상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다도 지연되고 있는지의 여부를 판정하는 제1 판정 스텝과,

상기 제1 판정 스텝에서, 상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 스텝과,

상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 상기 참조 정보에 기초하여, 상기 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 스텝과,

상기 제2 판정 스텝에서, 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 스텝

을 포함하는 것을 특징으로 하는 프로그램이 기록되어 있는 프로그램 기록 매체.

청구항 6

삭제

청구항 7

삭제

명 세 서

기 술 분 야

[0001] 본 발명은, 데이터 처리 장치 및 데이터 처리 방법, 프로그램 및 프로그램 기록 매체, 데이터 기록 매체, 및, 데이터 구조에 관한 것으로, 특히, 예를 들면, 편리성 등이 높은 데이터 처리를 가능하게 하는 데이터 처리 장치 및 데이터 처리 방법, 프로그램 및 프로그램 기록 매체, 데이터 기록 매체, 및 데이터 구조에 관한 것이다.

배 경 기 술

[0002] 최근, 대용량이며, 랜덤 액세스가 가능한 기록 미디어로서, 예를 들면, DVD(Digital Versatile Disc)가 보급되고, 또한, DVD를 이용하여 각종 처리를 행하는 DVD 장치도 널리 보급되고 있다.

[0003] DVD 장치로서는, 예를 들면, DVD에 대하여, 텔레비전 방송 프로그램의 데이터 등의 기록 재생을 행하는 DVD 레코더나, DVD에 지도 정보 등을 기록하고, 그 지도 정보의 표시를 행하는 카 네비게이션 시스템, DVD에 게임의 프로그램 등을 기록하고, 그 프로그램을 실행하는 게임 장치 등이 있다.

[0004] 또한, DVD에 대해서는, 예를 들면, 비특허문헌1 「DVD Specifications for Read-Only Disc Part3 : Version 1.1 Dececmber 1997」에, 그 상세가 기재되어 있다.

[0005] DVD와 같이, 대량의 데이터를 기록할 수 있는 기록 미디어나, 그것을 이용하여 각종 처리를 행하는 DVD 장치 등에는, 그러한 대량의 데이터에 대해서, 편리성 등이 높은 데이터 처리를 행하는 것이 요청된다.

[0006] <발명의 개시>

[0007] 본 발명은, 이러한 상황을 감안하여 이루어진 것으로서, 편리성 등이 높은 데이터 처리를 행하는 것이 가능하도록 한 것이다.

[0008] 본 발명의 데이터 처리 장치는, 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 부호화 비디오 데이터의 디코드에 이용되는 이용 정보가, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보를 포함하는 경우에, 비디오 데이터의 출력이, 그것과 동기하여 출력되어야할 출력 데이터의 출력보다 지연되고 있는지의 여부를 판정하는 제1 판정 수단과, 제1 판정 수단에서, 비디오 데이터의 출력이, 출력 데이터의 출력보다도 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 수단과, 지시 수단에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 참조 정보에 기초하여, 상기 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 수단과, 제2 판정 수단에서, 지시 수단에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 지시 수단에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 수단을 포함하는 것을 특징으로 한다.

[0009] 본 발명의 데이터 처리 방법은, 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 부호화 비디오 데이터의 디코드에 이용되는 이용 정보가, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보를 포함하는 경우에, 비디오 데이터의 출력이, 그것과 동기해서 출력되어야 할 출력

데이터의 출력보다도 지연되고 있는지의 여부를 판정하는 제1 판정 스텝과, 제1 판정 스텝에서, 비디오 데이터의 출력이, 출력 데이터의 출력보다도 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 스텝과, 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 참조 정보에 기초하여, 상기 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 스텝과, 제2 판정 스텝에서, 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 스텝을 포함하는 것을 특징으로 한다.

[0010] 본 발명의 프로그램은, 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 부호화 비디오 데이터의 디코드에 이용되는 이용 정보가, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보를 포함하는 경우에, 비디오 데이터의 출력이, 그것과 동기하여 출력되어야 할 출력 데이터의 출력보다도 지연되고 있는지의 여부를 판정하는 제1 판정 스텝과, 제1 판정 스텝에서, 비디오 데이터의 출력이, 출력 데이터의 출력보다 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 스텝과, 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 참조 정보에 기초하여, 상기 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 스텝과, 제2 판정 스텝에서, 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 스텝을 포함하는 것을 특징으로 한다.

[0011] 본 발명의 프로그램 기록 매체에 기록되어 있는 프로그램은, 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 부호화 비디오 데이터의 디코드에 이용되는 이용 정보가, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보를 포함하는 경우에, 비디오 데이터의 출력이, 그것과 동기하여 출력되어야 할 출력 데이터의 출력보다도 지연되고 있는지의 여부를 판정하는 제1 판정 스텝과, 제1 판정 스텝에서, 비디오 데이터의 출력이, 출력 데이터의 출력보다 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 스텝과, 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 참조 정보에 기초하여, 상기 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 스텝과, 제2 판정 스텝에서, 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 스텝을 포함하는 것을 특징으로 한다.

[0012] 본 발명의 데이터 기록 매체는, 부호화 데이터는, 소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와, 비디오 데이터와 동기해서 출력될 출력 데이터와, 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 부호화 비디오 데이터의 디코드에 이용되는 이용 정보를 포함하고, 이용 정보는, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보를 포함하는 것을 특징으로 한다.

[0013] 본 발명의 데이터 구조는, 부호화 데이터는, 소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와, 비디오 데이터와 동기해서 출력될 출력 데이터와, 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 부호화 비디오 데이터의 디

코드에 이용되는 이용 정보를 포함하고, 이용 정보는, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보를 포함하는 것을 특징으로 한다.

[0014] 본 발명의 데이터 처리 장치 및 데이터 처리 방법, 및 프로그램 및 프로그램 기록 매체에 기록되어 있는 프로그램에서는, 액세스 유닛 단위의 부호화 비디오 데이터에 있어서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 부호화 비디오 데이터의 디코드에 이용되는 이용 정보에, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지 여부를 나타내는 참조 정보가 포함되는 경우에, 비디오 데이터의 출력이, 그것과 동기해서 출력되어야 할 출력 데이터의 출력보다도 지연되고 있는지의 여부가 판정되어, 비디오 데이터의 출력이, 출력 데이터의 출력보다도 지연되고 있다고 판정된 경우, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵할 것이 지시된다. 또한, 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 참조 정보에 기초하여, 상기 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부가 판정되어, 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우, 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리가 스킵된다.

[0015] 본 발명의 데이터 기록 매체 및 데이터 구조에서는, 부호화 데이터는, 소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와, 비디오 데이터와 동기해서 출력되어야 할 출력 데이터와, 액세스 유닛 단위의 부호화 비디오 데이터에 있어서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 부호화 비디오 데이터의 디코드에 이용되는 이용 정보를 포함하고, 이용 정보는, 해당 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 해당 부호화 비디오 데이터 이외의 다른 부호화 비디오 데이터의 디코드에 참조되는지 여부를 나타내는 참조 정보를 포함하고 있다.

[0016] 본 발명에 따르면, 편리성 등이 높은 데이터 처리가 가능하게 된다. 특히, 비디오 데이터와, 그것과 동기해서 출력되어야 할 출력 데이터의 동기를 용이하게 취하는 것이 가능하게 된다.

도면의 간단한 설명

[0017] 도 1은 본 발명을 적용한 디스크 장치의 제1 실시예의 하드웨어 구성예를 도시하는 블록도.

[0018] 도 2A 및 도 2B는 CPU(112)가 실행하는 소프트웨어 모듈 군의 구성예를 도시하는 블록도.

[0019] 도 3은 버퍼 제어 모듈(215)의 구성예를 도시하는 블록도.

[0020] 도 4는 디스크(101)에서의 디렉토리 구성예를 도시하는 도면.

[0021] 도 5는 "PLAYLIST.DAT" 파일의 신택스를 도시하는 도면.

[0022] 도 6은 PlayItem()의 신택스를 도시하는 도면.

[0023] 도 7은 PlayListMark()의 신택스를 도시하는 도면.

[0024] 도 8은 mark_type의 값과, Mark()의 타입과의 관계를 도시하는 도면.

[0025] 도 9는 PlayList(),PlayItem(), 클립, 및 클립 스트림 파일에 저장된 프로그램 스트림의 관계를 도시하는 도면.

[0026] 도 10은 클립 정보 파일 Clip()의 신택스를 도시하는 도면.

[0027] 도 11은 엘리멘터리 스트림을 식별하는 stream_id 및 private_stream_id와, 엘리멘터리 스트림과의 관계를 도시하는 도면.

[0028] 도 12는 StaticInfo()의 신택스를 도시하는 도면.

[0029] 도 13은 DynamicInfo()의 신택스를 도시하는 도면.

[0030] 도 14는, EP_map()의 신택스를 도시하는 도면.

[0031] 도 15A 및 도 15B는 MPEG-2 System의 프로그램 스트림, 프로그램 스트림 팩, 및 프로그램 스트림 팩의 신택스를 도시하는 도면.

[0032] 도 16A 및 도 16B는 MPEG-2 System의 PES 패킷의 신택스를 도시하는 도면.

[0033] 도 17A, 도 17B 및 도 17C는 MPEG-2 System의 PES 패킷의 신택스를 도시하는 도면.

[0034] 도 18A 및 도 18B는 MPEG-2 System의 PES 패킷의 신택스를 도시하는 도면.

[0035] 도 19A 및 도 19B는 MPEG-2 System에서의 PES_packet()의 stream_id에 기술되는 값과, 엘리멘터리 스트림의 속성(종류)과의 관계를 도시하는 도면.

[0036] 도 20은 디스크 장치가 채용하는 stream_id를 도시하는 도면.

[0037] 도 21은 private_stream1_PES_payload()의 신택스를 도시하는 도면.

[0038] 도 22는 private_stream_id의 값과, private_payload()에 저장되는 엘리멘터리 스트림의 속성과의 관계를 도시하는 도면.

[0039] 도 23은 private_stream2_PES_payload()의 신택스를 도시하는 도면.

[0040] 도 24는 au_information()의 신택스를 도시하는 도면.

[0041] 도 25는 "PLAYLIST.DAT" 파일의 구체예를 도시하는 도면.

[0042] 도 26A 및 도 26B는 클립 정보 파일 "00001.CLP", "00002.CLP", "00003.CLP"의 구체예를 도시하는 도면.

[0043] 도 27은 클립 정보 파일 "00001.CLP" 중 EP_map()의 구체예를 도시하는 도면.

[0044] 도 28은 PlayList #0과 PlayList #1 중 PlayListMark()의 구체예를 도시하는 도면.

[0045] 도 29는 재생전 처리를 설명하는 플로우차트.

[0046] 도 30은 재생 처리를 설명하는 플로우차트.

[0047] 도 31은 PlayItem 갈아타기 처리를 설명하는 플로우차트.

[0048] 도 32는 타임 코드 표시 처리를 설명하는 플로우차트.

[0049] 도 33은 스트림 절환 처리를 설명하는 플로우차트.

[0050] 도 34는 버퍼 제어 모듈(215)의 처리를 설명하는 플로우차트.

[0051] 도 35는 버퍼 제어 모듈(215)의 처리를 설명하는 플로우차트.

[0052] 도 36은 비디오 스트림의 읽어내기의 처리를 설명하는 플로우차트.

[0053] 도 37은 오디오 스트림의 읽어내기의 처리를 설명하는 플로우차트.

[0054] 도 38은 자막 스트림의 읽어내기의 처리를 설명하는 플로우차트.

[0055] 도 39는 재동기 처리를 설명하는 플로우차트.

[0056] 도 40은 마크 처리를 설명하는 플로우차트.

[0057] 도 41은 출력 속성의 제어 처리를 설명하는 플로우차트.

[0058] 도 42는 클립 정보 파일 "00003.CLP"에 기술되어 있는 pts_change_point와 DynamicInfo()와의 세트의 구체예를 도시하는 도면.

[0059] 도 43은 자막 표시 제어 처리를 설명하는 플로우차트.

[0060] 도 44는 캡쳐 제어 처리와 백그라운드/스크린 세이버 처리를 설명하는 플로우차트.

[0061] 도 45는 private_stream2_PES_payload()의 다른 신택스를 도시하는 도면.

[0062] 도 46은 au_information()의 다른 신택스를 도시하는 도면.

[0063] <부호의 설명>

[0064] 101:디스크

[0065] 102:디스크 드라이버

[0066] 111:버스

[0067] 112:CPU

[0068] 113:메모리

[0069] 114:드라이브 인터페이스

[0070] 115:입력 인터페이스

[0071] 116:비디오 디코더

[0072] 117:오디오 디코더

[0073] 118:비디오 출력 인터페이스

[0074] 119:오디오 출력 인터페이스

[0075] 120:비디오 출력 단자

[0076] 121:오디오 출력 단자

[0077] 201:오퍼레이팅 시스템

[0078] 210:비디오 콘텐츠 재생 프로그램

[0079] 211:스크립트 제어 모듈

[0080] 212:플레이어 제어 모듈

[0081] 213:콘텐츠 데이터 공급 모듈

[0082] 214:디코드 제어 모듈

[0083] 214A:계시부

[0084] 215:버퍼 제어 모듈

[0085] 215A:버퍼

[0086] 216:비디오 디코더 제어 모듈

[0087] 217:오디오 디코더 제어 모듈

[0088] 218:자막 디코더 제어 모듈

[0089] 219:그래픽스 처리 모듈

[0090] 220:비디오 출력 모듈

[0091] 220A:FIFO

[0092] 221:오디오 출력 모듈

[0093] 221A:FIFO

[0094] 231:데이터 선두 포인터 기억부

[0095] 232:데이터 써내기 포인터 기억부

[0096] 233:비디오 읽어내기 기능부

[0097] 234:오디오 읽어내기 기능부

[0098] 235:자막 읽어내기 기능부

[0099] 241:비디오 읽어내기 포인터 기억부

[0100] 242:stream_id 레지스터

[0101] 243:au_information() 레지스터

[0102] 251:오디오 읽어내기 포인터 기억부

[0103] 252:stream_id 레지스터

[0104] 253:private_stream_id 레지스터

[0105] 261:자막 읽어내기 기능 플래그 기억부

[0106] 262:자막 읽어내기 포인터 기억부

[0107] 263:stream_id 레지스터

[0108] 264:private_stream_id 레지스터

[0109] S101:디스크 체크

[0110] S102:에러 처리

[0111] S103:SCRIPT.DAT와 PLAYLIST.DAT의 읽어들이기

[0112] S104:파일 공급

[0113] S105:PLAYLIST.DAT의 해석

[0114] S106:클립 정보 파일의 읽어내기

[0115] S107:파일 체크

[0116] S108:SCRIPT.DAT의 해석, 실행

[0117] S121:IN_time 인식

[0118] S122:재생 개시 위치 조사

[0119] S123:타임 코드 표시

[0120] S124:PlaylistMark()의 해석

[0121] S125:재생할 스트림의 결정

[0122] S126:출력 속성 제어

[0123] S127:재생 개시의 준비

[0124] S128:데이터 읽어들이기 개시

[0125] S129:디코더 제어 개시

[0126] S130:디코드 개시

[0127] S131:그래픽스 처리 개시

[0128] S132:출력 처리 개시

[0129] S151:디코드 중단 제어

[0130] S152:플레이 아이템 재생 개시

[0131] S171:메시지 출력

[0132] S172:타임 코드 환산

[0133] S173:타임 코드 출력

[0134] S191:스트림 수 체크

[0135] S192:재생하고 있는 스트림의 특정

[0136] S193:다음에 재생할 스트림의 인식

[0137] S194:다음에 재생할 스트림의 읽어내기 지시

[0138] S211:private_stream_2 패킷의 탐색

[0139] S212:지정된 stream_id인가?

[0140] S213:au_information()을 내부 레지스터에 읽어들이기

[0141] S214:비디오 읽어내기 포인터에 직후의 비디오 패킷의 선두를 삽입한다

[0142] S215:데이터 요구가 있는가?

[0143] S216:프로그램 스트림의 구문 해석을 행하면서, Au_length분의 비디오 데이터를 출력

[0144] S217:number-of-access-unit만 처리하였는가?

[0145] S230:private_stream_1인가?

[0146] S231:동기 코드 탐색

[0147] S232:MPEG 오디오 패킷의 탐색

[0148] S233:오디오 읽어내기 포인터 갱신

[0149] S234:private_stream_1 패킷의 탐색

[0150] S235:지정된 private_stream_id인가?

[0151] S236:오디오 읽어내기 포인터(234)←Au_locator의 직후의 위치+Au_locator

[0152] S237:데이터 요구가 있는가?

[0153] S238:프로그램 스트림의 구문 해석을 행하면서, 고정 길이의 오디오 액세스 유닛을 출력

[0154] S251:자막 읽어내기 기능 플래그인가?

[0155] S252:private_stream_1 패킷의 탐색

[0156] S253:지정된 private_stream_id인가?

[0157] S254:자막 읽어내기 포인터←Au_locator의 직후의 위치+Au_locator

[0158] S255:데이터 요구가 있는가?

[0159] S256:프로그램 스트림의 구문 해석을 행하면서, 자막 액세스 유닛의 선두에 기술을 하고 있는 길이분의 자막 액세스 유닛을 출력

[0160] S271:타임 스탬프 어긋남이 큰가?

[0161] S272:비디오가 지연되어 있는가?

[0162] S273:비디오 액세스 유닛의 스킵을 지시

[0163] S274:액세스 유닛의 au_ref_flag를 검사

[0164] S275:액세스 유닛은, 비참조 화상인가?

[0165] S276:비디오 액세스 유닛을 처리

[0166] S277:비디오 액세스 유닛의 처리를 스킵

[0167] S278:비디오의 반복 출력을 지시

[0168] S279:비디오의 반복 출력

[0169] S301:현재 시각이, mark_time_stamp에 일치하는가?

[0170] S302:처리 대상으로 하는 mark를 인식

[0171] S303:처리 대상 mark에, entry_ES_stream_id/entry_ES_private_stream_id가 기술되어 있는가?

[0172] S304:재생 중인 스트림에, entry_ES_stream_id/entry_ES_private_stream_id의 스트림이 포함되는가?

[0173] S305:처리 대상 mark의 mark_type을 판정

[0174] S306:챕터/인덱스의 갱신 처리

[0175] S307:이벤트 메시지와 mark_data를, 스크립트 제어 모듈(211)에 통지

[0176] S308:mark_data에 대응하는 처리

[0177] S320:pts_change_point를 건내준다

[0178] S321:현재 시각이, pts_change_point에 일치하는가?

[0179] S322:현재 시각에 일치하는 pts_change_point의 DynamicInfo()를 인식

[0180] S323:DynamicInfo()에 기술된 출력 속성을 공급

[0181] S324:출력 속성에 따라서, 비디오/오디오의 출력의 제어를 개시

[0182] S341:자막의 처리의 초기화를 지시

[0183] S342:자막의 표시에 대하여, 새로운 표시 방식 지시가 있는가?

[0184] S343:자막 스트림을 재생 중인가?

[0185] S345:디폴트 표시의 지시인가?

[0186] S346:재생 중인 자막 스트림의 StaticInfo()를 취득

[0187] S347:StaticInfo()의 configurable_flag를 판정

[0188] S348:에러 표시

[0189] S349:표시 방식 지시를 공급

[0190] S350:표시 방식 지시에 따라서, 자막의 처리를 개시

[0191] S351:PlayItem 갈아타기가 행하여졌는가?

[0192] S371:비디오를 재생 중인가?

[0193] S372:재생 중인 비디오의 PlayList()와 Clip()으로부터, capture_enable_flag를 취득

[0194] S373:capture_enable_flag에 기초하여, 캡처의 가부를 판정?

[0195] S374:에러 표시

[0196] S375:캡처 지시를 공급

[0197] S376:캡처, 기억

[0198] S381:픽처의 표시

[0199] <발명을 실시하기 위한 최량의 형태>

[0200] 이하에 본 발명의 실시예를 설명하지만, 청구의 범위에 기재된 구성 요건과, 발명의 실시예에서의 구체예와의 대응 관계를 예시하면, 다음과 같이 된다. 이 기재는, 청구의 범위에 기재되어 있는 발명을 서포트하는 구체예가, 발명의 실시예에 기재되어 있는 것을 확인하기 위한 것이다. 따라서, 발명의 실시예 중에는 기재되어 있지만, 구성 요건에 대응하는 것으로서, 여기에는 기재되지 않은 구체예가 있었다고 해도, 그것은, 그 구체예가, 그 구성 요건에 대응하는 것이 아님을 의미하는 것은 아니다. 반대로, 구체예가 구성 요건에 대응하는 것으로서 여기에 기재되어 있었다고 해도, 그것은, 그 구체예가, 그 구성 요건 이외의 구성 요건에는 대응하지 않음을 의미하는 것도 아니다.

[0201] 또한, 이 기재는, 발명의 실시예에 기재되어 있는 구체예에 대응하는 발명이, 청구의 범위에 모두 기재되어

있는 것을 의미하는 것은 아니다. 바꾸어 말하면, 이 기재는, 발명의 실시예에 기재되어 있는 구체예에 대응하는 발명으로서, 이 출원의 청구의 범위에는 기재되지 않은 발명의 존재, 즉, 장래, 분할 출원되거나, 보정에 의해 추가되는 발명의 존재를 부정하는 것은 아니다.

[0202] 본 발명의 청구의 범위 1에 기재된 데이터 처리 장치는,

[0203] 부호화 데이터를 처리하는 데이터 처리 장치(예를 들면, 도 1의 디스크 장치)로서,

[0204] 상기 부호화 데이터는,

[0205] 소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와,

[0206] 상기 비디오 데이터와 동기해서 출력될 출력 데이터와,

[0207] 상기 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 상기 부호화 비디오 데이터의 디코드에 이용되는 이용 정보(예를 들면, 도 23의 private_stream_2_PES_payload())

[0208] 를 포함하고,

[0209] 상기 이용 정보는, 그 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 상기 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보(예를 들면, 도 24의 au_ref_flag)를 포함하고,

[0210] 상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다도 지연되고 있는지의 여부를 판정하는 제1 판정 수단(예를 들면, 도 39의 스텝 S272의 처리를 행하는 도 2A 및 도 2B의 디코드 제어 모듈(214))과,

[0211] 상기 제1 판정 수단에서, 상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다도 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 수단(예를 들면, 도 39의 스텝 S273의 처리를 행하는 도 2A 및 도 2B의 디코드 제어 모듈(214))과,

[0212] 상기 지시 수단에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 상기 참조 정보에 기초하여, 그 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 수단(예를 들면, 도 39의 스텝 S275의 처리를 행하는 도 2A 및 도 2B의 디코드 제어 모듈(214))과,

[0213] 상기 제2 판정 수단에서, 상기 지시 수단에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 상기 지시 수단에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 수단(예를 들면, 도 39의 스텝 S277의 처리를 행하는 도 2A 및 도 2B의 비디오 디코더 제어 모듈(216))

[0214] 을 포함하는 것을 특징으로 한다.

[0215] 본 발명의 청구의 범위 2에 기재된 데이터 처리 장치는,

[0216] 상기 제1 판정 수단에서, 상기 출력 데이터의 출력이, 상기 비디오 데이터의 출력보다도 지연되고 있다고 판정된 경우에, 상기 비디오 데이터를 반복해 출력시키는 출력 제어 수단(예를 들면, 도 39의 스텝 S279의 처리를 행하는 도 2A 및 도 2B의 비디오 디코더 제어 모듈(216))을 더 구비하는 것을 특징으로 한다.

[0217] 본 발명의 청구의 범위 3에 기재된 데이터 처리 방법은,

[0218] 부호화 데이터를 처리하는 데이터 처리 방법으로서,

[0219] 상기 부호화 데이터는,

[0220] 소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와,

[0221] 상기 비디오 데이터와 동기해서 출력될 출력 데이터와,

[0222] 상기 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 상기 부호화 비디오 데이터의 디코드에 이용되는 이용 정보(예를 들면, 도 23의 private_stream_2_PES_payload())

[0223] 를 포함하고,

[0224] 상기 이용 정보는, 그 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 상기 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보(예를 들면, 도 24의 au_ref_flag)를 포함하고,

[0225] 상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다도 지연되고 있는지의 여부를 판정하는 제1 판정 스텝(예를 들면, 도 39의 스텝 S272)과,

[0226] 상기 제1 판정 스텝에서, 상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다도 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 스텝(예를 들면, 도 39의 스텝 S273)과,

[0227] 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 상기 참조 정보에 기초하여, 그 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 스텝(예를 들면, 도 39의 스텝 S275)과,

[0228] 상기 제2 판정 스텝에서, 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 스텝(예를 들면, 도 39의 스텝 S277)

[0229] 을 포함하는 것을 특징으로 한다.

[0230] 본 발명의 청구의 범위 4에 기재된 프로그램 및 청구의 범위 5에 기재된 프로그램 기록 매체에 기록되어 있는 프로그램은,

[0231] 부호화 데이터를 처리하는 데이터 처리를, 컴퓨터에 행하게 하는 프로그램으로서,

[0232] 상기 부호화 데이터는,

[0233] 소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와,

[0234] 상기 비디오 데이터와 동기해서 출력될 출력 데이터와,

[0235] 상기 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 상기 부호화 비디오 데이터의 디코드에 이용되는 이용 정보(예를 들면, 도 23의 private_stream_2_PES_payload())

[0236] 를 포함하고,

[0237] 상기 이용 정보는, 그 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 상기 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보(예를 들면, 도 24의 au_ref_flag)를 포함하고,

[0238] 상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다도 지연되고 있는지의 여부를 판정하는 제1 판정 스텝(예를 들면, 도 39의 스텝 S272)과,

[0239] 상기 제1 판정 스텝에서, 상기 비디오 데이터의 출력이, 상기 출력 데이터의 출력보다도 지연되고 있다고 판정된 경우에, 1개의 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵하는 것을 지시하는 지시 스텝(예를 들면, 도 39의 스텝 S273)과,

[0240] 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터에 대한 상기 참조 정보에 기초하여, 그 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 판정하는 제2 판정 스텝(예를 들면, 도 39의 스텝 S275)과,

[0241] 상기 제2 판정 스텝에서, 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되지 않는다고 판정된 경우에, 상기 지시 스텝에서 처리의 스킵이 지시된 액세스 유닛의 부호화 비디오 데이터의 처리를 스킵시키는 스킵 제어 스텝(예를 들면, 도 39의 스텝 S277)

[0242] 을 포함하는 것을 특징으로 한다.

[0243] 본 발명의 청구의 범위 6에 기재된 데이터 기록 매체는,

[0244] 부호화 데이터가 기록되어 있는 데이터 기록 매체로서,

[0245] 상기 부호화 데이터는,

[0246] 소정의 단위의 비디오 데이터를 부호화해서 얻어지는, 액세스 유닛 단위의 부호화 비디오 데이터와,

[0247] 상기 비디오 데이터와 동기해서 출력될 출력 데이터와,

[0248] 상기 액세스 유닛 단위의 부호화 비디오 데이터에서의 1 이상의 디코드 개시 가능점 각각의 직전에 배치되는, 상기 부호화 비디오 데이터의 디코드에 이용되는 이용 정보(예를 들면, 도 23의 private_stream_2_PES_payload())

[0249] 를 포함하고,

[0250] 상기 이용 정보는, 그 이용 정보로부터 다음 이용 정보까지의 사이에 배치되어 있는 1 이상의 상기 액세스 유닛 각각의 부호화 비디오 데이터에 대응하는 비디오 데이터가, 다른 부호화 비디오 데이터의 디코드에 참조되는지의 여부를 나타내는 참조 정보(예를 들면, 도 24의 au_ref_flag)를 포함하는 것을 특징으로 한다.

[0251] 이하, 도면을 참조하여, 본 발명의 실시예에 대하여 설명한다.

[0252] [하드웨어 구성]

[0253] 도 1은 본 발명을 적용한 디스크 장치의 제1 실시예의 하드웨어의 구성예를 도시하는 블록도이다.

[0254] 도 1의 디스크 장치는, 예를 들면, 디스크 플레이어나, 게임 장치, 카 네비게이션 시스템 그 밖에 적용할 수 있다.

[0255] 도 1의 디스크 장치에서, 디스크(101)는, 예를 들면, DVD 등의 광 디스크, 혹은 광 자기 디스크, 자기 디스크 등으로서, 비디오 데이터나, 오디오 데이터, 자막 데이터 등의 콘텐츠 데이터, 또한, 콘텐츠 데이터를 재생하는 데도 필요한 데이터가 기록되어 있다.

[0256] 또한, 디스크(101)에 기록되는 데이터(기록 데이터)에는, 필요에 따라, 컴퓨터가 실행 가능한 프로그램도 포함된다. 또한, 본 실시예에서는, 기록 매체로서, 디스크 형상의 기록 매체인 디스크(101)를 채용하지만, 그 밖의, 기록 매체로서는, 예를 들면, 반도체 메모리나, 테이프 형상의 기록 매체이어도 된다. 또한, 도 1의 디스크 장치에는, 먼 곳에 있는 디스크(101)로부터 읽어내어져 송신되어 오는 데이터를 입력받을 수 있다. 즉, 디스크(101)로부터의 데이터의 읽어내기는, 디스크 장치에 접속한 별도의 장치에서 행하고, 그 별도의 장치에서 읽어내어진 데이터를, 디스크 장치에서 수신하여 처리할 수 있다. 또한, 디스크 장치에서는, 디스크(101)에 기록된 데이터와 마찬가지의 데이터를 스토리지에 기억하고 있는 서버 등으로부터, 인터넷 등의 네트워크를 통하여, 데이터의 배신을 받아 처리하는 것도 가능하다. 또한, 디스크 장치에서는, 서버 그 밖의 장치로부터의 데이터를 수신하고, 일단, 디스크(101)에 기록하고나서, 그 디스크(101)에 기록된 데이터를 처리하는 것도 가능하다.

[0257] 디스크 드라이브(102)에는, 디스크(101)가 착탈 가능하게 되어 있다. 디스크 드라이브(102)는, 도시하지 않은 인터페이스를 내장하고, 그 인터페이스를 통하여, 드라이브 인터페이스(114)에 접속되어 있다. 디스크 드라이브(102)는, 거기에 장착된 디스크(101)를 구동하여, 드라이브 인터페이스(114)로부터의 읽어내기 등의 명령에 따라, 디스크(101)로부터 데이터를 읽어내어, 드라이브 인터페이스(114)에 공급하는 등의 처리를 행한다.

[0258] 버스(111)에는, CPU(Central Processing Unit)(112), 메모리(113), 드라이브 인터페이스(114), 입력 인터페이스(115), 비디오 디코더(116), 오디오 디코더(117), 비디오 출력 인터페이스(118), 오디오 출력 인터페이스(119)가 접속되어 있다.

[0259] CPU(112) 및 메모리(113)는, 컴퓨터 시스템을 형성하고 있다. 즉, CPU(112)는, 메모리(113)에 기억된 프로그램인, 후술하는 소프트웨어 모듈 군을 실행하여, 디스크 장치 전체를 제어함과 함께, 후술하는 각종 처리를 행한다. 메모리(113)는, CPU(112)가 실행하는 소프트웨어 모듈 군을 기억하고 있다. 또한, 메모리(113)는, CPU(112)의 동작상 필요한 데이터를 일시 기억한다. 또한, 메모리(113)는, 불휘발성 메모리만, 또는 휘발성 메모리와 불휘발성 메모리의 조합으로 구성하는 것이 가능하다. 또한, 도 1의 디스크 장치에, 하드디스크를 설치하고, 그 하드디스크에, CPU(112)가 실행하는 소프트웨어 모듈 군을 기록(인스톨)해 둘 경우에는, 메모리(113)는 휘발성 메모리만으로 구성하는 것이 가능하다.

[0260] 여기에서, CPU(112)가 실행하는 프로그램(소프트웨어 모듈 군)은, 디스크 장치에 내장되어 있는 기록 매체로서의 메모리(113)에 미리 기록해 둘(기억시켜 둘) 수 있다.

[0261] 또한, 프로그램은, 디스크(101), 나아가서는, 디스크(101) 이외의 플렉시블 디스크, CD-ROM(Compact Disc Read Only Memory), MO(Magneto Optical) 디스크, 자기 디스크, 메모리 카드 등의 리무버블 기록 매체에, 일시적 혹은 영속적으로 저장(기록)해 둘 수 있다. 이러한 리무버블 기록 매체는, 소위 팩키지 소프트 웨어로서 제공할 수 있다.

[0262] 또한, 프로그램은, 메모리(113)에 미리 기억시켜 두는 것, 혹은, 전술한 바와 같은 리무버블 기록 매체로부터 디스크 장치에 인스톨할 수 있다. 또한, 프로그램은, 다운로드 사이트로부터, 디지털 위성 방송용의 인공위성을 통하여, 디스크 장치에 무선으로 전송하거나, LAN(Local Area Network), 인터넷 등의 네트워크를 통하여, 디스크 장치에 유선으로 전송하고, 디스크 장치에서는, 그와 같이 하여 전송되어 오는 프로그램을, 입력 인터페이스(115)에서 수신하고, 내장한 메모리(113)에 인스톨할 수 있다.

[0263] 또한, 프로그램은, 하나의 CPU에 의해 처리되는 것이어도 되고, 복수의 CPU에 의해 분산 처리되는 것이어도 된다.

[0264] 드라이브 인터페이스(114)는, CPU(112)의 제어 하에, 디스크 드라이브(102)를 제어하고, 이에 의해, 디스크 드라이브(102)가 디스크(101)로부터 읽어낸 데이터를, 패스(111)를 통하여, CPU(112)나, 메모리(113), 비디오 디코더(116), 오디오 디코더(117) 등에 공급한다.

[0265] 입력 인터페이스(115)는, 도시하지 않은 키(버튼)나, 리모콘(리모트 컨트롤러)이 유저에 조작됨으로써 공급되는 신호를 수신하고, 버스(111)를 통하여, CPU(112)에 공급한다. 또한, 입력 인터페이스(115)는, 그 밖의, 예를 들면, 모뎀(ADSL(Asymmetric Digital Subscriber Line) 모뎀을 포함함)이나, NIC(Network Interface Card) 등의 통신 인터페이스로서도 기능한다.

[0266] 비디오 디코더(116)는, 디스크 드라이브(102)에 의해 디스크(101)로부터 읽어내어져, 드라이브 인터페이스(114) 및 버스(111)을 통하여 공급되는, 비디오 데이터의 부호화 데이터(부호화 비디오 데이터)를 디코드하고, 그 결과 얻어지는 비디오 데이터를, 버스(111)를 통하여, CPU(112)나 비디오 출력 인터페이스(118)에 공급한다.

[0267] 오디오 디코더(117)는, 디스크 드라이브(102)에 의해 디스크(101)로부터 읽어내어져, 드라이브 인터페이스(114) 및 버스(111)을 통하여 공급되는, 오디오 데이터의 부호화 데이터(부호화 오디오 데이터)를 디코드하고, 그 결과 얻어지는 오디오 데이터를, 버스(111)을 통하여, CPU(112)나 오디오 출력 인터페이스(119)에 공급한다.

[0268] 비디오 출력 인터페이스(118)는, 버스(111)를 통하여 공급되는 비디오 데이터에 필요한 처리를 실시하고, 비디오 출력 단자(120)로부터 출력한다. 오디오 출력 인터페이스(119)는, 버스(111)을 통하여 공급되는 오디오 데이터에 필요한 처리를 실시하여, 오디오 출력 단자(121)로부터 출력한다.

[0269] 비디오 출력 단자(120)는, 도시하지 않은 CRT(Cathode Ray Tube)나, 액정 패널 등의 비디오 출력 장치에 접속되어 있고, 따라서, 비디오 출력 단자(120)로부터 출력되는 비디오 데이터는, 비디오 출력 장치에 공급되어 표시된다. 오디오 출력 단자(121)는, 도시하지 않은 스피커나 앰프 등의 오디오 출력 장치에 접속되어 있고, 따라서, 오디오 출력 단자(121)로부터 출력되는 오디오 데이터는, 오디오 출력 장치에 공급되어서 출력된다.

[0270] 또한, 디스크 장치로부터, 비디오 출력 장치와 오디오 출력 장치에의 비디오 데이터와 오디오 데이터의 공급은, 유선 또는 무선 중 어느 하나에 의하여 행하는 것도 가능하다.

[0271] [소프트웨어 모듈 군의 구성]

[0272] 다음으로, 도 2A 및 도 2B는, 도 1의 CPU(112)가 실행하는 소프트웨어 모듈 군의 구성예를 도시하고 있다.

[0273] CPU(112)가 실행하는 소프트웨어 모듈 군은, 오퍼레이팅 시스템(OS)(201)과, 어플리케이션 프로그램으로서의 비디오 콘텐츠 재생 프로그램(210)으로 대별된다.

[0274] 「오퍼레이팅 시스템(201)」

[0275] 오퍼레이팅 시스템(201)은, 디스크 장치의 전원이 투입되면 최초로 기동하여(CPU(112)가 오퍼레이팅 시스템(201)을 실행하여), 초기 설정 등의 필요한 처리를 행하고, 어플리케이션 프로그램인 비디오 콘텐츠 재생 프로그램(210)을 불러낸다.

[0276] 오퍼레이팅 시스템(201)은, 비디오 콘텐츠 재생 프로그램(210)에 대하여, 파일의 읽어내기 등의 인프라(인프라스트럭처; infrastructure)적인 서비스를 제공한다. 즉, 오퍼레이팅 시스템(201)은, 예를 들면, 파일의 읽어내기에 관해서는, 비디오 콘텐츠 재생 프로그램(210)으로부터의 파일의 읽어내기의 리퀘스트에 대하여, 드라이브 인터페이스(114)를 통하여 디스크 드라이브(102)를 조작하여, 디스크(101)의 데이터를 읽어내어, 비디오 콘텐츠 재생 프로그램(210)에 건재주는 서비스를 제공한다. 또한, 오퍼레이팅 시스템(201)은, 파일 시스템의 해석 등도 행한다.

[0277] 또한, 오퍼레이팅 시스템(201)은, 멀티태스크 처리의 기능을 구비하고 있으며, 복수의 소프트웨어 모듈을, 시분할로 (외관상) 동시에 동작시킬 수 있다. 즉, 비디오 콘텐츠 재생 프로그램(210)은, 몇개의 소프트웨어 모듈로 구성되지만, 각 소프트웨어 모듈은, 병렬로 동작할 수 있다.

[0278] 「비디오 콘텐츠 재생 프로그램(210)」

[0279] 비디오 콘텐츠 재생 프로그램(210)은, 스크립트 제어 모듈(211), 플레이어 제어 모듈(212), 콘텐츠 데이터 공급 모듈(213), 디코드 제어 모듈(214), 버퍼 제어 모듈(215), 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 자막 디코더 제어 모듈(218), 그래픽스 처리 모듈(219), 비디오 출력 모듈(220), 및 오디오 출력 모듈(221)로 구성되어 있다.

[0280] 비디오 콘텐츠 재생 프로그램(210)은, 디스크(101)의 재생에 있어서 중심적인 역활을 하는 소프트웨어로서, 디스크(101)가 디스크 드라이브(102)에 장착(삽입)되면, 그 디스크(101)가, 콘텐츠가 기록된 후술하는 포맷의 디스크인지를 확인한다. 또한, 비디오 콘텐츠 재생 프로그램(210)은, 디스크(101)로부터, 후술하는 스크립 파일을 읽어내어 실행하고, 또한, 디스크(101)로부터, 그 디스크(101)에 기록된 콘텐츠를 재생하는 데에도 필요한 메타데이터(데이터베이스 정보)의 파일을 읽어내어, 그 메타데이터에 기초하여, 콘텐츠의 재생을 제어한다.

[0281] 이하, 도 2A 및 도 2B의 비디오 콘텐츠 재생 프로그램(210)을 구성하는 소프트웨어 모듈에 대하여 설명한다. 또한, 도 2A 및 도 2B에서는, 원칙으로서, 실선의 화살표는, 콘텐츠의 데이터를 나타내고, 점선의 화살표는, 제어의 데이터를 나타낸다.

[0282] 「스크립트 제어 모듈(211)」

[0283] 스크립트 제어 모듈(211)은, 디스크(101)에 기록된 스크립 파일에 기술되어 있는 스크립트 프로그램(스크립트)을 해석하여 실행한다. 스크립트 프로그램에서는, 예를 들면, 「그래픽 처리 모듈(219)을 조작하여, 메뉴 등의 화상을 작성하고 표시함」, 「리모콘 등의 UI(User Interface)로부터의 신호에 따라 메뉴의 표시를 변경함(예를 들면, 메뉴 상의 커서를 이동하는 등)」, 「플레이어 제어 모듈(212)을 제어함」 등의 동작을 기술할 수 있다.

[0284] 「플레이어 제어 모듈(212)」

[0285] 플레이어 제어 모듈(212)은, 디스크(101)에 기록되어 있는 메타데이터(데이터 페이스 정보) 등을 참조하여, 콘텐츠의 재생에 관한 제어를 행한다. 즉, 플레이어 제어 모듈(212)은, 예를 들면, 디스크(101)에 기록되어 있는, 후술하는 PlayList()나 Clip()을 해석하고, 그 해석 결과에 따라, 콘텐츠 데이터 공급 모듈(213)이나, 디코드 제어 모듈(214), 버퍼 제어 모듈(215)을 제어한다. 또한, 플레이어 제어 모듈(212)은, 스크립트 제어 모듈(211)이나 입력 인터페이스(115)로부터의 지시에 따라, 재생 대상의 스트림을 절환하는, 후술하는 스트림 절환 등의 제어를 행한다. 또한, 플레이어 제어 모듈(212)은, 디코드 제어 모듈(214)로부터 시각을 취득하고, 시각 표시나, 후술하는 마크(Mark())의 처리 등을 행한다.

[0286] 「콘텐츠 데이터 공급 모듈(213)」

[0287] 콘텐츠 데이터 공급 모듈(213)은, 플레이어 제어 모듈(212)의 제어에 따라,혹은, 버퍼 제어 모듈(215)에 축적된 데이터의 양에 기초하여, 디스크(101)로부터의 콘텐츠의 데이터나 메타데이터 등의 읽어내기를, 오퍼레이팅 시스템(201)에 요구한다.

[0288] 또한, 오퍼레이팅 시스템(201)이, 콘텐츠 데이터 공급 모듈(213)로부터의 요구에 따라 디스크(101)로부터 읽어낸 메타데이터 등은, 필요한 모듈에 공급된다. 또한, 오퍼레이팅 시스템(201)이, 콘텐츠 데이터 공급 모듈(213)로부터의 요구에 따라 디스크(101)로부터 읽어낸 콘텐츠의 데이터는, 버퍼 제어 모듈(215)에 공급된다.

[0289] 「디코드 제어 모듈(214)」

[0290] 디코드 제어 모듈(214)은, 플레이어 제어 모듈(212)로부터의 제어에 따라, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 및 자막 디코더 제어 모듈(218)의 동작을 제어한다. 또한, 디코드 제어 모듈(214)은, 시각을 계시하는 계시부(214A)를 내장하고, 비디오 디코더 제어 모듈(216)의 제어에 의해 출력되는 비디오 데이터의 출력과, 그 비디오 데이터와 동기하여 출력될 데이터(출력 데이터)의 출력, 즉, 여기에서는, 오디오 디코더 제어 모듈(217)의 제어에 의해 출력되는 오디오 데이터의 출력과의 동기를 관리한다.

[0291] 「버퍼 제어 모듈(215)」

[0292] 버퍼 제어 모듈(215)은, 도 1의 메모리(113)의 기억 영역의 일부인 버퍼(215A)를 내장하고 있으며, 그 버퍼(215A)에, 콘텐츠 데이터 공급 모듈(213)이 오퍼레이팅 시스템(201)에 요구를 행함으로써 디스크(101)로부터 읽어내어진 콘텐츠의 데이터를 일시 기억한다.

[0293] 또한, 버퍼 제어 모듈(215)은, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 또는 자막 디코더 제어 모듈(218)의 요구에 따라, 버퍼(215A)에 기억된 데이터를, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 또는 자막 디코더 제어 모듈(218)에 공급한다.

[0294] 즉, 버퍼 제어 모듈(215)은, 후술하는 도 3에서 설명하는 비디오 읽어내기 기능부(233), 오디오 읽어내기 기능부(234), 및 자막 읽어내기 기능부(235)를 내장하고 있다. 그리고, 버퍼 제어 모듈(215)은, 비디오 디코더 제어 모듈(216)로부터의 데이터의 요구를, 비디오 읽어내기 기능부(233)에서 처리함으로써, 버퍼(215A)에 기억된 데이터를, 비디오 디코더 제어 모듈(216)에 공급한다. 마찬가지로, 버퍼 제어 모듈(215)은, 오디오 디코더 제어 모듈(217)로부터의 데이터의 요구를, 오디오 읽어내기 기능부(234)에서 처리함으로써, 버퍼(215A)에 기억된 데이터를, 오디오 디코더 제어 모듈(217)에 공급함과 함께, 자막 디코더 제어 모듈(218)로부터의 데이터의 요구를, 자막 읽어내기 기능부(235)에서 처리함으로써, 버퍼(215A)에 기억된 데이터를, 자막 디코더 제어 모듈(218)에 공급한다.

[0295] 「비디오 디코더 제어 모듈(216)」

[0296] 비디오 디코더 제어 모듈(216)은, 버퍼 제어 모듈(215) 내의 비디오 읽어내기 기능부(233)(도 3)를 조작하여, 비디오 데이터를 부호화한 데이터(비디오 부호화 데이터)를, 비디오 액세스 유닛 단위로, 버퍼 제어 모듈(215)의 버퍼(215A)로부터 읽어내어, 도 1의 비디오 디코더(116)에 공급한다. 또한, 비디오 디코더 제어 모듈(216)은, 비디오 디코더(116)를 제어하고, 비디오 액세스 유닛 단위의 데이터를 디코드시킨다. 또한, 비디오 디코더 제어 모듈(216)은, 비디오 디코더(116)에서의 디코드의 결과 얻어지는 비디오 데이터를, 그래픽 처리 모듈(219)에 공급한다.

[0297] 여기에서, 비디오 액세스 유닛이란, 예를 들면, 비디오 데이터의 1픽처(1프레임 또는 1필드)분이다.

[0298] 「오디오 디코더 제어 모듈(217)」

[0299] 오디오 디코더 제어 모듈(217)은, 버퍼 제어 모듈(215) 내의 오디오 읽어내기 기능부(234)(도 3)를 조작하여, 오디오 데이터를 부호화한 데이터(오디오 부호화 데이터)를, 오디오 액세스 유닛 단위로, 버퍼 제어 모듈(215)의 버퍼(215A)로부터 읽어내어, 도 1의 오디오 디코더(117)에 공급한다. 또한, 오디오 디코더 제어 모듈(217)은, 오디오 디코더(117)를 제어하여, 오디오 액세스 유닛 단위의 데이터를 디코드시킨다. 또한, 오디오 디코더 제어 모듈(217)은, 오디오 디코더(117)에서의 디코드의 결과 얻어지는 오디오 데이터를, 오디오 출력 모듈(221)에 공급한다.

[0300] 여기에서, 오디오 액세스 유닛이란, 오디오 데이터의 소정의 데이터량분(예를 들면, 1픽처에 동기하여 출력되는 분)이다. 본 실시예에서는, 오디오 액세스 유닛은, 예를 들면, 기지의 고정 길이라고 한다.

[0301] 「자막 디코더 제어 모듈(218)」

[0302] 자막 디코더 제어 모듈(218)은, 버퍼 제어 모듈(215) 내의 자막 읽어내기 기능부(235)(도 3)를 조작하여, 자막 데이터를 부호화한 데이터(자막 부호화 데이터)를, 자막 액세스 유닛 단위로, 버퍼 제어 모듈(215)의 버퍼(215A)로부터 읽어낸다. 또한, 자막 디코더 제어 모듈(218)은, 내부에, 도시하지 않은 자막 디코드 소프트웨어를 구비하고 있으며, 버퍼(215A)로부터 읽어낸 데이터를 디코드한다. 또한, 자막 디코더 제어 모듈(218)은, 그 디코드의 결과 얻어지는 자막 데이터(자막의 화상 데이터)를, 그래픽 처리 모듈(219)에 공급한다.

[0303] 여기에서, 자막 액세스 유닛이란, 자막 데이터의 소정의 데이터량 분(예를 들면, 1픽처에 동기하여 출력되는 분)이다. 본 실시예에서는, 자막 액세스 유닛의 사이즈는, 예를 들면, 그 자막 액세스 유닛의 선두에 기술되

어 있는 것으로 한다.

[0304] 「그래픽 처리 모듈(219)」

[0305] 그래픽 처리 모듈(219)은, 플레이어 제어 모듈(212)의 제어(지시)에 따라, 자막 디코더 제어 모듈(218)로부터의 자막 데이터의 확대나 축소를 행하여, 비디오 디코더 제어 모듈(216)로부터의 비디오 데이터와 가산(오버레이)한다. 또한, 그래픽 처리 모듈(219)은, 자막 데이터와의 가산 후의 비디오 데이터의 사이즈(화면틀)를, 도 1의 비디오 출력 단자(120)에 접속된 비디오 출력 장치의 표시 화면에 맞추기 위한 확대 또는 축소 등을 행하고, 그 결과 얻어지는 비디오 데이터를, 비디오 출력 모듈(220)에 출력한다.

[0306] 또한, 그래픽 처리 모듈(219)은, 스크립트 제어 모듈(211)이나 플레이어 제어 모듈(212)의 지시(제어)에 따라, 메뉴나 메시지 등을 생성하고, 출력 비디오 데이터에 오버레이한다.

[0307] 또한, 그래픽 처리 모듈(219)은, 도 1의 비디오 출력 단자(120)에 접속된 비디오 출력 장치의 어스펙트비와, 디스크(101)에 기록된 비디오 데이터의 어스펙트비를 지시하는 정보 등에 기초하여, 비디오 출력 모듈(220)에 출력하는 비디오 데이터의 어스펙트비의 변환을 행한다.

[0308] 즉, 예를 들면, 비디오 출력 장치의 어스펙트비가 16:9인 경우에, 비디오 데이터의 어스펙트비를 지시하는 정보가 4:3의 어스펙트비를 나타내고 있을 때에는, 그래픽 처리 모듈(219)은, 비디오 출력 모듈(220)에 출력하는 비디오 데이터를, 가로 방향(수평 방향)으로 스퀴즈(축소) 처리하고, 좌우에 검은 부분을 넣어 출력한다. 또한, 예를 들면, 비디오 출력 장치의 어스펙트비가 4:3인 경우에, 비디오 데이터의 어스펙트비를 지시하는 정보가 16:9의 어스펙트비를 나타내고 있을 때에는, 그래픽 처리 모듈(219)은, 비디오 출력 모듈(220)에 출력하는 비디오 데이터를, 세로 방향(수직 방향)으로 스퀴즈(축소) 처리하고, 상하에 검은 부분을 넣어 출력한다.

[0309] 또한, 비디오 출력 장치의 어스펙트비와, 비디오 데이터의 어스펙트비를 지시하는 정보가 나타내는 어스펙트비가, 모두, 4:3이나 16:9이며, 동일한 경우, 그래픽 처리 모듈(219)은, 비디오 출력 모듈(220)에 출력하는 비디오 데이터를, 스퀴즈 처리하지 않고, 그대로 출력한다.

[0310] 그 밖의, 그래픽 처리 모듈(219)은, 예를 들면, 플레이어 제어 모듈(212)로부터의 요구에 따라, 지금 처리 중인 비디오 데이터를 캡쳐한다. 또한, 그래픽 처리 모듈(219)은, 그 캡쳐한 비디오 데이터를 기억하거나, 혹은, 플레이어 제어 모듈(212)에 공급한다.

[0311] 「비디오 출력 모듈(220)」

[0312] 비디오 출력 모듈(220)은, 도 1의 메모리(113)의 일부를 배타적으로 점유하여 FIFO(First In First Out)(220A)(버퍼)로서 사용하여, 그래픽 처리 모듈(219)로부터의 비디오 데이터를 일시적으로 기억하고, 또한 그 FIFO(220A)에 기억된 비디오 데이터를 적절히 읽어내어, 비디오 출력 단자(120)(도 1)에 출력한다.

[0313] 「오디오 출력 모듈(221)」

[0314] 오디오 출력 모듈(221)은, 도 1의 메모리(113)의 일부를 배타적으로 점유하여 FIFO(221A)(버퍼)로서 사용하고, 오디오 디코더 제어 모듈(217)(오디오 디코더(117))로부터의 오디오 데이터를 일시적으로 기억하고, 또한, 그 FIFO(221A)에 기억된 오디오 데이터를 적절히 읽어내어, 오디오 출력 단자(121)(도 1)에 출력한다.

[0315] 또한, 오디오 출력 모듈(221)은, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터가, 좌측 채널이 「주음성」의 오디오 데이터이고, 우측 채널의 「부음성」의 오디오 데이터인 이중(Dual)(2개 국어) 모드의 오디오 데이터인 경우, 미리 지정된 음성 출력 모드에 따라, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터를, 오디오 출력 단자(121)에 출력한다.

[0316] 즉, 음성 출력 모드로서, 예를 들면, 「주음성」이 지정되어 있을 때에는, 오디오 출력 모듈(221)은, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터 중의 좌측 채널의 오디오 데이터를, 우측 채널의 오디오 데이터로서 카피하고, 그 좌측 채널과 우측 채널의 오디오 데이터(「주음성」의 오디오 데이터)를, 오디오 출력 단자(121)에 출력한다. 또한, 음성 출력 모드로서, 「부음성」이 지정되어 있을 때에는, 오디오 출력 모듈(221)은, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터 중의 우측 채널의 오디오 데이터를, 좌측 채널의 오디오 데이터로서 카피하고, 그 좌측 채널과 우측 채널의 오디오 데이터(「부음성」의 오디오 데이터)를, 오디오 출력 단자(121)에 출력한다. 또한, 음성 출력 모드로서, 「주?부」가 지정되어 있을 때에는, 오디오 출력 모듈(221)은, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터를, 그대로, 오디오 출력 단자

(121)에 출력한다.

[0317] 또한, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터가, 스테레오(Stereo) 모드의 오디오 데이터인 경우, 오디오 출력 모듈(221)은, 음성 출력 모드의 지정에 관계없이, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터를, 그대로, 오디오 출력 단자(121)에 출력한다.

[0318] 여기에서, 음성 출력 모드의 지정은, 예를 들면, 비디오 콘텐츠 재생 프로그램(210)이 생성하는 메뉴가 표시된 화면 등에서, 유저가 리모콘 등을 조작함으로써 대화적으로 행할 수 있다.

[0319] [버퍼 제어 모듈(215)의 구성]

[0320] 다음으로, 도 3은, 도 2A 및 도 2B의 버퍼 제어 모듈(215)의 구성예를 도시하고 있다.

[0321] 버퍼 제어 모듈(215)은, 도 1의 메모리(113)의 일부를, 버퍼(215A)로서 배타적으로 사용하여, 그 버퍼(215A)에, 디스크(101)로부터 읽어내어진 데이터를 일시 기억시킨다. 또한, 버퍼 제어 모듈(215)은, 버퍼(215A)에 기억된 데이터를 읽어내어, 도 2A 및 도 2B의 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217),또는 자막 디코더 제어 모듈(218)에 공급한다.

[0322] 즉, 버퍼 제어 모듈(215)은, 버퍼(215A) 외, 메모리(113)의 일부인 데이터 선두 포인터 기억부(231), 및 데이터 써넣기 포인터 기억부(232)를 가짐과 함께, 내부 모듈로서, 비디오 읽어내기 기능부(233), 오디오 읽어내기 기능부(234), 자막 읽어내기 기능부(235)를 가진다.

[0323] 버퍼(215A)는, 예를 들면, 링 버퍼로서, 디스크(101)로부터 읽어내어진 데이터를 순차적으로 기억하고, 그 기억 용량분의 데이터를 기억한 후에는, 가장 오래된 데이터에 덮어쓰는 형식으로, 최신의 데이터를, 이른바 무한 루프 형상으로 기억해 간다.

[0324] 데이터 선두 포인터 기억부(231)는, 버퍼(215A)에 기억된 데이터 중, 아직, 버퍼(215A)로부터 읽어내어져 있지 않은 가장 오래된 데이터가 기억되어 있는 위치(어드레스)를 가리키는 데이터 선두 포인터를 기억한다.

[0325] 데이터 써넣기 포인터 기억부(232)는, 디스크(101)로부터 읽어내어진 최신의 데이터가 써넣는 버퍼(215A)의 위치(어드레스)를 가리키는 써넣기 포인터를 기억한다.

[0326] 여기에서, 데이터 써넣기 포인터가 가리키는 위치는, 버퍼(215A)에, 디스크(101)로부터 읽어내어진 데이터가 기억될 때마다, 도면에서, 오른쪽 방향(시계 방향)으로 갱신되어 가고, 데이터 선두 포인터가 가리키는 위치는, 버퍼(215A)로부터의 데이터의 읽어내기에 따라, 도면에서, 오른쪽 방향으로 갱신되어 간다. 따라서, 버퍼(215A)에 기억된 데이터 중, 말하자면 유효한 데이터는, 데이터 선두 포인터가 가리키는 위치로부터, 오른쪽 방향으로, 데이터 써넣기 포인터가 가리키는 위치까지 기억되어 있는 데이터이다.

[0327] 비디오 읽어내기 기능부(233)는, 도 2A 및 도 2B의 비디오 디코더 제어 모듈(216)로부터의 요구에 따라, 버퍼(215A)로부터 비디오 스트림(비디오 데이터에 관한 엘리멘터리 스트림)을 읽어내어, 비디오 디코더 제어 모듈(216)에 공급한다. 오디오 읽어내기 기능부(234)도, 도 2A 및 도 2B의 오디오 디코더 제어 모듈(217)로부터의 요구에 따라서, 버퍼(215A)로부터 오디오 스트림(오디오 데이터에 관한 엘리멘터리 스트림)을 읽어내어, 오디오 디코더 제어 모듈(217)에 공급한다. 자막 읽어내기 기능부(235)도, 도 2A 및 도 2B의 자막 디코더 제어 모듈(218)로부터의 요구에 따라, 버퍼(215A)로부터 자막 스트림(자막 데이터에 관한 엘리멘터리 스트림)을 읽어내어, 자막 디코더 제어 모듈(218)에 공급한다.

[0328] 즉, 광 디스크(101)에는, 예를 들면, MPEG(Moving Picture Experts Group)2의 규격에 준거한 프로그램 스트림(MPEG2-System Program Stream)이 기록되어 있으며, 버퍼(215A)에는, 광 디스크(101)로부터 읽어내어진 프로그램 스트림이 기억된다. 이 프로그램 스트림은, 비디오 스트림이나, 오디오 스트림, 자막 스트림 등의 1 이상의 엘리멘터리 스트림이 시분할 다중되어 있다. 비디오 읽어내기 기능부(233)는, 프로그램 스트림의 디멀티플렉스의 기능을 갖고, 버퍼(215A)에 기억된 프로그램 스트림으로부터, 비디오 스트림을 분리하여 읽어낸다.

[0329] 마찬가지로, 오디오 읽어내기 기능부(234)도, 프로그램 스트림의 디멀티플렉스의 기능을 갖고, 버퍼(215A)에 기억된 프로그램 스트림으로부터, 오디오 스트림을 분리하여 읽어낸다. 자막 읽어내기 기능부(235)도, 프로그램 스트림의 디멀티플렉스의 기능을 갖고, 버퍼(215A)에 기억된 프로그램 스트림으로부터, 자막 스트림을 분리하여 읽어낸다.

[0330] 여기에서, 비디오 읽어내기 기능부(233)는, 도 1의 메모리(113)의 일부인 비디오 읽어내기 포인터 기억부

(241), stream_id 레지스터(242), 및 au_information() 레지스터(243)를 가지고 있다.

[0331] 비디오 읽어내기 포인터 기억부(241)는, 버퍼(215A)의, 비디오 스트림이 기억된 위치(어드레스)를 가리키는 비디오 읽어내기 포인터를 기억하고, 비디오 읽어내기 기능부(233)는, 버퍼(215A)의, 비디오 읽어내기 포인터가 가리키는 위치에 기억되어 있는 데이터를, 비디오 스트림으로서 읽어낸다. stream_id 레지스터(242)는, 버퍼(215A)에 기억된 프로그램 스트림을 해석하고, 그 프로그램 스트림 중에서 읽어내는 비디오 스트림을 식별(특정)하기 위한 후술하는 stream_id를 기억한다. au_information() 레지스터(243)는, 버퍼(215A)로부터 비디오 스트림을 읽어내기 위하여 필요한 (비디오 스트림의 읽어내기에 이용되는) 데이터인 후술하는 au_information()을 기억한다.

[0332] 오디오 읽어내기 기능부(234)는, 도 1의 메모리(113)의 일부인 오디오 읽어내기 포인터 기억부(251), stream_id 레지스터(252), 및 private_stream_id 레지스터(253)를 가지고 있다.

[0333] 오디오 읽어내기 포인터 기억부(251)는, 버퍼(215A)의, 오디오 스트림이 기억된 위치(어드레스)를 가리키는 오디오 읽어내기 포인터를 기억하고, 오디오 읽어내기 기능부(234)는, 버퍼(215A)의, 오디오 읽어내기 포인터가 가리키는 위치에 기억되어 있는 데이터를, 오디오 스트림으로서 읽어낸다. stream_id 레지스터(252)와 private_stream_id 레지스터(253)는, 버퍼(215A)에 기억된 프로그램 스트림을 해석하여, 그 프로그램 스트림 중에서 읽어내는 오디오 스트림을 식별하기 위한 후술하는 stream_id와 private_stream_id를, 각각 기억한다.

[0334] 자막 읽어내기 기능부(235)는, 도 1의 메모리(113)의 일부인 자막 읽어내기 기능 플래그 기억부(261), 자막 읽어내기 포인터 기억부(262), stream_id 레지스터(263), 및 private_stream_id 레지스터(264)를 가지고 있다.

[0335] 자막 읽어내기 기능 플래그 기억부(261)는, 자막 읽어내기 기능 플래그를 기억한다. 자막 읽어내기 기능 플래그 기억부(261)에 기억된 자막 읽어내기 기능 플래그가, 예를 들면 0인 경우, 자막 읽어내기 기능부(235)는 기능 동작하지 않고, 자막 읽어내기 기능 플래그 기억부(261)에 기억된 자막 읽어내기 기능 플래그가, 예를 들면 1인 경우, 자막 읽어내기 기능부(235)는 기능한다.

[0336] 자막 읽어내기 포인터 기억부(262)는, 버퍼(215A)의, 자막 스트림이 기억된 위치(어드레스)를 가리키는 자막 읽어내기 포인터를 기억하고, 자막 읽어내기 기능부(235)는, 버퍼(215A)의, 자막 읽어내기 포인터가 가리키는 위치에 기억되어 있는 데이터를, 자막 스트림으로서 읽어낸다. stream_id 레지스터(263)와 private_stream_id 레지스터(264)는, 버퍼(215A)에 기억된 프로그램 스트림을 해석하고, 그 프로그램 스트림 중에서 읽어내는 자막 스트림을 식별하기 위한 후술하는 stream_id와 private_stream_id를 각각 기억한다.

[0337] [디스크(101)에 기록된 데이터의 데이터 포맷의 설명]

[0338] 다음으로, 디스크(101)에 기록된 데이터의 데이터 포맷에 대하여 설명한다.

[0339] 도 4는, 디스크(101)의 디렉토리 구조를 모식적으로 도시하고 있다.

[0340] 디스크(101)의 파일 시스템으로서는, 예를 들면, ISO(International Organization for Standardization)-9660이나, UDF(Universal Disk Format:http://www.osta.org/specs/) 등에서 규정된 파일 시스템이 이용되고 있으며, 디스크(101)에 기록된 데이터의 파일은 디렉토리 구조에 의해 계층적으로 관리되고 있다. 여기에서, 파일 시스템은, 전술한 파일 시스템에 한정되는 것은 아니다.

[0341] 도 4에서는, 파일 시스템의 기점을 나타내는 루트(root) 디렉토리에, "VIDEO" 디렉토리가 놓이고, "VIDEO" 디렉토리에는, "CLIP" 디렉토리와, "STREAM" 디렉토리의 2개의 디렉토리가 놓여 있다.

[0342] "VIDEO" 디렉토리에는, "CLIP" 디렉토리와 "STREAM" 디렉토리의 2개의 디렉토리 외에, "SCRIPT.DAT" 파일과, "PLAYLIST.DAT" 파일의 2개의 데이터 파일이 놓여 있다.

[0343] "SCRIPT.DAT" 파일은, 스크립트 프로그램이 기술된 스크립 파일이다. 즉, "SCRIPT.DAT" 파일에는, 디스크(101)의 재생 형태를 인터랙티브한 것으로 하기 위해 사용하는 스크립트 프로그램이 기술되어 있다. 이 "SCRIPT.DAT" 파일에 기술된 스크립트 프로그램은, 도 2A 및 도 2B의 스크립트 제어 모듈(211)에 의해 해석, 실행된다.

[0344] "PLAYLIST.DAT" 파일에는, 디스크(101)에 기록된 비디오 데이터 등의 콘텐츠의 재생 수순이 기술된 플레이 리스트(후술하는 도 5의 PlayList())가 1 이상 저장되어 있다.

[0345] "CLIP" 디렉토리에는, 1 이상의 클립 정보 파일이 놓이고, "STREAM" 디렉토리에는, 1 이상의 클립 스트림 파일이 놓인다. 즉, 도 4에서는, "CLIP" 디렉토리에는, 3개의 클립 정보 파일 "00001.CLP", "00002.CLP", "00003.CLP"이 놓여 있어, "STREAM" 디렉토리에는, 3개의 클립 스트림 파일 "00001.PS", "00002.PS", "00003.PS"가 놓여 있다.

[0346] 클립 스트림 파일에는, 비디오 데이터, 오디오 데이터, 자막 데이터 등의 1 이상의 데이터(스트림)를 압축, 부호화하여 얻어지는 1 이상의 엘리멘터리 스트림을 시분할 다중화한 프로그램 스트림이 저장되어 있다.

[0347] 클립 정보 파일에는, 대응하는 클립 스트림 파일의 성질 등의, 클립 스트림에 관한 (파일) 메타데이터가 기술되어 있다.

[0348] 즉, 클립 스트림 파일과 클립 정보 파일은, 1 대 1로 대응하고 있다. 도 4에서는, 클립 스트림 파일에는, 5문자의 숫자+피리오드+"PS"라고 하는 명명 규칙에 따라, 파일명이 붙여져 있고, 클립 정보 파일에는, 대응하는 클립 스트림 파일과 동일한 5문자의 숫자+피리오드+"CLP"이라고 하는 명명 규칙에 따라, 파일명이 붙여져 있다.

[0349] 따라서, 파일이, 클립 스트림 파일 또는 클립 정보 파일 중 어느 하나는, 파일명의 확장자(피리오드로부터 우측의 부분)에 의해 식별할 수 있으며, 또한, 대응하는 클립 스트림 파일과 클립 정보 파일은, 파일명의 확장자 이외의 부분(피리오드로부터 좌측의 부분)이 일치하는지의 여부에 의해 식별할 수 있다.

[0350] 이하, 디스크(101)에 기록된 각 파일의 상세에 대하여 설명한다.

[0351] 「PLAYLIST.DAT 」

[0352] 도 5는, 도 4의 "VIDEO" 디렉토리 아래의 "PLAYLIST.DAT" 파일의 내부 구조(신택스(syntax))를 도시하고 있다.

[0353] 여기에서, 도 5에서, "Syntax"의 란의 기재가 데이터 구조를 나타내고, "No.of bits"의 란의 기재는, 대응하는 행의 "Syntax"의 란의 데이터의 비트 길이를 나타낸다. 또한, "Mnemonic"의 란의 기재 중의 "bslbf"(bit string left bit first)는, 대응하는 행의 "Syntax"의 란의 데이터가 좌의 비트로부터 송출되는 것을 의미하고, "uimsbf"(unsigned integer most significant bit first)는, 대응하는 행의 "Syntax"의 란의 데이터가, 부호 없는 정수값이며, 최상위 비트부터 송출되는 것을 의미한다. 이하 설명하는, 도 5와 마찬가지의 도면에 대해서도, 마찬가지이다.

[0354] "PLAYLIST.DAT" 파일에서는, 그 선두부터, 그 명칭(파일명) 등의 정보를 기술하기 위한 name_length(8비트)와 name_string(255바이트)이 순차적으로 배치된다.

[0355] 즉, name_length는, 그 후에 배치되는 name_string의 사이즈를, 바이트수로 나타낸다. name_string은, "PLAYLIST.DAT" 파일의 명칭(파일명)을 나타낸다.

[0356] 또한, name_string에 대해서는, 그 선두부터, name_length로 나타내는 바이트 수까지가 유효한 명칭으로서 사용된다. 예를 들면 name_length가 값 10인 경우에는, name_string의 선두로부터 10바이트분이 유효한 명칭으로서 해석된다.

[0357] name_string 후에는, number_of_PlayLists(16비트)가 배치된다. number_of_PlayLists는, 계속되는 PlayList()의 개수를 나타낸다. number_of_PlayLists 후에, 그 number_of_PlayLists의 수만큼의 PlayList()이 배치된다.

[0358] PlayList()는, 디스크(101)에 기록된 클립 스트림 파일의 재생 수순이 기술된 플레이 리스트이며, 이하와 같은 내부 구조를 가진다.

[0359] 즉, PlayList()의 선두에는, PlayList_data_length(32비트)가 배치된다. PlayList_data_length은, 그 PlayList()의 사이즈를 나타낸다.

[0360] PlayList_data_length 후에는, reserved_for_word_alignment(15비트)와 capture_enable_flag_PlayList(1비트)가 순차적으로 배치된다. 15 비트의 reserved_for_word_alignment는, 그 후에 배치되는 1비트의 capture_enable_flag_PlayList의 위치에서, 소위 워드 얼라인먼트(word alignment)를 취하기 위하여(16비트의 위치에 일치시키기 위하여) 배치된다. capture_enable_flag_PlayList는, PlayList()에 의해 재생되는 비디오 스트림에 대응하는 비디오 데이터(PlayList()에 속하는 비디오 데이터)의, 광 디스크(101)가 재생되는 도 1의 디스크 장치 내에서의 2차 이용이 허가되는지의 여부를 나타내는 1비트의 플래그이다.

capture_enable_flag_PlayList가, 0 또는 1 중, 예를 들면 1인 경우, PlayList()에 속하는 비디오 데이터의 2차 이용이 허가되어 있는 것을 나타내고, capture_enable_flag_PlayList가, 0 또는 1 중, 예를 들면 0인 경우, PlayList()에 속하는 비디오 데이터의 2차 이용이 허가되어 있지 않은(금지되어 있는) 것을 나타낸다.

[0361] 또한, 도 5에서는, capture_enable_flag_PlayList를 1비트라고 했지만, 기타, capture_enable_flag_PlayList는, 복수 비트로 구성하고, PlayList()에 속하는 비디오 데이터의 2차 이용을, 말하자면 단계적으로 허가하도록 하는 것이 가능하다. 즉, capture_enable_flag_PlayList는, 예를 들면, 2비트로 구성할 수 있다. 그리고, capture_enable_flag_PlayList의 값이 00B(B은, 그 전의 숫자가 2진수인 것을 나타냄)인 경우에는, 비디오 데이터의 2차 이용을 금지하고, capture_enable_flag_PlayList의 값이 01B인 경우에는, 비디오 데이터를, 64×64 픽셀 이하의 사이즈로 축소하여 이용하는 2차 이용만을 허가할 수 있다. 또한, capture_enable_flag_PlayList의 값이 10B인 경우에는, 사이즈의 제한 없음으로, 비디오 데이터의 2차 이용을 허가할 수 있다.

[0362] 또한, 전술된 바와 같이, 비디오 데이터의 2차 이용에 있어서, 사이즈에 제한을 설정하는 것은 아니고, 용도에 제한을 설정하도록 하는 것도 가능하다. 즉, capture_enable_flag_PlayList의 값이 01B인 경우에는, 비디오 콘텐츠 재생 어플리케이션(210)(도 2A 및 도 2B)만으로의 2차 이용을 허가하고, capture_enable_flag_PlayList의 값이 10B인 경우에는, 도 1의 디스크 장치 내의, 비디오 콘텐츠 재생 어플리케이션(210)을 포함하는 임의의 어플리케이션에 의한 2차 이용을 허가할 수 있다. 여기에서, 도 1의 디스크 장치 내의 비디오 콘텐츠 재생 어플리케이션(210) 이외의 어플리케이션으로서는, 예를 들면, 벽지(백그라운드)나 스크린 세이버의 표시의 처리를 행하는 어플리케이션 등이 있다.

[0363] 또한, capture_enable_flag_PlayList를, 전술된 바와 같이, 2비트라고 한 경우, 그 전에 배치되는 reserved_for_word_alignment는, 워드 얼라인먼트를 취하기 위해서, 14비트로 된다.

[0364] 또한, capture_enable_flag_PlayList에 의해, 비디오 데이터의 디스크 장치내에서의 2차 이용을 허가하는 것 외에, 디스크 장치 외에서의 2차 이용을 허가하도록 하는 것도 가능하다. 여기에서, 비디오 데이터의, 디스크 장치 외에서의 2차 이용을 허가하는 경우에는, 비디오 데이터는, 예를 들면, 디스크 장치에 착탈 가능한 기록 매체나 디스크 장치에 접속 가능한 다른 장치에 착탈 가능한 기록 매체에 기록되거나, 혹은 인터넷 등의 네트워크를 통하여, 다른 장치에 송신(배신)된다. 이 경우, 비디오 데이터에는, 그 비디오 데이터를 기록 매체에 기록하는 횟수나 배신하는 횟수를 제한하는 정보를 부가하도록 할 수 있다.

[0365] capture_enable_flag_PlayList에 이어서는, PlayList_name_length(8비트)와 PlayList_name_string(255바이트)가 순차적으로 배치된다. PlayList_name_length 는, 그 후에 배치되는 PlayList_name_string의 사이즈를, 바이트 수로 나타내고, PlayListname_string은, PlayList()의 명칭을 나타낸다.

[0366] PlayList_name_string 후에는, number_of_PlayItems(16비트)가 배치된다. numberofPlayItems은, 계속되는 PlayItem()의 개수를 나타낸다.

[0367] number_of_PlayItems 후에는, 그 number_of_PlayItems의 수만큼의 PlayItem()의 구조가 기술된다.

[0368] 여기에서, 1개의 PlayList()에서는, PlayItem() 단위로, 콘텐츠의 재생 수순을 기술할 수 있다.

[0369] 또한, PlayList() 중, number_of_PlayItems의 수만큼의 PlayItem() 각각에 대해서는, 그 PlayList() 중에서 유니크한 ID(Identification)가 붙여진다. 즉, PlayList() 중 최초의 PlayItem()에는, ID로서 0번이 붙여지고, 이하, 계속되는 PlayItem()에 대하여, 그 출현순으로, 1번, 2번, …로 일련 번호의 ID가 붙여진다.

[0370] number_of_PlayItems의 수만큼의 PlayItem() 후에는, 1개의 PlayListMark()이 배치된다. PlayListMark()는, PlayList()에 따라 행해지는 재생의 시간축 상의 표시로 되는 후술하는 Mark()의 집합으로서, 그 상세에 대해서는, 도 7면을 참조하여 후술한다.

[0371] 「PlayItem()의 설명」

[0372] 다음으로, 도 6은, 도 5의 PlayList()에 포함되는 PlayItem()의 내부 구조를 나타내고 있다.

[0373] PlayItem()의 선두에는, length(16비트)가 배치되고, length는, 그것을 포함하는 PlayItem()의 사이즈를 나타낸다.

[0374] length에 이어서는, Clip_Information_file_name_length(16비트)와 Clip_Information_file_name(가변 길이)이 순차적으로 배치된다. Clip_Information_file_name_length는, 그 후에 배치되는 Clip_Information_file_name

의 사이즈를, 바이트 수로 나타낸다. Clip_Information_file_name은, PlayItem()에 의해 재생하는 클립 스트림 파일(도 4의 확장자가 PS의 파일)에 대응하는 클립 정보 파일(도 4의 확장자가 CLP의 파일)의 파일명을 나타낸다. 또한, 클립 스트림 파일 및 클립 정보 파일의 파일명의, 전술한 명명 규칙에 의해, Clip_Information_file_name으로부터, PlayItem()에 의해 재생하는 클립 정보 파일의 파일명을 인식하고, 그 클립 스트림 파일을 특정할 수 있다.

[0375] Clip_Information_file_name에 이어서는, IN_time(32비트)과 OUT_time(32비트)이 순차적으로 배치된다.

[0376] IN_time 과 OUT_time은, 각각, Clip_Information_file_name으로부터 특정되는 클립 스트림 파일의 재생 개시 위치와 재생 종료 위치를 지정하는 시각 정보이다.

[0377] IN_time에 의하면, 클립 스트림 파일의 (선두를 포함함) 도중의 위치를 재생 개시 위치로서 지정할 수 있으며, OUT_time에 의하면, 클립 스트림 파일의 (최후를 포함함) 도중의 위치를 재생 종료 위치로서 지정할 수 있다.

[0378] 여기에서, PlayItem()에 의하면, Clip_Information_file_name으로부터 특정되는 클립 스트림 파일의, IN_time으로부터 OUT_time 까지의 사이의 콘텐츠가 재생된다. 이 PlayItem()에 의해 재생되는 콘텐츠를, 이하, 적절하게, 클립이라고 한다.

[0379] 「PlayListMark()의 설명」

[0380] 다음으로, 도 7은, 도 5의 PlayList0에 포함되는 PlayListMark()의 내부 구조를 도시하고 있다.

[0381] PlayListMark()는, 전술한 바와 같이, 그 PlayListMark()를 포함하는 PlayList()(도 5)에 따라 행해지는 재생의 시간축 상의 표시로 되는, 0 이상의 Mark()의 집합이다. 1 개의 Mark()는, PlayList()에 따라 행해지는 재생의 시간축 상의 1개의 시각(위치)을 나타내는 시각 정보, Mark()의 타입을 나타내는 타입 정보, 및 타입 정보가 이벤트를 발생시키는 타입을 나타내고 있을 때의, 그 이벤트의 인수로 되는 인수 정보를, 적어도 가진다.

[0382] 즉, PlayListMark()의 선두에는, length(32비트)가 배치된다. length는, 그것을 포함하는 PlayListMark()의 사이즈를 나타낸다.

[0383] length 후에는, number_of_PlayList_marks(16비트)가 배치되고, number_of_PlayList_marks는, 그에 후속하여 배치되는 Mark()의 개수를 나타낸다. number_of_PlayList_marks 후에는, 그 number_of_PlayList_marks의 수만큼 Mark()의 구조가 기술된다.

[0384] Mark()의 선두에는, mark_type(8비트)이 배치된다. mark_type은, 전술한 타입 정보로서, 그것을 포함하는 Mark()의 타입을 나타낸다.

[0385] 본 실시예에서는, Mark()의 타입으로서, 예를 들면, 챕터(Chapter), 인덱스(Index), 이벤트(Event)의 3종류가 준비되어 있다.

[0386] 타입이 챕터의 Mark()(이하, 적절하게, 챕터 마크라고 함)는, PlayList()를 분할하는 첫머리 탐색의 단위인 챕터의 선두 위치의 표시이다. 또한, 타입이 인덱스의 Mark()(이하, 적절하게, 인덱스 마크라고 함)는, 챕터를 세분화한 단위인 인덱스의 선두 위치의 표시이다. 타입이 이벤트의 Mark()(이하, 적절하게, 이벤트 마크라고 함)는, PlayList()에 따른 콘텐츠의 재생 중에 있어서 이벤트를 발생시키는 위치의 표시이다. 이벤트 마크에 의한 이벤트의 발생은, 후술하는 바와 같이, 스크립트 제어 모듈(211)에 통지된다.

[0387] 여기에서, mark_type의 값과, Mark()의 타입과의 관계를, 도 8에 도시한다. 도 8에 따르면, 챕터 마크의 mark_type에는, 1이 세트된다. 또한, 인덱스 마크의 mark_type에는, 2가 세트되고, 이벤트 마크의 mark_type에는, 3이 세트된다. 또한, 도 8에서는, mark_type으로 나타내는 8비트의 값 중, 0과, 4 내지 255는, 장래의 확장을 위한 예약(reserved)으로 되어 있다.

[0388] 도 7로 되돌아가서, mark_type 후에는, mark_name_length(8비트)가 배치된다. 또한, Mark()의 마지막에는, markname_string(24바이트)이 배치된다. mark_name_length와 mark_name_string은, Mark()의 명칭을 기술하기 위한 것이며, mark_name_length는, mark_name_string의 유효한 사이즈를, mark_name_string은, Mark()의 명칭을, 각각 나타낸다. 따라서, mark_name_string의 선두로부터 mark_name_length가 나타내는 바이트 수까지, Mark()의 유효한 명칭을 나타낸다.

[0389] mark_name_length에 이어서는, PlayList() 상에서 정의되는 Mark()를 클립 스트림 파일과 대응지어지는 4개의

요소 ref_to_PlayItem_id(16비트), mark_time_stamp(32비트), entry_ES_stream_id(8비트)가 순차적으로 배치된다.

[0390] ref_to_PlayItem_id에는, Mark()가 속하는 PlayItem()에 대하여 일련 번호로 붙여진 ID가 기술된다. ref_to_PlayItem_id에 의해, Mark()가 속하는 PlayItem()(도 6)이 특정되고, 나아가서는, 도 6에서 설명한 바와 같이, 클립 정보 파일과 클립 스트림 파일이 특정된다.

[0391] mark_time_stamp은, ref_to_PlayItem_id에 의해 특정되는 클립 스트림 파일내에서의 Mark()이 나타내는 위치(시각)를 나타낸다.

[0392] 여기에서, 도 9는, PlayList(),PlayItem(), 클립, 및 클립 스트림 파일에 저장된 프로그램 스트림의 관계를 도시하고 있다.

[0393] 도 9에서는, PlayList()는, 3개의 PlayItem()으로 구성되어 있고, 그 3개의 PlayItem() 각각에는, 일련 번호로 붙여지는 ID#0, #1, #2가 붙여져 있다. 여기에서, 이하, 적절하게, ID#i가 붙여진 PlayItem()을, PlayItem#i라고 기술한다.

[0394] 또한, 도 9에서는, PlayItem#0, PlayItem#1, PlayItem#2에 의해 재생되는 콘텐츠인 클립이, 각각, 클립A, 클립B, 클립C로서 기재되어 있다.

[0395] 클립의 실체는, 도 6의 PlayItem()에서의 Clip_Information_file_name으로부터 특정되는(클립 정보 파일로부터, 더욱 특정되는) 클립 스트림 파일에 저장된 프로그램 스트림 중의, IN_time부터 OUT_time까지의 프로그램 스트림이다. 도 9에서는, 클립A, 클립B, 클립C의 실체로서의 프로그램 스트림이, 프로그램 스트림 A, 프로그램 스트림 B, 프로그램 스트림 C로서, 각각 기재되어 있다.

[0396] 예를 들면, 도 9에서, PlayList()에 따라 행해지는 재생의 시간축 상의 위치(시각)t0의 표시로 되는 Mark()에서는, 그 ref_to_PlayItem_id와 mark_time_stamp은, 다음과 같이 기술된다.

[0397] 즉, 시각 t0은, PlayItem#1의 재생이 행해지는 시각이기 때문에, ref_to_PlayItem_id에는, 그 PlayItem#1의 ID인 1이 기술된다.

[0398] 재차, 시각 t0에서는, PlayItem#1에 의해, 클립B의 실체인 프로그램 스트림 B가 재생되기 때문에, mark_time_stamp에는, 프로그램 스트림B가 저장된 클립 스트림 파일에서의 시각 t0에 상당하는 시각이 기술된다.

[0399] 또한, 도 7로 되돌아가서, entry_ES_stream_id와, entry_ES_private_stream_id는, Mark()를, 특정한 엘리멘터리 스트림에 관련지은 경우에, 그 엘리멘터리 스트림을 특정하기 위하여 사용된다. 즉, entry_ES_stream_id에는, Mark()를 관련지은 엘리멘터리 스트림(이 배치되는, 후술하는 도 16A 및 도 16B 내지 도 18A 및 도 18B에 도시하는 PES_packet())의 후술하는 stream_id가 기술된다. 또한, entry_ES_private_stream_id에는, 필요에 따라, Mark()를 관련지은 엘리멘터리 스트림(이 배치되는, 후술하는 도 21에 도시하는 private_stream1_PES_payload()에서의 private_header())의 후술하는 private_stream_id가 기술된다.

[0400] 예를 들면, 비디오 스트림#1과 비디오 스트림#2가 다중화되어 있는 클립에서, 비디오 스트림#1을 재생하고 있는 경우와, 비디오 스트림#2를 재생하고 있는 경우에서 챕터의 발생 시각을 변경하려고 할 때에는, 비디오 스트림#1 재생 시의 챕터 마크 발생 시각의 Mark()의 entry_ES_stream_id와 entry_ES_private_stream_id에, 비디오 스트림#1의 stream_id와 private_stream_id가 기술되고, 또한, 비디오 스트림#2 재생 시의 챕터 마크 발생 시각의 Mark()의 entry_ES_stream_id와 entry_ES_private_stream_id에, 비디오 스트림#2의 stream_id와 private_stream_id가 기술된다.

[0401] 또한, 특정한 엘리멘터리 스트림에 관련짓지 않은 Mark()의 entry_ES_stream_id와, entry_ES_private_stream_id에는, 예를 들면, 모두 0이 기술된다.

[0402] entry_ES_private_stream_id 후에는, mark_data(32비트)가 배치된다. mark_data는, Mark()가 이벤트 마크인 경우에, 그 이벤트 마크에 의해 발생되는 이벤트의 인수로 되는 인수 정보이다. 또한, mark_data는, Mark()가 챕터 마크나 인덱스 마크인 경우에, 그 챕터 마크나 인덱스 마크가 나타내는 칩터나 인덱스의 번호로서 사용하는 것도 가능하다.

[0403] 「Clip()의 설명」

[0404] 다음으로, 도 4의 "CLIP" 디렉토리에 놓인, 확장자가 CLP인 클립 정보 파일의 내부 구조에 대하여 설명한다.

[0405] 도 4에서는, "CLIP" 디렉토리에, 3개의 클립 정보 파일 "00001.CLP", "00002.CLP", "00003.CLP"이 놓여 있고, 각각에는, "STREAM" 디렉토리에 놓인 클립 스트림 파일 "00001.PS", "00002.PS", "00003.PS"의 성질 등을 나타내는 메타데이터가 저장되어 있다.

[0406] 도 10은, 그러한 클립 정보 파일 Clip()의 내부 구조를 도시하고 있다.

[0407] 클립 정보 파일 Clip()의 선두에는, presentation_start_time과 presentation_end_time(모두 32비트)이, 순차적으로 배치된다. presentation_start_time과 presentation_end_time은, 클립 정보 파일 Clip()에 대응하는 클립 스트림 파일(에 저장되어 있는 프로그램 스트림)의 선두와 최후의 시각을 나타낸다. 또한, 클립 스트림 파일의 시각은, MPEG2-System의 시각에서 사용되고 있는 90㎑의 배수로 기술된다.

[0408] presentation_end_time에 이어서는, reserved_for_word_alignment(7비트)와 capture_enable_flag_Clip(1비트)이 순차적으로 배치된다. 7비트의 reserved_for_word_alignment는, 워드 얼라인먼트를 취하기 위한 것으로, capture_enable_flag_Clip은, 전술한 도 5의 capture_enable_flag_PlayList와 마찬가지로, 비디오 데이터의 2차 이용을 허가하는지의 여부를 나타내는 플래그이다.

[0409] 단, 도 5의 capture_enable_flag_PlayList는, PlayList()에 의해 재생되는 비디오 스트림에 대응하는 비디오 데이터(PlayList()에 속하는 비디오 데이터)의 2차 이용을 허가하는지의 여부를 나타내는 것에 대해서, 도 10의 capture_enable_flag_Clip은, 클립 정보 파일 Clip()에 대응하는 클립 스트림 파일에 저장되어 있는 비디오 스트림(비디오의 엘리멘터리 스트림)에 대응하는 비디오 데이터의 2차 이용을 허가하는지의 여부를 나타낸다. 따라서, 도 5의 capture_enable_flag_PlayList와, 도 10의 capture_enable_flag_Clip에서는, 2차 이용을 허가하는 비디오 데이터의 단위(범위)가 상이하다.

[0410] 또한, 도 10의 capture_enable_flag_Clip도, 도 5의 capture_enable_flag_PlayList에서 설명한 바와 같이, 1비트가 아니고, 복수 비트라고 하는 것이 가능하다.

[0411] capture_enable_flag_Clip 후에는, number_of_streams(8비트)가 배치되고, 이 number_of_streams에는, 그에 후속하는 StreamInfo() 구조의 개수가 기술된다. 따라서, number_of_streams에 이어서는, 그 number_of_Streams의 수만큼, StreamInfo()의 구조가 기술된다.

[0412] StreamInfo()의 선두에는, length(16비트)이 배치되고, 이 length는, 그것을 포함하는 StreamInfo()의 사이즈를 나타낸다. length에 이어서는, stream_id(8비트)와 private_stream_id(8비트)가 배치되어 있고, 이 stream_id와 private_stream_id에 의해, StreamInfo()에 관련지은 엘리멘터리 스트림이 특정(식별)된다.

[0413] 여기에서, 도 11은, 엘리멘터리 스트림을 식별하는 stream_id 및 private_stream_id와, 엘리멘터리 스트림과의 관계를 나타내고 있다.

[0414] stream_id는, MPEG2-System 규격에서 규정되어 있는 것과 동일한 것으로, 그 값은, MPEG2-System 규격에서, 엘리멘터리 스트림(데이터)의 속성(종류)마다, 미리 정해져 있다. 따라서, MPEG2-System 규격에서 정의되어 있는 속성의 엘리멘터리 스트림은, stream_id만으로 특정할 수 있다.

[0415] 본 실시예에서는, MPEG2-System 규격에서 규정되어 있지 않은 속성의 엘리멘터리 스트림도 취급하는 것이 가능하게 되어 있으며, private_stream_id는, MPEG2-System 규격에서 규정되어 있지 않은 속성의 엘리멘터리 스트림을 식별하기 위한 정보이다.

[0416] 도 11에서는, MPEG로 규정되어 있는 부호화(복호) 방식으로 인코드된 비디오의 엘리멘터리 스트림, ATRAC(Adaptive TRansform Acoustic Coding) 방식으로 인코드된 오디오의 엘리멘터리 스트림(이하, 적절하게, ATRAC 오디오 스트림이라고 함), LPCM(Linear Pulse Code Modulation) 방식으로 인코드된 오디오의 엘리멘터리 스트림(이하, 적절하게, LPCM 오디오 스트림이라고 함), 자막의 엘리멘터리 스트림(이하, 적절하게, 자막 스트림이라고 함)의 4개의 속성의 엘리멘터리 스트림에 대해서, stream_id 및 private_stream_id와의 관계를 나타내고 있다.

[0417] MPEG2-System 규격에서는, MPEG로 규정되어 있는 부호화 방식으로 인코드된 비디오의 엘리멘터리 스트림은, 0xE0 내지 0xEF의 범위의 값을 (0x는, 그에 후속하는 문자열이 16진수인 것을 나타냄), 엘리멘터리 스트림을 식별하는 stream_id로서 이용하여 다중화하는 것이 규정되어 있다. 따라서, MPEG로 규정되어 있는 부호화 방식으로 인코드된 비디오의 엘리멘터리 스트림에 대해서는, 0xE0 내지 0xEF의 범위의 값의 stream_id로 식별할 수 있는 16개의 비디오의 엘리멘터리 스트림을, 프로그램 스트림에 다중화할 수 있다.

[0418] 또한, MPEG로 규정되어 있는 부호화 방식으로 인코드된 비디오의 엘리멘터리 스트림의 식별은, 0xE0 내지

0xEF의 범위의 값의 stream_id로 행할 수 있으므로, private_stream_id는 불필요하다(무시할 수 있다).

[0419] 한편, MPEG2-System에서는, ATRAC 오디오 스트림, LPCM 오디오 스트림, 자막 스트림에 대해서는, stream_id는 정의되어 있지 않다.

[0420] 따라서, 본 실시예에서는, MPEG2-System에서 stream_id가 정의되어 있지 않은 엘리멘터리 스트림에 대해서는, 그 stream_id에, MPEG2-System에서 private_stream_1이라고 하는 속성을 나타내는 값인 0xBD를 채용하고, 또한 도 11에 도시한 바와 같이, private_stream_id를 이용하여 식별(특정)을 행하는 것으로 하고 있다.

[0421] 즉, ATRAC 오디오 스트림의 식별에는, 0x00 내지 0x0F의 범위의 값의 private_stream_id가 사용된다. 따라서, 프로그램 스트림에는, 16개의 ATRAC 오디오 스트림을 다중화할 수 있다. 또한, LPCM 오디오 스트림의 식별에는, 0x10 내지 0x1F의 범위의 값 private_stream_id가 사용된다. 따라서, 프로그램 스트림에는, 16개의 LPCM 오디오 스트림을 다중화할 수 있다. 또한, 자막 스트림의 식별에는, 0x80 내지 0x9F의 범위의 값의 private_stream_id가 사용된다. 따라서, 프로그램 스트림에는, 32개의 자막 스트림을 다중화할 수 있다.

[0422] 또한, stream_id 및 privatestream_id에 대해서는, 더욱 후술한다.

[0423] 도 10으로 되돌아가서, private_stream_id 후에는, StaticInfo(), reserved_for_word_alignment(8비트)가 순차적으로 배치된다. StaticInfo()에는, (그 StaticInfo()를 포함하는 StreamInfo()에 기술된) stream_id 및 private_stream_id에 의해 특정되는 엘리멘터리 스트림의 재생 중에 변화하지 않는 정보가 기술된다. StaticInfo()의 상세에 대해서는, 도 12를 참조하여 후술한다.

[0424] reserved_for_word_alignment는, 워드 얼라인먼트를 취하기 위해 사용된다.

[0425] reserved_for_word_alignment에 이어서는, number_of_DynamicInfo(8비트)가 배치되고, number_of_DynamicInfo는, 그에 후속하여 배치되는 pts_change_point(32 비트)와 DynamicInfo()의 세트의 수를 나타낸다.

[0426] 따라서, number_of_DynamicInfo에 이어서는, 그 number_of_DynamicInfo의 수만큼의 세트수의 pts_change_point와 DynamicInfo()의 구조가 기술된다.

[0427] pts_change_point는, 그것과 세트로 되어 있는 DynamicInfo()의 정보가 유효해지는 시각을 나타낸다. 여기에서, 엘리멘터리 스트림의 선두의 시각을 나타내는 pts_change_point는, 그 엘리멘터리 스트림이 저장된 클립 스트림 파일에 대응하는 클립 정보 파일 Clip()의 처음에 기술되는 presentation_start_time과 동일하다.

[0428] DynamicInfo()에는, stream_id 및 private_stream_id에 의해 특정되는 엘리멘터리 스트림의 재생 중에 변화하는, 말하자면 동적인 정보가 기술된다. DynamicInfo()에 기술된 정보는, 그것과 세트로 되어 있는 pts_change_point가 나타내는 재생 시각으로 되었을 때에 유효로 된다. 또한, DynamicInfo()의 상세 내용에 대해서는, 도 13을 참조하여 후술한다.

[0429] number_of_DynamicInfo의 수만큼의 세트의 pts_change_point와 DynamicInfo() 후에는, EP_map()이 배치된다. 또한, EP_map()에 대하여는, 도 14를 참조하여 후술한다.

[0430] 「StaticInfo()의 설명」

[0431] 다음으로, 도 12를 참조하여, 도 10의 StaticInfo()의 상세 내용에 대해서 설명한다.

[0432] 도 12는, StaticInfo()의 신택스를 도시하고 있다.

[0433] StaticInfo() 은, 대응하는 엘리멘터리 스트림의 속성(종류)에 따라 내용이 상이하다. StaticInfo()에 대응하는 엘리멘터리 스트림의 속성은, 그 StaticInfo()을 포함하는 도 10의 StreamInfo()에 포함되는 stream_id와 private_stream_id에 의해 판단된다.

[0434] StaticInfo()에 대응하는 엘리멘터리 스트림이 비디오 스트림인 경우 (stream==VIDEO), StaticInfo()은, picture_size(4비트), frame_rate(4비트), 및 cc_flag(1비트)과, 워드 얼라인을 취하기 위한 reserved_for_word_alingment으로 구성된다.

[0435] picture_size는, 비디오 스트림에 대응하는 비디오 데이터(에 의해 표시되는 화상)의 크기를 나타낸다. frame_rate는, 비디오 스트림에 대응하는 비디오 데이터의 프레임 주파수를 나타낸다. cc_flag는, 비디오 스트림에 클로즈드 캡션(Closed Caption) 데이터가 포함되어 있는지 여부를 나타낸다. 즉, 예를 들면, 비디오 스트림에 클로즈드 캡션 데이터가 포함되어 있는 경우에는, CC_flag는 1로 되고, 비디오 스트림에 클로즈드 캡션 데이터가 포함되어 있지 않은 경우에는, cc_flag는 0으로 된다.

[0436] StaticInfo()에 대응하는 엘리멘터리 스트림이 오디오 스트림인 경우 (stream==AUDIO), StaticInfo()은, audio_language_code(16비트), channel_configuration(8비트), lfe_existence(1비트), 및 sampling_frequency(4비트)과, 워드 얼라인을 취하기 위한 reserved_for_word_alingment으로 구성된다.

[0437] audio_language_code에는, 오디오 스트림에 포함되어 있는 오디오 데이터의 언어를 나타내는 코드가 기술된다. channel_configuration은, 모노럴(mono)/스테레오(stereo)/멀티채널 등의, 오디오 스트림에 포함되어 있는 오디오 데이터의 속성을 나타낸다. lfe_existence는, 오디오 스트림에 저역 강조 채널이 포함되어 있는지 여부를 나타내고, 포함되어 있으면 1로 되고, 포함되어 있지 않으면 0으로 된다. sampling_frequency는, 오디오 스트림에 포함되어 있는 오디오 데이터의 샘플링 주파수를 나타내는 정보이다.

[0438] StaticInfo()에 대응하는 엘리멘터리 스트림이 자막 스트림인 경우 (stream==SUBTITLE), StaticInfo()은, subtitle_language_code(16비트) 및 configurable_flag(1비트)과, 워드 얼라인을 취하기 위한 reserved_for_word_alingment으로 구성된다.

[0439] subtitle_language_code에는, 자막 스트림에 포함되어 있는 자막 데이터의 언어를 나타내는 코드가 기술된다. configurable_flag는, 자막 스트림에 포함되어 있는 자막 데이터의 표시를 디폴트의 표시 방식으로부터 변경하는 것을 허가하는지의 여부를 나타내는 정보로서, 예를 들면, 표시 방식의 변경이 허가되어 있는 경우에는 1이 기술되고, 허가되어 있지 않은 경우에는 0이 기술된다. 또한, 자막 데이터의 표시 방식으로서는, 자막 데이터의 표시 사이즈나, 표시 위치, 표시색, 표시 패턴(예를 들면, 점멸 표시 등), 표시 방향(예를 들면, 수직 방향이나 수평 방향) 등이 있다.

[0440] 「DynamicInfo()의 설명」

[0441] 다음으로, 도 13을 참조하여, 도 10의 DynamicInfo()의 상세 내용에 대해서 설명한다.

[0442] 도 13는, DynamicInfo()의 신택스를 도시하고 있다.

[0443] DynamicInfo()의 선두에는, 워드 얼라인을 위한 reserved_for_word_alignment(8비트)이 배치되어 있고, 그에 후속하는 요소는, DynamicInfo()에 대응하는 엘리멘터리 스트림의 속성에 따라 내용이 상이하다. DynamicInfo()에 대응하는 엘리멘터리 스트림의 속성은, 도 12에서 설명한 StaticInfo()에 있어서의 경우와 마찬가지로, DynamicInfo()을 포함하는 도 10의 StreamInfo()에 포함되는 stream_id와 private_stream_id에 의해 판단된다.

[0444] DynamicInfo()에는, 도 10에서 설명한 바와 같이, 엘리멘터리 스트림의 재생 중에 변화하는 동적인 정보가 기술된다. 이 동적인 정보는, 특별히 한정되는 것은 아니지만, 도 13의 실시 형태에서는, DynamicInfo()에 대응하는 엘리멘터리 스트림에 대응하는 데이터, 즉, 엘리멘터리 스트림이 처리됨으로써 출력되는 데이터의 출력 속성(엘리멘터리 스트림으로부터 얻어지는 데이터의 출력 속성)이, DynamicInfo()에 기술된다.

[0445] 구체적으로는, DynamicInfo()에 대응하는 엘리멘터리 스트림이 비디오 스트림인 경우(stream==VIDEO), DynamicInfo()은, display_aspect_ratio(4비트)와, 워드 얼라인을 위한 reserved_for_word_alingment로 구성된다. diplay_aspect_ratio에는, 비디오 스트림에 대응하는 비디오 데이터의 출력 속성(표시 방식)로서의, 예를 들면, 그 비디오 데이터의 어스펙트비가 기술된다. 즉, diplay_aspect_ratio에는, 예를 들면, 어스펙트비가, 16:9 또는 4:3 중의 어느 것인지를 나타내는 정보가 기술된다. 또한, 비디오 스트림의 DynamicInfo()에는, 어스펙트비 외, 예를 들면, 비디오 데이터에 의해 표시되는 화상의 사이즈(X화소×Y화소) 등을 기술하는 것이 가능하다.

[0446] DynamicInfo()에 대응하는 엘리멘터리 스트림이 오디오 스트림인 경우 (stream==AUDIO), DynamicInfo()은, channel_assignment(4비트)와, 워드 얼라인을 취하기 위한 reserved_for_word_alingment으로 구성된다. channel_assignment에는, 오디오 스트림에 2채널의 오디오 데이터가 포함되어 있는 경우에, 그 2채널의 출력 속성(출력 방식)이 기술된다. 즉, channel_assignment에는, 오디오 데이터가, 스테레오 또는 듀얼(2개 국어) 중의 어느 채널 할당이 되어 있는 것인지를 나타내는 정보가 기술된다.

[0447] DynamicInfo()에 대응하는 엘리멘터리 스트림이 자막 스트림인 경우 (stream==SUBTITLE), DynamicInfo()은, 워드 얼라인을 취하기 위한 reserved_for_word_alingment으로 구성된다. 즉, 도 13의 실시 형태에서는, 자막 스트림에 관해서는, 동적인 정보로서의 출력 속성은 정의되어 있지 않다.

[0448] 「EP_map()의 설명」

[0449] 다음으로, 도 14를 참조하여, 도 10의 EP_map()의 상세 내용에 대해서 설명한다.

[0450] 도 14는, EP_map()의 신택스를 도시하고 있다.

[0451] EP_map()에는, 그 EP_map()을 포함하는 도 10의 클립 정보 파일 Clip()에 대응하는 클립 스트림 파일에 저장된 프로그램 스트림에 다중화되어 있는 엘리멘터리 스트림마다, 각 엘리멘터리 스트림의, 디코드를 개시할 수 있는 디코드 개시 가능점(엔트리 포인트)의 정보가 기술된다.

[0452] 여기에서, 고정 레이트의 스트림에 대해서는, 디코드 개시 가능점은, 계산에 의해 구할 수 있지만, 가변 레이트의 스트림, 혹은 MPEG규격에 따라서 부호화된 비디오 스트림과 같이, 비디오 액세스 유닛마다에 사이즈가 서로 다른 스트림에 대해서는, 디코드 개시 가능점은, 계산에 의해 구하는 것은 불가능하고, 실제로 스트림을 해석하지 않으면 찾아낼 수 없다. 디코드 개시 가능점을 신속하게 인식하는 것은, 랜덤 액세스를 행하기 위해 중요하며, EP_map()에 따르면, 디코드 개시 가능점을 신속하게 인식할 수 있다.

[0453] 또한, MPEG2-Video에서는, Sequence_header()(시퀀스 헤더) 등을 포함시킨 인트라 픽처의 선두가, 디코드 개시 가능점이다.

[0454] EP_map()의 선두에는, 워드 얼라인을 위한 reserved_for_word_alignment(8비트)이 배치되어 있고, 계속해서 number_of_stream_id_entries(8비트)가 배치되어 있다. number_of_stream_id_entries는, EP_map()에 디코드 개시 가능점의 정보가 기술되어 있는 엘리멘터리 스트림의 개수를 나타낸다.

[0455] number_of_stream_id_entries 후에는, 엘리멘터리 스트림을 식별하기 위한 정보와, 그 엘리멘터리 스트림의 디코드 개시 가능점의 정보가, number_of_stream_id_entries가 나타내는 수만큼 반복해서 배치된다.

[0456] 즉, number_of_stream_id_entries의 직후에는, 엘리멘터리 스트림을 식별하는 정보로서의 stream_id(8비트)과 private_stream_id(8비트)이 배치되고, 그에 후속하여, number_of_EP_entries(32비트)가 배치된다. number_of_EP_entries는, 그 직전의 stream_id와 private_stream_id에 의해 식별(특정)되는 엘리멘터리 스트림의 디코드 개시 가능점의 수를 나타낸다.

[0457] number_of_EP_entries 후에는, 그 직전의 stream_id와 private_stream_id에 의해 특정되는 엘리멘터리 스트림의 디코드 개시 가능점의 정보로서의 PTS_EP_start(32비트)과 RPN_EP_start(32비트)의 세트가, number_of_EP_entries가 나타내는 수만큼 반복해서 배치된다.

[0458] 디코드 개시 가능점의 정보의 하나인 PTS_EP_start는, 전술된 바와 같이 stream_id와 private_stream_id에 의해 특정되는 엘리멘터리 스트림이 다중화되어 있는 프로그램 스트림이 저장된 클립 스트림 파일 내에서의, 디코드 개시 가능점의 시각(재생 시각)을 나타낸다.

[0459] 디코드 개시 가능점의 정보의 다른 하나인 RPN_EP_start에는, 전술된 바와 같이 stream_id와 private_stream_id에 의해 특정되는 엘리멘터리 스트림이 다중화되어 있는 프로그램 스트림이 저장된 클립 스트림 파일 내에서의, 디코드 개시 가능점의 위치를, 프로그램 스트림의 pack() 단위로 세었을 때의 값이 기술된다. 또한, 본 실시 형태에서는, pack()의 사이즈는 2048바이트로 고정인 것으로 한다. 또한, 본 실시 형태에서는, 디스크(101)(도 1)의 1섹터가, 2048바이트인 것으로 한다.

[0460] 여기에서, 비디오 스트림에 대해서는, 그 디코드 개시 가능점(엔트리 포인트)의 직전에, 후술하는 private_stream_2 패킷(private_stream_2의 속성의 PES_packet())이 배치되어 있다. private_stream_2 패킷은, 그 private_stream_2 패킷부터, 다음 private_stream_2 패킷까지의 사이에 배치되어 있는 비디오 스트림을 디코드하는데에도 이용되는 정보가 저장되어 있다. 이 때문에, 비디오 스트림에 대해서는, 디코드 개시 가능점의 정보로서의 RPN_EP_start에는, 실제의 디코드 개시 가능점 그 자체가 아니라, 실제의 디코드 개시 가능점의 직전에 배치되어 있는 private_stream_2 패킷의 선두의 위치가 기술된다.

[0461] 또한, EP_map()에서, 디코드 개시 가능점의 정보로서의 PTS_EP_start와 RPN_EP_start의 세트는, stream_id와 private_stream_id에 의해 특정되는 엘리멘터리 스트림마다에, 미리, 오름순으로 소트되어 있다. 이에 의해, 디코드 개시 가능점의 정보로서의 PTS_EP_start와 RPN_EP_start의 세트는, 이분 검색이 가능하게 되어 있다.

[0462] 또한, 가변 레이트의 스트림이나, 비디오 액세스 유닛 마다에 사이즈가 서로 다른 스트림을 대상으로 한 랜덤 액세스의 방법은, 예를 들면, 일본 특개2000-341640호 공보(일본 특원평11-317738호) 등에 기재되어 있다.

[0463] 「클립 스트림 파일의 설명」

[0464] 다음으로, 도 4의 "STREAM" 디렉토리에 놓이는, 확장자가 PS인 클립 스트림 파일(도 4에서는, "00001.PS",

"00002.PS", "00003.PS")의 내부 구조에 대해서 설명한다.

[0465] 클립 스트림 파일은, MPEG-2 System(ISO/IEC 13818-1)에 정의된 MPEG2_Program_Stream()을 베이스로 구성되어 있다.

[0466] 즉, 도 15A 및 도 15B은, MPEG-2 System(ISO/IEC 13818-1:2000) 규격에 기술되어 있는 Table2-31, Table2-32, Table2-33을 도시하고 있다.

[0467] 클립 스트림 파일에 저장된 프로그램 스트림은, MPEG-2 System 규격의 Table2-31에 정의되어 있는 MPEG2_Program_Stream()으로서, 1개 이상의 pack()과, 1개의 MPEG_program_end_code로 구성된다. 또한, MPEG2_Program_Stream()의 설명은, 일본 특허 제2785220호 등에도 기재되어 있다.

[0468] 1개의 pack()는, MPEG-2 System 규격의 Table2-32에 정의되어 있는 바와 같이, 1개의 Pack_header()과, 임의의 수의 PES_packet()으로 구성된다. Pack_header()의 상세 내용은, MPEG-2 System 규격의 Table2-33에 정의되어 있다.

[0469] 여기에서, MPEG-2 System 규격에서는, pack()의 사이즈는, 가변 길이로서 정의되어 있지만, 여기에서는, 도 14에서 설명한 바와 같이, 2048바이트로 고정된 것으로 한다. 또한, 여기에서는, 1개의 pack()의 PES_packet()의 수는, 1개, 2개, 또는 3개로 한다. Pack()이 후술하는 private_stream_2 패킷으로 시작되는 경우, 그 직후(동일한 Pack() 내)에 대응하는 비디오 스트림의 PES_packet()이 반드시 존재한다. 또 이에 덧붙여서 3개째의 PES_packet()으로서 padding_packet(패딩 패킷)을 둘 수 있다. 또 private_stream_2 패킷은 반드시 Pack()의 선두에 놓인다.

[0470] Pack()이 private_stream_2 패킷으로 시작하지 않는 경우에는, Pack()의 선두에는 비디오, 오디오, 자막 등의 콘텐츠 데이터가 저장된 PES_packet()이 놓인다. 이에 덧붙여서 2개째의 PES_packet()로서 padding_packet(패딩 패킷)을 둘 수 있다.

[0471] 도 16A 및 도 16B 내지 도 18A 및 도 18B은, MPEG-2 System 규격의 Table2-17에서 정의되어 있는 PES_packet()을 도시하고 있다.

[0472] PES_packet()은, packet_start_code_prefix, stream_id, 및 PES_packet_length(도 16A 및 도 16B)과, stream_id 등에 의해 구조가 변화되는 헤더 부분(stuffing_byte를 포함함)(도 16A 및 도 16B 내지 도 18A 및 도 18B)과, PES_packet_data_byte(도 18A 및 도 18B)로 대별할 수 있다. 또한, PES_packet()이, padding_packet인 경우(stream_id==padding_stream), PES_packet_data_byte 대신에, padding_byte(0xFF)(도 18A 및 도 18B)이 필요한 수만큼 반복해서 배치된다.

[0473] 여기에서, PES_packet()의 헤더 부분에는, 도 16A 및 도 16B, 및, 도 17A, 도 17B 및 도 17C에 도시한 바와 같이 PTS(Presentation Time Stamp)로 불리는 표시 타이밍을 나타내는 정보와, DTS(Decoding Time Stamp)로 불리는 디코드 타이밍을 나타내는 정보를 배치할 수 있다. 본 실시 형태에서는, 모든 액세스 유닛(MPEG2-System으로 정의된, 엘리멘터리 스트림을 구성하는 디코드 단위)에 대하여 PTS가 부가되고, MPEG2-System으로 정하는 경우에 DTS가 부가되는 것으로 한다.

[0474] 프로그램 스트림에 다중화되는 엘리멘터리 스트림은, PES_packet()의 PES_packet_data_byte(도 18A 및 도 18B)에 저장된다. 그리고, PES_packet()의 stream_id에는, 그 PES_packet_data_byte에 저장된 엘리멘터리 스트림을 식별하기 위해, 그 엘리멘터리 스트림의 속성에 따른 값이 기술된다.

[0475] PES_packet()의 stream_id에 기술되는 값과, 엘리멘터리 스트림의 속성(종류)의 관계는, MPEG-2 System 규격의 Table2-18에 정의되어 있다. 여기에서, 도 19A 및 도 19B에, MPEG-2 System 규격의 Table2-18을 나타낸다.

[0476] 본 실시 형태에서는, 도 19A 및 도 19B에 나타낸 MPEG-2 System 규격에서 정의되어 있는 stream_id 중의, 예를 들면, 도 20에 나타내는 값을 채용한다.

[0477] 즉, 본 실시 형태에서는, 10111101B, 10111110B, 10111111B, 110xxxxxB, 1110xxxxB의 5패턴을, stream_id의 값으로서 채용한다. 또한, "x"은, 0 또는 1 중의 임의의 값을 나타낸다.

[0478] 그리고, private_stream_1로 불리는 속성의 엘리멘터리 스트림의 PES_packet()의 stream_id는, 도 20에 따라, 10111101B로 된다. 또한, padding_packet의 PES_packet()의 stream_id는, 도 20에 따라, 10111110B로 된다. 또한, private_stream_2로 불리는 속성의 엘리멘터리 스트림의 PES_packet()의 stream_id는, 도 20에 따라,

10111111B로 된다.

[0479] 또한, MPEG로 정의된 오디오 스트림(오디오의 엘리멘터리 스트림)의 PES_packet()의 stream_id는, 110xxxxxB로 된다. 또한, 110xxxxxB 중의 하위 5비트 xxxxx는, 오디오 스트림을 구별하는 오디오 스트림 넘버이며, 프로그램 스트림(MPEG2_Program_Stream())에는, 이 오디오 스트림 넘버에 의해 구별하는 것이 가능한 수인 32(=2^5)개의 오디오 스트림(MPEG로 정의된 오디오 스트림)을 다중화할 수 있다.

[0480] 또한, MPEG로 정의된 비디오 스트림(비디오의 엘리멘터리 스트림)의 PES_packet()의 stream_id는, 1110xxxxB로 된다. 또한, 1110xxxxB 중의 하위 4비트 xxxx는, 비디오 스트림을 구별하는 비디오 스트림 넘버이며, 프로그램 스트림에는, 이 비디오 스트림 넘버에 의해 구별할 수 있는 수인 16(=2^4)개의 비디오 스트림(MPEG로 정의된 비디오 스트림)을 다중화할 수 있다.

[0481] 그런데, stream_id가 1110xxxxB인 PES_packet()은, MPEG로 정의된 비디오 스트림을 저장하기 위해서 사용되고, stream_id이 110xxxxxB인 PES_packet()은, MPEG로 정의된 오디오 스트림을 저장하기 위해서 사용된다. 한편, MPEG로 정의되어 있지 않은 부호화 방식(예를 들면 ATRAC 방식)의 엘리멘터리 스트림을 저장하는 데에도 사용하는 PES_packet()의 stream_id는, MPEG에서는 규정되어 있지 않고, 따라서, MPEG로 정의되어 있지 않은 부호화 방식의 엘리멘터리 스트림은, MPEG로 정의된 비디오 스트림이나 오디오 스트림과 마찬가지로, 단순하게, stream_id를 지정하고, PES_packet()에 저장할 수는 없다.

[0482] 따라서, 본 실시 형태에서는, private_stream_1의 PES_packet()의 PES_packet_data_byte를 확장하고, MPEG로 정의되어 있지 않은 부호화 방식의 엘리멘터리 스트림을 저장한다.

[0483] 여기에서, private_stream_1의 PES_Packet()의, 확장한 PES_packet_data_byte를, private_stream1_PES_Payload()로 기술한다.

[0484] 「private_stream1_PES_payload()의 설명」

[0485] 도 21는, private_stream1_PES_payload()의 신택스를 도시하고 있다.

[0486] private_stream1_PES_payload()는, private_header()와 private_payload()로 구성된다. private_payload()에는, ATRAC 오디오 스트림이나, LPCM 오디오 스트림, 자막 스트림 등의, MPEG로 정의되어 있지 않은 부호화 방식의 엘리멘터리 스트림이 저장된다.

[0487] private_header()의 선두에는, private_stream_id(8비트)이 배치된다. private_stream_id는, private_payload()에 저장되는 엘리멘터리 스트림을 식별하는 식별 정보로서, 그 속성(종류)에 따라서, 예를 들면, 이하와 같은 값으로 된다.

[0488] 즉, 도 22는, private_stream_id의 값과, private_payload()에 저장되는 엘리멘터리 스트림의 속성과의 관계를 도시하고 있다.

[0489] 도 22에서는, 0000xxxxB, 0001xxxxB, 100xxxxxB의 3패턴이, private_stream_id의 값으로서 채용되어 있다. 또한, "x"는, 도 20에 있어서의 경우와 마찬가지로, 0 또는 1 중의 임의의 값을 나타낸다.

[0490] 도 22에 의하면, ATRAC 오디오 스트림이 private_payload()에 저장되는 private_stream1_PES_payload()의 private_stream_id는, 0000xxxxB으로 된다. 또한, 0000xxxxB 중의 하위 4비트 xxxx는, ATRAC 오디오 스트림를 구별하는 오디오 스트림 넘버이며, 프로그램 스트림(MPEG2_Program_Stream())에는, 이 오디오 스트림 넘버에 의해 구별할 수 있는 수인 16(=2^4)개의 ATRAC 오디오 스트림을 다중화할 수 있다.

[0491] 또한, 도 22에 따르면, LPCM 오디오 스트림이 private_payload()에 저장되는 private_stream1_PES_payload()의 private_stream_id는, 0001xxxxB으로 된다. 또한, 0001xxxxB 중의 하위 4비트 xxxx는, LPCM 오디오 스트림을 구별하는 오디오 스트림 넘버이며, 프로그램 스트림에는, 이 오디오 스트림 넘버에 의해 구별할 수 있는 수인 16(=2^4)개의 LPCM 오디오 스트림을 다중화할 수 있다.

[0492] 또한, 도 22에 따르면, 자막 스트림이 private_payload()에 저장되는 private_stream1_PES_payload()의 private_stream_id는, 100xxxxxB로 된다. 또한, 100xxxxxB 중의 하위 5비트 xxxxx는, 자막 스트림을 구별하는 자막 스트림 넘버이며, 프로그램 스트림에는, 이 자막 스트림 넘버에 의해 구별할 수 있는 수인 32(=2^5)개의 자막 스트림을 다중화할 수 있다.

[0493] 여기에서, 도 20와 도 22의 관계를 정리한 것이, 전술한 도 11이다.

[0494] 도 21로 되돌아가서, private_stream1_PES_payload()에서, private_stream_id에 계속되는 요소는, private_payload()에 저장되는 엘리멘터리 스트림의 속성에 따라 내용이 상이하다. private_payload()에 저장되는 엘리멘터리 스트림의 속성은, private_header()의 선두의 private_stream_id에 의해 판단된다.

[0495] private_payload()에 저장되는 엘리멘터리 스트림이 ATRAC 오디오 스트림인 경우(private_stream_id==ATRAC), 장래의 확장용의 reserved_for_future_use(8비트)가 배치되고, 그 후, AU_locator(16비트)가 배치된다. AU_locator는, 그 AU_locator의 직후의 위치를 기준으로 하여, private_payload()에 저장된 ATRAC 오디오 스트림의 오디오 액세스 유닛(ATRAC 오디오 액세스 유닛)(오디오 프레임)의 선두 위치를 나타낸다. private_payload()에 오디오 액세스 유닛이 존재하지 않는 경우, AU_locator에는, 예를 들면 0xFFFF가 기술된다.

[0496] private_payload()에 저장되는 엘리멘터리 스트림이 LPCM 오디오 스트림인 경우(private_stream_id==LPCM), fs_flag(1비트), reserved_for_future_use(3비트), ch_flag(4비트), 및 AU_locator(16비트)이 순차적으로 배치된다.

[0497] fs_flag는, private_payload()에 저장되는 LPCM 오디오 스트림의 샘플링 주파수를 나타낸다. 즉, 예를 들면, 샘플링 주파수가 48KHz인 경우, fs_flag는 0으로 되고, 샘플링 주파수가 44.1KHz인 경우, fs_flag는 1로 된다.

[0498] ch_flag는, private_payload()에 저장되는 LPCM 오디오 스트림의 채널수를 나타낸다. 예를 들면, LPCM 오디오 스트림이 모노럴인 경우, ch_flag는 1로 되고, LPCM 오디오 스트림이 스테레오인 경우, ch_flag은 2로 된다.

[0499] AU_locator는, 그 AU_locator의 직후의 위치를 기준으로 하여, private_payload()에 저장되는 LPCM 오디오 스트림의 오디오 액세스 유닛(LPCM 오디오 액세스 유닛)(오디오 프레임)의 선두 위치를 나타낸다. private_payload()에 오디오 액세스 유닛이 존재하지 않는 경우, AU_locator에는, 예를 들면 0xFFFF가 기술된다.

[0500] private_payload()에 저장되는 엘리멘터리 스트림이 자막 스트림인 경우 (private_stream_id==SUBTITLE), 장래의 확장을 위한 reserved_for_future_use(8비트)이 배치되고, 그 후에, AU_locator(16비트)이 배치된다. AU_locator는, 그 AU_locator의 직후의 위치를 기준으로 하여, private_payload()에 저장되는 자막 스트림의 자막 액세스 유닛의 선두 위치를 나타낸다. private_payload()에 자막 액세스 유닛이 존재하지 않는 경우, AU_locator에는, 예를 들면 0xFFFF가 기술된다.

[0501] 「private_stream2_PES_payload()의 설명」

[0502] 다음으로, 도 23는, private_stream2_PES_payload()의 신택스를 도시하고 있다.

[0503] private_stream2_PES_payload()는, private_stream_2의 PES_packet()의 PES_packet_data_byte(도 18A 및 도 18B)을 확장한 것, 즉, private_stream_2의 PES_packet()의, 확장한 PES_packet_data_byte이며, 비디오 스트림의 디코드에 이용되는 정보가 기술된다.

[0504] 본 실시 형태에서는, private_stream_2의 PES_packet()는, 비디오 스트림에 있어서의 디코드 개시 가능점의 직전에 배치된다. 따라서, 본 실시 형태에서는, 프로그램 스트림으로부터 private_stream_2의 PES_packet()을 찾아내면, 그 직후의 비디오 스트림으로부터 디코드를 개시할 수 있다.

[0505] 여기에서, 전술한 도 14의 EP_map()의 RPN_EP_start는, 비디오 스트림에 대해서는, private_stream_2의 PES_packet()의 선두의 위치를 나타낸다.

[0506] private_stream2_PES_payload()의 선두에는, 장래의 확장용의 reserved_for_future_use(8비트)가 배치되고, 계속하여, video_stream_id(8비트), 1stRef_picture(16비트), 2ndRef_picture(16비트), 3rdRef_picture(16비트), 4thRef_picture(16비트), au_information(), 및 VBI()가, 순차적으로 배치된다.

[0507] video_stream_id에는, private_stream_2의 PES_packet()의 직후에 배치되는 비디오 스트림의 PES_packet()의 stream_id(와 동일한 값)가 기술된다. 이 video_stream_id에 의해, private_stream_2의 PES_packet()(의 private_stream2_PES_payload())에 저장된 정보를 이용해서 디코드되는 비디오 스트림(이 저장된 PES_packet())이 특정된다.

[0508] 1stRef_picture, 2ndRef_picture, 3rdRef_picture, 4thRef_picture는, video_stream_id에 의해 특정되는 비디오 스트림의, private_stream_2의 PES_packet()부터 다음의 private_stream_2의 PES_packet()까지 중의 1, 2, 3, 4번째의 참조 화상을 포함하는 최후의 pack()의 위치를 상대값으로, 각각 나타낸다. 또한, 1stRer_picture, 2ndRef_picture, 3rdRef_picture, 4thRef_picture에 대해서는, 예를 들면, 일본 특개평09-46712호공보(일본 특원평07-211420호)에, bytes_to_first_P_pic bytes_to_second_P_pic으로서, 그 상세 내용이 개시되어 있다.

[0509] au_information()에는, private_stream_2의 PES_packet()로부터, 다음 private_stream_2의 PES_packet()까지의 비디오 스트림 중의 비디오 액세스 유닛에 관한 정보가 기술된다. au_information()의 상세 내용에 대해서는, 도 24를 참조하여 후술한다.

[0510] VBI()는, Closed Caption의 정보를 기술하기 위해 사용된다.

[0511] 이상과 같은 private_stream2_PES_payload()을 가지는 private_stream_2의 PES_packet()은, 비디오 스트림마다의, 디코드 개시 가능점마다에 배치된다.

[0512] 다음으로, 도 24는, 도 23의 au_information()의 신택스를 도시하고 있다.

[0513] au_information()의 선두에는, length(16비트)이 배치된다. length는, 그것을 포함하는 au_information()의 사이즈를 나타낸다. length에 이어서는, reserved_for_word_alignment(8비트), 및 number_of_access_unit(8비트)이 순차적으로 배치된다. reserved_for_word_alignment는, 워드 얼라인을 취하기 위해서 사용된다.

[0514] number_of_access_unit은, 그것을 포함하는 private_stream_2의 PES_packet()로부터, 다음 private_stream_2의 PES_packet()까지의 사이에 포함되는 비디오 액세스 유닛(픽쳐)의 수를 나타낸다.

[0515] 즉, number_of_access_unit는, 도 23의 private_stream2_PES_payload()이 동일한 video_stream_id를 가지는 private_stream_2의 PES_packet() 중에서, 이 au_informatnio()로부터 다음 au_information()의 직전까지(이 au_infromation()이 클립 스트림 파일에서 최후의 au_information()이면, 클립 스트림 파일의 최후까지), video_stream_id로 기재되는 비디오 스트림에 포함되는 액세스 유닛(픽쳐)의 수를 나타낸다.

[0516] number_of_access_unit 후에는, 그 number_of_access_unit의 수만큼 for 루프의 내용이 배치된다. 즉, number_of_access_unit를 포함하는 private_stream_2의 PES_packet()로부터, 다음 private_stream_2의 PFS_packet()까지의 사이에 포함되는 1이상의 비디오 액세스 유닛 각각에 관한 정보가 배치된다.

[0517] for 루프 내에 배치되는 정보(비디오 액세스 유닛에 관한 정보)는, 아래와 같이 되어 있다.

[0518] 즉, for 루프 내에는, pic_struct_copy(4비트), au_ref_flag(1비트), AU_length(21비트), reserved가 배치된다.

[0519] pic_struct_copy에는, 비디오 스트림이 MPEG4-AVC(ISO/IEC 14496-10)인 경우에, 대응하는 비디오 액세스 유닛에 대하여 설정되어 있는, ISO/IEC 14496-10, D.2.2에 정의되어 있는 pic_struct()의 카피가 기술된다. 또한, pic_struct()은, 예를 들면, 픽쳐를 프레임으로서 표시하거나, 혹은, 픽쳐의 톱 필드를 표시하고, 그 후, 보텀 필드를 표시하는 등과 같은 정보이다.

[0520] au_ref_flag는, 대응하는 액세스 유닛이, 다른 액세스 유닛(의 픽쳐)의 디코드에 있어서 참조되는 참조 화상인지의 여부를 나타내고, 참조 화상인 경우에는 1로 되고, 참조 화상이 아닌 경우에는 0으로 된다.

[0521] AU_length는, 대응하는 액세스 유닛의 사이즈를 바이트 단위로 나타낸다.

[0522] [디스크(101)에 기록된 데이터의 구체예]

[0523] 다음으로, 도 25 내지 도 28는, 도 1의 디스크(101)에 기록된, 전술한 바와 같은 포맷의 데이터의 구체예를 도시하고 있다.

[0524] 여기에서, 도 25 내지 도 28에서는, 비디오 스트림으로서는, MPEG2-Video를 채용하고, 오디오 스트림으로서는, ATRAC 오디오 스트림을 채용하고 있다. 단, 비디오 스트림이나 오디오 스트림은, 이것에 한정되는 것은 아니다. 즉, 비디오 스트림으로서는, 예를 들면, MPEG4-Visual이나 MPEG4-AVC등을 채용할 수 있다. 또한, 오디오 스트림으로서는, 예를 들면, MPEG1/2/4 오디오나 LPCM 오디오 스트림 등을 채용할 수 있다.

[0525] 또한, 자막 스트림은, 비디오 스트림이나 오디오 스트림과 달리, 동일한 간격으로 연속적인

디코드?표시(출력)이 행해진다고는 한정할 수 없다. 즉, 자막 스트림은, 때때로, 도 2A 및 도 2B의 버퍼 제어 모듈(215)로부터 자막 디코더 제어 모듈(218)에 공급되어서 디코드된다.

[0526] 도 25 내지 도 28는, 디스크(101)에서, 도 4에 도시한 바와 같이, "CLIP" 디렉토리에, 3개의 클립 정보 파일 "00001.CLP", "00002.CLP", "00003.CLP"이 기록되고, "STREAM" 디렉토리에, 그 3개의 클립 정보 파일 "00001.CLP", "00002.CLP, "00003.CLP" 각각에 대응하는 3개의 클립 스트림 파일 "00001.PS", "00002.PS", "00003.PS"이 기록되어 있는 경우의, "LAYLIST.DAT" 파일, 클립 정보 파일 "00001.CLP", "00002CLP", 및 "00003.CLP"등의 구체적인 예를 도시하고 있다. 단, 도 25 내지 도 28에서는, "PLAYLIST.DAT" 파일 등의 데이터의 일부를 생략하고 있다.

[0527] 즉, 도 25는, 도 5에서 설명한 "PLAYLIST.DAT" 파일의 구체예를 도시하고 있다.

[0528] 도 25에서는, number_of_PlayLists는 2로 되어 있고, 따라서, "PLAYLIST.DAT" 파일에 포함되는 PlayList()의 수는 2이다. 도 25에서는, 그 2개의 PlayList() 중의 1번째가 PlayList #0로, 2번째가 PlayList #1로, 각각 기재되어 있다.

[0529] 1번째의 PlayList()인 PlayList #0에 대해서는, capture_enable_flag_PlayList가 1로 되어 있고, 따라서, PlayList #0에 따라서 재생되는 비디오 데이터의 2차 이용이 허가되어 있다. 또한, PlayList #0에 대해서는, number_of_PlayItems가 2로 되어 있고, 따라서, PlayList #0에 포함되는 PlayItem()의 수는 2이다. 도 25에서는, 그 2개의 PlayItem()인 PlayItem#0와 PlayItem#1의 구체예가, 「PlayList #0」의 란의 아래쪽에 기재되어 있다.

[0530] PlayList #0에 포함되는 1번째의 PlayItem()인 PlayItem#0에서는, 도 6에서 설명한 Clip_Information_file_name이 "00001.CLP"로, In_time이 180,090으로, OUT_time이 27,180,090으로, 각각 되어 있다. 따라서, PlayList #0의 PlayItem#0에 의해 재생되는 클립은, 클립 정보 파일 "00001.CLP"에 대응하는 클립 스트림 파일 "00001.PS"의, 시각 180,090로부터 27,180,090까지이다.

[0531] PlayList #0에 포함되는 2번째의 PlayItem()인 PlayItem#1에서는, 도 6에서 설명한 Clip_Information_file_name이 "00002.CLP"로, In_time이 90,000으로, OUT_time이 27,090,000으로, 각각 되어 있다. 따라서, PlayList #0의 PlayItem#1에 의해 재생되는 클립은, 클립 정보 파일 "00002.CLP"에 대응하는 클립 스트림 파일 "00002.PS"의, 시각 90,000로부터 27,090,000까지이다.

[0532] 한편, 도 25에 있어서, 2번째의 PlayList()인 PlayList #1에 대해서는, capture_enable_flag_PlayList가 0으로 되어 있고, 따라서, PlayList #1에 따라서 재생되는 비디오 데이터의 2차 이용이 허가되어 있지 않다(금지되어 있다). 또한, PlayList #1에 대해서는, number_of_PlayItems가 1로 되어 있고, 따라서, PlayList #1에 포함되는 PlayItem()의 수는 1이다. 도 25에서는, 그 1개의 PlayItem()인 PlayItem#0의 구체예가, 「PlayList #1」의 란의 아래쪽에 기재되어 있다.

[0533] PlayList #1에 포함되는 1개의 PlayItem()인 PlayItem#0에서는, 도 6에서 설명한 Clip_Information_file_name이 "00003.CLP"으로, In_time이 90,000으로, OUT_time이 81,090,000으로, 각각 되어 있다. 따라서, PlayList #1의 PlayItem#0에 의해 재생되는 클립은, 클립 정보 파일 "00003.CLP"에 대응하는 클립 스트림 파일 "00003.PS"의, 시각 90,000로부터 81,090,000까지이다.

[0534] 다음으로, 도 26A 및 도 26B은, 도 10에서 설명한 클립 정보 파일 Clip()의 구체예를 도시하고 있다. 즉, 도 26A 및 도 26B은, 도 4의 클립 정보 파일 "00001.CLP", "00002.CLP", "00003.CLP"의 구체예를 도시하고 있다.

[0535] 클립 정보 파일 "00001.CLP"에서는, presentation_start_time이 90,000으로, presentation_end_time이 27,990,000으로, 각각 되어 있다. 따라서, 클립 정보 파일 "00001.CLP"에 대응하는 클립 스트림 파일 "00001.PS"에 저장된 프로그램 스트림에 따르면, 310초분((27,990,000-90,000)/90㎑)의 콘텐츠가 이용 가능하다.

[0536] 또한, 클립 정보 파일 "00001.CLP"에서는, capture_enable_flag_Clip이 1로 되어 있고, 따라서, 클립 정보 파일 "00001.CLP"에 대응하는 클립 스트림 파일 "00001.PS"에 저장된 프로그램 스트림에 다중화된 비디오 스트림(에 대응하는 비디오 데이터)에 대해서는, 그 2차 이용이 허가되어 있다.

[0537] 또한, 도 26A 및 도 26B에서, 클립 정보 파일 "00001.CLP"의 number_of_streams는 4로 되어 있고, 따라서, 대응하는 클립 스트림 파일 "00001.PS"에 저장된 프로그램 스트림에는, 4개의 엘리멘터리 스트림이 다중화되어

있다.

[0538] 지금, 그 4개의 엘리멘터리 스트림을, stream#0, stream#1, stream#2, stream#3으로 하면, 도 26A 및 도 26B에서는, 그 4개의 엘리멘터리 스트림 stream#0, stream#1, stream#2, stream#3 각각의 StreamInfo()(도 10)의 구체예가, 「"00001.CLP"」의 란의 아래쪽에 기재되어 있다.

[0539] 클립 스트림 파일 "00001.PS"의 1개째의 엘리멘터리 스트림 stream#0에 대해서는, stream_id가 0xE0로 되어 있고, 따라서, 이 엘리멘터리 스트림 stream#0는, 도 20 및 도 22(혹은 도 11)에서 설명한 바와 같이, 비디오 스트림이다. 또한, 본 실시 형태에서는, 비디오 스트림에 대해서는, 전술한 바와 같이, private_stream_id는 무관계이지만, 도 26A 및 도 26B에서는, 0x00으로 되어 있다.

[0540] 또한, 클립 스트림 파일 "00001.PS"의 1개째의 엘리멘터리 스트림인 비디오 스트림 stream#0에 대해서는, 그 StreamInfo()에 포함되는 StaticInfo()(도 12)의 picture_size가 '720×480'으로, frame_rate가 '29.97Hz'로, cc_flag가 'Yes'로, 각각 되어 있다. 따라서, 이 비디오 스트림 stream#0은, 720×480픽셀의, 프레임 주기가 29.97Hz인 비디오 데이터이며, 또한, 클로즈드 캡션 데이터를 포함하고 있다.

[0541] 또한, 클립 스트림 파일 "00001.PS"의 1개째의 엘리멘터리 스트림인 비디오 스트림 stream#0에 대해서는, StreamInfo()(도 10)의 number_of_DynamicInfo가 0으로 되어 있고, pts_change_point와 DynamicInfo()의 세트는 존재하지 않는다.

[0542] 다음으로, 클립 스트림 파일 "00001.PS"의 2개째의 엘리멘터리 스트림 stream#1에 대해서는, stream_id가 0xBD가 되어 있고, private_stream_id가 0x00로 되어 있다. 따라서, 이 엘리멘터리 스트림 stream#1은, 도 20 및 도 22에서 설명한 바와 같이, ATRAC 오디오 스트림이다.

[0543] 또한, 클립 스트림 파일 "00001.PS"의 2개째의 엘리멘터리 스트림인 ATRAC 오디오 스트림 stream#1에 대해서는, 그 StreamInfo()에 포함되는 StaticInfo()(도 12)의 audio_language_code가 '일본어'로, channel_configuration이 'STEREO'로, lfe_existence가 'NO'로, sampling_frequency가 '48㎑'로, 각각 되어 있다. 따라서, 이 ATRAC 오디오 스트림 stream#1은, 일본어이고, 또한 스테레오의 오디오 데이터이다. 또한, 저역 강조 채널은 포함되어 있지 않고, 샘플링 주파수는, 48㎑이다.

[0544] 또한, 클립 스트림 파일 "00001.PS"의 2개째의 엘리멘터리 스트림인 ATRAC 오디오 스트림 stream#1에 대해서는, StreamInfo()(도 10)의 number_of_DynamicInfo가 0으로 되어 있고, pts_change_point와 DynamicInfo()의 세트는 존재하지 않는다.

[0545] 다음으로, 클립 스트림 파일 "00001.PS"의 3개째의 엘리멘터리 스트림 stream#2에 대해서는, stream_id가 0xBD로 되어 있고, private_stream_id이 0x80으로 되어 있다. 따라서, 이 엘리멘터리 스트림 stream#2은, 도 20 및 도 22에서 설명한 바와 같이, 자막 스트림이다.

[0546] 또한, 클립 스트림 파일 "00001.PS"의 3개째의 엘리멘터리 스트림인 자막 스트림 stream#2에 대해서는, 그 StreamInfo()에 포함되는 StaticInfo()(도 12)의 subtitle_language_code가 '일본어'로, configurable_flag가 0으로, 각각 되어 있다. 따라서, 이 자막 스트림 stream#2은, 일본어의 자막 데이터이며, 또한, 그 표시 방식을 변경하는 것은 허가되어 있지 않다(금지되어 있다).

[0547] 또한, 클립 스트림 파일 "00001.PS"의 3개째의 엘리멘터리 스트림인 자막 스트림 stream#2에 대해서는, StreamInfo()(도 10)의 number_of_DynamicInfo가 0으로 되어 있어, pts_change_point와 DynamicInfo()의 세트는 존재하지 않는다.

[0548] 다음으로, 클립 스트림 파일 "00001.PS"의 4개째의 엘리멘터리 스트림 stream#3에 대해서는, stream_id가 0xBD로 되어 있고, private_stream_id가 0x81로 되어 있다. 따라서, 이 엘리멘터리 스트림 stream#3은, 도 20 및 도 22에서 설명한 바와 같이, 자막 스트림이다.

[0549] 또한, 클립 스트림 파일 "00001.PS"의 3개째의 엘리멘터리 스트림인 자막 스트림 stream#2과, 4개째의 엘리멘터리 스트림인 자막 스트림 stream#3를 구별하기 위해, 각각의 private_stream_id는, 0x80과 0x81로 되어 있다.

[0550] 또한, 클립 스트림 파일 "00001.PS"의 4개째의 엘리멘터리 스트림인 자막 스트림 stream#3에 대해서는, 그 StreamInfo()에 포함되는 StaticInfo()(도 12)의 subtitle_language_code가 '일본어'로, configurable_flag가 1로, 각각 되어 있다. 따라서, 이 자막 스트림 stream#3은, 일본어의 자막 데이터이며, 또한, 그 표시 방식

을 변경하는 것이 허가되어 있다.

[0551] 또한, 클립 스트림 파일 "00001.PS"의 4개째의 엘리멘터리 스트림인 자막 스트림 stream#3에 대해서는, StreamInfo()(도 10)의 number_of_DynamicInfo가 0으로 되어 있어, pts_change_point와 DynamicInfo()의 세트는 존재하지 않는다.

[0552] 다음으로, 도 26A 및 도 26B에서, 클립 정보 파일 "00002.CLP"에 대해서는, presentation_start_time이 90,000으로, presentation_end_time이 27,090,000으로, 각각 되어 있다. 따라서, 클립 정보 파일 "00002.CLP"에 대응하는 클립 스트림 파일 "00002.PS"에 저장된 프로그램 스트림에 따르면, 300초분((27,090,000-90,000)/90㎑)의 콘텐츠가 이용 가능하다.

[0553] 또한, 클립 정보 파일 "00002.CLP"에서는, capture_enable_flag_Clip이 0으로 되어 있고, 따라서, 클립 정보 파일 "00002.CLP"에 대응하는 클립 스트림 파일 "00002.PS"에 저장된 프로그램 스트림에 다중화된 비디오 스트림(에 대응하는 비디오 데이터)에 대해서는, 그 2차 이용이 허가되어 있지 않다(금지되어 있다).

[0554] 또한, 도 26A 및 도 26B에서, 클립 정보 파일 "00002.CLP"의 number_of_streams는 4로 되어 있고, 따라서, 대응하는 클립 스트림 파일 "00002.PS"에 저장된 프로그램 스트림에는, 전술한 클립 스트림 파일 "00001.PS"에 있어서의 경우와 마찬가지로, 4개의 엘리멘터리 스트림이 다중화되어 있다.

[0555] 지금, 그 4개의 엘리멘터리 스트림을, stream#0, stream#1, stream#2, stream#3으로 하면, 도 26A 및 도 26B에서는, 그 4개의 엘리멘터리 스트림 stream#0, stream#1, stream#2, stream#3 각각의 StreamInfo()(도 10)의 구체예가, 「"00002.CLP"」의 란의 아래쪽에 기재되어 있다.

[0556] 여기에서, 도 26A 및 도 26B에서는, 클립 스트림 파일 "00002.PS"의 1 내지 4개째의 엘리멘터리 스트림 stream#0 내지 #3 각각의 StreamInfo()의 내용은, 전술한 클립 스트림 파일 "00001.PS"의 1 내지 4개째의 엘리멘터리 스트림 stream#0 내지 #3 각각의 StreamInfo()의 내용과 동일하게 되어 있으므로, 그 설명은 생략한다.

[0557] 또한, 전술된 바와 같이, 클립 스트림 파일 "00002.PS"의 1 내지 4개째의 엘리멘터리 스트림 stream#0 내지 #3 각각의 StteamInfo()의 내용은, 클립 스트림 파일 "00001.PS"의 1 내지 4개째의 엘리멘터리 스트림 stream#0 내지 #3 각각의 StreamInfo()의 내용과 동일하므로, 클립 스트림 파일 "00002.PS"의 1개째의 엘리멘터리 스트림 stream#0는 비디오 스트림이며, 2개째의 엘리멘터리 스트림 stream#1은 ATRAC 오디오 스트림이다. 또한, 그 3개째의 엘리멘터리 스트림 stream#2과, 4개째의 엘리멘터리 스트림 stream#3은, 모두 자막 스트림이다.

[0558] 다음으로, 도 26A 및 도 26B에서, 클립 정보 파일 "00003.CLP"에 대해서는, presentation_start_time이 90,000으로, presentation_end_time이 81,090,000으로, 각각 되어 있다. 따라서, 클립 정보 파일 "00003.CLP"에 대응하는 클립 스트림 파일 "00003.PS"에 저장된 프로그램 스트림에 따르면, 900초분((81,090,000-90,000)/90㎑)의 콘텐츠가 이용 가능하다.

[0559] 또한, 클립 정보 파일 "00003.CLP"에서는, capture_enable_flag_Clip이 1로 되어 있고, 따라서, 클립 정보 파일 "00003.CLP"에 대응하는 클립 스트림 파일 "00003.PS"에 저장된 프로그램 스트림에 다중화된 비디오 스트림에 대해서는, 그 2차 이용이 허가되어 있다.

[0560] 또한, 도 26A 및 도 26B에서, 클립 정보 파일 "00003.CLP"의 number_of_streams는 3으로 되어 있고, 따라서, 대응하는 클립 스트림 파일"00003.PS"에 저장된 프로그램 스트림에는, 3개의 엘리멘터리 스트림이 다중화되어 있다.

[0561] 지금, 그 3개의 엘리멘터리 스트림을, stream#0, stream#1, stream#2로 하면, 도 26A 및 도 26B에서는, 그 3개의 엘리멘터리 스트림 stream#0, stream#1, stream#2 각각의 StreamInfo()(도 10)의 구체예가, 「"00003.CLP"」의 란의 아래쪽에 기재되어 있다.

[0562] 클립 스트림 파일 "00003.PS"의 1개째의 엘리멘터리 스트림 stream#0에 대해서는, stream_id가 0xE0로 되어 있고, 따라서, 이 엘리멘터리 스트림 stream#0는, 도 20 및 도 22(혹은 제11도)에서 설명한 바와 같이, 비디오 스트림이다. 또한, 클립 스트림 파일 "00001.PS"의 1개째의 엘리멘터리 스트림 stream#0와 마찬가지로, private_stream_id는, 0x00으로 되어 있다.

[0563] 또한, 클립 스트림 파일 "00003.PS"의 1개째의 엘리멘터리 스트림인 비디오 스트림 stream#0에 대해서는, 그 StreamInfo()에 포함되는 StaticInfo()(도 12)의 picture_size가 '720×480'으로, frame_rate가 '29.97㎐'으

로, cc_flag가 'No'로, 각각 되어 있다. 따라서, 이 비디오 스트림 stream#0은, 720×480 픽셀의, 프레임 주기가 29.97Hz인 비디오 데이터이며, 또한, 클로즈드 캡션 데이터를 포함하고 있지 않다.

[0564] 또한, 클립 스트림 파일 "00003.PS"의 1개째의 엘리멘터리 스트림인 비디오 스트림 stream#0에 대해서는, StreamInfo()(도 10)의 number_of_DynamicInfo가 2로 되어 있고, 따라서, 그 StreamInfo()에는, pts_change_point와 DynamicInfo()의 세트가 2세트 기술되어 있다.

[0565] 다음으로, 클립 스트림 파일 "00003.PS"의 2개째의 엘리멘터리 스트림 stream#1에 대해서는, stream_id가 0xE1로 되어 있고, 따라서, 이 엘리멘터리 스트림 stream#1은, 도 20 및 도 22(혹은 도 11)에서 설명한 바와 같이, 비디오 스트림이다. 또한, 클립 스트림 파일 "00003.PS"의 1개째의 엘리멘터리 스트림인 비디오 스트림 stream#0와, 2개째의 엘리멘터리 스트림인 비디오 스트림 stream#1를 구별하기 위해, 각각의 stream_id는, 0xE0와 0xE1로 되어 있다. 또한, 클립 스트림 파일 "00001.PS"의 1개째의 엘리멘터리 스트림 stream#0와 마찬가지로, private_stream_id는, 0x00으로 되어 있다.

[0566] 또한, 클립 스트림 파일 "00003.PS"의 2개째의 엘리멘터리 스트림인 비디오 스트림 stream#1에 대해서는, 그 StreamInfo()에 포함되는 StaticInfo()(도 12)의 picture_size, frame_rate, cc_flag이, 1개째의 엘리멘터리 스트림인 비디오 스트림 stream#0에 관한 것과 동일하게 되어 있다. 따라서, 클립 스트림 파일 "00003.PS"의 2개째의 엘리멘터리 스트림인 비디오 스트림 stream#1은, 720×480픽셀의, 프레임 주기가 29.97Hz의 비디오 데이터이며, 또한, 클로즈드 캡션 데이터를 포함하고 있지 않다.

[0567] 또한, 클립 스트림 파일 "00003.PS"의 2개째의 엘리멘터리 스트림인 비디오 스트림 stream#1에 대해서는, StreamInfo()(도 10)의 number_of_DynamicInfo가 0으로 되어 있어, pts_change_point와 DynamicInfo()의 세트는 존재하지 않는다.

[0568] 다음으로, 클립 스트림 파일 "00003.PS"의 3개째의 엘리멘터리 스트림 stream#2에 대해서는, stream_id가 0xBD로 되어 있고, private_stream_id가 0x00가 되어 있다. 따라서, 이 엘리멘터리 스트림 stream#2은, 도 20 및 도 22에서 설명한 바와 같이, ATRAC 오디오 스트림이다.

[0569] 또한, 클립 스트림 파일 "00003.PS"의 3개째의 엘리멘터리 스트림인 ATRAC 오디오 스트림 stream#2에 대해서는, 그 StreamInfo()에 포함되는 StaticInfo()(도 12)의 audio_language_code, channel_configuration, lfe_existence, sampling_frequency가, 클립 스트림 파일 "00001.PS"의 2개째의 엘리멘터리 스트림인 ATRAC 오디오 스트림 stream#1의 것과 동일하게 되어 있다. 따라서, "00003.PS"의 3개째의 엘리멘터리 스트림인 ATRAC 오디오 스트림 stream#2은, 일본어이고, 또한 스테레오의 오디오 데이터이다. 또한, 저역 강조 채널은 포함되어 있지 않고, 샘플링 주파수는, 48㎑이다.

[0570] 또한, 클립 스트림 파일 "00003.PS"의 3개째의 엘리멘터리 스트림인 ATRAC 오디오 스트림 stream#2에 대해서는, StreamInfo()(도 10)의 number_of_DynamicInfo가 3으로 되어 있고, 따라서, 그 StreamInfo()에는, pts_change_point와 DynamicInfo()의 세트가 3세트 기술되어 있다.

[0571] 다음으로, 도 27은, 도 10에서 설명한 클립 정보 파일 Clip() 중의 EP_map()구체예를 도시하고 있다. 즉, 도 27은, 도 4의 클립 정보 파일 "00001.CLP" 중의, 도 14의 EP_map()의 구체예를 도시하고 있다.

[0572] 도 27에 있어서, EP_map()의 number_of_stream_id_entries는 1로 되어 있고, 따라서, 이 EP_map()에는, 1개의 엘리멘터리 스트림에 관한 디코드 개시 가능점의 정보가 기술되어 있다.

[0573] 또한, 도 27의 EP_map()에서는, stream_id가 0xE0으로 되어 있다. 따라서, 도 20 및 도 22에서 설명한 것으로부터, EP_map()에는, 0xE0로 되어 있는 stream_id에 의해 특정되는 비디오 스트림에 관한 디코드 개시 가능점의 정보(PTS_EP_start와 RPN_EP_start(도 14))이 기술되어 있다. 즉, 도 27은, 클립 정보 파일 "00001.CLP"의 EP_map()이며, 클립 정보 파일 "00001.CLP"에 대응하는 클립 스트림 파일 "00001.PS"에서, stream_id가 0xE0의 엘리멘터리 스트림은, 도 26A 및 도 26B에서 설명한 바와 같이, 클립 스트림 파일 "00001.PS"의 1개째의 비디오 스트림 stream#0이기 때문에, 도 27의 EPmap()에 기술되어 있는 정보는, 그 비디오 스트림 stream#0의 디코드 개시 가능점의 PTS_EP_start와 RPN_EP_start이다.

[0574] 도 27에서는, 클립 스트림 파일 "00001.PS"의 1개째의 비디오 스트림 stream#0의 디코드 개시 가능점 중의, 선두로부터 5점의 PTS_EP_start와 RPN_EP_start가 기재되어 있고, 6점째 이후의 PTS_EP_start와 RPN_EP_start의 기재는 생략하고 있다.

[0575] 또한, 도 27의 EP_map()에서, private_stream_id는 0x00로 되어 있지만, stream_id이 비디오 스트림을 나타내

고 있는 경우에는, 전술한 바와 같이, private_stream_id는 무관계이다(무시된다).

[0576] 다음으로, 도 28는, 도 25에서 설명한 PlayList #0와 PlayList #1(도 5의 PlayList()) 중의 PlayListMark()의 구체예를 도시하고 있다.

[0577] 도 28 상측은, PlayList #0의 PlayListMark()(도 7)을 도시하고 있다.

[0578] 도 28 상측에서, PlayList #0의 PlayListMark()에 있어서의 number_of_PlayList_marks는 7로 되어 있고, 따라서, PlayList #0(의 PlayListMark())에 포함되는 Mark()의 수는 7이다.

[0579] 또한, 도 28 상측에서, PlayList #0에 포함되는 7개의 Mark() 중의 1번째의 Mark()인 Mark#0은, mark_type(도 7)이 'Chapter'로 되어 있으므로, 챕터 마크이다. 또한, Mark#0은, ref_to_PlayItem_id(도 7)이 0으로 되어 있으므로, PlayList #0에 포함되는 도 25의 2개의 PlayItem#0와 #1 중의, PlayItem#0에 속한다. 또한, Mark#0은, mark_time_stamp이 180,090으로 되어 있으므로, PlayList #0에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일의 시각(재생 시각) 180,090상의 표시이다. 또한, Mark#0는, entry_ES_stream_id 및 entry_ES_private_stream_id가 모두 0으로 되어 있으므로, 어느쪽의 엘리멘터리 스트림에도 관련지어져 있지 않다. 또한, Mark#0은, mark_data가 1로 되어 있으므로, 번호가 1인 챕터를 나타낸다.

[0580] 또한, 여기에서는, PlayList #0에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일은, 도 25에서 설명한, 그 PlayItem#0의 Clip_Information_file_name에 기술되어 있는 "00001.CLP"로부터 특정되는 클립 스트림 파일 "00001.PS"이며, 따라서, Mark#0의 mark_time_stamp이 나타낸, 전술한 시각 180,090은, 클립 스트림 파일 "00001.PS"의 시각이다.

[0581] 도 28 상측에서, PlayList #0에 포함되는 7개의 Mark() 중의 5번째의 Mark()인 Mark#4도, 1번째의 Mark#0와 마찬가지의 챕터 마크이다.

[0582] 즉, 5번째의 Mark()인 Mark#4은, mark_type(도 7)이 'Chapter'로 되어 있으므로, 챕터 마크이다. 또한, Mark#4은, ref_to_PlayItem_id(도 7)이 1로 되어 있으므로, PlayList #0에 포함되는 도 25의 2개의 PlayItem#0와 #1 중의, PlayItem#1에 속한다. 또한, Mark#4은, mark_time_stamp이 90,000으로 되어 있으므로, PlayList #0에 포함되는 PlayItem#1에 의해 재생되는 클립 스트림 파일의 시각 90,000 상의 표시이다. 또한, Mark#4은, entry_ES_stream_id 및 entry_ES_private_stream_id가 모두 0으로 되어 있으므로, 어느쪽의 엘리멘터리 스트림에도 관련지어져 있지 않다. 또한, Mark#4은, mark_data가 2로 되어 있으므로, 번호가 2인 챕터를 나타낸다.

[0583] 또한, 여기에서는, PlayList #0에 포함되는 PlayItem#1에 의해 재생되는 클립 스트림 파일은, 도 25에서 설명한, 그 PlayItem#1의 Clip_Information_file_name에 기술되어 있는 "00002.CLP"로부터 특정되는 클립 스트림 파일 "00002.PS"이며, 따라서, Mark#4의 mark_time_stamp이 나타내는, 전술한 시각 90,000은, 클립 스트림 파일 "00002.PS"의 시각이다.

[0584] 또한, 도 28 상측에서, PlayList #0에 포함되는 7개의 Mark() 중의 2번째의 Mark()인 Mark#1은, mark_type(도 7)이 'Index'로 되어 있으므로, 인덱스 마크이다. 또한, Mark#1은, ref_to_PlayItem_id(도 7)이 0으로 되어 있으므로, PlayList #0에 포함되는 도 25의 2개의 PlayItem#0와 #1 중의, PlayItem#0에 속한다. 또한, Mark#1은, mark_time_stamp이 5,580,090으로 되어 있으므로, PlayList #0에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일의 시각 상의 표시이다. 또한, Mark#1은, entry_ES_stream_id 및 entry_ES_private_stream_id가 모두 0으로 되어 있으므로, 어느쪽의 엘리멘터리 스트림에도 관련지어져 있지 않다. 또한, Mark#1은, mark_data가 1로 되어 있으므로, 번호가 1인 인덱스를 나타낸다.

[0585] 또한, 여기에서는, PlayList #0에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일은, 전술한 바와 같이, 클립 스트림 파일 "00001.PS"이며, 따라서, Mark#1의 mark_time_stamp이 나타낸다, 전술한 시각 5,580,090은, 클립 스트림 파일 "00001.PS"의 시각이다.

[0586] 도 28 상측에서, PlayList #0에 포함되는 7개의 Mark() 중의 3번째, 6번째, 7번째의 Mark()인 Mark#2, Mark#5, Mark#6도, 2번째의 Mark#1와 마찬가지의 인덱스 마크이다.

[0587] 또한, 도 28 상측에서, PlayList #0에 포함되는 7개의 Mark() 중의 4번째의 Mark0인 Mark#3은, mark_type(도 7)이 'Event'로 되어 있으므로, 이벤트 마크이다. 또한, Mark#3은, ref_to_PlayItem_id(도 7)이 0으로 되어 있으므로, PlayList #0에 포함되는 도 25의 2개의 PlayItem#0와 #1 중의, PlayItem#0에 속한다. 또한, Mark#3은, mark_time_stamp이 16,380,090으로 되어 있으므로, PlayList #0에 포함되는 PlayItem#0에 의해 재

생되는 클립 스트림 파일의 시각 16,380,090 상의 표시이다. 또한, Mark#3은, entry_ES_stream_id 및 entry_ES_private_stream_id가 모두 0으로 되어 있으므로, 어느쪽의 엘리멘터리 스트림에도 관련지어져 있지 않다. 또한, Mark#3은, mark_data가 0으로 되어 있으므로, 인수로서 0을 수반하는 이벤트를 발생시킨다.

[0588] 또한, 여기에서는, PlayList #0에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일은, 전술한 바와 같이, 클립 스트림 파일 "00001.PS"이며, 따라서, Mark#3의 mark_time_stamp가 나타내는, 전술한 시각 16,380,090은, 클립 스트림 파일 "00001.PS"의 시각이다.

[0589] 여기에서, 도 28 상측에서는, PlayList #0의 PlayListMark()의 일람표의 난외의 우측에, Mark()이 속하는 PlayItem()의 선두로부터의 시간을 도시하고 있고, 또한 그 우측에, PlayList #0의 선두로부터의 시각을 도시하고 있다.

[0590] 다음으로, 도 28 하측은, PlayList #1의 PlayListMark()(도 7)을 도시하고 있다.

[0591] 도 28 하측에서, PlayList #1의 PlayListMark()에 있어서의 number_of_PlayList_marks는 3으로 되어 있고, 따라서, PlayList #1(의 PlayListMark())에 포함되는 Mark()의 수는 3이다.

[0592] 또한, 도 28 하측에서, PlayList #1에 포함되는 3개의 Mark() 중의 1번째의 Mark()인 Mark#0은, mark_type(도 7)이 'Chapter'로 되어 있으므로, 챕터 마크이다. 또한, Mark#0은, ref_to_PlayItem_id(도 7)가 0으로 되어 있으므로, PlayList #1에 포함되는 도 25의 1개의 PlayItem#0에 속한다. 또한, Mark#0은, mark_time_stamp이 90,000으로 되어 있으므로, PlayList #1에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일의 시각 90,000 상의 표시이다. 또한, Mark#0은, entry_ES_stream_id 및 entry_ES_private_stream_id가 모두 0으로 되어 있으므로, 어느쪽의 엘리멘터리 스트림에도 관련지어져 있지 않다. 또한, Mark#0은, mark_data가 0으로 되어 있으므로, 번호가 0인 챕터를 나타낸다.

[0593] 또한, 여기에서는, PlayList #1에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일은, 도 25에서 설명한, 그 PlayItem#0의 Clip_Information_file_name에 기술되어 있는 "00003.CLP"로부터 특정되는 클립 스트림 파일 "00003.PS"이며, 따라서, Mark#0의 mark_time_stamp가 나타내는, 전술한 시각 90,000은, 클립 스트림 파일 "00003.PS"의 시각이다.

[0594] 도 28 하측에서, PlayList #1에 포함되는 3개의 Mark() 중의 2번째의 Mark()인 Mark#1은, mark_type(도 7)이 'Event'로 되어 있으므로, 이벤트 마크이다. 또한, Mark#1은, ref_to_PlayItem_id(도 7)이 0으로 되어 있으므로, PlayList #1에 포함되는 도 25의 1개의 PlayItem#0에 속한다. 또한, Mark#1은, mark_time_stamp이 27,090,000 로 되어 있으므로, PlayList #1에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일의 시각 27,090,000 상의 표시이다. 또한, Mark#1은, entry_ES_stream_id가 0xE0이고, entry_ES_private_stream_id가 0으로 되어 있으므로, stream_id가 0xE0에 의해 특정(식별)되는 엘리멘터리 스트림, 즉, 도 20 및 도 22에서 설명한 바와 같이, 비디오 스트림에 관련지어져 있다. 또한, Mark#1은, mark_data가 1로 되어 있으므로, 인수로서 1을 수반하는 이벤트를 발생시킨다.

[0595] 또한, 여기에서는, PlayList #1에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일은, 전술한 바와 같이, 클립 스트림 파일 "00003.PS"이며, 따라서, Mark#1의 mark_time_stamp이 나타내는, 전술한 시각 27,090,000은, 클립 스트림 파일 "00003.PS"의 시각이다.

[0596] 또한, Mark#1이 관련지어져 있는, stream_id가 0xE0인 비디오 스트림은, 그 Mark#1이 속하는, 도 25의 PlayList #1에 포함되는 PlayItem#0의 Clip_Information_file_name에 기술되어 있는 "00003.CLP"에 기술되어 있는 stream_id가 0xE0의 비디오 스트림, 즉, 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"로부터 특정되는, 클립 스트림 파일"00003.PS"에 다중화되어 있는 3개의 엘리멘터리 스트림 stream#0 내지 #2 중의, 1개째의 엘리멘터리 스트림(비디오 스트림) stream#0다.

[0597] 다음으로, 도 28 하측에서, PlayList #1에 포함되는 3개의 Mark() 중의 3번째의 Mark()인 Mark#2은, mark_type(도 7)이 'Event'로 되어 있으므로, 이벤트 마크이다. 또한, Mark#2은, ref_to_PlayItem_id(도 7)이 0으로 되어 있으므로, PlayList #1에 포함되는 도 25의 1개의 PlayItem#0에 속한다. 또한, Mark#1은, mark_time_stamp이 27,540,000으로 되어 있으므로, PlayList #1에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일의 시각 27,540,000 상의 표시이다. 또한, Mark#2은, entry_ES_stream_id가 0xE1이고, entry_ES_private_stream_id가 0으로 되어 있으므로, stream_id가 0xE1에 의해 특정(식별)되는 엘리멘터리 스트림, 즉, 도 20 및 도 22에서 설명한 바와 같이, 비디오 스트림에 관련지어져 있다. 또한, Mark#2은, mark_data가 2로 되어 있으므로, 인수로서 2를 수반하는 이벤트를 발생시킨다.

[0598] 또한, 여기에서는, PlayList #1에 포함되는 PlayItem#0에 의해 재생되는 클립 스트림 파일은, 전술한 바와 같이, 클립 스트림 파일 "00003.PS"이며, 따라서, Mark#2가 나타내는, 전술한 시각 27,540,000은, 클립 스트림 파일 "00003.PS"의 시각이다.

[0599] 또한, Mark#2이 관련지어져 있는, stream_id가 0xE1인 비디오 스트림은, 그 Mark#2이 속하는, 도 25의 PlayList #1에 포함되는 PlayItem#0의 Clip_Information_file_name에 기술되어 있는 "00003.CLP"에 기술되어 있는 stream_id가 0xE1인 비디오 스트림, 즉, 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"로부터 인식되는, 클립 스트림 파일"00003.PS"에 다중화되어 있는 3개의 엘리멘터리 스트림 stream#0 내지 #2 중의, 2개째의 엘리멘터리 스트림(비디오 스트림) stream#1이다.

[0600] 여기에서, 도 28 하측에서는, PlayList #1의 PlayListMark0의 일람표의 난외의 우측에, Mark()이 속하는 PlayItem()의 선두로부터의 시각을 도시하고 있다.

[0601] 또한, 도 28에서는, 챕터 마크나 인덱스 마크가 나타내는 챕터나 인덱스의 번호가, mark_data에 기술되어 있지만, 챕터 마크나 인덱스 마크가 나타내는 챕터나 인덱스의 번호는, mark_data에 기술하지 않아도, 예를 들면, PlayListMark()에 있어서의 챕터 마크나 인덱스 마크의 수를 카운트 함으로써 인식할 수 있다.

[0602] [디스크 장치의 동작 설명]

[0603] 다음으로, 도 1의 디스크(101)에, 도 25 내지 도 28에서 설명한 바와 같은 데이터(파일)이 기록되어 있는 것으로 하여, 도 1의 디스크 장치의 동작에 대해서 설명한다.

[0604] 디스크(101)가 디스크 드라이브(102)에 삽입되면, 그 취지를 나타내는 메시지가 드라이브 인터페이스(114), 나아가서는, 도 2A 및 도 2B의 오퍼레이팅 시스템(201)을 경유하여, 비디오 콘텐츠 재생 프로그램(210)에 전해진다. 비디오 콘텐츠 재생 프로그램(210)은, 디스크(101)가 디스크 드라이브(102)에 삽입된 취지의 메시지를 오퍼레이팅 시스템(201)으로부터 수신하면, 도 29의 재생 전처리를 시작한다.

[0605] 「재생 전처리」

[0606] 즉, 도 29는, 비디오 콘텐츠 재생 프로그램(210)이 행하는 재생 전처리를 설명하는 플로우차트이다.

[0607] 여기에서, 이하, 플로우차트에 의해 설명하는 디스크 장치의 동작 또는 처리는, 반드시 플로우차트로서 기재된 순서에 따라 시계열로 행해질 필요는 없고, 병렬적 혹은 개별로 행해지는 경우도 있다. 단, 본 명세서에서는, 편의상, 디스크 장치의 동작 또는 처리를, 플로우차트를 참조하여 설명한다.

[0608] 재생 전처리에서는, 비디오 콘텐츠 재생 프로그램(210)은, 스텝 S101에서, 오퍼레이팅 시스템(201)의 파일 시스템 기능을 사용하여, 디스크(101)을 체크하여, 디스크(101)가, 비디오 콘텐츠 재생 프로그램(210)용의 정상의 디스크인지의 여부를 판정한다.

[0609] 여기에서, 전술한 바와 같이, 디스크(101)에의 액세스(파일의 읽어내기)는, 오퍼레이팅 시스템(201)의 파일 시스템 기능을 사용해서 행해지는데, 이하에서는, 그 설명은, 적절하게 생략한다.

[0610] 스텝 S101에서, 디스크(101)가 정상의 디스크가 아닌 것으로 판정된 경우, 즉, 예를 들면, 디스크(101)에 채용되어 있는 파일 시스템이, 오퍼레이팅 시스템(201)이 대응하지 않는 타입이었던 경우나, 루트 디렉토리에 "VIDEO" 디렉토리가 놓여져 있지 않은 경우, 비디오 콘텐츠 재생 프로그램(210)은 디스크(101)에 대응해 있지 않다고 판단하고, 스텝 S102로 진행하여, 그래픽스 처리 모듈(219)이, 에러 처리를 행하여, 재생 전처리를 종료한다.

[0611] 즉, 그래픽스 처리 모듈(219)은, 에러 처리로서, 디스크(101)가 정상이 아니라는 취지의 에러 메시지(의 비디오 데이터)을 생성하고, 비디오 출력 모듈(220)로부터 출력시키고, 이에 의해, 에러 메시지를 표시시킨다. 또한, 에러 처리로서는, 기타, 예를 들면, 오디오 출력 모듈(221)로부터의 고음의 출력이나, 디스크 드라이브(102)로부터의 디스크(101)의 배출 등을 행하도록 하는 것이 가능하다.

[0612] 한편, 스텝 S101에서, 디스크(101)가 정상의 디스크라고 판정된 경우, 스텝 S103으로 진행하여, 비디오 콘텐츠 재생 프로그램(210)은, 콘텐츠 데이터 공급 모듈(213)에 의해, 오퍼레이팅 시스템(201)에 대하여, 디스크(101)(도 4)의 "VIDEO" 디렉토리에 놓여 있는 "SCRIPT.DAT" 파일과, "PLAYLIST.DAT" 파일의 2개의 데이터 파일을 요구해서 읽어들이고, 스텝 S104로 진행한다. 스텝 S104에서는, "SCRIPT.DAT" 파일이, 스크립트 제어 모듈(211)에 공급됨과 함께, "PLAYLIST.DAT" 파일이, 플레이어 제어 모듈(212)에 공급된다.

[0613] 그 후, 스텝 S104로부터 S105 내지 S107로 진행하여, 플레이어 제어 모듈(212)은, 초기화 처리를 행한다. 또

한, 스크립트 제어 모듈(211)은, 플레이어 제어 모듈(212)의 초기화 처리가 종료할 때까지 대기한다.

[0614] 「플레이어 제어 모듈(212)의 초기화 처리」

[0615] 초기화 처리에서는, 스텝 S105에서, 플레이어 제어 모듈(212)이, "PLAYLIST.DAT" 파일의 해석을 행하고, "PLAYLIST.DAT" 파일 중에서 사용되어져 있는 클립 정보 파일의 수와 그 파일명을 조사한다.

[0616] 즉, 지금의 경우, "PLAYLIST.DAT" 파일은, 도 25에 도시한 것으로 되어 있고, 플레이어 제어 모듈(212)은, 도 25의 "PLAYLIST.DAT" 파일에 있어서 number_of_PlayLists가 2로 되어 있으므로, 1번째의 PlayList #0와 2번째의 PlayList #1의 2개의 PlayList()가 존재하는 것을 인식한다. 또한, 플레이어 제어 모듈(212)은, 도 25의 "PLAYLIST.DAT" 파일에 있어서의 1번째의 PlayList #0에 대해서, number_of_PlayItems가 2로 되어 있으므로, 그 PlayList #0에는, 1번째의 PlayItem#0와 2번째의 PlayItem#1과의 2개의 PlayItem()이 존재하는 것을 인식한다. 그리고, 플레이어 제어 모듈(212)은, 도 25의 "PLAYLIST.DAT" 파일에 있어서의, PlayList #0에 포함되는 1번째의 PlayItem#0와 2번째의 PlayItem#1 각각의 Clip_Information_file_name을 참조함으로써, PlayList #0에 포함되는 1번째의 PlayItem#0의 클립 정보 파일(의 파일명)이 "00001.CLP"이며, 2번째의 PlayItem#1의 클립 정보 파일이 "00002.CLP"인 것을 인식한다.

[0617] 플레이어 제어 모듈(212)은, 2번째의 PlayList #1에 대해서도 마찬가지로 하여, 그 number_of_PlayItems가 1이므로, 1개의 PlayItem()(PlayItem#0)이 존재하고, 또한, 그 PlayItem#0 중의 Clip_Information_file_name으로부터, 그 PlayItem#0의 클립 정보 파일이 "00003.CLP"인 것을 인식한다.

[0618] 그 후, 스텝 S105로부터 S106으로 진행하여, 플레이어 제어 모듈(212)은, 디스크(101)로부터, 스텝 S105에서 인식한 클립 정보 파일, 즉, 디스크(101)의 "VIDEO" 디렉토리 내의 "CLIP" 디렉토리로부터 3개의 클립 정보 파일 "00001.CLP", "00002.CLP", "00003.CLP"을 읽어들인다.

[0619] 여기에서, 스텝 S106에서의 클립 정보 파일의 읽어들이기는, 처음에 재생되는 PlayList()의 PlayItem의 클립 정보 파일만으로 충분하지만, 본 실시 형태에서는, 전술한 바와 같이, 모든 PlayList()의 PlayItem()의 클립 정보 파일을 미리 읽기해 두는 것으로 한다.

[0620] 스텝 S106의 처리 후에는, 스텝 S107로 진행하여, 플레이어 제어 모듈(212)은, 스텝 S105에서 인식한 클립 정보 파일의 읽어들이기에 성공한 것인지 여부를 판정하고, 또한, 그 읽어들인 클립 정보 파일에 대응하는 클립 스트림 파일이, 디스크(101)에 존재하는지의 여부를 판정(체크) 한다. 즉, 스텝 S107에서는, 클립 정보 파일 "00001.CLP", "00002.CLP", "00003.CLP"의 읽어들이기에 성공하고, 또한, 그 클립 정보 파일 "00001.CLP", "00002.CLP", "00003.CLP" 각각과 파일명의 확장자만이 상이한 클립 스트림 파일 "00001.PS", "00002.PS", "00003.PS"이, 디스크(101)의 "VIDEO" 디렉토리 아래에 있는 "STREAM" 디렉토리에 존재하는지의 여부를 판정한다.

[0621] 스텝 S107에서, 스텝 S105에서 인식한 클립 정보 파일의 읽어들이기에 실패한 것으로 판정되거나, 또는, 클립 정보 파일에 대응하는 클립 스트림 파일이, 디스크(101)에 존재하지 않는 것으로 판정된 경우, 즉, 예를 들면, "PLAYLIST.DAT" 파일에 따른 재생에 필요한 클립 정보 파일이나 클립 스트림 파일이, 디스크(101)에 기록되어 있지 않은 경우, 비디오 콘텐츠 재생 프로그램(210)은, 디스크(101)에 대응해 있지 않다고 판단하고, 스텝 S102로 진행하여, 전술한 에러 처리가 행해지고, 재생 전처리를 종료한다.

[0622] 한편, 스텝 S107에서, 스텝 S105에서 인식한 클립 정보 파일의 읽어들이기에 성공하고, 또한, 클립 정보 파일에 대응하는 클립 스트림 파일이, 디스크(101)에 존재하는 것으로 판정된 경우, 플레이어 제어 모듈(212)은, 초기화 처리를 종료하고, 스텝 S108로 진행한다.

[0623] 스텝 S108에서는, 스크립트 제어 모듈(211)이, "SCRIPT.DAT" 파일의 해석과 실행을 시작한다.

[0624] 예를 들면, 지금, 스크립트 제어 모듈(211)이, "SCRIPT.DAT" 파일을 실행함으로써, 1번째의 PlayList()(PlayList #0)의 재생이, 플레이어 제어 모듈(212)에 지시된 것으로 하면, 도 30의 재생 처리가 행해진다.

[0625] 「재생 처리」

[0626] 즉, 도 30는, 비디오 콘텐츠 재생 프로그램(210)이 행하는 재생 처리를 설명하는 플로우차트이다.

[0627] 「재생 준비 처리」

[0628] 재생 처리에서는, 플레이어 제어 모듈(212)은, 우선 스텝 S121과 S122에서, 스크립트 제어 모듈(211)로부터

재생이 지시된 PlayList(), 즉, 1번째의 PlayList()(PlayList #0)의 재생 준비 처리를 행한다.

[0629] 즉, 플레이어 제어 모듈(212)은, 스텝 S121에서, 1번째의 PlayList #0에 포함되는 1번째의 PlayItem#0의 IN_time(도 6)을 확인하고, 스텝 S122로 진행하여, 1번째의 PlayList #0에 포함되는 1번째의 PlayItem#0에 의해 재생되는 클립 스트림 파일 "00001.PS" 상의, 그 PlayItem#0의 IN_time에 상당하는, 재생을 시작할 재생 개시 위치를 조사한다.

[0630] 여기에서, PlayItem()의 IN_time(도 6)이, 클립 스트림 파일의 선두를 지시하고 있는 경우에는, 클립 스트림 파일의 선두로부터, 프로그램 스트림을 읽어내면 되지만, IN_time이, 클립 스트림 파일의 선두 이외를 지시하고 있는 경우에는, IN_time에 대응하는 위치를 찾아내어(조사하여), 거기로부터 프로그램 스트림을 읽어낼 필요가 있다.

[0631] 구체적으로는, 도 25에 도시한 경우, 1번째의 PlayList #0에 포함되는 1번째의 PlayItem#0의 IN_time은, 180,090이다. 플레이어 제어 모듈(212)은, 1번째의 PlayList #0에 포함되는 1번째의 PlayItem#0에 의해 재생되는 클립 스트림 파일 "00001.CLP"의, 도 27에 나타낸 EP_map()로부터, PlayItem#0의 IN_time인 180,090에 알맞은 재생 개시 위치를 찾아낸다.

[0632] 즉, 플레이어 제어 모듈(212)은, 예를 들면, EP_map()에 기술된 디코드 개시 가능점을 나타내는 PTS_EP_start 중의, 식 PTS_EP_start≤IN_time을 만족하는 최대의 PTS_EP_start를, 이분 탐색(바이너리 서치) 등을 이용해서 검색한다. 여기에서, IN_time 이하의 PTS_EP_start를 검색하는 것은, IN_time에 의해 나타내는 위치가, 디코드 개시 가능점이다라고는 할 수 없기 때문이다.

[0633] 지금의 경우, IN_time은, 전술한 바와 같이, 180,090이다. 또한, 1번째의 PlayList #0에 포함되는 1번째의 PlayItem#0에 의해 재생되는 클립 스트림 파일 "00001.CLP"의, 도 27에 나타낸 EP_map()에서는, 식 PTS_EP_start≤IN_time을 만족하는 최대의 PTS_EP_start로서, 180,090이 기술되어 있다. 따라서, 플레이어 제어 모듈(212)에서는, 도 27에 나타낸 EP_map()로부터, 그 180,090이 되어 있는 PTS_EP_start가 검색된다.

[0634] 또한, 플레이어 제어 모듈(212)에서는, 그 검색된 PTSEP_start에 대응하는 RPN_EP_start인 305(섹터)가 읽어내어지고, 그 305인 RPN_EP_start에 의해 나타내는 클립 스트림 파일 "00001.PS" 상의 위치가, 재생 개시 위치로서 결정된다.

[0635] 플레이어 제어 모듈(212)은, 이상과 같이 해서 재생 개시 위치를 결정하면, 스텝 S122로부터 S123으로 진행하여, 타임 코드를 표시하도록, 그래픽스 처리 모듈(219)을 제어한다. 그래픽스 처리 모듈(219)은, 플레이어 제어 모듈(212)의 제어에 따라, 타임 코드(의 비디오 데이터)을 생성해서 비디오 출력 모듈(220)에 출력한다. 이에 의해, 타임 코드의 표시가 개시된다.

[0636] 여기에서, 스텝 S123에서 표시가 개시되는 타임 코드는, 예를 들면, PlayList()의 선두를 00:00:00(시간:분:초)으로 환산한 값으로 한다. 또한, 타임 코드와 함께, 또는 타임 코드가 아니라, 챕터나 인덱스의 번호를 표시하도록 해도 된다.

[0637] 「PlaylistMark()의 해석 처리」

[0638] 스텝 S123에서 타임 코드의 표시가 개시된 후에는, 스텝 S124로 진행하여, 플레이어 제어 모듈(212)은, 스크립트 제어 모듈(211)로부터 재생이 지시된 PlayList(), 즉, 1번째의 PlayList()(PlayList #0)에 기술되어 있는 PlayListMark()(도 7)을 해석하는 해석 처리를 행한다.

[0639] 구체적으로는, 플레이어 제어 모듈(212)은, 이미 읽어들인 "PLAYLIST.DAT" 파일에 있어서의 1번째의 PlayList #0의, 도 28 상측에 나타낸 PlayListMark()에서, number_of_PlayList_marks가 7로 되어 있으므로, 그 PlayList #0에 포함되는 Mark()의 수가 7인 것을 인식한다.

[0640] 또한, 플레이어 제어 모듈(212)은, PlayList #0에 포함되는, 도 28 상측의 7개의 Mark()을 해석하고, 그 ref_to_PlayItem_id로부터, 7개의 Mark() 중의 1번째로부터 4번째까지의 4개의 Mark()가, PlayList #0의 1번째의 PlayItem()(PtayItem#0)에 속해 있는 것을 인식한다.

[0641] 그 후, 플레이어 제어 모듈(212)은, PlayList #0의 1번째의 PlayItem#0에 속해 있는 4개의 Mark()의 mark_time_stamp을 취출하고, 요소수가 4의 배열로서, 디코드 제어 모듈(214)에 건네준다. 즉, 이에 의해, 도 28 상측의 7개의 Mark() 중의 1번째로부터 4번째까지의 4개의 Mark() 각각의 mark_time_stamp인 {180,090}, {5,580,090}, {10,980,090}, {16,380,090}의 4개의 시각이, 플레이어 제어 모듈(212)로부터, 디코드 제어 모듈(214)에 건네진다. 이 때 이들 시각의 속성은 「마크 처리」인 것도, 플레이어 제어 모듈

(212)로부터, 디코드 제어 모듈(214)에 전해진다. 디코드 제어 모듈(214)은, 계시부(214A)에 의해 시간을 재고 있는 시각이, 「마크 처리」의 속성의 시각에 일치한 때, 그 취지를 나타내는 메시지, 「마크 처리」의 속성의 시각에 일치한 시각, 및 「마크 처리」의 속성을, 플레이어 제어 모듈(212)에 전한다.

[0642] 「재생할 엘리멘터리 스트림의 결정 처리」

[0643] 다음으로, 스텝 S124로부터 S125로 진행하여, 플레이어 제어 모듈(212)은, 재생할 엘리멘터리 스트림을 결정한다.

[0644] 즉, 플레이어 제어 모듈(212)은, 스크립트 제어 모듈(211)로부터 재생이 지시된 PlayList()인 1번째의 PlayList #0에 있어서의 1번째의 PlayItem#0(도 25)의 Clip_Information_fime_name에 파일명이 기술되어 있는, 도 26A 및 도 26B의 클립 정보 파일 "00001.CLP"에서, number_of_streams가 4로 되어 있으므로, 대응하는 클립 스트림 파일 "00001.PS"에, 4개의 엘리멘터리 스트림이 다중화되어 있는 것을 인식한다. 또한, 플레이어 제어 모듈(212)은, 그 4개의 엘리멘터리 스트림에 대한, 도 26A 및 도 26B의 클립 정보 파일 "00001.CLP"의 StaticInfo()의 stream_id와 필요한 private_stream_id를 순서대로 조사하고, 그 4개의 엘리멘터리 스트림이, 1개의 비디오 스트림, 1개의 ATRAC 오디오 스트림, 및 2개의 자막 스트림인 것을 인식한다. 즉, 클립 스트림 파일"00001.PS"에 다중화되어 있는 각 속성의 엘리멘터리 스트림의 개수가 인식된다.

[0645] 또한, 클립 스트림 파일에 다중화되어 있는 각 속성의 엘리멘터리 스트림의 개수의 정보는, 재생 중에서의 엘리멘터리 스트림의 절환(오디오 절환, 자막 절환 등)에 사용된다. 또한, 자막 스트림은, 클립 스트림 파일 중에 존재하지 않는(콘텐츠에 자막이 포함되지 않는) 경우가 있고, 자막 스트림이 존재하는지의 여부의 판단에, 「자막 스트림」의 속성의 엘리멘터리 스트림의 개수의 정보가 사용된다.

[0646] 플레이어 제어 모듈(212)은, 이상과 같은 StaticInfo()의 조사의 결과에 기초하여, 재생할 엘리멘터리 스트림을 선택, 결정하지만, 지금의 경우, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림 중에, 「비디오 스트림」과 「오디오 스트림」의 속성의 엘리멘터리 스트림은, 각각 1개밖에 없으므로, 「비디오 스트림」과 「오디오 스트림」의 속성의 엘리멘터리 스트림에 대해서는, 선택의 여지가 없고, 그 1개의 비디오 스트림과 오디오 스트림(ATRAC 오디오 스트림)이, 재생할 엘리멘터리 스트림으로서 결정된다.

[0647] 또한, 「자막 스트림」의 속성의 엘리멘터리 스트림에 대해서는, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림 중에 2개 존재하므로, 그 2개의 자막 스트림 중의 어느쪽인가 1개의 자막 스트림이, 재생할 엘리멘터리 스트림으로서 선택, 결정된다. 여기에서는, 예를 들면, 2개의 자막 스트림 중의, 클립 정보 파일 "00001.CLP"에서의 출현순으로 최초의 자막 스트림이 선택되는 것으로 한다.

[0648] 여기에서, 전술된 바와 같이, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림의 속성과 개수를 인식하는데 있어서는, 그 4개의 엘리멘터리 스트림 각각을 특정할 필요가 있지만, 플레이어 제어 모듈(212)은, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림의 특정을, stream_id와 필요한 private_stream_id에 의해 행한다.

[0649] 즉, 플레이어 제어 모듈(212)은, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림 중의, 「비디오 스트림」의 속성의 엘리멘터리 스트림인 비디오 스트림을, 도 26A 및 도 26B에서 클립 정보 파일 "00001.CLP"에 대해서 설명한 바와 같이, 0xE0로 되어 있는 stream_id에 의해 특정한다.

[0650] 또한, 플레이어 제어 모듈(212)은, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림 중의, 「오디오 스트림」의 속성의 엘리멘터리 스트림인 ATRAC 오디오 스트림을, 도 26A 및 도 26B에서 클립 정보 파일 "00001.CLP"에 대해서 설명한 바와 같이, 0xBD로 되어 있는 stream_id, 및 0x00로 되어 있는 private_stream_id에 의해 특정한다.

[0651] 또한, 플레이어 제어 모듈(212)은, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림에 있어서의 「자막 스트림」의 속성의 엘리멘터리 스트림인 2개의 자막 스트림 각각을, 도 26A 및 도 26B에서 클립 정보 파일 "00001.CLP"에 대해서 설명한 바와 같이, 0xBD로 되어 있는 stream_id, 및 0x80이 되어 있는 private_stream_id와, 0xBD로 되어 있는 stream_id, 및 0x81로 되어 있는 private_stream_id에 의해, 각각 특정한다.

[0652] 이상과 같이, 클립 스트림 파일에 대응하는 클립 정보 파일의 메타데이터로서 기술되는 stream_id와 private_stream_id의 조합에 의해, 그 클립 스트림 파일에 다중화되어 있는 엘리멘터리 스트림을 특정할 수 있다.

[0653] 여기에서, stream_id와 private_stream_id의 조합은, MPEG2-System의 다중화를 확장하기 위해서 설치한 메카니즘이다. 이 stream_id와 private_stream_id의 조합을, 메타데이터(데이터 베이스)에서, 엘리멘터리 스트림을 특정하기 위해서 사용함으로써, 엘리멘터리 스트림을 확실하게 특정하는 것이 가능하게 된다. 또한, 장래, private_stream_id의 의미를 확장하고, 대응하는 엘리멘터리 스트림의 개수나 종류(속성)를 늘린 경우에도 지금의 메카니즘을 그대로 사용가능하기 때문에, 확장성에 있어서 우월하다.

[0654] 즉, 예를 들면, BD(Blue ray Disc) 규격에서는, 데이터의 특정에, MPEG2 규격의 트랜스포트 스트림(Transport Stream)의 PID(Packet ID)가 이용되기 때문에, MPEG2 규격에 구속된다. 또한, 예를 들면, DVD_Video규격에서는, private_stream_id에 유사한 sub_stream_id가 정의되어 있지만, sub_stream_id은, 스트림의 특정 때문에 데이터베이스 상에 기술할 수 있도록 되어 있지 않고, 8개 혹은 32개의 스트림 정보를 기술하는 고정적인 영역에 기술할 수 있는 것에 지나지 않기 때문에(예를 들면 VI4-49, Table4.2.1-2(VTS-AST_ATRT)이나 VI4-52, Table4.2.1-3(VTS_SPST_ATRT)등 참조), 확장성이 부족하다.

[0655] 이것에 대하여, stream_id와 private_stream_id의 조합은, 메타데이터가 기술되는, 예를 들면, 도 10의 클립 정보 파일 Clip()에서, number_of_streams으로 나타낼 수 있는 수만큼 기술할 수가 있고, 따라서, 클립 스트림 파일에 다중화되어 있는 엘리멘터리 스트림을, 그 수에 상관없이(단, number_of_streams로 나타낼 수 있는 수의 범위), 클립 정보 파일 Clip()에 기술된 메타데이터로서의 stream_id와 private_stream_id의 조합으로부터 특정하는 것이 가능하게 된다.

[0656] 또한, 본 실시 형태에서는, stream_id와 private_stream_id의 조합은, 도 10의 클립 정보 파일에서, 대응하는 클립 스트림 파일에 다중화되어 있는 엘리멘터리 스트림을 특정하는데에도 사용되는 것 외에, 예를 들면, 도 7의 PlayListMark()에 있어서의 entry_ES_stream_id와 entry_ES_private_stream_id의 조합으로서, Mark()을 관련짓는 엘리멘터리 스트림의 특정에도 사용된다. 또한, stream_id와 private_stream_id의 조합은, 그 외, 예를 들면, 도 14의 EP_map()에서, 디코드 가능 개시점의 정보를 기술하는 엘리멘터리 스트림의 특정에도 사용된다.

[0657] 「출력 속성의 제어 처리」

[0658] 그 후, 스텝 S125로부터 S126으로 진행하여, 플레이어 제어 모듈(212)은, 재생 대상의 엘리멘터리 스트림, 즉, 스텝 S125에서 재생한다고 결정한 엘리멘터리 스트림의 출력 속성의 제어 처리를 행한다.

[0659] 구체적으로는, 플레이어 제어 모듈(212)은, 우선, 재생 대상의 엘리멘터리 스트림, 즉, 스텝 S125에서 재생한다고 결정한 비디오 스트림, ATRAC 오디오 스트림, 자막 스트림의 각각에 대해서, 출력 속성이 기술되는 DynamicInfo()(도 13)의 수를 나타내는 number_of_DynamicInfo(도 10)을 조사한다.

[0660] 여기서, 지금의 경우, 재생 대상의 비디오 스트림, ATRAC 오디오 스트림, 자막 스트림은, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 엘리멘터리 스트림이며, 그들의 number_of_DynamicInfo는, 도 26A 및 도 26B의 "00001.CLP"에서 설명한 바와 같이, 모두 0으로 되어 있다. 이와 같이, 재생 대상의 엘리멘터리 스트림의 전부에 대해서, number_of_DynamicInfo가 0인 경우, 플레이어 제어 모듈(212)은, 재생 대상의 엘리멘터리 스트림의 출력 속성의 제어 처리로서는, 특별히 처리를 행하지 않는다.

[0661] 또한, 재생 대상의 엘리멘터리 스트림에 대한 number_of_DynamicInfo가 0이 아닌 경우에, 그 엘리멘터리 스트림의 출력 속성의 제어로서 행해지는 처리에 대해서는, 후술한다.

[0662] 「재생 개시의 준비 처리」

[0663] 스텝 S126의 처리 후에는, 스텝 S127로 진행하여, 플레이어 제어 모듈(212)은, 재생 대상의 엘리멘터리 스트림의 재생 개시의 준비 처리를 행한다.

[0664] 즉, 플레이어 제어 모듈(212)은, 콘텐츠 데이터 공급 모듈(213)에 대하여, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일 "00001.PS"의 파일명과, 스텝 S122에서 결정한 재생 개시 위치인 EP_map()에 기술된 RPN_EP_start(=305)를 제공한다.

[0665] 또한, 플레이어 제어 모듈(212)은, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일 "00001_PS에 저장된 프로그램 스트림의, 버퍼 제어 모듈(215)에의 공급이 개시되기 전에, 버퍼 제어 모듈(215)을 초기화한다.

[0666] 구체적으로는, 버퍼 제어 모듈(215)(도 3)에서는, 데이터 선두 포인터 기억부(231)에 기억되는 데이터 선두 포인터, 데이터 써넣기 포인터 기억부(232)에 기억되는 데이터 써넣기 포인터, 비디오 읽어내기 포인터 기억

부(241)에 기억되는 비디오 읽어내기 포인터, 오디오 읽어내기 포인터 기억부(251)에 기억되는 오디오 읽어내기 포인터, 자막 읽어내기 포인터 기억부(262)에 기억되는 자막 읽어내기 포인터에, 동일한 값이 대입된다.

[0667] 이에 의해, 데이터 선두 포인터 기억부(231)에 기억된 데이터 선두 포인터와, 데이터 써넣기 포인터(232)에 기억된 데이터 써넣기 포인터는, 버퍼 제어 모듈(215)의 버퍼(215A)의 동일 위치를 가리킨다. 이것은, 버퍼(215A)에, 유효한 데이터가 축적되어 있지 않은 상태를 나타낸다.

[0668] 또한, 플레이어 제어 모듈(212)은, 재생 대상의 엘리멘터리 스트림을 식별(특정)하기 위한 식별 정보로서의 stream_id, 또한, 필요에 따라, private_stream_id를, 버퍼 제어 모듈(215)에 공급한다.

[0669] 즉, 전술한 바와 같이, 재생 대상의 엘리멘터리 스트림 중의, 「비디오 스트림」의 속성의 비디오 스트림은, 0xE0으로 되어 있는 stream_id에 의해 특정되며, 「오디오 스트림」의 속성의 ATRAC 오디오 스트림은, 0xBD로 되어 있는 stream_id, 및 0x00으로 되어 있는 private_stream_id에 의해 특정되고, 「자막 스트림」의 속성의 자막 스트림은, 0xBD로 되어 있는 stream_id, 및 0x80으로 되어 있는 private_stream_id에 의해 특정된다. 플레이어 제어 모듈(212)은, 이들 stream_id와 private_stream_id를, 버퍼 제어 모듈(215)에 공급한다.

[0670] 버퍼 제어 모듈(215)(도 3)에서는, 비디오 읽어내기 기능부(233)가, 플레이어 제어 모듈(212)로부터의, 비디오 스트림에 대한 0xE0으로 되어 있는 stream_id를, stream_id 레지스터(242)에 기억시킨다. 또한, 오디오 읽어내기 기능부(234)가, 플레이어 제어 모듈(212)로부터의, 0xBD로 되어 있는 stream_id와, 0x00으로 되어 있는 private_stream_id를, stream_id 레지스터(252)와 private_stream_id 레지스터(253)에, 각각 기억시킨다. 또한, 자막 읽어내기 기능부(235)가, 플레이어 제어 모듈(212)로부터의, 0xBD로 되어 있는 stream_id와, 0x80으로 되어 있는 private_stream_id를, stream_id 레지스터(263)와 private_stream_id 레지스터(264)에, 각각 기억시킨다.

[0671] 또한, 플레이어 제어 모듈(212)은, 버퍼 제어 모듈(215)에 공급한 재생 대상의 엘리멘터리 스트림의 stream_id와 private_stream_id를, 금후의 처리를 위해 기억한다. 플레이어 제어 모듈(212)은, 이들 stream_id나 Private_stream_id를, 후술하는 스트림 절환을 요구하는 메시지의 발생 시나, 후술하는 마크 처리에서 지금재생 중인 스트림을 특정하기 위해 사용한다.

[0672] 플레이어 제어 모듈(212)은, 버퍼 제어 모듈(215)(도 3)의 초기화로서, 또한, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일에 따른 값의 자막 읽어내기 기능 플래그를, 자막 읽어내기 기능 플래그 기억부(261)에 세트한다.

[0673] 즉, 지금의 경우, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일 "00001.PS"에는, 자막 스트림이 포함되기 때문에, 자막 읽어내기 기능부(235)를 기능시키기 위해, 값이 1인 자막 읽어내기 기능 플래그가, 자막 읽어내기 기능 플래그 기억부(261)에 세트된다. 또한, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일에 자막 스트림이 포함되어 있지 않은 경우, 자막 읽어내기 기능 플래그 기억부(261)에는, 값이 0인 자막 읽어내기 기능 플래그가 세트된다. 이 경우, 자막 읽어내기 기능부(235)는 기능하지 않는다(특별히 처리를 행하지 않는다).

[0674] 또한, 플레이어 제어 모듈(212)은, 스크립트 제어 모듈(211)로부터 재생을 지시받은 1번째의 PlayList#0에 포함되는 1번째의 PlayItem#0(도 25)의 IN_time인 180,090과, OUT_time인 27,180,090을, 디코드 제어 모듈(214)에 대하여 제공한다. 디코드 제어 모듈(214)에서는, IN_time은, PlayItem()에 의해 재생되는 클립의 디코드 개시의 제어에, OUT_time은, 그 클립의 디코드 종료, 또한, 후술하는 PlayItem 갈아타기의 제어에, 각각 사용된다.

[0675] 또한, 플레이어 제어 모듈(212)은, 그래픽 처리 모듈(219)에 대한 자막 스트림의 표시 방식의 지시를 초기화한다. 즉, 플레이어 제어 모듈(212)은, 자막 스트림의 표시 방식을, 예를 들면, 디폴트의 표시 방식으로 하도록, 그래픽 처리 모듈(219)을 제어한다.

[0676] 「데이터 읽어들이기 개시」

[0677] 그 후, 스텝 S127로부터 S128로 진행하여, 플레이어 제어 모듈(212)은, 콘텐츠 데이터 공급 모듈(213)을 제어하고, 이에 의해, 콘텐츠 데이터 공급 모듈(213)은, 오퍼레이팅 시스템(201)의 기능을 사용하여, 재생 대상의 엘리멘터리 스트림이 다중화된 프로그램 스트림이 저장된 클립 스트림 파일을 읽어낸다. 즉, 콘텐츠 데이터 공급 모듈(213)은, 디스크(101)(도 4)의 "VIDEO" 디렉토리 아래에 있는 "STREAM" 디렉토리의 클립 스트림 파일 "00001.PS"를 지정하고, 또한, 스텝 S122에서 결정된 재생 개시 위치인 305섹터를 지정하고, 오퍼레이팅 시스템(201)에 대하여 파일 읽어내기를 요구한다. 또한, 콘텐츠 데이터 공급 모듈(213)은, 디스크(101)로부

터 읽어낸 데이터를, 버퍼 제어 모듈(215)에 공급하도록 지정한다.

[0678] 이에 의해, 디스크(101)로부터의, 클립 스트림 파일 "00001.PS"에 저장된 프로그램 스트림의 읽어내기가 개시되고, 그 프로그램 스트림은, 버퍼 제어 모듈(215)에 공급된다.

[0679] 버퍼 제어 모듈(215)(도 3)은, 디스크(101)로부터 읽어내어져 공급된 프로그램 스트림을, 버퍼(215A)의 데이터 써넣기 포인터 기억부(232)의 데이터 써넣기 포인터가 가리키는 위치에 써넣고, 써넣은 데이터의 사이즈만큼 데이터 써넣기 포인터를 인크리먼트한다.

[0680] 여기서, 이하, 특별히 언급하지 않는 한, 콘텐츠 데이터 공급 모듈(213)은, 버퍼 제어 모듈(215)의 버퍼(215A)에 빈 부분이 있으면, 디스크(101)로부터 데이터를 읽어내어, 버퍼 제어 모듈(215)의 버퍼(215A)에 공급하여 기억시키는 것으로 한다. 따라서, 버퍼(215A)에는, 항상, 충분한 데이터가 축적되어 있는 것으로 한다.

[0681] 「디코더 제어 개시」

[0682] 이상과 같이 하여, 디스크(101)로부터의 데이터의 읽어내기가 개시되어, 그 데이터가 버퍼 제어 모듈(215)의 버퍼(215A)에 축적되기 시작하면, 스텝 S128로부터 S129로 진행하여, 디코드 제어 모듈(214)은, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 자막 디코더 제어 모듈(218)을 제어하고, 디코드 동작의 전 단계로서, 버퍼(215A)로부터의 데이터의 읽어내기를 개시시킨다.

[0683] 즉, 이에 의해, 비디오 디코더 제어 모듈(216)은, 버퍼 제어 모듈(215)(도 3)의 비디오 읽어내기 기능부(233)에 데이터를 요구하고, 그 요구에 따라서 버퍼 제어 모듈(215)로부터 건네 받은, 버퍼(215A)에 기억된 1개의 비디오 액세스 유닛, 그 비디오 액세스 유닛에 부가되어 있는 PTS와 DTS(이하, 적절하게, 타임 스탬프ㄹ라고 함), 및 디코드 개시 가능점의 직전에 배치되어 있는 private_stream_2의 PES_packet()에 기술된 정보(이하, 적절하게, 부가 정보라고도 함)인 pic_struct_copy나, au_ref_flag, AU_length 등을 얻는다. 또한, 타임 스탬프는, 비디오 디코더 제어 모듈(216)이 비디오 액세스 유닛을 얻을 때마다, 비디오 디코더 제어 모듈(216)로부터 디코드 제어 모듈(214)에 건네진다.

[0684] 한편, 오디오 디코더 제어 모듈(217)도, 버퍼 제어 모듈(215)(도 3)의 오디오 읽어내기 기능부(234)에 데이터를 요구하고, 그 요구에 따라서 버퍼 제어 모듈(215)로부터 건네 받은, 버퍼(215A)에 기억된 1개의 (ATRAC) 오디오 액세스 유닛과, 그 오디오 액세스 유닛에 부가되어 있는 타임 스탬프(PTS, DTS)를 얻는다. 또한, 타임 스탬프는, 오디오 디코더 제어 모듈(217)이 오디오 액세스 유닛을 얻을 때마다, 오디오 디코더 제어 모듈(217)로부터 디코드 제어 모듈(214)에 건네진다.

[0685] 또한, 자막 디코더 제어 모듈(218)은, 버퍼 제어 모듈(215)(도 3)의 자막 읽어내기 기능부(235)에 데이터를 요구하고, 그 요구에 따라서 버퍼 제어 모듈(215)로부터 건네 받은, 버퍼(215A)에 기억된 1개의 자막 액세스 유닛과, 그 자막 액세스 유닛에 부가되어 있는 타임 스탬프를 얻는다. 또한, 타임 스탬프는, 자막 디코더 제어 모듈(218)이 자막 액세스 유닛을 얻을 때마다, 자막 디코더 제어 모듈(218)로부터 디코드 제어 모듈(214)로 건네진다. 또한, 재생 대상의 엘리멘터리 스트림에, 자막 스트림이 존재하지 않는 경우나, 버퍼(215A)에, 자막 액세스 유닛이 기억되어 있지 않은 경우에는, 버퍼 제어 모듈(215)로부터 자막 디코더 제어 모듈(218)에는, 데이터는 건네지지 않는다.

[0686] 여기서, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 및 자막 디코더 제어 모듈(218)은, 버퍼 제어 모듈(215)에 대하여 데이터를 요구할 때마다, 그 데이터의 요구에 대하는 결과를, 디코드 제어 모듈(214)에 건네진다.

[0687] 또한, 버퍼 제어 모듈(215)로부터, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 및 자막 디코더 제어 모듈(218)에 대하여 데이터가 건네질 때의, 그 데이터의 버퍼(215A)로부터의 읽어내기의 상세에 대해서는 후술한다.

[0688] 「디코드 개시」

[0689] 이상과 같이, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 자막 디코더 제어 모듈(218)이, 버퍼 제어 모듈(215)의 버퍼(215A)로부터 데이터를 읽어내기 시작하면, 스텝 S129로부터 S130으로 진행하여, 그 데이터의 디코드가 개시된다.

[0690] 즉, 디코드 제어 모듈(214)은, 스텝 S127에서 플레이어 제어 모듈(212)로부터 공급된, PlayList#0에 포함되는 1번째의 PlayItem#0의 IN_time인 180,090, 또한, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈

(217), 자막 디코더 제어 모듈(218)로부터 스텝 S129에서 설명한 바와 같이 건네 받는 타임 스탬프에 기초하여, 동기를 확보하기 위해 필요하면 타이밍을 어긋나게 하여, 디코드 개시를, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 및 자막 디코더 제어 모듈(218)에 명령한다.

[0691] 여기서, 동기를 확보하기 위한 타이밍을 어긋나게 한 디코드 개시의 명령의 방법은, 예를 들면, 특허 제3496725호에 기재되어 있으며, 간단히는, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 자막 디코더 제어 모듈(218)의 각각으로부터 받은 타임 스탬프 중의 최소값을, 계시부(214A)에 의해 계시되는 시각의 초기값으로서 설정하여 시각의 계시를 개시하고, 계시부(214A)에 의해 계시되는 시각과, 타임 스탬프가 일치한 시점에서, 디코드 개시를 명령하는 방법이 있다.

[0692] 비디오 디코더 제어 모듈(216)은, 디코드 제어 모듈(214)로부터의 디코드 개시의 지시를 받고, 그 지시에 따라서, 버퍼 제어 모듈(215)(도 3)의 비디오 읽어내기 기능부(233)로부터 얻은 1개의 비디오 액세스 유닛을, 비디오 디코더(116)(도 1)에 건네주어 디코드시킨다. 또한, 비디오 디코더 제어 모듈(216)은, 비디오 디코더(116)에 의한 디코드의 결과 얻어지는 비디오 데이터를, 그래픽 처리 모듈(219)에 공급한다.

[0693] 이후, 비디오 디코더 제어 모듈(216)은, 버퍼 제어 모듈(215)의 비디오 읽어내기 기능부(233)로부터 얻어지는 1개씩의 비디오 액세스 유닛을, 비디오 디코더(116)에서 순차적으로 디코드하고, 그 디코드의 결과 얻어지는 비디오 데이터를, 그래픽 처리 모듈(219)에 공급해 간다.

[0694] 한편, 오디오 디코더 제어 모듈(217)도, 디코드 제어 모듈(214)로부터의 디코드 개시의 지시를 받고, 그 지시에 따라서, 버퍼 제어 모듈(215)(도 3)의 오디오 읽어내기 기능부(234)로부터 얻은 1개의 오디오 액세스 유닛을, 오디오 디코더(117)(도 1)에 건네주어 디코드시킨다. 또한, 오디오 디코더 제어 모듈(217)은, 오디오 디코더(117)에 의한 디코드의 결과 얻어지는 오디오 데이터를, 오디오 출력 모듈(221)에 공급한다.

[0695] 이후, 오디오 디코더 제어 모듈(217)은, 버퍼 제어 모듈(215)의 오디오 읽어내기 기능부(234)로부터 얻어지는 1개씩의 오디오 액세스 유닛을, 오디오 디코더(117)에서 순차적으로 디코드하고, 그 디코드의 결과 얻어지는 오디오 데이터를, 오디오 출력 모듈(221)에 공급해 간다.

[0696] 또한, 자막 디코더 제어 모듈(218)도, 디코드 제어 모듈(214)로부터의 디코드 개시의 지시를 받고, 그 지시에 따라서, 버퍼 제어 모듈(215)(도 3)의 자막 읽어내기 기능부(235)로부터 얻은 1개의 자막 액세스 유닛을, 내부에 갖는 자막 디코드 소프트웨어로 디코드하고, 그 디코드의 결과 얻어지는 자막 데이터(자막의 화상 데이터)를, 그래픽 처리 모듈(219)에 공급한다.

[0697] 이후, 자막 디코더 제어 모듈(218)은, 버퍼 제어 모듈(215)의 자막 읽어내기 기능부(235)로부터 얻어지는 1개씩의 자막 액세스 유닛을, 내부에 갖는 자막 디코드 소프트웨어로 순차적으로 디코드하고, 그 디코드의 결과 얻어지는 자막 데이터를, 그래픽 처리 모듈(219)에 공급해 간다.

[0698] 「그래픽 처리」

[0699] 그 후, 스텝 S130로부터 S131로 진행하여, 그래픽 처리 모듈(219)은, 전술한 바와 같이 하여, 비디오 디코더 제어 모듈(216)로부터 공급되는 비디오 데이터, 또한, 필요에 따라, 자막 디코더 제어 모듈(218)로부터 공급되는 자막 데이터를 대상으로, 그래픽 처리를 행한다.

[0700] 즉, 그래픽 처리 모듈(219)은, 우선 자막 디코더 제어 모듈(218)로부터의 자막 데이터를, 플레이어 제어 모듈(212)로부터의 표시 방식의 지시에 따라, 확대나 축소하는 등 자막 처리를 행한다. 플레이어 제어 모듈(212)로부터, 표시 방식의 지시가 없는 경우 또는 디폴트의 표시 방식의 지시가 있던 경우, 그래픽 처리 모듈(219)은, 자막 디코더 제어 모듈(218)로부터의 자막 데이터를, 그대로 보존한다.

[0701] 또한, 그래픽 처리 모듈(219)은, 비디오 디코더 제어 모듈(216)로부터의 비디오 데이터와, 자막 디코더 제어 모듈(218)로부터의 자막 데이터, 또는 자막 처리 후의 자막 데이터를 가산하고, 비디오 디코더 제어 모듈(216)로부터의 비디오 데이터에 자막 데이터가 오버레이된 출력 비디오 데이터를 얻어, 비디오 출력 모듈(220)에 공급한다.

[0702] 또한, 그래픽 처리 모듈(219)은, 스크립트 제어 모듈(211)이나 플레이어 제어 모듈(212)로부터, 예를 들면, 메뉴나, 메시지, 타임 코드, 챕터 또는 인덱스의 번호 등의 정보의 표시의 지시를 받은 경우에는, 그 정보를 생성하고, 출력 비디오 데이터에 오버레이하여, 비디오 출력 모듈(220)에 공급한다.

[0703] 「출력 처리」

[0704] 스텝 S131의 처리 후에는, 스텝 S132로 진행하여, 비디오 출력 모듈(220)은, 스텝 S131에서 설명한 바와 같이 하여 그래픽 처리 모듈(219)로부터 공급되는 출력 비디오 데이터를, FIFO(220A)에 순차적으로 기억시키고, 그 FIFO(220A)에 기억된 출력 비디오 데이터를, 미리 정해진 출력 레이트로 순차적으로 출력한다.

[0705] 비디오 출력 모듈(220)은, FIFO(220A)의 기억 용량(잔량)에 여유가 있는 한, 그래픽 처리 모듈(219)로부터의 출력 비디오 데이터를 받아들이지만, 여유가 없는 경우에는, 출력 비디오 데이터의 받아들임의 정지를, 그래픽 처리 모듈(219)에 요구한다. 이에 의해, 그래픽 처리 모듈(219)은, 처리를 정지함과 함께, 처리의 정지를, 비디오 디코더 제어 모듈(216) 및 자막 디코더 제어 모듈(218)에 요구한다. 이에 의해, 비디오 디코더 제어 모듈(216) 및 자막 디코더 제어 모듈(218)이 처리를 정지한다.

[0706] 비디오 출력 모듈(220)은, 출력 비디오 데이터의 받아들임의 정지를, 그래픽 처리 모듈(219)에 요구한 후에, FIFO(220A)로부터의 출력 비디오 데이터의 출력이 진행되어, FIFO(220A)에 여유가 생긴 시점에서, 출력 비디오 데이터의 받아들임을, 그래픽 처리 모듈(219)에 요구한다. 이 요구는, 출력 비디오 데이터의 받아들임의 정지의 요구와 마찬가지로, 그래픽 처리 모듈(219)로부터, 비디오 디코더 제어 모듈(216) 및 자막 디코더 제어 모듈(218)에 전해진다. 이에 의해, 그래픽 처리 모듈(219), 또한, 비디오 디코더 제어 모듈(216) 및 자막 디코더 제어 모듈(218)은, 정지하고 있던 처리를 재개한다.

[0707] 한편, 오디오 출력 모듈(221)도, 스텝 S130에서 설명한 바와 같이 하여 오디오 디코더 제어 모듈(217)로부터 공급되는 오디오 데이터를, FIFO(221A)에 순차적으로 기억시키고, 그 FIFO(221A)에 기억된 오디오 데이터를, 미리 정해진 출력 레이트(샘플링 주파수)로 순차적으로 출력한다. 오디오 출력 모듈(221)은, FIFO(221A)의 기억 용량(잔량)에 여유가 있는 한, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터를 받아들이지만, 여유가 없는 경우에는, 오디오 데이터의 받아들임의 정지를, 오디오 디코더 제어 모듈(217)에 요구한다. 이에 의해, 오디오 디코더 제어 모듈(217)은, 처리를 정지한다.

[0708] 오디오 출력 모듈(221)은, 오디오 데이터의 받아들임의 정지를, 오디오 디코더 제어 모듈(217)에 요구한 후에, FIFO(221A)로부터의 오디오 데이터의 출력이 진행되어, FIFO(221A)에 여유가 생긴 시점에서, 오디오 데이터의 받아들임을, 오디오 디코더 제어 모듈(217)에 요구한다. 이에 의해, 오디오 디코더 제어 모듈(217)은, 정지하고 있던 처리를 재개한다.

[0709] 이상과 같이 하여, 비디오 출력 모듈(220) 및 오디오 출력 모듈(221)로부터 데이터가 출력됨에 따라서, 엘리멘터리 스트림의 디코드가 행해져 간다.

[0710] 도 1의 디스크 장치가 디스크(101)를 재생할 때의 전체의 처리 또는 동작의 흐름은, 도 29 및 도 30에서 설명한 바와 같지만, 이하, 디스크 장치에서 디스크(101)의 재생이 행해지고 있을 때의, 그 밖의 처리 또는 동작에 대해서 설명한다.

[0711] [PlayItem 갈아타기]

[0712] 도 29 및 도 30에서 설명한 바와 같이 하여, 도 25에서의 1번째의 PlayList#0의 1번째의 PlayItem#0의 재생이 시작되지만, PlayList#0에 따르면, 그 1번째의 PlayItem#0의 재생이 종료하면, 2번째의 PlayItem#1의 재생이 개시된다. 즉, PlayItem#0으로부터 PlayItem#1로 PlayItem을 갈아타는 PlayItem 갈아타기가 행해진다.

[0713] 따라서, 도 31의 플로우차트를 참조하여, 이 PlayItem 갈아타기의 처리에 대해서 설명한다.

[0714] 도 29 및 도 30에서 설명한 바와 같이 하여, 도 25에서의 PlayList#0의 1번째의 PlayItem#0(의 클립)의 재생이 개시되면, 디코드 제어 모듈(214)(도 2A 및 도 2B)은, 그 1번째의 PlayItern#0의 재생이 행해지고 있는 동안, 내장한 계시부(214A)가 계시하고 있는 시각을 계속해서 확인하고 있다.

[0715] 「PlayItem#0의 재생 종료」

[0716] 그리고, 디코드 제어 모듈(214)은, 계시부(214A)가 시간을 재고 있는 시각이, 도 30의 스텝127에서 플레이어 제어 모듈(212)로부터 공급된 1번째의 PlayItem#0의 OUT_time인 27,180,090(도 25)과 동일하게 되지만, 스텝 S151에서, 디코드 중단 제어를 행하여, PlayItem#0의 재생을 종료한다.

[0717] 즉, 디코드 제어 모듈(214)은, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 자막 디코더 제어 모듈(218)을 조작하여, 디코드 동작을 정지시킨다. 또한, 디코드 제어 모듈(214)은, 비디오 출력 모듈(220)을 제어하여, 지금 출력 중인 출력 비디오 데이터를 계속하여 출력시킨다.

[0718] 또한, 디코드 제어 모듈(214)은, 1번째의 PlayItem#0의 재생이 종료된 취지의 메시지를, 플레이어 제어 모듈

(212)에 전해진다.

[0719] 「PlayItem#1의 재생 개시」

[0720] 플레이어 제어 모듈(212)은, 전술한 바와 같이, 도 29의 스텝 S105에서, 1번째의 PlayList#0에, 1번째의 PlayItem#0와 2번째의 PlayItem#1이 존재하는 것을 인식하고 있고, 디코드 제어 모듈(214)로부터, 1번째의 PlayItem#0의 재생이 종료된 취지의 메시지가 전해지면, 스텝 S151로부터 S152로 진행하여, 2번째의 PlayItem#1의 재생을, 전술한 1번째의 PlayItem#0에서의 경우와 마찬가지로 하여 개시한다.

[0721] 즉, 2번째의 PlayItem#1의 재생 수순을 개설하면, 우선, 플레이어 제어 모듈(212)은, 도 30의 스텝 S122에서의 경우와 마찬가지로 하여, 2번째의 PlayItem#1에 대해서, EP_map()에 기술된 RPN_EP_start 중 중 어느 하나를, 재생 개시 위치로서 결정한다.

[0722] 또한, 플레이어 제어 모듈(212)에서는, 도 30의 스텝 S124에서 설명한 바와 같이 하여, 2번째의 PlayItem#1에 속하는 Mark()의 인식이나, 도 30의 스텝 S125에서 설명한 바와 같이 하여, PlayItem#1에 의해 재생되는 클립 스트림 파일"00002.PS"에 다중화되어 있는 각 속성의 엘리멘터리 스트림의 개수의 인식, 또한, 재생할 (재생 대상의) 엘리멘터리 스트림의 결정 등이 행해진다.

[0723] 그리고, 플레이어 제어 모듈(212)은, 도 30의 스텝 S127에서의 경우와 마찬가지의 처리를 행한다.

[0724] 즉, 플레이어 제어 모듈(212)은, 재생 개시 위치로서 결정한 EP_map0의 RPN_EP_start와, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일의 파일명, 즉, 지금의 경우, 2번째의 PlayItem#1(도 25)의 Clip_Information_file_name에 기술된 "00002.CLP"에 대응하는 클립 스트림 파일"00002.PS"의 파일명을, 콘텐츠 데이터 공급 모듈(213)에 대하여 제공한다.

[0725] 또한, 플레이어 제어 모듈(212)은, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클림 스트림 파일 "00002.PS"에 저장된 프로그램 스트림의, 버퍼 제어 모듈(215)에의 공급이 개시되기 전에, 버퍼 제어 모듈(215)을 초기화한다.

[0726] 즉, 이에 의해, 버퍼 제어 모듈(215)(도 3)에서, 데이터 선두 포인터 기억부(231)에 기억되는 데이터 선두 포인터, 데이터 써넣기 포인터 기억부(232)에 기억되는 데이터 써넣기 포인터, 비디오 읽어내기 포인터 기억부(241)에 기억되는 비디오 읽어내기 포인터, 오디오 읽어내기 포인터 기억부(251)에 기억되는 오디오 읽어내기 포인터, 자막 읽어내기 포인터 기억부(262)에 기억되는 자막 읽어내기 포인터에, 동일한 값이 대입된다

[0727] 또한, 플레이어 제어 모듈(212)은, 재생 대상의 엘리멘터리 스트림을 식별 하기 위한 식별 정보로서의 stream_id, 나아가서는, 필요에 따라, private_stream_id를, 버퍼 제어 모듈(215)에 공급한다.

[0728] 버퍼 제어 모듈(215)(도 3)에서는, 비디오 읽어내기 기능부(233)가, 플레이어 제어 모듈(212)로부터의, 재생 대상의 엘리멘터리 스트림 중의 비디오 스트림에 대한 stream_id를, stream_id 레지스터(242)에 기억시킨다. 또한, 오디오 읽어내기 기능부(234)가, 플레이어 제어 모듈(212)로부터의, 재생 대상의 엘리멘터리 스트림 중의 오디오 스트림의 stream_id와 private_stream_id를, stream_id 레지스터(252)와 private_stream_id 레지스터(253)에, 각각 기억시킨다.

[0729] 또한, 지금 재생 대상으로 되어 있는 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일 "00002.PS"에는, 자막 스트림이 포함되기 때문에, 플레이어 제어 모듈(212)로부터 자막 읽어내기 기능부(235)에는, 재생 대상의 엘리멘터리 스트림 중의 자막 스트림의 stream_id와 private_stream_id가 공급되며, 자막 읽어내기 기능부(235)는, 그 stream_id와 private_stream_id를, stream_id 레지스터(263)와 private_stream_id 레지스터(264)에, 각각 기억시킨다.

[0730] 그리고, 플레이어 제어 모듈(212)은, 버퍼 제어 모듈(215)(도 3)의 초기화로서, 또한, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일에 따른 값의 자막 읽어내기 기능 플래그를, 자막 읽어내기 기능 플래그 기억부(261)에 세트한다.

[0731] 즉, 지금의 경우, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일 "00002.PS"에는, 자막 스트림이 포함되기 때문에, 자막 읽어내기 기능부(235)를 기능시키기 위해, 값이 1인 자막 읽어내기 기능 플래그가, 자막 읽어내기 기능 플래그 기억부(261)에 세트된다.

[0732] 또한, 플레이어 제어 모듈(212)은, 재생하고자 하는 2번째의 PlayItem#1(도 25)의 IN_time인 90,000과, OUT_time인 27,090,000을, 디코드 제어 모듈(214)에 대하여 공급한다.

[0733] 또한, 플레이어 제어 모듈(212)은, 그래픽 처리 모듈(219)에 대한 자막 스트림의 표시 방식의 지시를 초기화한다. 즉, 플레이어 제어 모듈(212)은, 자막 스트림의 표시 방식을 디폴트의 표시 방식으로 하도록, 그래픽 처리 모듈(219)을 제어한다.

[0734] 또한, 재생 대상의 자막 스트림에 대해서, configurable_flag(도 12)가, 표시 방식의 변경을 허가하는 1로 되어 있는 경우에는, 플레이어 제어 모듈(212)로부터 그래픽 처리 모듈(219)에 대한 자막 스트림의 표시 방식의 지시는, 지금의 표시 방식 그대로로 하도록 해도 된다.

[0735] 이하, 2번째의 PlayItem#1의 재생은, 1번째의 PlayItem#0의 재생과 마찬가지로 하여 행해져 간다. 그리고, 디코드 제어 모듈(214)은, 그 2번째의 PlayItem#1의 재생이 행해지고 있는 동안, 내장한 계시부(214A)가 계시하고 있는 시각을 계속해서 확인하고 있으며, 계시부(214A)가 계시하고 있는 시각이, 스텝 S152(도 31)에서 플레이어 제어 모듈(212)로부터 공급된 2번째의 PlayItem#1의 OUT_time인 27,090,000(도 25)과 동일하게 되면, 스텝 S151에서의 경우와 마찬가지의 디코드 중단 제어를 행하여, PlayItem#1의 재생을 종료한다.

[0736] [타임 코드의 표시]

[0737] 다음으로, 전술한 바와 같이, 도 30의 스텝 S123에서, 타임 코드의 표시가 개시되지만, 이 타임 코드의 표시는, 순차적으로 갱신되어 간다.

[0738] 따라서, 도 32의 플로우차트를 참조하여, 타임 코드의 표시의 처리에 대해서 설명한다.

[0739] 디코드 제어 모듈(214)(도 2A 및 도 2B)은, 그 내장한 계시부(214A)에 의해 1초가 계시되면, 스텝 S171에서, 1초가 경과한 취지의 메시지와 함께, 그 계시부(214A)에 의해 계시되어 있는 지금 시각을, 플레이어 제어 모듈(212)에 공급하고, 스텝 S172로 진행한다. 스텝 S172에서는, 플레이어 제어 모듈(212)은, 디코드 제어 모듈(214)로부터의 메시지와 지금 시각을 수신하고, 그 지금 시각을, 타임 코드로 환산하고, 스텝 S173으로 진행한다.

[0740] 스텝 S173에서는, 플레이어 제어 모듈(212)은, 스텝 S172에서 얻은 타임 코드를 표시하도록, 그래픽 처리 모듈(219)을 제어하고, 스텝 S171로 되돌아간다.

[0741] 이에 의해, 타임 코드는, 1초마다 갱신된다. 또한, 타임 코드의 갱신의 간격은, 1초로 한정되는 것은 아니다.

[0742] [스트림 절환]

[0743] 다음으로, 도 25에서 설명한 1번째의 PlayList#0을 구성하는 1번째의 PlayItem#0에 의해 재생되는 클립 스트림 파일 "00001.PS"나, 2번째의 PlayItem#1에 의해 재생되는 클립 스트림 파일 "00002.PS"에는, 도 26A 및 도 26B에서 설명한 바와 같이, 2개의 자막 스트림이 다중화되어 있다.

[0744] 이와 같이, 클립 스트림 파일에, 복수의, 동일한 속성의 엘리멘터리 스트림이 다중화되어 있는 경우에는, 재생 대상의 엘리멘터리 스트림을, 그 복수의, 동일한 속성의 엘리멘터리 스트림 중의 1개로부터, 다른 1개로 절환하는 스트림 절환을 행할 수 있다.

[0745] 따라서, 도 33의 플로우차트를 참조하여, 스트림 절환의 처리에 대해서 설명한다.

[0746] 스트림 절환의 요구는, 예를 들면, "SCRIPT.OAT" 파일(도 4)에, 스트림 절환의 지시가 스크립트 프로그램으로서 기술되어 있는 경우에, 스크립트 제어 모듈(211)이, 그 스크립트 프로그램을 실행함으로써, 혹은, 유저가 리모콘을 조작함으로써, 플레이어 제어 모듈(212)에 공급된다.

[0747] 즉, 스크립트 제어 모듈(211)은, 스트림 절환의 지시가 기술되어 있는 스크립트 프로그램을 실행하면, 스트림 절환을 요구하는 메시지를, 플레이어 제어 모듈(212)에 공급한다. 또한, 입력 인터페이스(115)는, 유저가 리모콘을 조작함으로써, 리모콘으로부터, 스트림 절환을 지시하는 신호를 수신하면, 스트림 절환을 요구하는 메시지를, 플레이어 제어 모듈(212)에 공급한다. 예를 들면, 지금, 플레이어 제어 모듈(212)에 대하여, 자막 스트림의 절환을 요구하는 메시지인 자막 스트림 절환의 메시지가 공급된 것으로 하면, 플레이어 제어 모듈(212)은, 스텝 S191에서, 도 30의 스텝 S125에서 행해진 재생 대상의 엘리멘터리 스트림의 결정 시에 인식한 자막 스트림의 개수를 체크한다.

[0748] 플레이어 제어 모듈(212)은, 자막 스트림의 개수를 체크한 결과, 그 개수가 1개 이하인 경우, 자막 스트림 절환의 메시지를 무시하고, 따라서, 이후의 스텝 S192 내지 스텝 S194의 처리는 행해지지 않는다.

[0749] 한편, 자막 스트림의 개수가 2개 이상인 경우, 스텝 S192 내지 스텝 S194로 순차적으로 진행하여, 재생하는 자막 스트림이, 지금 재생되고 있는 자막 스트림으로부터, 다른 자막 스트림으로 절환된다.

[0750] 즉, 스텝 S192에서, 플레이어 제어 모듈(212)은, 지금 재생 중인 자막 스트림을, 클립 정보 파일 상에 의해 특정한다. 구체적으로는, 예를 들면, 지금, 도 25에서 설명한 1번째의 PlayList#0을 구성하는 2번째의 PlayItem#1에 의해, 클립 스트림 파일 "00002.PS"에 다중화된, stream_id가 0xBD이고, private_stream_id가 0x80인 자막 스트림이 재생되고 있는 것으로 하면, 스텝 S192에서는, 지금 재생 중인 자막 스트림이, 클립 스트림 파일 "00002.PS"에 다중화된 2개의 자막 스트림 중의, 도 26A 및 도 26B의 클립 정보 파일 "00002.CLP" 상에서 3개째의 자막 스트림인 stream#2인 것이 특정된다.

[0751] 그리고, 스텝 S193으로 진행하여, 플레이어 제어 모듈(212)은, 스텝 S192에 의해 특정한 자막 스트림의, 클립 정보 파일 상에서 다음 자막 스트림을, 다음에 재생할 자막 스트림으로서 인식(특정)한다. 도 26A 및 도 26B에서는, 클립 정보 파일 "00002.CLP" 상에서, 3개째의 자막 스트림 stream#2의 다음 자막 스트림은, 4개째의 자막 스트림 stream#3이기 때문에, 스텝 S193에서는, 이 4개째의 자막 스트림 stream#3이, 다음에 재생할 자막 스트림으로서 인식된다.

[0752] 또한, 지금 재생 중인 자막 스트림이, 클립 스트림 파일 "00002.PS"에 다중화된 2개의 자막 스트림 중의, 도 26A 및 도 26B의 클립 정보 파일 "00002.CLP" 상에서 4개째의 자막 스트림인 stream#3인 것이 특정된 경우에는, 예를 들면, 3개째의 자막 스트림 stream#2가, 다음에 재생할 자막 스트림으로서 인식된다.

[0753] 그 후, 스텝 S194로 진행하여, 플레이어 제어 모듈(212)은, 스텝 S193에서 인식한 다음에 재생할 자막 스트림의 stream_id와 private_stream_id를, 버퍼 제어 모듈(215)(도 3)의 자막 읽어내기 기능부(235)에 대하여 제공하고, 그 stream_id와 private_stream_id를, 차회로부터의, 자막 액세스 유닛의 버퍼(215A)로부터의 읽어내기로부터 사용하도록 지시한다.

[0754] 버퍼 제어 모듈(215)(도 3)의 자막 읽어내기 기능부(235)에서는, 스텝 S194에서 플레이어 제어 모듈(212)로부터 공급되는 stream_id와 private_stream_id를, stream_id 레지스터(263)와 private_stream_id 레지스터(264)에, 각각 새롭게 세트하고, 차회 이후의 버퍼(215A)로부터의 읽어내기는, 그 stream_id 레지스터(263)와 private_stream_id 레지스터(264)에 각각 새롭게 세트된 stream_id와 private_stream_id에 의해 특정되는 자막 액세스 유닛을 대상으로 하여 행해진다.

[0755] 이상과 같이 하여, 재생하는 자막 스트림이, 지금 재생되고 있는 자막 스트림으로부터, 다른 자막 스트림으로 절환된다.

[0756] [버퍼 제어 모듈(215)의 처리]

[0757] 다음으로, 도 34 내지 도 38을 참조하여, 버퍼 제어 모듈(215)(도 3)의 처리, 즉, 버퍼(215A)에의 데이터의 써넣기와, 버퍼(215A)로부터의 데이터의 읽어내기에 대해서 설명한다.

버퍼 제어 모듈(215)은, 도 3에서 설명한 바와 같이, 버퍼(215A)에 대한 데이터의 써넣기 및 읽어내기를 행하기 위한 5개의 포인터를 갖고 있다.

[0758] 즉, 도 34 및 도 35에 도시한 바와 같이, 버퍼 제어 모듈(215)은, 데이터 선두 포인터 기억부(231)에 기억되는 데이터 선두 포인터, 데이터 써넣기 포인터 기억부(232)에 기억되는 데이터 써넣기 포인터, 비디오 읽어내기 포인터 기억부(241)에 기억되는 비디오 읽어내기 포인터, 오디오 읽어내기 포인터 기억부(251)에 기억되는 오디오 읽어내기 포인터, 및 자막 읽어내기 포인터 기억부(262)에 기억되는 자막 읽어내기 포인터를 갖고 있다.

[0759] 또한, 도 34 및 도 35에서는, 도 3에서의 비디오 읽어내기 기능부(233)의 stream_id 레지스터(242) 및, au_information() 레지스터(243), 오디오 읽어내기 기능부(234)의 stream_id 레지스터(252) 및 private_stream_id 레지스터(253), 및 자막 읽어내기 기능부(235)의 자막 읽어내기 기능 플래그 기억부(261), stream_id 레지스터(263), 및 private_stream_id 레지스터(264)의 도시는, 생략하고 있다.

[0760] 데이터 선두 포인터 기억부(231)에 기억된 데이터 선두 포인터는, 버퍼(215A)에 남는 가장 오래된 데이터(읽어낼 필요가 있는 데이터로서, 아직 읽어내어지지 않은 데이터 중의 가장 오래된 데이터)의 위치를 나타낸다. 데이터 써넣기 포인터 기억부(232)에 기억된 데이터 써넣기 포인터는, 버퍼(215A)에의 데이터의 써넣기의 위치를 나타내고, 이 위치는, 버퍼(215A)에서 가장 새로운 데이터가 써넣어지는 위치이다.

[0761] 비디오 읽어내기 포인터 기억부(241)에 기억된 비디오 읽어내기 포인터는, 버퍼(215A)로부터 읽어내는 비디오

스트림의 위치를 나타낸다. 또한, 오디오 읽어내기 포인터 기억부(251)에 기억된 오디오 읽어내기 포인터는, 버퍼(215A)로부터 읽어내는 오디오 스트림의 위치를 나타내고, 자막 읽어내기 포인터 기억부(262)에 기억된 자막 읽어내기 포인터는, 버퍼(215A)로부터 읽어내는 자막 스트림의 위치를 나타낸다.

[0762] 또한, 도 3에서 설명한 바와 같이, 데이터 선두 포인터, 데이터 써넣기 포인터, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 및 자막 읽어내기 포인터는, 모두, 버퍼(215A)를 오른쪽 방향으로 이동한다.

[0763] 또한, 본 실시 형태에서는, 데이터 선두 포인터는, 도 35에 도시한 바와 같이, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 또는 자막 읽어내기 포인터 중의, 가장 오래된 데이터의 위치를 가리키고 있는 것과 동일한 위치를 가리키도록, 항상 갱신되는 것으로 한다. 여기서, 도 35에서는, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 또는 자막 읽어내기 포인터 중의, 오디오 읽어내기 포인터가, 가장 오래된 데이터의 위치를 가리키고 있고, 데이터 선두 포인터는, 그 오디오 읽어내기 포인터와 일치하고 있다. 이상과 같은 데이터 선두 포인터, 데이터 써넣기 포인터, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 및 자막 읽어내기 포인터를 갖는 버퍼 제어 모듈(215)에서는, 데이터 써넣기 포인터는, 디스크(101)로부터 새로운 데이터가 읽어내어져, 버퍼(215A)에 써넣어지면, 그 써넣어진 새로운 데이터의 직후의 위치를 가리키도록, 오른쪽 방향으로 갱신된다.

[0764] 또한, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 또는 자막 읽어내기 포인터는, 버퍼(215A)로부터, 비디오 스트림, 오디오 스트림, 또는 자막 스트림이 읽어내어지면, 그 읽어내기량에 따른 분만큼, 각각, 오른쪽 방향으로 갱신된다. 여기서 읽어내기에 따른 분이란, 실제로 읽어낸 비디오, 오디오, 자막의 데이터에 대응하는 부분과, 읽어낸 데이터 사이에 포함되어 있으며, 읽어내기 시에는 띄어 읽기를 행한, 다른 스트림의 데이터의 부분을 합친 것으로 된다.

[0765] 또한, 데이터 선두 포인터는, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터,또는 자막 읽어내기 포인터가 갱신되면, 그 비디오 읽어내기 포인터, 오디오 읽어내기 포인터,또는 자막 읽어내기 포인터 중의, 가장 오래된 데이터의 위치를 가리키고 있는 것과 동일한 위치를 가리키도록 갱신된다.

[0766] 여기서, 버퍼 제어 모듈(215)은, 버퍼(215A)에의 데이터의 써넣기에 대해서는, 데이터 써넣기 포인터가 데이터 선두 포인터를 추월하지 않도록, 버퍼(215A)에의 데이터의 써넣기를 제어한다.

[0767] 즉, 데이터 써넣기 포인터에 의한 데이터 선두 포인터의 추월이 발생하지 않는 한, 버퍼 제어 모듈(215)에서는, 디스크(101)로부터 읽어낸 데이터가, 데이터 써넣기 포인터가 가리키는 버퍼(215A)의 위치에 써넣어져, 데이터 써넣기 포인터가 갱신되어 간다. 한편, 데이터 써넣기 포인터에 의한 데이터 선두 포인터의 추월이 발생할려고 하면, 버퍼 제어 모듈(215)에서는, 콘텐츠 데이터 공급 모듈(213)에 대하여, 디스크(101)로부터의 데이터의 읽어내기의 정지(중단)가 요구되며, 또한, 버퍼(215A)에의 데이터의 써넣기가 정지된다. 이에 의해, 버퍼(215A)의 오버플로우를 방지할 수 있다.

[0768] 이상과 같이, 디스크(101)로부터 읽어낸 데이터의, 버퍼(215A)에의 써넣기는, 데이터 선두 포인터와 데이터 써넣기 포인터의 2개의 포인터의 위치 관계만으로 제어된다.

[0769] 한편, 버퍼 제어 모듈(215)은, 버퍼(215A)로부터의 데이터의 읽어내기에 대해서는, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 및 자막 읽어내기 포인터, 나아가서는, 데이터 선두 포인터가, 데이터 써넣기 포인터를 추월하지 않도록, 버퍼(215A)로부터의 데이터의 읽어내기를 제어한다.

[0770] 즉, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 또는 자막 읽어내기 포인터에 의한 데이터 써넣기 포인터의 추월이 발생하지 않는 한, 버퍼 제어 모듈(215)에서는, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 또는 자막 디코더 제어 모듈(218)로부터의 요구에 따라서, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 또는 자막 읽어내기 포인터가 가리키는 버퍼(215A)의 위치로부터 데이터가 읽어내어져, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 또는 자막 읽어내기 포인터가 갱신됨과 함께, 필요에 따라, 데이터 선두 포인터가 갱신된다. 한편, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 또는 자막 읽어내기 포인터에 의한 데이터 써넣기 포인터의 추월이 발생할려고 하면, 버퍼 제어 모듈(215)에서는, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 또는 자막 디코더 제어 모듈(218)로부터의 요구가, 예를 들면 동결되어, 충분한 데이터가 준비될 때까지 대기된다. 이에 의해, 버퍼(215A)의 언더 플로우를 방지할 수 있다.

[0771] 이상으로부터, 버퍼(215A)에는, 데이터 선두 포인터가 가리키는 위치로부터, 오른쪽 방향으로, 데이터 써넣기 포인터가 가리키는 위치까지의 범위(도 34 및 도 35에서 음영이 표시되어 있는 부분)에, 비디오 디코더 제어

모듈(216), 오디오 디코더 제어 모듈(217), 및 자막 디코더 제어 모듈(218)에 공급할 데이터가 기억되어 있으며, 또한, 그 범위 내에, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 및 자막 읽어내기 포인터는 존재한다.

[0772] 또한, 전술한 경우에는, 데이터 선두 포인터는, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 또는 자막 읽어내기 포인터가 가리키고 있는 위치 중의, 가장 오래된 데이터의 위치를 가리키도록 갱신하는 것으로 하였지만, 그 밖에, 데이터 선두 포인터의 갱신은, 예를 들면, 그 가장 오래된 데이터의 위치로부터, 소정 시간(예를 들면, 1초)분만큼 과거의 데이터의 위치를 가리키도록 행하는 것이 가능하다.

[0773] 즉, 일반적으로는, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 또는 자막읽어내기 포인터 중의, 비디오 읽어내기 포인터나 오디오 읽어내기 포인터가, 가장 오래된 데이터의 위치를 가리키는 경우가 많다고 예상된다.

[0774] 따라서, 데이터 선두 포인터를, 비디오 읽어내기 포인터 또는 오디오 읽어내기 포인터가 가리키는 가장 오래된 데이터의 위치로부터, 예를 들면 1초분만큼 과거의 데이터의 위치를 가리키도록 갱신한 경우, 도 34에 도시한 바와 같이, 비디오 읽어내기 포인터 또는 오디오 읽어내기 포인터가 가리키는 가장 오래된 데이터의 위치로부터 과거 1초분의 데이터를, 버퍼(215A)에 남겨 둘 수 있다. 여기서, 도 34에서는, 오디오 읽어내기 포인터가, 가장 오래된 데이터의 위치를 가리키고 있고, 데이터 선두 포인터는, 그 위치로부터 1초분만큼 과거의 데이터의 위치를 가리키고 있다.

[0775] 이상과 같이, 1초분만큼 과거의 데이터의 위치를 가리키도록, 데이터 선두 포인터를 갱신함으로써, 디스크 장치의 응답성을 향상시킬 수 있다.

[0776] 즉, 도 35에 도시한 바와 같이, 오디오 읽어내기 포인터가 가리키고 있는 가장 오래된 데이터의 위치를 가리키도록, 데이터 선두 포인터를 갱신하는 경우에는, 예를 들면, 리버스 방향으로의 특수 재생이 지시되었을 때에, 버퍼(215A)로부터의 읽어내기가 종료된 데이터를, 디스크(101)로부터 다시 읽어낼 필요가 있기 때문에, 특수 재생이 지시되고 나서, 그 특수 재생이 가능하게 될 때까지, 어느 정도의 시간이 걸린다.

[0777] 이에 대하여, 도 34에 도시한 바와 같이, 오디오 읽어내기 포인터가 가리키고 있는 가장 오래된 데이터의 위치로부터 1초분만큼 과거의 데이터의 위치를 가리키도록, 데이터 선두 포인터를 갱신하는 경우에는, 리버스 방향으로의 특수 재생이 지시되었을 때에, 그 특수 재생을 개시하는 데에 필요한 데이터가, 버퍼(215A)에 기억되어 있는 1초분만큼 과거의 데이터이면, 전술한 바와 같은 디스크(101)로부터의 데이터의 재읽어내기를 행하지 않고, 바로, 특수 재생을 개시하는 것이 가능하게 된다.

[0778] 또한, 오디오 읽어내기 포인터가 가리키고 있는 가장 오래된 데이터의 위치로부터 1초분만큼 과거의 데이터의 위치를 가리키도록, 데이터 선두 포인터를 갱신하는 경우에도, 특수 재생을 개시하는 데에 필요한 데이터가, 버퍼(215A)에 기억되어 있지 않은 경우가 있을 수 있다. 이 경우에는, 특수 재생을 개시하는 데에 필요한 데이터가, 디스크(101)로부터 재차 읽어내어진다.

[0779] 다음으로, 버퍼(215A)로부터의 비디오 스트림, 오디오 스트림, 자막 스트림의 각각의 읽어내기의 상세에 대해서 설명한다.

[0780] 도 30의 스텝 S127에서 설명한 바와 같이, 클립 스트림 파일의 재생이 개시될 때에, 버퍼 제어 모듈(215)에서는, 데이터 선두 포인터, 데이터 써넣기 포인터, 비디오 읽어내기 포인터, 오디오 읽어내기 포인터, 자막 읽어내기 포인터가, 모두, 버퍼(215A) 상의 동일한 위치를 가리키도록 초기화된다.

[0781] 그리고, 디스크(101)로부터 클립 스트림 파일에 저장된 프로그램 스트림(MPEG2-System Program Stream)이 읽어내어져, 버퍼 제어 모듈(215)에 공급되면, 버퍼 제어 모듈(215)에서는, 그 프로그램 스트림이, 버퍼(215A)의 데이터 써넣기 포인터가 가리키는 위치에 기억됨과 함께, 데이터 써넣기 포인터가, 오른쪽 방향으로 갱신되어 간다.

[0782] 또한, 버퍼 제어 모듈(215)(도 3)에서는, 비디오 읽어내기 기능부(233)가, 버퍼(215A)에 기억된 프로그램 스트림의 구문 해석을 행하고, 비디오 디코더 제어 모듈(216)로부터의 요구에 따라서, 버퍼(215A)에 기억된 프로그램 스트림으로부터, 비디오 스트림(비디오 액세스 유닛)을 추출(분리)하여 읽어내어, 비디오 디코더 제어 모듈(216)에 공급한다.

[0783] 마찬가지로, 오디오 읽어내기 기능부(234)도, 버퍼(215A)에 기억된 프로그램 스트림의 구문 해석을 행하고, 오디오 디코더 제어 모듈(217)로부터의 요구에 따라서, 버퍼(215A)에 기억된 프로그램 스트림으로부터, 오디

오 스트림(오디오 액세스 유닛)을 추출하여 읽어내어, 오디오 디코더 제어 모듈(217)에 공급한다. 자막 읽어내기 기능부(235)도, 버퍼(215A)에 기억된 프로그램 스트림의 구문 해석을 행하고, 자막 디코더 제어 모듈(218)로부터의 요구에 따라서, 버퍼(215A)에 기억된 프로그램 스트림으로부터, 자막 스트림(자막 액세스 유닛)을 추출하여 읽어내어, 자막 디코더 제어 모듈(218)에 공급한다.

[0784] 「비디오 스트림의 읽어내기」

[0785] 다음으로, 도 36의 플로우차트를 참조하여, 비디오 읽어내기 기능부(233)(도 3)에 의한, 버퍼(215A)로부터의 비디오 스트림의 읽어내기 처리의 상세에 대해서 설명한다.

[0786] 비디오 읽어내기 기능부(233)는, 우선 최초로, 스텝 S211에서, 버퍼(215A)에 기억된 프로그램 스트림 중의 private_stream_2의 PES_packet()을 탐색하여 찾아낸다. 즉, private_stream_2의 PES_packet()의 stream_id는, 도 20에서 설명한 바와 같이, 10111111B(=0xBF)이며, 비디오 읽어내기 기능부(233)는, stream_id가 10111111B로 되어 있는 PES_packet()를 탐색하여 찾아낸다.

[0787] 여기서, 예를 들면, 지금, 전술한 바와 같이, 클립 스트림 파일 "00001.PS" 에 저장된 프로그램 스트림에 다중화된 엘리멘터리 스트림이, 재생 대상의 엘리멘터리 스트림인 것으로 하면, 그 프로그램 스트림을 디스크(101)로부터 읽어내어, 버퍼(215A)에 기억시킬 때에, 도 30의 스텝 S122에서, 클립 스트림 파일 "00001.PS"의 EP_map()(도 27)에 기술된 디코드 개시 가능점의 정보로부터, 305섹터가 재생 개시 위치로서 결정되며, 또한, 도 30의 스텝 S128에서, 재생 개시 위치인 305섹터가 지정되며, 오퍼레이팅 시스템(201)에 대하여, 클립 스트림 파일 "00001.PS"에 저장된 프로그램 스트림의 읽어내기가 요구된다.

[0788] 또한, 비디오 스트림에 대해서는, EP_map()에 기술된 디코드 개시 가능점의 정보는, 실제의 디코드 개시 가능점의 직전에 배치된 private_stream_2의 PES_packet()의 위치를 나타낸다.

[0789] 따라서, 클립 스트림 파일 "00001.PS"에 저장된 프로그램 스트림이 디스크(101)로부터 읽어내어져, 버퍼(215A)에 기억된 직후에서는, 데이터 선두 포인터나 비디오 읽어내기 포인터가 가리키는 버퍼(215A)의 위치에는, private_stream_2의 PES_packet()이 기억되어 있다.

[0790] 비디오 읽어내기 기능부(233)는, 스텝 S211에서, private_stream_2의 PES_packet()이 발견되면, 스텝 S212로 진행하여, 그 private_stream_2의 PES_packet()의 PES_packet_data_byte인 private_stream2_PES_payload()(도 23)에 기술되어 있는 video_stream_id를 추출하고, 그 video_stream_id가, 도 30의 스텝 S127에서 stream_id 레지스터(242)(도 3)에 기억된, 재생 대상의 비디오 스트림의 stream_id와 일치하는지의 여부를 판정한다.

[0791] 스텝 S212에서, private_stream2_PES_payload()에 기술되어 있는 video_stream_id가, stream_id 레지스터(242)에 기억된 stream_id와 일치하지 않는다고 판정된 경우, 즉, 직전의 스텝 S211에서 찾아내어진 private_stream_2의 PES_packet()이, 재생 대상의 비디오 스트림의 디코드 개시점에 배치된 것이 아닌 경우, 스텝 S211로 되돌아가서, 버퍼(215A)에 기억된 프로그램 스트림 중의 다른 private_stream_2의 PES_packet()의 탐색이 행해지고, 이하, 마찬가지의 처리가 반복된다.

[0792] 한편, 스텝 S212에서, private_stream2_PES_payload()에 기술되어 있는 video_stream_id가, stream_id 레지스터(242)에 기억된 stream_id와 일치한다고 판정된 경우, 즉, 직전의 스텝 S211에서 찾아내어진 private_stream_2의 PES_packet()이, 재생 대상의 비디오 스트림의 디코드 개시점에 배치된 것인 경우, 스텝 S213으로 진행하여, 비디오 읽어내기 기능부(233)는, 그 private_stream_2의 PES_packet()의 private_stream2_PES_payload()에 기술되어 있는 au_information()을, 버퍼(215A)로부터 읽어내어, au_information() 레지스터(243)(도 3)에 기억시키고, 스텝 S214로 진행한다.

[0793] 스텝 S214에서는, 비디오 읽어내기 기능부(233)는, 직전의 스텝 S211에서 찾아낸 private_stream_2의 PES_packet()(video_stream_id(도 23)가, stream_id 레지스터(242)(도 3)에 기억된 stream_id와 일치하는 private_stream_2의 PES_packet())의 사이즈만큼, 비디오 읽어내기 포인터 기억부(231)에 기억된 비디오 읽어내기 포인터를 갱신한다.

[0794] 즉, 클립 스트림 파일에서는, private_stream_2의 PES_packet() 직후에, 그 video_stream_id와 일치하는 stream_id의 비디오 스트림(PES_packet())이 배치되어 있고, 따라서, 스텝 S214에서는, 비디오 읽어내기 포인터는, 비디오 스트림의 실제의 디코드 개시 가능점의 위치를 가리키도록 갱신된다.

[0795] 그 후, 스텝 S214로부터 S215로 진행하여, 비디오 읽어내기 기능부(233)는, 비디오 디코더 제어 모듈(216)로부터, 데이터의 요구가 있었는지의 여부를 판정하고, 없다고 판정한 경우, 스텝 S215로 되돌아가서, 마찬가지

의 처리를 반복한다.

[0796] 또한, 스텝 S215에서, 비디오 디코더 제어 모듈(216)로부터, 데이터의 요구가 있었다고 판정된 경우, 스텝 S216으로 진행하여, 비디오 읽어내기 기능부(233)는, 비디오 읽어내기 포인터가 가리키고 있는 버퍼(215A)의 위치로부터의 프로그램 스트림의 구문 해석을 행하면서, au-information() 레지스터(243)에 기억된 au_information()의 AU_length에 기술된 바이트수의 데이터, 즉 1개의 비디오 액세스 유닛을, 버퍼(215A)로부터 읽어내어, 비디오 디코더 제어 모듈(216)에 공급함과 함께, 비디오 읽어내기 포인터를, 버퍼(215A)로부터 읽어낸 1개의 비디오 액세스 유닛의 사이즈분만큼 갱신한다.

[0797] 즉, au_information()에는, 도 24에서 설명한 바와 같이, 그것을 포함하는 private_stream_2의 PES_packet()으로부터, 다음 private_stream_2의 PES_packet()까지의 사이에 포함되는 비디오 액세스 유닛(픽처)의 수를 나타내는 number_of_access_unit가 기술되어 있다.

[0798] 또한, au_information()에는, 도 24에서 설명한 바와 같이, 그 number_of_access_unit의 수만큼의 비디오 액세스 유닛 각각에 관한 정보로서의 pic_struct_copy, au_ref_flag, 및 AU_length가 기술되어 있다.

[0799] au_information()에 nurnber_of_access_unit의 수만큼 기술되어 있는 AU_length 각각은, 도 24에서 설명한 바와 같이, 그것을 포함하는 private_stream_2의 PES_packet()로부터, 다음 private_stream_2의 PES_packet()까지의 사이에 포함되는, number_of_access_unit의 수의 비디오 액세스 유닛 각각의 사이즈를 나타내기 때문에, 비디오 읽어내기 기능부(233)는, 그 AU_length를 이용함으로써, 비디오 스트림의 구문 해석을 행하지 않고, 액세스 유닛의 잘라내기를 행할 수 있다.

[0800] 즉, 종래, MPEG2-Video나 MPEG4-AVC의 액세스 유닛을 잘라내는 경우에는, 비디오 스트림의 구문을 안 후에, 비디오 스트림의 구문 해석을 행할 필요가 있었지만, 디스크(101)에 기록된 클립 스트림 파일에 저장된 프로그램 스트림은, 비디오 액세스 유닛 단위의 비디오 스트림에서의 1 이상의 실제의 디코드 개시 가능점 각각의 직전에, 비디오 액세스 유닛의 사이즈를 나타내는 AU_length가 기술된 private_stream_2의 PES_packet()fmf 포함하고 있으므로, 비디오 읽어내기 기능(233)은, 그 private_stream_2의 PES_packet()에 기술된 AU_length에 기초하여, 비디오 스트림의 구문 해석을 행하지 않고, 버퍼(215A)로부터, 비디오 액세스 유닛(단위의 비디오 스트림)을 읽어내어, 비디오 디코더 제어 모듈(216)에 공급할 수 있다.

[0801] 또한, 비디오 읽어내기 기능부(233)는, 스텝 S216에서, 비디오 액세스 유닛을, 비디오 디코더 제어 모듈(216)에 공급할 때에, 그 비디오 액세스 유닛에 관한 정보로서 au_information()에 기술되어 있는 pic_struct_copy, au_ref_flag, 및 AU_length와, 비디오 액세스 유닛 단위에 부가되어 있는 타임 스탬프(PTS, DTS)도, 비디오 디코더 제어 모듈(216)에 공급한다.

[0802] 스텝 S216에서, 버퍼(215A)로부터 1개의 비디오 액세스 유닛이 읽어내어져, 비디오 디코더 제어 모듈(216)에 공급된 후에는, 스텝 S217로 진행하여, 비디오 읽어내기 기능부(233)는, au_information() 레지스터(243)에 기억된 au_information()(도 24)의 nurnber_of_access_unit가 나타내는 수만큼의 액세스 유닛을 처리하였는지의 여부를 판정한다.

[0803] 스텝 S217에서, nurnber_of_access_unit가 나타내는 수만큼의 액세스 유닛을, 아직 처리하지 않았다고 판정된 경우, 즉, number_of_access_unit가 나타내는 수만큼의 액세스 유닛을, 아직, 버퍼(215A)로부터 읽어내어 비디오 디코더 제어 모듈(216)에 공급하지 않은 경우, 스텝 S215로 되돌아가서, 이하, 마찬가지의 처리가 반복된다.

[0804] 또한, 스텝 S217에서, number_of_access_unit가 나타내는 수만큼의 액세스 유닛을 처리하였다고 판정된 경우, 즉, number_of_access_unit가 나타내는 수만큼의 액세스 유닛을, 버퍼(215A)로부터 읽어내어 비디오 디코더 제어 모듈(216)에 공급한 경우, 스텝 S211로 되돌아가서, 다음 private_stream_2의 PES_packet()의 탐색이 행해지고, 이하, 마찬가지의 처리가 반복된다.

[0805] 「오디오 스트림의 읽어내기」

[0806] 다음으로, 도 37의 플로우차트를 참조하여, 오디오 읽어내기 기능부(234)(도 3)에 의한, 버퍼(215A)로부터의 오디오 스트림의 읽어내기 처리의 상세에 대해서 설명한다.

[0807] 오디오 읽어내기 기능부(234)는, 우선 처음에, 스텝 S230에서, 도 30의 스텝 S127에서 stream_id 레지스터(252)(도 3)에 기억된, 재생 대상의 오디오 스트림의 stream_id가, private_stream_1의 PES_packet()를 나타내고 있는지의 여부를 판정한다.

[0808] 스텝 S230에서, stream_id 레지스터(252)에 기억된 stream_id가, private_stream_1의 PES_packet()를 나타내고 있지 않다고 판정된 경우, 즉, stream_id 레지스터(252)에 기억된 stream_id가, 도 20에서 설명한 바와 같이, MPEG 규격에 따라서 부호화된 오디오 스트림에 할당되는 110xxxxxB인 경우, 스텝 S231로 진행하여, 오디오 읽어내기 기능부(234)는, 버퍼(215A)에 기억된 프로그램 스트림으로부터, MPEG Audio로 정해진 오디오 프레임 선두를 나타내는 동기 코드를 찾는다. 동기 코드의 위치가 오디오 프레임 선두이므로, 오디오 읽어내기 기능부(234)는, 오디오 읽어내기 포인터를, 오디오 프레임 선두의 위치를 나타내도록 갱신하고, 스텝 S231로부터 S232로 진행한다. 스텝 S232에서는, 오디오 읽어내기 기능부(234)는, 버퍼(215A)에 기억된 프로그램 스트림 중의, stream_id 레지스터(252)에 기억된 stream_id에 일치하는 PES_packet()를, 오디오 읽어내기 포인터가 나타내는 위치로부터 탐색해서 찾아내고, 스텝 S233으로 진행한다.

[0809] 스텝 S233에서는, 오디오 읽어내기 기능부(234)는, 오디오 읽어내기 포인터 기억부(251)에 기억된 오디오 읽어내기 포인터를, 직전의 스텝 S232에서 찾아낸 PES_packet()의 PES_packet_data_byte(도 16A 및 도 16B 내지 도 18A 및 도 18B)의 선두를 가리키도록 갱신하고, 스텝 S237로 진행한다.

[0810] 스텝 S237에서는, 오디오 읽어내기 기능부(234)는, 오디오 디코더 제어 모듈(217)로부터, 데이터의 요구가 있었는지의 여부를 판정하고, 없다고 판정한 경우, 스텝 S237로 되돌아가서, 마찬가지의 처리를 반복한다.

[0811] 또한, 스텝 S237에서, 오디오 디코더 제어 모듈(217)로부터, 데이터의 요구가 있었다고 판정된 경우, 스텝 S238로 진행하여, 오디오 읽어내기 기능부(234)는, 오디오 읽어내기 포인터가 가리키고 있는 버퍼(215A)의 위치로부터의 프로그램 스트림의 구문 해석을 행하면서, 기지의 고정 길이의 1개의 오디오 액세스 유닛을, 버퍼(215A)로부터 읽어내고, 그 오디오 액세스 유닛에 부가되어 있는 타임 스탬프(PTS, DTS)와 함께, 오디오 디코더 제어 모듈(217)에 공급한다.

[0812] 그리고, 오디오 읽어내기 기능부(234)는, 버퍼(215A)로부터 읽어낸 1개의 오디오 액세스 유닛의 사이즈분만큼, 오디오 읽어내기 포인터를 갱신하고, 스텝 S237로 되돌아가서, 이하, 마찬가지의 처리가 반복된다.

[0813] 한편, 스텝 S230에서, stream_id 레지스터(252)에 기억된 stream_id가, private_stream_1의 PES_packet()를 나타내고 있다고 판정된 경우, 즉, stream_id 레지스터(252)에 기억된 Stream_id가, 10111101B(=0xBD)이며, 도 20에서 설명한 바와 같이, private_stream_1의 PES_packet()를 나타내고 있는 경우, 스텝 S234로 진행하여, 오디오 읽어내기 기능부(234)는, 버퍼(215A)에 기억된 프로그램 스트림 중의 private_stream_1의 PES_packet()를 탐색하여 찾아낸다. 즉, 오디오 읽어내기 기능부(234)는, stream_id가 10111101B로 되어 있는 PES_packet()를 탐색하여 찾아낸다.

[0814] 오디오 읽어내기 기능부(234)는, 스텝 S234에 있어서, private_stream_1의 PES_packet()가 발견되면, 스텝 S235로 진행하여, 그 private_stream_1의 PES_packet()의 PES_packet_data_byte인 private_stream1_PES_payload()(도 21)에 기술되어 있는 private_stream_id를 추출하고, 그 private_stream_id가, 도 30의 스텝 S127에서 private_stream_id 레지스터(253)(도 3)에 기억된, 재생 대상의 오디오 스트림의 private_stream_id와 일치하는지의 여부를 판정한다.

[0815] 스텝 S235에서, private_stream1_PES_payload()에 기술되어 있는 private_stream_id가, private_stream_id 레지스터(253)에 기억된 private_stream_id와 일치하지 않는다고 판정된 경우, 즉, 직전의 스텝 S234에서 찾아내어진 private_stream_1의 PES_packet()가, 재생 대상의 오디오 스트림이 아닌 경우, 스텝 S234로 되돌아가서, 버퍼(215A)에 기억된 프로그램 스트림 중의 다른 private_stream_1의 PES_packet()의 탐색이 행해지고, 이하, 마찬가지의 처리가 반복된다.

[0816] 한편, 스텝 S235에서, private_stream1_PES_payload()에 기술되어 있는 private_stream_id가, private_stream_id 레지스터(253)에 기억된 private_stream_id와 일치한다고 판정된 경우, 즉, 직전의 스텝 S234에서 찾아내어진 private_stream_1의 PES_packet()가, 재생 대상의 오디오 스트림인 경우, 스텝 S236으로 진행하여, 오디오 읽어내기 기능부(234)는, 그 private_stream_1의 PES_packet()의 private_stream1_PES_payload()(도 21)에 기술되어 있는 AU_locator를, 버퍼(215A)로부터 읽어내고, 그 AU_locator 직후의 위치와, 그 AU_locator가 나타내는 값을 가산함으로써, 오디오 액세스 유닛의 선두 위치를 구한다.

[0817] 즉, AU_locator는, 도 21에서 설명한 바와 같이, 그 AU_locator 직후의 위치를 기준으로 하여, private_stream1_PES_payload()의 private_payload()에 저장되는 오디오 액세스 유닛(혹은 자막 액세스 유

닛)의 선두 위치를 나타내기 때문에, AU_locator 직후의 위치에, 그 AU_locator가 나타내는 값을 가산함으로써, 오디오 액세스 유닛의 (절대적인) 선두 위치를 구할 수 있다.

[0818] 오디오 읽어내기 기능부(234)는, 또한, 스텝 S236에서, 이상과 같이 하여 구한 오디오 액세스 유닛의 선두 위치를 가리키도록, 오디오 읽어내기 포인터 기억부(251)에 기억된 오디오 읽어내기 포인터를 갱신하고, 스텝 S237로 진행한다.

[0819] 스텝 S237에서는, 오디오 읽어내기 기능부(234)는, 오디오 디코더 제어 모듈(217)로부터, 데이터의 요구가 있었는지의 여부를 판정하고, 없다고 판정한 경우, 스텝 S237로 되돌아가서, 마찬가지의 처리를 반복한다.

[0820] 또한, 스텝 S237에서, 오디오 디코더 제어 모듈(217)로부터, 데이터의 요구가 있었다고 판정된 경우, 스텝 S238로 진행하여, 오디오 읽어내기 기능부(234)는, 오디오 읽어내기 포인터가 가리키고 있는 버퍼(215A)의 위치로부터의 프로그램 스트림의 구문 해석을 행하면서, 기지의 고정 길이의 1개의 오디오 액세스 유닛을, 버퍼(215A)로부터 읽어내어, 그 오디오 액세스 유닛에 부가되어 있는 타임 스탬프와 함께, 오디오 디코더 제어 모듈(217)에 공급한다.

[0821] 그리고, 오디오 읽어내기 기능부(234)는, 버퍼(215A)로부터 읽어낸 1개의 오디오 액세스 유닛의 사이즈 분만큼, 오디오 읽어내기 포인터를 갱신하고, 스텝 S237로 되돌아가서, 이하, 마찬가지의 처리가 반복된다.

[0822] 「자막 스트림의 읽어내기」

[0823] 다음으로, 도 38의 플로우차트를 참조하여, 자막 읽어내기 기능부(235)(도 3)에 의한, 버퍼(215A)로부터의 자막 스트림의 읽어내기 처리의 상세에 대해서 설명한다.

[0824] 자막 읽어내기 기능부(235)는, 우선 처음에, 스텝 S251에서, 도 30의 스텝 S127에서 자막 읽어내기 기능 플래그 기억부(261)에 기억된 자막 읽어내기 기능 플래그를 판정한다. 스텝 S251에서, 자막 읽어내기 기능 플래그가 0이라고 판정된 경우, 즉, 예를 들면, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일에 자막 스트림이 포함되어 있지 않아, 도 30의 스텝 S127에서 자막 읽어내기 기능 플래그 기억부(261)에, 0이 세트된 경우, 자막 읽어내기 기능부(235)는 특별히 처리를 행하지 않는다.

[0825] 한편, 스텝 S251에서, 자막 읽어내기 기능 플래그가 1이라고 판정된 경우, 즉, 예를 들면, 재생 대상의 엘리멘터리 스트림이 다중화되어 있는 클립 스트림 파일에 자막 스트림이 포함되어 있어, 도 30의 스텝 S127에서 자막 읽어내기 기능 플래그 기억부(261)에, 1이 세트된 경우, 스텝 S252로 진행하여, 자막 읽어내기 기능부(235)는, stream_id 레지스터(263)(도 3)에 기억된, 재생 대상의 자막 스트림의 stream_id에 일치하는 PES_packet()를, 버퍼(215A)에 기억된 프로그램 스트림으로부터 탐색한다.

[0826] 여기서, 도 30의 스텝 S127에서 설명한 바와 같이, stream_id 레지스터(263)(도 3)에는, 재생 대상의 자막 스트림의 stream_id가 기억되지만, 자막 스트림의 stream_id는, 도 20에서 설명한 바와 같이, private_stream_1의 PES_packet()을 나타내는 10111101B(=0xBD)이다.

[0827] 따라서, 스텝 S252에서는, 버퍼(215A)에 기억된 프로그램 스트림 중의 private_stream_1의 PES_packet()이 탐색되게 된다.

[0828] 스텝 S252에서, private_stream_1의 PES_packet()의 탐색이 행해져, private_stream_1의 PES_packet()가 발견되면, 스텝 S253로 진행하여, 자막 읽어내기 기능부(235)는, 그 private_stream_1의 PES_packet()의 PES_packet_data_byte인 private_stream1_PES_payload()(도 21)에 기술되어 있는 private_stream_id를 뽑아내고, 그 private_stream_id가, 도 30의 스텝 S127에서 private_stream_id 레지스터(264)(도 3)에 기억된, 재생 대상의 자막 스트림의 private_stream_id와 일치하는지의 여부를 판정한다.

[0829] 스텝 S253에서, private_stream1_PES_payload()에 기술되어 있는 private_stream_id가, private_stream_id 레지스터(264)에 기억된 private_stream_id와 일치하지 않는다고 판정된 경우, 즉, 직전의 스텝 S252에서 발견된 private_stream_1의 PESpacket()이, 재생 대상의 자막 스트림이 아닌 경우, 스텝 S252로 되돌아가고, 버퍼(215A)에 기억된 프로그램 스트림중의 다른 private_stream_1의 PES_packet()의 탐색이 행해지고, 이하, 마찬가지의 처리가 반복된다.

[0830] 한편, 스텝 S253에서, private_stream1_PES_payload()에 기술되어 있는 private_stream_id가, private_stream_id 레지스터(264)에 기억된 private_stream_id와 일치한다고 판정된 경우, 즉, 직전의 스텝 S252에서 발견된 private_stream_1의 PES_packet()가, 재생 대상의 자막 스트림인 경우, 스텝 S254로 진행하여, 자막 읽어내기 기능부(235)는, 그 private_stream_1의 PES_packet()의 private_stream1_PES_payload()(도

21)에 기술되어 있는 AU_locator를, 버퍼(215A)로부터 읽어내고, 그 AU_locator 직후의 위치와, 그 AU_locator가 나타내는 값을 가산함으로써, 자막 액세스 유닛의 선두 위치를 구한다.

[0831] 즉, AU_locator는, 도 21에서 설명한 바와 같이, 그 AU_locator 직후의 위치를 기준으로 하여, private_stream1_PES_payload()의 private_payload()에 저장되는 자막 액세스 유닛(혹은 오디오 액세스 유닛)의 선두 위치를 나타내기 때문에, AU_locator 직후의 위치에, 그 AU_locator가 나타내는 값을 가산함으로써, 자막 액세스 유닛의 (절대적인) 선두 위치를 구할 수 있다.

[0832] 자막 읽어내기 기능부(235)는, 또한, 스텝 5254에서, 이상과 같이 하여 구한 자막 액세스 유닛의 선두 위치를 가리키도록, 자막 읽어내기 포인터 기억부(262)에 기억된 자막 읽어내기 포인터를 갱신하고, 스텝 S255로 진행한다.

[0833] 스텝 S255에서는, 자막 읽어내기 기능부(235)는, 자막 디코더 제어 모듈(218)로부터, 데이터의 요구가 있었는지의 여부를 판정하고, 없다고 판정한 경우, 스텝 S255로 되돌아가서, 마찬가지의 처리를 반복한다.

[0834] 또한, 스텝 S255에서, 자막 디코더 제어 모듈(218)로부터, 데이터의 요구가 있었다고 판정된 경우, 스텝 S256으로 진행하여, 자막 읽어내기 기능부(235)는, 자막읽어내기 포인터가 가리키고 있는 버퍼(215A)의 위치로부터의 프로그램 스트림의 구문 해석을 행하면서, 자막 액세스 유닛의 선두에 기술되어 있는 사이즈분의 1개의 자막 액세스 유닛을, 버퍼(215A)로부터 읽어내어, 그 자막 액세스 유닛에 부가되어 있는 타임 스탬프와 함께, 자막 디코더 제어 모듈(218)에 공급한다. 즉, 자막 액세스 유닛의 선두에는, 도 2A 및 도 2B에서 설명한 바와 같이, 그 자막 액세스 유닛의 사이즈가 기술되어 있고, 자막 읽어내기 기능부(235)는, 그 사이즈분의 데이터를, 자막 읽어내기 포인터가 가리키고 있는 버퍼(215A)의 위치로부터 읽어내고, 그 읽어낸 데이터인 자막 액세스 유닛을, 그 자막 액세스 유닛에 부가되어 있는 타임 스탬프와 함께, 자막 디코더 제어 모듈(218)에 공급한다.

[0835] 그리고, 자막 읽어내기 기능부(235)는, 버퍼(215A)로부터 읽어낸 1개의 자막 액세스 유닛의 사이즈분만큼, 자막 읽어내기 포인터를 갱신하고, 스텝 S255으로 되돌아가서, 이하, 마찬가지의 처리가 반복된다.

[0836] [재동기 처리]

[0837] 다음으로, 도 2A 및 도 2B의 디코드 제어 모듈(214)에 의한, 비디오 데이터와 오디오 데이터의 동기 제어에 대해서 설명한다.

[0838] 도 30의 S130에서 설명한 바와 같이, 디코드 제어 모듈(214)은, 동기를 확보 하기 위해 필요하면 타이밍을 어긋나게 하여, 디코드 개시를, 비디오 디코더 제어 모듈(216), 오디오 디코더 제어 모듈(217), 및 자막 디코더 제어 모듈(218)에 명령하지만, 예를 들면, 그 후의 비디오 디코더(116)와 오디오 디코더(117)의 디코드 처리의 진행 정도에 의해, 비디오 데이터의 출력과, 그 비디오 데이터와 동기하여 출력될 출력 데이터로서의 오디오 데이터의 출력이 어긋나는 경우가 있다.

[0839] 따라서, 디코드 제어 모듈(214)에서는, 비디오 데이터의 출력과, 그 비디오 데이터와 동기하여 출력될 오디오 데이터의 출력에 발생한 어긋남을 보정하고, 비디오 데이터와 오디오 데이터가 동기하여 출력되도록 하기 위한 재동기 처리가 행해진다.

[0840] 도 39의 플로우차트를 참조하여, 재동기 처리에 대해서 설명한다.

[0841] 재동기 처리에서는, 우선 처음에, 스텝 S271에서, 디코드 제어 모듈(214)은, 비디오 디코더 제어 모듈(216)로부터의 비디오 액세스 유닛의 타임 스탬프와, 오디오 제어 모듈(217)로부터의 오디오 액세스 유닛의 타임 스탬프의 어긋남이 큰지의 여부를 판정한다.

[0842] 즉, 도 30의 스텝 S129에서 설명한 바와 같이, 비디오 디코더 제어 모듈(216)은, 버퍼 제어 모듈(215)로부터 비디오 액세스 유닛를 얻을 때마다, 그 비디오 액세스 유닛의 타임 스탬프를, 디코드 제어 모듈(214)에 공급한다. 마찬가지로, 오디오 제어 모듈(217)도, 버퍼 제어 모듈(215)로부터 오디오 액세스 유닛을 얻을 때마다, 그 오디오 액세스 유닛의 타임 스탬프를, 디코드 제어 모듈(214)에 공급한다.

[0843] 스텝 S271에서는, 디코드 제어 모듈(214)은, 비디오 디코더 제어 모듈(216)과 오디오 제어 모듈(217)의 각각으로부터, 동일 타이밍에서(동일 타이밍으로 간주할 수 있는, 임의의 시간 내에) 공급되는 타임 스탬프끼리를 비교하고, 그들의 타임 스탬프의 어긋남이 큰지의 여부를 판정한다. 스텝 S271에서, 비디오 디코더 제어 모듈(216)로부터의 비디오 액세스 유닛의 타임 스탬프와, 오디오 제어 모듈(217)로부터의 오디오 액세스 유닛의 타임 스탬프의 어긋남이 크지 않다고 판정된 경우, 즉, 비디오 액세스 유닛의 타임 스탬프와, 오디오 액세스

유닛의 타임 스탬프의 어긋남이, 미리 정해진 동기가 취해져 있다고 간주할 수 있는 범위 내인, 예를 들면, 2 비디오 프레임(약 66밀리초)인 경우, 스텝 S271로 되돌아가서, 타임 스탬프끼리의 어긋남의 판정(감시)이 속행된다.

[0844] 한편, 스텝 S271에서, 비디오 디코더 제어 모듈(216)로부터의 비디오 액세스 유닛의 타임 스탬프와, 오디오 제어 모듈(217)로부터의 오디오 액세스 유닛의 타임 스탬프의 어긋남이 크다고 판정된 경우, 즉, 비디오 액세스 유닛의 타임 스탬프와, 오디오 액세스 유닛의 타임 스탬프의 어긋남이, 미리 정해진 동기가 취해져 있다고 간주할 수 있는 범위 밖인 경우, 스텝 S272로 진행하여, 디코드 제어 모듈(214)은, 비디오 디코더 제어 모듈(216)로부터의 비디오 액세스 유닛의 타임 스탬프와, 오디오 제어 모듈(217)로부터의 오디오 액세스 유닛의 타임 스탬프를 비교함으로써, 비디오 데이터의 출력(디코드)과, 오디오 데이터의 출력 중 어느 것이 지연되어 있는지를 판정한다.

[0845] 스텝 S272에서, 비디오 데이터의 출력이, 오디오 데이터의 출력보다도 지연되어 있다고 판정된 경우, 스텝 S273로 진행하여, 디코드 제어 모듈(214)은, 1비디오 액세스 유닛만큼, 비디오 액세스 유닛의 처리를 진행시키기 위해서, 비디오 디코더 제어 모듈(216)에 대하여, 비디오 액세스 유닛의 디코드와 출력(표시)을 행하지 않는 취지의 지시, 즉, 비디오 액세스 유닛의 처리의 스킵의 지시를 출력하고, 스텝 S274로 진행한다.

[0846] 스텝 S274에서는, 비디오 디코더 제어 모듈(216)은, 디코드 제어 모듈(214)로부터의 스킵의 지시를 수신하고, 그 스킵의 지시에 따라서, 버퍼 제어 모듈(215)로부터의 비디오 액세스 유닛과 함께 공급되는 au_ref_flag(도 24)를 검사한다.

[0847] 즉, private_stream_2의 PES_packet()의 private_stream2_PES_payload()(도 23)에 배치된 au_information()(도 24)에는, 액세스 유닛에 관한 정보로서의 au_ref_flag가 포함되어 있고, 버퍼 제어 모듈(215)은, 도 30의 스텝 S129나, 도 36의 스텝 S216에서 설명한 바와 같이, 비디오 액세스 유닛과 함께, 그 비디오 액세스 유닛의 au_ref_flag를 비디오 디코더 제어 모듈(216)에 공급한다.

[0848] 스텝 S274에서는, 이와 같이, 액세스 유닛과 함께 공급되는, 그 액세스 유닛의 au_ref_flag가 검사된다.

[0849] 그리고, 스텝 S274로부터 S275로 진행하여, 비디오 디코더 제어 모듈(216)은, 버퍼 제어 모듈(215)로부터 공급된 비디오 액세스 유닛의 au_ref_flag의 검사의 결과에 기초하여, 그 비디오 액세스 유닛이, 다른 픽쳐의 디코드 시에 참조되지 않는 비참조 화상인지의 여부를 판정한다.

[0850] 여기서, 도 24에서 설명한 바와 같이, 비디오 액세스 유닛의 au_ref_flag는, 그 액세스 유닛이 참조 화상인지의 여부를 나타내고, 참조 화상인 경우에는 1로 되고, 참조 화상이 아닌 경우, 즉, 비참조 화상인 경우에는 0으로 된다.

[0851] 스텝 S275에서, 버퍼 제어 모듈(215)로부터 공급된 비디오 액세스 유닛이 비참조 화상(의 비디오 액세스 유닛)이 아니라고 판정된 경우, 즉, 버퍼 제어 모듈(215)로부터 공급된 비디오 액세스 유닛이 참조 화상인 경우, 스텝 S276으로 진행하여, 비디오 디코더 제어 모듈(216)은, 통상대로, 그 비디오 액세스 유닛을, 비디오 디코더(116)에 처리시키고, 다음 비디오 액세스 유닛이, 버퍼 제어 모듈(215)로부터 공급되는 것을 대기하고, 스텝 S274로 되돌아간다. 또한, 스텝 S275에서, 버퍼 제어 모듈(215)로부터 공급된 비디오 액세스 유닛이 비참조 화상으로 판정된 경우, 스텝 S277로 진행하여, 비디오 디코더 제어 모듈(216)은, 그 비디오 스트림 유닛의, 비디오 디코더(116)에 의한 처리를 스킵시키고, 다음 비디오 액세스 유닛이, 버퍼 제어 모듈(215)로부터 공급되는 것을 대기하고, 스텝 S271로 되돌아간다.

[0852] 이와 같이, 비디오 액세스 유닛의 처리가 스킵됨으로써, 비디오 액세스 유닛의 처리가, 거의 1비디오 액세스 유닛분만큼 진행된다(처리 시간이 단축된다). 그 결과, 오디오 데이터의 출력보다도 지연되어 있던 비디오 데이터의 출력이 빨라지게 된다.

[0853] 한편, 스텝 S272에서, 비디오 데이터의 출력이, 오디오 데이터의 출력보다도 지연되어 있지 않다고 판정된 경우, 즉, 오디오 데이터의 출력이, 비디오 데이터의 출력보다도 지연되어 있는 경우, 스텝 S278로 진행하여, 디코드 제어 모듈(214)은, 비디오 액세스 유닛의 처리를 대기시키기 위해서, 비디오 디코더 제어 모듈(216)에 대하여, 지금 디코드되어 있는 비디오 액세스 유닛에 대응하는 비디오 데이터를 반복하여 출력하는 반복 출력의 지시를 출력하고, 스텝 S279로 진행한다.

[0854] 스텝 S279에서는, 비디오 디코더 제어 모듈(216)은, 디코드 제어 모듈(214)로부터의 반복 출력의 지시를 수신하고, 그 반복 출력의 지시에 따라서, 지금 비디오 디코더(116)에서 디코드되어 있는 비디오 액세스 유닛에 대응하는 비디오 데이터를 반복하여, 그래픽 처리 모듈(219)에 출력하고, 다음 비디오 액세스 유닛이, 버퍼

제어 모듈(215)로부터 공급되는 것을 대기하고, 스텝 S271로 되돌아간다.

[0855] 이상과 같이, 디코드 제어 모듈(214)에서는, 비디오 데이터의 출력이, 오디오 데이터의 출력보다도 지연되어 있는지의 여부를 판정하고, 비디오 데이터의 출력이, 오디오 데이터의 출력보다도 지연되어 있는 경우에는, 1개의 액세스 유닛의 처리의 스킵을, 비디오 디코더 제어 모듈(216)에 지시한다. 그리고, 비디오 디코더 제어 모듈(216)에서는, 스킵이 지시된 액세스 유닛의 au_ref_flag에 기초하여, 그 액세스 유닛이 참조 화상인지,또는 비참조 화상인지를 판정하고, 비참조 화상인 경우에, 비디오 디코더(116)에, 스킵이 지시된 액세스 유닛의 처리를 스킵시킨다. 따라서, 비디오 데이터의 출력과, 오디오 데이터의 출력의 동기를, 용이하게 취할 수 있다.

[0856] 즉, 처리를 스킵하는 액세스 유닛이 참조 화상인 경우, 그 액세스 유닛에 대응하는 비디오 데이터는, 그 후에 디코드되는 다른 액세스 유닛의 디코드 시에 참조하기 위해 디코드할 필요가 있다. 따라서, 비디오 데이터의 출력과, 오디오 데이터의 출력의 동기를 취하기 위한 동기 제어에서, 참조 화상의 액세스 유닛의 처리를 스킵하게 되면, 그 참조 화상을 참조하는 다른 액세스 유닛을 디코드할 수 없고, 그 결과, 비디오 데이터의 표시에서, 동기 제어가 노이즈로서 나타나게 된다.

[0857] 이 때문에, 처리를 스킵하는 것은, 참조 화상이 아닌 액세스 유닛, 즉, 비참조 화상의 액세스 유닛으로 하는 것이 바람직하다.

[0858] 한편, 종래의 엘리멘터리 스트림에 대해서, 비참조 화상의 액세스 유닛을 찾기 위해서는, 엘리멘터리 스트림의 구문 해석을 행할 필요가 있지만, 예를 들면, MPEG4-AYC 등에 따른 부호화에 의해 얻어지는 엘리멘터리 스트림은, 구문이 매우 복잡하기 때문에, 구문 해석에, 많은 코스트가 든다.

[0859] 이에 대하여, 디스크(101)에 기록된 클립 스트림 파일에 저장된 프로그램 스트림에는, 비디오 액세스 유닛이 PES_packet_data_byte에 배치되는 PES_packet()(도 16A 및 도16B 내지 도 18A 및 도 18B)와는 별도로, PES_packet_data_byte를 확장한 private_stream2_PES_payload()(도 23)가 배치된 private_stream_2의 PES_packet()가 다중화되어 있고, 그 private_stream2_PES_payload()의 au_information()(도 24)에는, 비디오 액세스 유닛마다, 그 비디오 액세스 유닛이 참조 화상인지, 또는 비참조 화상인지를 나타내는 au_ref_flag가 기술되어 있다. 그리고, 그 au_ref_flag는, 대응하는 비디오 액세스 유닛과 함께, 버퍼 제어 모듈(215)로부터 비디오 디코더 제어 모듈(216)에 공급된다. 따라서, 비디오 디코더 제어 모듈(216)은, 비디오 액세스 유닛과 함께 공급되는, 그 비디오 액세스 유닛의 au_ref_flag를 검사함으로써, 코스트를 거의 들이지 않고, 비디오 액세스 유닛이 참조 화상인지, 또는 비참조 화상인지를 인식할 수 있다.

[0860] [마크 처리]

[0861] 다음으로, 도 40의 플로우차트를 참조하여, PlayListMark()(도 7)에 기술된 Mark()에 기초하여 행해지는 마크 처리에 대해서 설명한다.

[0862] 디코드 제어 모듈(214)은, 내장한 계시부(214A)에 의해 계시되어 있는 지금 시각을, 항상 확인하고 있으며, 스텝 S301에서, 지금 시각이, PlayListMark()(도 7)에 기술된 어느 하나의 Mark()의 mark_time_stamp에 일치하였는지의 여부를 판정한다.

[0863] 즉, 도 30의 스텝 S124에서 설명한 바와 같이, 플레이어 제어 모듈(212)은, 도 25에 도시한 1번째의 PlayList#0의 1번째의 PlayItem#0을 재생하자고 할 때에, 도 28 상측에 도시한 PlayListMark()에 포함되는 7개의 Mark() 중의 1번째로부터 4번째까지의 4개의 Mark()가, PlayList#0의 1번째의 PlayItem#0에 속해 있는 것을 인식하고, 그 4개의 Mark()의 mark_time_stamp인 {180,090}, {5, 580, 090}, {10, 980, 090}, {16, 380, 090}을, 그 mark_time_stamp가 나타내는 시각의 속성이 「마크 처리」라는 취지와 함께, 디코드 제어 모듈(214)에 건네주고 있다.

[0864] 스텝 S301에서는, 디코드 제어 모듈(214)에서, 지금 시각이, 전술한 바와 같이 하여 플레이어 제어 모듈(212)로부터 공급된 「마크 처리」의 속성의 시각(mark_time_stamp) 중 어느 하나와 일치하는지의 여부가 판정된다.

[0865] 스텝 S301에서, 지금 시각이, 「마크 처리」의 속성의 시각 중 어느 것과도 일치하지 않는다고 판정된 경우, 스텝 S301로 되돌아가서, 마찬가지의 처리가 반복된다.

[0866] 또한, 스텝 S301에서, 지금 시각이, 「마크 처리」의 속성의 시각 중 어느 하나와 일치한다고 판정된 경우, 디코드 제어 모듈(214)은, 지금 시각이, 「마크 처리」의 속성의 시각으로 되었다는 취지의 메시지와, 지금

시각과 일치한 「마크 처리」의 속성의 시각을, 플레이어 제어 모듈(212)에 공급하고, 스텝 S302로 진행한다.

[0867] 스텝 S302에서는, 플레이어 제어 모듈(212)이, 지금 시각이, 「마크 처리」의 속성의 시각으로 되었다는 취지의 메시지와, 지금 시각과 일치한 「마크 처리」의 속성의 시각(mark_time_stamp)을, 디코드 제어 모듈(214)로부터 수신하고, mark_time_stamp가 지금 시각에 일치한 Mark()를, 마크 처리의 처리 대상으로 하는 Mark()(이하, 적절하게, 처리 대상 mark라고 함)로서 인식한다.

[0868] 즉, 플레이어 제어 모듈(212)은, 지금 재생되고 있는 PlayList()의 PlayItem()을 인식하고 있으며, 그 PlayList() 및 PlayItem()과, 디코드 제어 모듈(214)로부터의, 지금 시각과 일치한 「마크 처리」의 속성의 시각(mark_time_stamp)(이하, 적절하게, 마크 시각이라고 함)으로부터, "PLAYLIST. DAT" 파일(도 5)의 PlayList Mark()(도 7)를 참조함으로써, 처리 대상 mark를 인식한다.

[0869] 구체적으로는, 예를 들면, 지금, 도 25에 도시한 1번째의 PlayList#0의 1번째의 PlayItem#0이 재생되고 있다고 하면, 그에 의해, 플레이어 제어 모듈(212)은, 마크 시각이, 도 28 상측에 도시한 PlayListMark()에 포함되는 7개의 Mark() 중 1번째로부터 4번째까지의 4개의 Mark() 중 어느 하나의 mark_time_stamp인 것을 인식한다.

[0870] 그리고, 디코드 제어 모듈(214)로부터 플레이어 제어 모듈(212)에 공급된 마크 시각이, 예를 들면, 16,380,090이었다라고 하면, 플레이어 제어 모듈(212)은, 도 28 상측에 도시한 PlayListMark()에 포함되는 1번째로부터 4번째까지의 4개의 Mark() 중, mark_time_stamp가, 마크 시각인 16,380,090에 일치하는 4번째의 Mark()를, 처리 대상 mark로서 인식한다.

[0871] 플레이어 제어 모듈(212)은, 이상과 같이 하여, 처리 대상 mark를 인식하면, 스텝 S302로부터 S303으로 진행하여, 처리 대상 mark에서, 엘리멘터리 스트림을 특정하는 entry_ES_stream_id와 entry_ES_private_stream_id(도 7)가 기술되어 있는지의 여부를 판정한다.

[0872] 스텝 S303에서, 처리 대상 mark에, 엘리멘터리 스트림을 특정하는 entry_ES_stream_id와 entry_ES_private_stream_id(도 7)가 기술되어 있지 않다고 판정된 경우, 즉, entry_ES_stream_id와 entry_ES_private_stream_id가, 모두 0x00인 경우, 스텝 S304를 스킵하여, 스텝 S305로 진행하고, 이하, 처리 대상 mark에 따른 처리가 행해진다.

[0873] 또한, 스텝 S303에서, 처리 대상 mark에, 엘리멘터리 스트림을 특정하는 entry_ES_stream_id와 entry_ES_private_stream_id(도 7)가 기술되어 있다고 판정된 경우, 스텝 S304로 진행하여, 플레이어 제어 모듈(212)은, 재생 중인 엘리멘터리 스트림에, 그 entry_ES_stream_id, 나아가서는, 필요에 따라서 entry_ES_private_stream_id에 의해 특정되는 엘리멘터리 스트림이 포함되는지의 여부를 판정한다.

[0874] 스텝 S304에서, 재생 중인 엘리멘터리 스트림에, 처리 대상 mark의 entry_ES_stream_id와 entry_ES_private_stream_id에 의해 특정되는 엘리멘터리 스트림이 포함되지 않는다고 판정된 경우, 스텝 S301로 되돌아간다. 즉, 처리 대상 mark의 entry_ES_stream_id와 entry_ES_private_stream_id에 의해 특정되는 엘리멘터리 스트림이 재생되고 있지 않은 경우, 처리 대상 mark는, 무시된다.

[0875] 한편, 스텝 S304에서, 재생 중인 엘리멘터리 스트림에, 처리 대상 mark의 entry_ES_stream_id와 entry_ES_private_stream_id에 의해 특정되는 엘리멘터리 스트림이 포함된다고 판정된 경우, 즉, 처리 대상 mark의 entry_ES_stream_id와 en y_ES_private_stream_id에 의해 특정되는 엘리멘터리 스트림이 재생되고 있는 경우, 처리 대상 mark는 유효라고 하여, 스텝 S305로 진행하고, 이하, 그 처리 대상 mark에 따른 처리가 행해진다.

[0876] 즉, 스텝 S305에서는, 플레이어 제어 모듈(212)은, 처리 대상 mark의 mark_type(도 7)을 참조함으로써, 그 처리 대상 mark를 판정한다.

[0877] 스텝 S305에서, 처리 대상 mark가, 챕터 마크 또는 인덱스 마크라고 판정된 경우, 즉, 처리 대상 mark의 mark_type이, 'Chapter' 또는 'Index'인 경우, 스텝 S306으로 진행하여, 플레이어 제어 모듈(212)은, 그래픽 처리 모듈(219)에 명령하여, 챕터 또는 인덱스 번호의 표시를, 처리 대상 mark인 챕터 마크 또는 인덱스 마크가 나타내는 챕터 또는 인덱스의 번호에 갱신시키고, 스텝 S301로 되돌아간다.

[0878] 또한, 스텝 S305에서, 처리 대상 mark가, 이벤트 마크라고 판정된 경우, 즉, 처리 대상 mark의 mark_type이, 'Event'인 경우, 스텝 S307로 진행하여, 플레이어 제어 모듈(212)은, 이벤트의 발생을 나타내는 이벤트 메시

지와, 처리 대상 mark의 mark_data를, 스크립트 제어 모듈(211)에 통지(공급)하고, 스텝 S308로 진행한다.

[0879] 스텝 S308에서는, 스크립트 제어 모듈(211)이, 플레이어 제어 모듈(212)로부터의 이벤트 메시지와 mark_data를 수신하고, 이벤트 메시지를 인터럽트 요구로 하여, 미리 "SCRIPT.DAT" 파일에 기술된 일련의 처리를, mark_data를 인수로 하여 행하고, 스텝 S301로 되돌아간다.

[0880] 즉, 스크립트 제어 모듈(211)에서는, mark_data에 대응한 처리가 행해진다.

[0881] 구체적으로는, 예를 들면, 도 28 하측에 도시한 PlayList#1의 PlayListMark()에서는, 2번째의 Mark()(Mark#1)과 3번째의 Mark()(Mark#2)는, 모두, mark_type이 'Event'이지만, mark_data는, 각각 1(Mark#1)과 2(Mark#2)로 서로 다르다.

[0882] 스크립트 제어 모듈(211)은, 2번째의 Mark()에 대응하는 이벤트 메시지를 수신한 경우와, 3번째의 Mark()에 대응하는 이벤트 메시지를 수신한 경우의, 어느 경우에도, 그 이벤트 메시지에 따라서, 동일한 이벤트 핸들러(인터럽트 처리 루틴)에 의해 처리를 행하지만, 이벤트 핸들러 내에서, 이벤트 메시지와 함께 공급되는 mark_data를 검사함으로써, 이벤트 메시지에 대하여, mark_data마다 서로 다른 처리를 행한다.

[0883] 구체적으로는, 예를 들면, mark_data가 1인 경우에는, 스크립트 제어 모듈(211)은, 그래픽 처리 모듈(219)을 제어하여, 제1 종류의 아이콘의 표시를 행하게 한다. 또한, 예를 들면, mark_data가 2인 경우, 스크립트 처리 모듈(211)은, 그래픽 처리 모듈(219)을 제어하여, 제2 종류의 아이콘의 표시를 행하게 한다.

[0884] 또한, mark_data는, 1이나 2에 한정되는 것은 아니고, 또한, mark_data에 대응하여 행하여지는 처리도, 전술한 바와 같은, 단순한 아이콘의 표시 한정되는 것은 아니다.

[0885] 즉, 예를 들면, mark_data가 3 내지 18의 범위의 값인 경우에는, 스크립트 제어 모듈(211)은, 그래픽 처리 모듈(219)을 제어하여, 제1 종류의 아이콘의 표시를, mark_data로부터 2를 감한 값(1~16의 수치)에 대응하는 밝기로 행하게 한다. 또한, 예를 들면, mark_data가 19 내지 34의 범위의 값인 경우에는, 스크립트 제어 모듈(211)은, 그래픽 처리 모듈(219)을 제어하여, 제2 종류의 아이콘의 표시를, mark_data로부터 18을 감한 값(1~16의 수치)에 대응하는 밝기로 행하게 한다.

[0886] 그 외에, 예를 들면, 입력 인터페이스(115)(도 1)에, 유저가 조작하는 컨트롤러가 접속되고 있고, 그 컨트롤러가, DC(Direct Current) 모터의 축으로 편심시킨 추를 부착한, DC 모터를 동작시키면 진동이 발생하는 진동 모터를 내장하는 경우에는, mark_data가 35 내지 42의 범위의 값일 때에, 그 진동 모터를, mark_data로부터 34를 감한 값(1~8의 수치)에 따른 동작 시간만큼 동작시킬 수 있다.

[0887] mark_data는 수치이며, 그 사용법이나 알고리즘은, 스크립트 제어 모듈(211)이 실행하는 스크립트 프로그램에 의해 기술할 수 있다. 따라서, mark_data는, 사전에 정해진 룰에 의해 사용하는 것 외에, 디스크(101)의 제조자, 혹은 디스크(101)에 기록되는 데이터를 제공하는 콘텐츠 프로바이더 등이 독자적으로 설정한 룰에 의해 사용하는 것이 가능하다.

[0888] 이상과 같이, 마크 처리에서는, 지금 시각이, 「마크 처리」의 속성의 시각과 일치하면, 그 「마크 처리」의 속성의 시각인 마크 시각으로부터, 처리 대상 mark가 인식된다. 또한, 처리 대상 mark에서, 엘리멘터리 스트림을 특정하는 entry_ES_stream_id와 entry_ES_private_stream_id가 기술되어 있지 않은 경우에는, 처리 대상 mark의 mark_type에 따른 처리가 행해진다. 또한, 처리 대상 mark에서, 엘리멘터리 스트림을 특정하는 entry_ES_stream_id와 entry_ES_private_stream_id가 기술되어 있는 경우에도, 그 entry_ES_stream_id와 entry_ES_private_stream_id에 의해 특정되는 엘리멘터리 스트림이 재생 중이면, 처리 대상 mark의 mark_type에 따른 처리가 행해진다.

[0889] 따라서, 예를 들면, 지금, 도 25에 도시한 2번째의 PlayList#1의 재생이 행해지고 있다고 하면, 이하와 같은 마크 처리가 행해진다.

[0890] 즉, 2번째의 PlayList#1의 PlayListMark()에서는, 도 28 하측에 도시한 바와 같이, mark_time_stamp가 각각 90,000, 27,090,000, 27,540,000으로 지정되어 있는 1번째의 Mark()(Mark#0), 2번째의 mark()(Mark#1), 3번째의 Mark()(Mark#2)가 기술되어 있다.

[0891] 또한, 도 28 하측의 PlayListMark()에서는, 2번째의 Mark()와 3번째의 Mark()의 entry_ES_stream_id에는, 각각, 0xE0과 0xE1이 기술되어 있기 때문에, 2번째의 Mark()과 3번째의 Mark()는, 각각, stream_id가 0xE0과 0xE1로 특정되는 엘리멘터리 스트림이 관련지어져 있다.

[0892] 여기에서, 도 25에서 설명한 바와 같이, 2번째의 PlayList#1에는, 1개의 PlayItem()(PlayItem#0)만이 기술되고, 그 PlayItem#0에 따르면, 클립 스트림 파일 "00003.PS"가 재생된다. 그리고, 클립 스트림 파일 "00003.PS"에는, 그 클립 스트림 파일 "00003.PS"에 대응하는 도 26A 및 도 26B의 클립 정보 파일 "00003.LP"에서 설명한 바와 같이, 0xE0으로 되어 있는 Stream_id에 의해 특정되는 비디오 스트림 stream#0, 0xE1로 되어 있는 stream_id에 의해 특정되는 비디오 스트림 stream#1, 0xBD로 되어 있는 stream_id 및 0x00으로 되어 있는 private_stream_id에 의해 특정되는 오디오 스트림 stream#2의 3개의 엘리멘터리 스트림이 다중화되어 있다.

[0893] 따라서, 도 28 하측의 PlayListMark()의 2번째의 Mark()에는, 클립 스트림 파일 "00003.PS"에 다중화되어 있는, stream_id가 0xE0으로 되어 있는 비디오 스트림 stream#0이 관련지어져 있으며, 3번째의 Mark()에는, 클립 스트림 파일 "00003.PS"에 다중화되어 있는, stream_id가 0xE1로 되어 있는 비디오 스트림 stream#1이 관련지어져 있다.

[0894] 도 25의 2번째의 PlayList#1의 PlayItem#0의 재생이 개시되는 경우, 도 30의 스텝 S124에서 설명한 바와 같이 하여, 플레이어 제어 모듈(212)은, 도 28 하측에 도시한 PlayListMark()에 포함되는 3개의 Mark()가, PlayList#1의 PlayItem#0에 속해 있는 것을 인식하고, 그 3개의 Mark()의 mark_time_Stamp인 {90,000}, {27,090,000}, {27,540,000}을, 그 mark_time_Stamp이 나타내는 시각의 속성이 「마크 처리」라는 취지와 함께, 디코드 제어 모듈(214)에 건네주고 있다.

[0895] 마크 처리에서는, 디코드 제어 모듈(214)이, PlayList#1의 PlayItem#0의 재생 중에, 계시부(214A)에 의해 계시되는 지금 시각이, 속성이 「마크 처리」의 시각 {90,000}, {27,090,000}, {27,540,000} 중 어느 하나에 일치하는지를, 항상 확인하고 있으며(스텝 S301), 지금 시각이, 속성이 「마크 처리」의 시각에 일치하면, 지금 시각과 일치한 「마크 처리」의 속성의 시각인 마크 시각과, 지금 시각이, 「마크 처리」의 속성의 시각으로 되었다는 취지의 메시지를, 플레이어 제어 모듈(212)에 공급한다.

[0896] 즉, 예를 들면, 지금, 지금 시각이, 「마크 처리」의 속성의 시각 {90,000}, {27,090,000}, {27,540,000} 중, 27,090,000에 일치했다고 하면, 디코드 제어 모듈(214)은, 지금 시각과 일치한 「마크 처리」의 속성의 시각인 마크 시각 27,090,000과, 지금 시각이, 「마크 처리」의 속성의 시각으로 되었다는 취지의 메시지를, 플레이어 제어 모듈(212)에 공급한다.

[0897] 플레이어 제어 모듈(212)은, PlayList#1의 PlayItem#0이 지금 재생되고 있는 것을 인식하고 있으며, 그 PlayList#1의 도 28 하측에 도시한 PlayListMark()에 기술된 Mark() 중, PlayItem#0에 속하는 3개의 Mark()의 mark_time_stamp인 90,000, 27,090,000, 27,540,000 각각과, 디코드 제어 모듈(214)로부터의 마크 시각인 27,090,000을 비교함으로써, 그 마크 시각 27,090,000에, mark_time_stamp가 일치하는 Mark(), 즉, 도 28 하측의 PlayListMark()에 기술된 2번째의 Mark()(Mark#1)를, 처리 대상 mark로서 인식한다(스텝 S302).

[0898] 처리 대상 mark인, 도 28 하측의 PlayListMark()에 기술된 2번째의 Mark()에서는, entry_ES_stream_id로서, 0xE0이 지정되어 있다. 이 0xE0으로 되어 있는 entry_ES_stream_id는, 전술한 것으로부터, 클립 스트림 파일 "00003.PS"에 다중화되어 있는, stream_id가 0xE0으로 되어 있는 비디오 스트림 stream#0(도 26A 및 도 26B)을 나타내는 것이며, 플레이어 제어 모듈(212)은, 재생 중인 엘리멘터리 스트림 중에, 그 비디오 스트림 stream#0이 포함되는지의 여부를 판정한다(스텝 S303, S304).

[0899] 그리고, 재생 중인 엘리멘터리 스트림 중에, 비디오 스트림 stream#0이 포함되지 않은 경우에는, 처리 대상 mark는 무시된다(스텝 S304).

[0900] 한편, 재생 중인 엘리멘터리 스트림 중에, 비디오 스트림 stream#0이 포함되는 경우에는, 처리 대상 mark는 유효라고 하여, 그 처리 대상 mark에 따른 처리가 행해진다(스텝 S305 내지 S308).

[0901] 즉, 지금의 경우, 처리 대상 mark인, 도 28 하측의 PlayListMark()에 기술된 2번째의 Mark()는, 그 mark_type이 'Event'로 되어 있기 때문에 이벤트 마크이며, 따라서, 플레이어 제어 모듈(212)은, 이벤트의 발생을 나타내는 이벤트 메시지와, 처리 대상 mark의 mark_data를, 스크립트 제어 모듈(211)에 공급한다(스텝 S305, S307). 그리고, 스크립트 제어 모듈(211)에서는, 플레이어 제어 모듈(212)로부터의 이벤트 메시지를 인터럽트 요구로 하여, 미리 "SCRIPT.DAT" 파일에 기술된 일련의 처리를, 그 이벤트 메시지와 함께 공급된 mark_data를 인수로 하여 행한다(스텝 S308).

[0902] 이상과 같이, 마크 처리에 따르면, PlayList()의 시간축 상의 1개의 재생 시각을 나타내는 mark_time_Stamp와, Mark()의 타입을 나타내는 mark_type과, 이벤트 마크의 인수로 되는 mark_data를 포함하

는 0이상의 Mark()를 갖는 PlayListMark()(도 7)을 포함하는 PlayList()(도 5)에 따라서 재생되고 있는 클립 스트림 파일의 재생 시각인 지금 시각이, mark_time_Stamp에 일치하는지의 여부가 판정되고, 지금 시각이, mark_time_stamp에 일치하는 경우에, 그 일치한 지금 시각인 마크 시각과 동등한 mark_time_Stamp를 갖는 Mark()가, 처리 대상 mark로서 인식된다. 또한, 그 처리 대상 mark가 갖는 mark_type이, 이벤트를 발생시키는 타입을 나타내고 있는 경우, 즉, 처리 대상 mark가, 이벤트 마크인 경우, 처리 대상 mark가 갖는 mark_data와 이벤트 메시지가 통지되고, 그 mark_data에 따른 처리가 실행된다. 따라서, 클립 스트림 파일의 재생 시간에 따라서, mark_data에 따른 처리를 실행하는 것이 가능해진다.

[0903] [출력 속성의 제어 처리]

[0904] 다음으로, 도 41의 플로우차트를 참조하여, 도 30의 스텝 S126 등에서 행하여지는 출력 속성의 제어 처리의 상세에 대하여 설명한다.

[0905] 도 30의 스텝 S126에서 설명한 바와 같이, 플레이어 제어 모듈(212)은, 우선, 재생 대상의 1이상의 엘리멘터리 스트림, 즉, 도 30의 스텝 S125에서 재생한다고 결정한 1이상의 엘리멘터리 스트림 각각에 대해서, 출력 속성이 기술되는 DynamicInfo()(도 13)의 수를 나타내는 number_of_DynamicInfo(도 10)를 조사한다.

[0906] 그리고, 재생 대상의 1이상의 엘리멘터리 스트림의 전부에 대해서, number_of_DynamicInfo가 0으로 되어 있는 경우, 플레이어 제어 모듈(212)은, 특별히 처리를 행하지 않는다.

[0907] 한편, 재생 대상의 엘리멘터리 스트림에 대한 number_of_DynamicInfo가 0이 아닌 경우, 플레이어 제어 모듈(212)은, 도 41의 플로우차트에 따른 출력 속성의 제어 처리를 행한다.

[0908] 따라서, 디스크(101)에 기록된 3개의 클립 정보 파일 "00001.CLP", "00002.CLP", "00003.CLP"이, 예를 들면, 도 26A 및 도 26B에 도시한 바와 같이 되어 있는 경우에, 클립 정보 파일 "00001.CLP"에 대응하는 클립 스트림 파일 "00001.PS"(를 재생하는 1번째의 PlayList#0의 1번째의 PlayItem#0)가 재생될 때에는, 클립 정보 파일 "00001.CLP"(도 26A 및 도 26B)에서는, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림 stream#0 내지 stream#3의 전부에 대해서, number_of_DynamicInfo가 0으로 되어 있기 때문에, 출력 속성의 제어 처리는 행하여지지 않는다.

[0909] 마찬가지로, 클립 정보 파일 "00002.CLP"에 대응하는 클립 스트림 파일 "00002.PS"(를 재생하는 1번째의 PlayList#0의 2번째의 PlayItem#1)가 재생될 때도, 클립 정보 파일 "00002.CLP"(도 26A 및 도 26B)에서는, 클립 스트림 파일 "00002.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림 stream#0 내지 stream#3의 전부에 대해서, number_of_DynamicInfo가 0으로 되어 있기 때문에, 출력 속성의 제어 처리는 행하여지지 않는다.

[0910] 한편, 클립 정보 파일 "00003.CLP"에 대응하는 클립 스트림 파일 "00003.PS"(를 재생하는 2번째의 PlayList#1의 PlayItem#0)가 재생될 때는, 클립 정보 파일 "00003.CLP"(도 26A 및 도 26B)에서, 클립 스트림 파일 "00003.PS"에 다중화되어 있는 3개의 엘리멘터리 스트림 stream#0 내지 stream#2 중, 1번째의 엘리멘터리 스트림인 비디오 스트림 stream#0과, 3번째의 엘리멘터리 스트림인 오디오 스트림 stream#2에 대해서, number_of_DynamicInfo가 0이 아닌 2와 3으로, 각각 되어 있기 때문에, 출력 속성의 제어 처리가 행해진다.

[0911] 즉, 출력 속성의 제어 처리에서는, 우선 최초로, 스텝 S320에서, 플레이어 제어 모듈(212)은, 재생 대상의 클립 스트림 파일에 대응하는 클립 정보 파일 Clip()(도 10)에 기술된 pts_change_point를, 「DynamicInfo() 처리」의 속성의 시각이라는 취지와 함께, 디코드 제어 모듈(214)에 건네주고, 디코드 제어 모듈(214)은, 플레이어 제어 모듈(212)로부터의 「DynamicInfo() 처리」의 속성의 시각인 pts_change_point를 수신하고, 스텝 S321로 진행한다.

[0912] 스텝 S321에서는, 디코드 제어 모듈(214)이, 계시부(214A)에 의해 계시되어 있는 지금 시각이, 「DynamicInfo() 처리」의 속성의 시각인 pts_change_point(중의 어느 하나)에 일치하였는지의 여부를 판정하고, 일치하지 않았다고 판정한 경우, 스텝 S321로 되돌아간다.

[0913] 또한, 스텝 S321에서, 지금 시각이, 「DynamicInfo() 처리」의 속성의 시각(중의 어느 하나)에 일치했다고 판정된 경우, 디코드 제어 모듈(214)은, 지금 시각이, 「DynamicInfo() 처리」의 속성의 시각으로 되었다는 취지의 메시지와, 지금 시각과 일치한 「DynamicInfo() 처리」의 속성의 시각(이하, 적절하게, DynamicInfo 시각이라고 함)을, 플레이어 제어 모듈(212)에 공급하고, 스텝 S322로 진행한다.

[0914] 스텝 S332에서는, 플레이어 제어 모듈(212)이, 지금 시각이, 「DynamicInfo() 처리」의 속성의 시각으로 되었다는 취지의 메시지와, DynamicInfo 시각을, 디코드 제어 모듈(214)로부터 수신하고, 그 DynamicInfo 시각에

일치하는 pts_change_point(도 10)와 세트로 되어 있는 DynamicInfo()를, 처리 대상의 DynamicInfo()인 처리 대상 DynamicInfo()로서 인식하고, 스텝 S323으로 진행한다.

[0915] 스텝 S323에서는, 플레이어 제어 모듈(212)은, 처리 대상 DynamicInfo()로 되어 있는 DynamicInfo()(도 13)에 기술된 출력 속성을, 그래픽 처리 모듈(219) 또는 오디오 출력 모듈(221)에 공급하고, 스텝 S324로 진행한다.

[0916] 스텝 S324에서는, 그래픽 처리 모듈(219) 또는 오디오 출력 모듈(221)이, 직전의 스텝 S323에서 플레이어 제어 모듈(212)로부터 공급된 출력 속성에 따라서, 비디오 데이터 또는 오디오 데이터의 출력의 제어를, 각각 개시하고, 스텝 S321로 되돌아간다.

[0917] 이에 의해, 비디오 데이터가, 출력 속성(표시 방식)으로서 기술된, 예를 들면 어스펙트비에 따라서 출력되고, 혹은 오디오 데이터가, 출력 속성(출력 방식)으로서 기술된, 예를 들면, 스테레오 또는 듀얼(2개 국어)에 따라서 출력된다.

[0918] 다음으로, 도 42를 참조하여, 출력 속성의 제어 처리의 상세에 대해서, 더 설명한다.

[0919] 즉, 도 42는, 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"에 기술되어 있는 pts_change_point와 DynamicInfo()와의 세트(도 10)를 나타내고 있다.

[0920] 여기에서, 전술한 바와 같이, 클립 스트림 파일 "00003.PS"에 다중화되어 있는 3개의 엘리멘터리 스트림 stream#0 내지 stream#2 중, 1번째의 엘리멘터리 스트림인 비디오 스트림 stream#0과, 3번째의 엘리멘터리 스트림인 오디오 스트림stream#2에 대해서는, 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"에서, number_of_DynamicInfo가, 각각 2와 3으로 되어 있다. 따라서, 클립 정보 파일 "00003.CLP "에서, 클립 스트림 파일 "00003.PS"의 1번째의 비디오 스트림 stream#0에 대해서는, 2세트의 pts_change_point 및 DynamicInfo()가 기술되어 있으며, 3 번째의 오디오 스트림 stream#2에 대해서는, 3세트의 pts_change_point 및 DynamicInfo()가 기술되어 있다.

[0921] 도 42 상측은, 클립 스트림 파일 "00003.PS"의 1번째의 비디오 스트림 stream#0에 대하여 기술되어 있는 2세트의 pts_change_point 및 DynamicInfo()를 나타내고 있으며, 도 42 하측은, 클립 스트림 파일 "00003.PS"의 3번째의 오디오 스트림 stream#2에 대하여 기술되어 있는 3세트의 pts_change_point 및, DynamicInfo()를 나타내고 있다.

[0922] 또한, 도 42 상측에서는, 1번째의 비디오 스트림 stream#0에 대하여 기술되어 있는 2세트의 pts_change_point 및 DynamicInfo() 외에, 그 비디오 스트림 stream#0에 대해서, 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"에 기술되어 있는 Stream_id(=0xE0), private_stream_id(=0x00), number_of_DynamicInfo(=2)도, 도시하고 있다. 마찬가지로, 도 42 하측에서도, 3번째의 오디오 스트림 stream#2에 대하여 기술되어 있는 3세트의 pts_change_point 및, DynamicInfo() 외에, 그 오디오 스트림 stream#2에 대해서, 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"에 기술되어 있는 stream_id(=0xBD), private_stream_id(=0x00), number_of_DynamicInfo(=3)도, 도시하고 있다.

[0923] 도 42 상측에서, 비디오 스트림 stream#0에 대하여 기술되어 있는 2세트의 pts_change_point 및 DynamicInfo() 중의 1세트째에서는, pts_change_point가 90,000으로 되어 있고, DynamicInfo()의 display_aspect_ratio(도 13)가, '4:3'으로 되어 있다. 또한, 그 2세트째에서는, pts_change_point가 54,090,000으로 되어 있으며, DynamicInfo()의 display_aspect_ratio가, '16:9'로 되어 있다.

[0924] 한편, 도 42 하측에서, 오디오 스트림 stream#2에 대하여 기술되어 있는 3세트의 pts_change_point 및 DynamicInfo() 중 1세트째에서는, pts_change_point가 90,000으로 되어 있으며, DynamicInfo()의 channel_assignment(도 13)가 'Dual'로 되어 있다. 또한, 그 2세트째에서는, pts_change_point가 27,090,000으로 되어 있으며, DynamicInfo()의 channel_assignment가 'Stereo'로 되어 있다. 또한, 그 3세트째에서는, pts_change_point가 32,490,000으로 되어 있으며, DynamicInfo()의 channel_assignment가 'Dual'로 되어 있다.

[0925] 예를 들면, 지금, 도 30의 스텝 S125에서, 클립 스트림 파일 "00003.PS"의, 0xE0로 되어 있는 stream_id에 의해 특정되는 1번째의 비디오 스트림 stream#0과, 0xBD로 되어 있는 stream_id 및 0x00으로 되어 있는 private_stream_id에 의해 특정되는 3번째의 오디오 스트림 stream#2가, 재생 대상의 스트림으로서 결정되었다고 한다.

[0926] 이 경우, 플레이어 제어 모듈(212)은, 0xE0으로 되어 있는 Stream_id에 의해 특정되는 비디오 스트림

stream#0에 대하여 기술되어 있는 도 42 상측의 2세트의 pts_change_point 및 DynamicInfo()와, 0xBD로 되어 있는 stream_id 및 0x00으로 되어 있는 private_stream_id에 의해 특정되는 오디오 스트림 stream#2에 대하여 기술되어 있는 도 42 하측의 3세트의 pts_change_point 및, DynamicInfo() 중 pts_change_point를 조사하고, 초기값을 인식한다.

[0927] 즉, 0xE0으로 되어 있는 stream_id에 의해 특정되는 비디오 스트림 stream#0에 대하여 기술되어 있는 도 42 상측의 2세트의 pts_change_point 및 DynamicInfo() 중의 1세트째에서는, pts_change_point가 90,000으로 되어 있다. 그리고, 이 90,000이라고 하는 시각은, 비디오 스트림 stream#0이 다중화되어 있는 클립 스트림 파일 "00003.PS"에 대응하는 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"에서, 클립 스트림 파일 "00003.PS"의 선두의 시각을 나타내는 presentation_start_time에 기술되어 있는 시각 90,000에 일치한다.

[0928] 마찬가지로, 0xBD가 되어 있는 stream_id 및 0x00으로 되어 있는 private_stream_id에 의해 특정되는 오디오 스트림 stream#2에 대하여 기술되어 있는 도 42 하측의 3세트의 pts_change_point 및, DynamicInfo() 중 1세트째에서는, pts_change_point가 90,000으로 되어 있다. 그리고, 이 90,000이라고 하는 시각은, 오디오 스트림 stream#2가 다중화되어 있는 클립 스트림 파일 "00003.PS"에 대응하는 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"에서, 클립 스트림 파일 "00003.PS"의 선두의 시각을 나타내는 presentation_start_time에 기술되어 있는 시각 90,000에 일치한다.

[0929] 플레이어 제어 모듈(212)은, 클립 스트림 파일 "00003.PS"의 선두의 시각을 나타내는 presentation_start_time에 기술되어 있는 시각 90,000에 일치하는 pts_change_point를, 초기값으로서 인식한다. 따라서, 도 42 상측의 2세트의 pts_change_point 및 DynamicInfo() 중 1세트째의 pts_change_point와, 도 42 하측의 3세트의 pts_change_point 및 DynamicInfo() 중 1세트째의 pts_change_point가, 초기값으로서 인식된다.

[0930] 그리고, 플레이어 제어 모듈(212)은, 클립 스트림 파일 "00003.PS"의 재생이 개시되기 전에(도 30의 스텝 S126에서), 초기값으로서 인식한 pts_change_point와 세트로 되어 있는 DynamicInfo()에 따라서, 대응하는 엘리멘터리 스트림의 출력 속성을 지시한다.

[0931] 즉, 0xE0으로 되어 있는 stream_id에 의해 특정되는 비디오 스트림 stream#0에 대해서는, 도 42 상측에서, 초기값인 90,000으로 되어 있는 pts_change_point와 세트로 되어 있는 DynamicInfo()에서, display_aspect_ratio가 '4:3'으로 되어 있다. 이 경우, 플레이어 제어 모듈(212)은, display_aspect_ratio가 '4:3'으로 되어 있다 취지, 즉 비디오 스트림 stream#0이, 4:3의 어스펙트비의 비디오 데이터라는 취지의 출력 속성의 정보를 가지고, 그래픽 처리 모듈(219)을 제어한다.

[0932] 또한, 0xBD로 되어 있는 stream_id 및 0x00으로 되어 있는 private_stream_id에 의해 특정되는 오디오 스트림 stream#2에 대해서는, 도 42 하측에서, 초기값인 90,000으로 되어 있는 pts_change_point와 세트로 되어 있는 DynamicInfo()에서, channel_assignment가 'Dual'로 되어 있다. 이 경우, 플레이어 제어 모듈(212)은, channel_assignment가 'Dual'로 되어 있다는 취지, 즉, 오디오 스트림 stream#2가, 듀얼의 오디오 데이터라는 취지의 출력 속성의 정보를, 오디오 출력 모듈(221)에 공급한다.

[0933] 여기서, 도 30의 스텝 S126에서는, 이상과 같은 초기값으로서의 pts_change_point를 대상으로 하는 출력 속성의 제어 처리가 행해진다.

[0934] 그 후, 플레이어 제어 모듈(212)은, 비디오 스트림 stream#0에 대한 도 42 상측의 2개의 pts_change_point인 90,000 및 54,090,000와, 오디오 스트림 stream#2에 대한 도 42 하측의 3개의 pts_change_point인 90,000, 27,090,000, 및 32,490,000 중, 초기값 90,000 이외의 시각인 {27,090,000}, {32,490,000}, {54,090,000}을, 「DynamicInfo() 처리」의 속성의 시각이라는 취지와 함께, 디코드 제어 모듈(214)로 건네준다(스텝 S320) .

[0935] 디코드 제어 모듈(214)은, 플레이어 제어 모듈(212)로부터의, 「DynamicInfo() 처리」의 속성의 시각 {27,090,000}, {32,490,000}, {54,090,000}을 수신하고, 또한, 비디오 스트림 stream#0 및 오디오 스트림 stream#2의 재생(클립 스트림 파일 "00003.PS"를 재생하는 2번째의 PlayList#1의 PlayItem#0의 재생)의 개시 후, 계시부(214A)에 의해 계시되어 있는 지금 시각의 감시를 개시한다.

[0936] 그리고, 디코드 제어 모듈(214)은, 지금 시각이, 「DynamicInfo() 처리」의 속성의 시각 {27,090,000}, {32,490,000}, {54,090,000} 중 어느 하나에 일치한 경우, 그 지금 시각과 일치한 「DynamicInfo() 처리」의 속성의 시각인 DynamicInfo 시각을, 플레이어 제어 모듈(212)에 공급한다(스텝 S321).

[0937] 즉, 예를 들면, 지금 시각이, 27,090,000으로 되었다고 하면, 디코드 제어 모듈(214)은, 「DynamicInfo() 처

리」의 속성의 시각 중, 지금 시각과 일치하는 27,090,000을, DynamicInfo 시각으로서, 플레이어 제어 모듈(212)에 공급한다.

[0938] 플레이어 제어 모듈(212)은, 디코드 제어 모듈(214)로부터의 DynamicInfo 시각인 27,090,000을 수신하고, 비디오 스트림 stream#0에 대한 도 42 상측의 2개의 pts_change_point와, 오디오 스트림 stream#2에 대한 도 42 하측의 3개의 pts_change_point 중에서, DynamicInfo 시각인 27,090,000에 일치하는 pts_change_point를 조사하고, 그 27,090,000에 일치하는 pts_change_point와 세트로 되어 있는 DynamicInfo(), 즉, 오디오 스트림 stream#2에 대한 도 42 하측의 2번째의 DynamicInfo()를, 처리 대상 DynamicInfo()로서 인식한다(스텝 S322).

[0939] 처리 대상 DynamicInfo()가, 비디오 스트림에 대한 DynamicInfo()인 경우, 플레이어 제어 모듈(212)은, 처리 대상 DynamicInfo()에 기술되어 있는 출력 속성을, 그래픽 처리 모듈(219)에 공급한다(스텝 S323). 또한, 처리 대상 DynamicInfo()가, 오디오 스트림에 대한 DynamicInfo()인 경우, 플레이어 제어 모듈(212)은, 처리 대상 DynamicInfo()에 기술되어 있는 출력 속성을, 오디오 출력 모듈(221)에 공급한다(스텝 S323).

[0940] 그래픽 처리 모듈(219)은, 플레이어 제어 모듈(212)로부터 출력 속성이 공급 되면, 그 출력 속성에 따라서, 비디오 데이터의 출력의 제어를 개시한다(스텝 S324).

[0941] 즉, 그래픽 처리 모듈(219)은, 예를 들면, 플레이어 제어 모듈(212)로부터의 출력 속성이 나타내는, 비디오 데이터의 어스펙트비의 지시(display_aspect_ratio(도 13))와, 도 1의 비디오 출력 단자(120)에 접속된 비디오 출력 장치의 어스펙트비에 기초하여, 비디오 출력 모듈(220)에 출력하는 비디오 데이터의 어스펙트비의 변환을 행한다.

[0942] 구체적으로는, 예를 들면, 비디오 출력 장치의 어스펙트비가 16:9인 경우에, 출력 속성으로서의 비디오 데이터의 어스펙트비의 지시가 4:3인 어스펙트비를 나타내고 있을 때에는, 그래픽 처리 모듈(219)은, 비디오 출력 모듈(220)에 출력하는 비디오 데이터를, 가로 방향으로 스퀴즈 처리하고, 좌우에 검은 부분을 넣어서 출력한다. 또한, 예를 들면, 비디오 출력 장치의 어스펙트비가 4:3인 경우에서, 출력 속성으로서의 비디오 데이터의 어스펙트비의 지시가 16:9인 어스펙트비를 나타내고 있을 때에는, 그래픽 처리 모듈(219)은, 비디오 출력 모듈(220)에 출력하는 비디오 데이터를, 세로 방향으로 스퀴즈 처리하고, 상하에 검은 부분을 넣어서 출력한다. 또한, 예를 들면, 비디오 출력 장치의 어스펙트비와, 출력 속성으로서의 비디오 데이터의 어스펙트비의 지시가 나타내는 어스펙트비가, 모두, 4:3이나 16:9로, 동일한 경우, 그래픽 처리 모듈(219)은, 비디오 출력 모듈(220)에 출력하는 비디오 데이터를, 스퀴즈 처리하지 않고, 그대로 출력한다.

[0943] 여기에서, 도 42 상측에서, 0xE0으로 되어 있는 stream_id에 의해 특정되는 비디오 스트림 stream#0에 대하여 기술되어 있는 2세트의 pts_change_point 및 DynamicInfo()에 따르면, 비디오 스트림 stream#0의 재생 개시 시인 시각 90,000으로부터, 시각 54,090,000의 직전까지는, 비디오 스트림 stream#0으로부터, 4:3의 어스펙트비의 비디오 데이터가 얻어진다. 그리고, 시각 54,090,000 이후에는, 비디오 스트림 stream#0으로부터, 16:9의 어스펙트비의 비디오 데이터가 얻어진다.

[0944] 따라서, 예를 들면, 도 1의 비디오 출력 단자(120)에 접속된 비디오 출력 장치의 어스펙트비가 4:3이라고 하면, 그래픽 처리 모듈(219)에서는, 시각 90,000으로부터 시각 54,090,000의 직전까지는 비디오 스트림 stream#0으로부터 얻어지는 4:3의 어스펙트비의 비디오 데이터가, 그대로 4:3의 어스펙트비의 비디오 출력 장치에 공급되어 표시된다.

[0945] 그리고, 시각 54,090,000 이후는, 비디오 스트림 stream#0으로부터 얻어지는 16:9의 어스펙트비의 비디오 데이터가, 세로 방향으로 스퀴즈 처리되고, 또한, 상하에 검은 부분이 들어 간 4:3의 어스펙트비의 비디오 신호로 변환되고, 4:3의 어스펙트비의 비디오 출력 장치에 공급되어 표시된다.

[0946] 한편, 오디오 출력 모듈(221)은, 플레이어 제어 모듈(212)로부터 출력 속성이 공급되면, 그 출력 속성에 따라서, 오디오 데이터의 출력의 제어를 개시한다(스텝 S324).

[0947] 즉, 오디오 출력 모듈(221)은, 예를 들면, 플레이어 제어 모듈(212)로부터의 출력 속성이 나타내는, 오디오 데이터의 채널 할당의 지시(channel_assignment(도 13))와, 유저가 리모콘을 조작함으로써 입력 인터페이스(115)(도 1)를 통하여 플레이어 제어 모듈(212)로부터 공급되는 음성 출력 모드에 기초하여, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터를 처리하고, 오디오 출력 단자(121)(도 1)에 출력한다.

[0948] 구체적으로는, 예를 들면, 출력 속성이 나타내는 오디오 데이터의 채널 할당의 지시가, 좌채널이 「주음성」의 오디오 데이터이고, 우채널이 「부음성」의 오디오 데이터인 듀얼(2개 국어) 모드를 나타내고 있는 경우,

오디오 출력 모듈(221)은, 플레이어 제어 모듈(212)로부터 공급되는 음성 출력 모드에 따라서, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터를 처리하여, 오디오 출력 단자(121)에 출력한다.

[0949] 즉, 음성 출력 모드로서, 예를 들면, 「주음성」이 지정되어 있을 때에는, 오디오 출력 모듈(221)은, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터 중의 좌채널의 오디오 데이터를, 우채널의 오디오 데이터로서 카피하고, 그 좌채널과 우채널의 오디오 데이터(「주음성」의 오디오 데이터)를, 오디오 출력 단자(121)에 출력한다. 또한, 음성 출력 모드로서, 「부음성」이 지정되어 있을 때에는, 오디오 출력 모듈(221)은, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터 중 우채널의 오디오 데이터를, 좌채널의 오디오 데이터로서 카피하고, 그 좌채널과 우채널의 오디오 데이터(「부음성」의 오디오 데이터)를, 오디오 출력 단자(121)에 출력한다. 또한, 음성 출력 모드로서, 「주?부」가 지정되어 있을 때에는, 오디오 출력 모듈(221)은, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터를, 그대로, 오디오 출력 단자(121)에 출력한다.

[0950] 또한, 예를 들면, 출력 속성이 나타내는 오디오 데이터의 채널 할당의 지시가, 스테레오 모드를 나타내고 있는 경우, 오디오 출력 모듈(221)은, 플레이어 제어 모듈(212)로부터 공급되는 음성 출력 모드에 상관없이, 오디오 디코더 제어 모듈(217)로부터의 오디오 데이터를, 그대로, 오디오 출력 단자(121)에 출력한다.

[0951] 여기에서, 도 42 하측에서, 0xBD로 되어 있는 stream_id 및, 0x00으로 되어 있는 private_stream_id에 의해 특정되는 오디오 스트림 stream#2에 대하여 기술되어 있는 3세트의 pts_change_point 및 DynamicInfo()에 따르면, 오디오 스트림 stream#2의 재생 개시 시인 시각 90,000으로부터, 시각 27,090,000의 직전까지는, 오디오 스트림 stream#2로부터, 듀얼의 오디오 데이터가 얻어진다. 또한, 시각 27,090,000로부터, 시각 32,490,000의 직전까지는, 오디오 스트림 stream#2로부터, 스테레오의 오디오 데이터가 얻어지며, 시각 32,490,000 이후는, 오디오 스트림 stream#2로부터, 듀얼의 오디오 데이터가 얻어진다.

[0952] 따라서, 예를 들면, 음성 출력 모드로서, 「주음성」이 지정되어 있다고 하면, 오디오 출력 모듈(221)에서는, 시각 90,000으로부터, 시각 27,090,000의 직전까지는, 오디오 스트림 stream#2로부터 얻어지는 듀얼의 오디오 데이터 중 좌채널의 오디오 데이터가, 우채널의 오디오 데이터로서 카피되고, 그 좌채널과 우채널의 오디오 데이터가, 오디오 출력 단자(121)에 출력된다.

[0953] 또한, 시각 27,090,000으로부터, 시각 32,490,000의 직전까지는, 오디오 스트림 stream#2로부터 얻어지는 스테레오의 오디오 데이터가, 그대로, 오디오 출력 단자(121)에 출력된다.

[0954] 그리고, 시각 32,490,000 이후는, 오디오 스트림 stream#2로부터 얻어지는 듀얼의 오디오 데이터 중의 좌채널의 오디오 데이터가, 우채널의 오디오 데이터로서 카피되어, 그 좌채널과 우채널의 오디오 데이터가, 오디오 출력 단자(121)에 출력된다.

[0955] 이상과 같이, 출력 속성의 제어 처리에서는, 클립 스트림 파일에 다중화되어 있는 엘리멘터리 스트림마다에, 그 엘리멘터리 스트림의 재생 시각을 나타내는 pts_change_point와, 그 엘리멘터리 스트림의 출력 속성을 포함하는 DynamicInfo()와의 세트를 0세트 이상 포함하는 클립 정보 파일 Clip()(도 10)의 기술에 기초하여, 재생 중인 엘리멘터리 스트림의 재생 시각이, pts_change_point에 일치하는지의 여부가 판정된다. 그리고, 재생 중인 엘리멘터리 스트림의 재생 시각이, pts_change_point에 일치하는 경우, 그 ptschange_point와 세트로 되어 있는 DynamicInfo()가 인식되고, 그 인식된 DynamicInfo()에 포함되는 출력 속성에 따라서, 재생 중인 엘리멘터리 스트림의 출력이 제어된다. 따라서, 엘리멘터리 스트림의 재생 시각과 출력 속성에 따라서, 그 엘리멘터리 스트림의 출력을 제어하는 것이 가능하게 된다.

[0956] [자막 표시 제어 처리]

[0957] 다음으로, 도 43의 플로우차트를 참조하여, 자막 스트림에 대응하는 자막 데이터의 표시를 제어하는 자막 표시 제어 처리에 대하여 설명한다.

[0958] PlayList()(도 5)(의 PlayItem())의 재생이 개시되면, 플레이어 제어 모듈(212)은, 스텝 S341에서, 그래픽 처리 모듈(219)에 대한 자막 데이터의 표시 방식의 지시를 초기화한다. 즉, 플레이어 제어 모듈(212)은, 자막 데이터의 표시 방식을 디폴트의 표시 방식으로 하도록, 그래픽 처리 모듈(219)을 제어한다. 또한, 스텝 S341에서 행하여지는 표시 방식의 지시의 초기화는, 도 30의 스텝 S127에서 설명한 표시 방식의 지시의 초기화에 대응한다.

[0959] 스텝 S341의 처리 후는, 스텝 S342로 진행하여, 플레이어 제어 모듈(212)은, 유저가 리모콘을 조작함으로써 입력 인터페이스(115)로부터, 자막 데이터의 표시에 대해서, 새로운 표시 방식의 지시가 있었는지의 여부를

판정한다.

[0960] 스텝 S342에서, 새로운 표시 방식의 지시가 있었다고 판정된 경우, 스텝 S343으로 진행하여, 플레이어 제어 모듈(212)은, 자막 스트림(에 대응하는 자막 데이터)을, 지금 재생하고 있는지의 여부를 판정한다.

[0961] 스텝 S343에서, 자막 스트림이 재생되지 않았다고 판정된 경우, 스텝 S342로 되돌아간다.

[0962] 또한, 스텝 S343에서, 자막 스트림이 재생되고 있다고 판정된 경우, 스텝 S345로 진행하여, 플레이어 제어 모듈(212)은, 새로운 표시 방식의 지시가, 디폴트의 표시 방식의 지시인지의 여부를 판정한다. 스텝 S343에서, 새로운 표시 방식의 지시가, 디폴트의 표시 방식의 지시라고 판정된 경우, 스텝 S341로 되돌아가서, 전술한 바와 같이, 플레이어 제어 모듈(212)은, 자막 데이터의 표시 방식을 디폴트의 표시 방식으로 하도록, 그래픽 처리 모듈(219)을 제어한다.

[0963] 한편, 스텝 S345에서, 새로운 표시 방식의 지시가, 디폴트의 표시 방식의 지시가 아니라고 판정된 경우, 즉, 새로운 표시 방식의 지시가, 예를 들면, 자막 데이터를 확대나 축소하여 표시하거나, 혹은 밝기를 바꾸어서 보기 쉽게 하는 등, 디폴트가 아닌 표시 방식의 지시인 경우, 스텝 S346으로 진행하여, 플레이어 제어 모듈(212)은, 지금 재생하고 있는 자막 스트림이 다중화된 클립 스트림 파일에 대응하는 클립 정보 파일 Clip()(도 10)의 StaticInfo()(도 12) 중, 지금 재생하고 있는 자막 스트림에 대한 StaticInfo()를 취득하고, 스텝 S347로 진행한다.

[0964] 스텝 S347에서는, 플레이어 제어 모듈(212)은, 스텝 S346에서 취득한 StaticInfo()의 configurable_flag를 판정한다.

[0965] 스텝 S347에서, configurable_flag가, 자막 데이터의 표시 방식의 변경을 허가하지 않는다는 취지의 0으로 되어 있다고 판정된 경우, 스텝 S348로 진행하여, 플레이어 제어 모듈(212)은, 그래픽 처리 모듈(219)을 제어함으로써, 출력 비디오 데이터에, 자막 데이터의 표시 방식을 변경할 수 없다는 취지의 에러 메시지를 오버레이시키고, 스텝 S342로 되돌아간다. 이에 의해, 에러 메시지가 표시된다.

[0966] 한편, 스텝 S347에서, configurable_flag가, 자막 데이터의 표시 방식의 변경을 허가한다는 취지의 1로 되어 있다고 판정된 경우, 스텝 S349로 진행하여, 플레이어 제어 모듈(212)은, 유저가 리모콘을 조작함으로써 입력 인터페이스(115)로부터 공급된 새로운 표시 방식의 지시를, 그래픽 처리 모듈(219)에 공급하고, 스텝 S350로 진행한다.

[0967] 스텝 S350에서는, 그래픽 처리 모듈(219)은, 자막 디코더 제어 모듈(218)로부터 공급되는 자막 데이터를, 직전의 스텝 S349에서 플레이어 제어 모듈(212)로부터 공급된 표시 방식의 지시에 따라서 확대 또는 축소 등 혹은 밝기를 바꾸는 등의 처리를 개시하고, 스텝 S342로 되돌아간다. 이에 의해, 자막 데이터는, 유저가 리모콘을 조작함으로써 지시한 표시 방식에 따른 표시 사이즈나, 표시 위치, 표시색등에 의해 표시된다.

[0968] 한편, 스텝 S342에서, 새로운 표시 방식의 지시가 없었다고 판정된 경우, 스텝 S351로 진행하여, 플레이어 제어 모듈(212)은, 도 31에서 설명한 PlayItem()의 갈아타기가 행하여졌는지의 여부를 판정하고, 행하여지지 않았다고 판정한 경우, 스텝 S342로 되돌아간다.

[0969] 또한, 스텝 S351에서, PlayItem()의 갈아타기가 행하여졌다고 판정된 경우, 스텝 S341로 되돌아가고, 전술한 바와 같이, 플레이어 제어 모듈(212)은, 자막 데이터의 표시 방식을 디폴트의 표시 방식으로 하도록, 그래픽 처리 모듈(219)을 제어한다. 즉, 이 경우, PlayItem()의 갈아타기가 행하여졌을 때에는, 자막 데이터의 표시 방식은, 디폴트의 표시 방식으로 되돌아간다.

[0970] 이상과 같이, 자막 표시 제어 처리에서는, 자막 스트림의 configurable_flag가, 표시 방식의 변경을 허가한다는 취지의 1로 되어 있는 경우에만, 그 자막 스트림에 대응하는 자막 데이터의 표시 방식이, 예를 들면, 유저가 리모콘을 조작함으로써 입력되는 표시 방식의 지시에 따라서 변경된다.

[0971] 따라서, 예를 들면, 도 26A 및 도 26B에 도시한 클립 정보 파일 "00001.CLP"에 따르면, 대응하는 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림 중 3개째의 엘리멘터리 스트림인 자막 스트림 stream#2에 대한 configurable_flag는, 표시 방식의 변경을 허가하지 않는다는 취지의 0으로 되어 있기 때문에, 그 자막 스트림 stream#2가 표시되어 있을 때에, 유저가 자막의 표시를 변경하도록 리모콘을 조작하여도, 그 표시는 변경되지 않는다.

[0972] 한편, 예를 들면, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림 중 4개째의 엘리멘터리 스트림인 자막 스트림 stream#3에 대한 configurable_flag는, 표시 방식의 변경을 허가한다는 취지의

1로 되어 있기 때문에, 그 자막 스트림 stream#3이 표시되어 있을 때에, 유저가 자막의 표시를 변경하도록 리모콘을 조작하면, 그 조작에 따라서, 자막의 표시 사이즈 등이 변경된다.

[0973] 즉, 예를 들면, 지금, 도 25의 1번째의 PlayList#0의 1번째의 PlayItem#0에 따라, 클립 스트림 파일 "00001.PS" 가 재생되고 있다고 한다. 또한, 도 26A 및 도 26B에서 클립 정보 파일 "00001.CLpn에 대하여 설명한 바와 같이, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림 중, 3개째와 4개째가 자막 스트림인데, 그 3개째의 자막 스트림 stream#2와, 4개째의 자막 스트림 stream#3 중, 예를 들면, 3개째의 자막 스트림 stream#2가, 지금 재생되고 있다고 한다.

[0974] 유저가, 리모콘을 조작함으로써, 자막의 표시 방식의 지시를 입력하면(스텝 S342), 그 표시 방식의 지시는, 입력 인터페이스(115)(도 1)로부터 플레이어 제어 모듈(212)에 공급된다. 플레이어 제어 모듈(212)은, 표시 방식의 지시가 공급되면, 재생 중인 자막 스트림에 대응하는 StaticInfo()(도 10)를, 클립 정보 파일로부터 찾아낸다(스텝 S346).

[0975] 즉, 지금의 경우, 재생 중인 자막 스트림은, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 3개째의 자막 스트림 stream#2이며, 플레이어 제어 모듈(212)은, 대응하는 클립 정보 파일 "00001.CLP"으로부터, 3개째의 자막 스트림 stream#2에 대한 StaticInfo()를 찾아낸다.

[0976] 또한, 플레이어 제어 모듈(212)은, 도 26A 및 도 26B에서 3개째의 자막 스트림 stream#2에 대한 StaticInfo()에 기술되어 있는, 0으로 되어 있는 configurable_flag를 판정하고(스텝 S347), 이에 의해, 3개째의 자막 스트림 stream#2에 대해서는, 표시 방식의 변경이 허가되어 있지 않은 것을 인식한다.

[0977] 이 경우, 플레이어 제어 모듈(212)은, 재생 중인 자막 스트림(에 대응하는 자막 데이터)이 확대 축소 등에 대응하고 있지 않다고 판단하고, 그래픽 처리 모듈(219)을 제어함으로써, 그 취지의 에러 메시지를 생성시켜(스텝 S348), 비디오 데이터에 오버레이하여 출력시킨다.

[0978] 한편, 클립 스트림 파일 "00001.PS"에 다중화되어 있는 4개의 엘리멘터리 스트림의 3개째의 자막 스트림 stream#2와, 4개째의 자막 스트림 stream#3 중, 3개째의 자막 스트림 stream#2가 아닌, 4개째의 자막 스트림 stream#3이, 지금 재생되고 있는 경우에는, 유저가 리모콘을 조작함으로써 표시 방식의 지시의 공급을 받은 플레이어 제어 모듈(212)은, 대응하는 클립 정보 파일 "00001.CLP"로부터, 4개째의 자막 스트림 stream#3에 대한 StaticInfo()를 찾아낸다.

[0979] 또한, 플레이어 제어 모듈(212)은, 도 26A 및 도 26B에서 4개째의 자막 스트림 stream#3에 대한 StaticInfo()에 기술되어 있는, 1로 되어 있는 configurable_flag를 판정하고(스텝 S347), 이에 의해, 4개째의 자막 스트림 stream#3에 대해서는, 표시 방식의 변경이 허가되어 있는 것을 인식한다.

[0980] 이 경우, 플레이어 제어 모듈(212)은, 재생 중인 자막 스트림(에 대응하는 자막 데이터)이 확대 축소 등에 대응하고 있다고 판단하고, 유저가 리모콘을 조작 함으로써 공급된 표시 방식의 지시를, 그래픽 처리 모듈(219)에 공급한다(스텝 S349).

[0981] 이에 의해, 그 후, 그래픽 처리 제어 모듈(219)은, 플레이어 제어 모듈(212)로부터의 표시 방식의 지시에 따라, 자막 디코더 제어 모듈(218)로부터의 자막 데이터를 확대 또는 축소 등으로 하여, 비디오 디코더 제어 모듈(216)로부터의 비디오 데이터에 오버레이하여 출력한다.

[0982] 또한, 플레이어 제어 모듈(212)은, PlayList()의 최초의 PlayItem()의 재생 개시 시에, 그래픽 처리 모듈(219)에 대한 자막 데이터의 표시 방식의 지시를 초기화한다(스텝 S341). 즉, 플레이어 제어 모듈(212)은, 자막 데이터의 표시 방식을 디폴트의 표시 방식으로 하도록, 그래픽 처리 모듈(219)을 제어한다.

[0983] 또한, 플레이어 제어 모듈(212)은, PlayItem()의 갈아타기 시에도, 그래픽 처리 모듈(219)에 대한 자막 데이터의 표시 방식의 지시를 초기화한다(스텝 S341, 5351).

[0984] 단, PlayItem()의 갈아타기 시에서는, 그 후에 새롭게 재생되는 PlayItem()에 따라서 재생되는 새로운 자막 스트림에 대한 configurable_flag를 조사하고, configurable_flag가 0인 경우에는, 그래픽 처리 모듈(219)에 대한 자막 데이터의 표시 방식의 지시를 초기화하고, configurable_flag가 1인 경우에는, 그래픽 처리 모듈(219)에 대한 표시 방식의 지시를, PlayItem()의 갈아타기 전인 채로 유지하도록 하는 것이 가능하다.

[0985] 또한, 도 43의 자막 표시 제어 처리에서는, 유저가 리모콘을 조작함으로써, 새로운 표시 방식의 지시가 입력된 경우에, 그 새로운 표시 방식의 지시를, 그래픽 처리 모듈(219)에 공급하도록 했지만(스텝 S349), 표시 방식의 지시는, 예를 들면, 메모리(113)(도 1)를 구성하는 불휘발성 메모리에 기억하고, 그 불휘발성 메모리에

기억된 표시 방식의 지시를, 그래픽 처리 모듈(219)에 공급하도록 하는 것이 가능하다.

[0986] 즉, 예를 들면, 도 1의 디스크 장치의 초기 설정으로서, 불휘발성 메모리에, 유저 설정의 표시 방식의 지시를 기억시켜 두고, 유저가 리모콘을 조작함으로써, 새로운 표시 방식의 지시가 입력된 경우에는, 불휘발성 메모리에 기억된 표시 방식의 지시를, 새로운 표시 방식의 지시에 갱신하는 한편, 그 불휘발성 메모리에 기억된 표시 방식의 지시를, 그래픽 처리 모듈(219)에 공급하도록 하는 것이 가능하다. 이 경우, 불휘발성 메모리에는, 전회의 재생 종료 시에서의 표시 방식의 지시가 유지되므로, 다음 회의 PlayList()의 재생 시에, 유저가 리모콘을 조작함으로써, 전회의 재생 종료 시에서의 표시 방식의 지시를 입력하지 않아도, 그 표시 방식으로, 자막 데이터의 표시가 개시된다.

[0987] 또한, 이 경우, 불휘발성 메모리에 기억시키는 표시 방식의 지시에는, 예를 들면, 자막 데이터를 확대 또는 축소할 때의 확대율 또는 축소율 등이 포함되는 것으로 한다.

[0988] 이상과 같이, 자막 표시 제어 처리에 따르면, 클립 정보 파일 Clip()(도 10)에 포함되는, 엘리멘터리 스트림마다의, 그 엘리멘터리 스트림의 재생 중에 변화하지 않는 StaticInfo() 중, 자막 데이터의 StaticInfo()가 취득되고, 그 StaticInfo()에 포함되는, 자막 데이터의 표시를 디폴트의 표시 방식으로부터 변경하는 것을 허가하는지의 여부를 나타내는 configurable_flag에 기초하여, 재생 중인 자막 데이터의 표시를 디폴트의 표시 방식으로부터 변경하는 것이 허가되어 있는지의 여부가 판정된다. 그리고, 재생 중인 자막 데이터의 표시를 디폴트의 표시 방식으로부터 변경하는 것이 허가되어 있는 경우에는, 자막 데이터의 표시 방식의 변경의 지시에 따라서, 그 자막 데이터의 표시 처리, 즉, 예를 들면, 자막 데이터를 확대 또는 축소, 혹은 표시색을 변경하는 등으로 하여 표시하는 처리가 행하여진다. 따라서, 자막 데이터의 표시 방식의 변경을 제어할 수 있다.

[0989] [캡쳐 제어 처리]

[0990] 다음으로, 도 44의 플로우차트를 참조하여, 비디오 스트림에 대응하는 비디오 데이터의 캡쳐를 제어하는 캡쳐 제어 처리에 대하여 설명한다. 또한, 도 44에는, 캡쳐 제어 처리를 설명하는 플로우차트와 함께, 그 캡쳐 제어 처리에 의해 캡쳐 된 비디오 데이터를 2차 이용하는 처리의 예인 백그라운드/스크린 세이버 처리를 설명하는 플로우차트도, 도시하고 있다.

[0991] 캡쳐 제어 처리는, 예를 들면, 유저가 리모콘을 조작함으로써, 비디오 데이터의 캡쳐를 지시하는 캡쳐 지시가, 입력 인터페이스(115)(도 1)를 통하여, 플레이어 제어 모듈(212)에 공급되면 개시된다.

[0992] 즉, 캡쳐 제어 처리에서는, 우선 최초로, 스텝 S371에서, 플레이어 제어 모듈(212)이, 비디오 스트림을 재생 중 인지의 여부를 판정하고, 재생 중이 아니라고 판정한 경우, 캡쳐 제어 처리는 종료한다.

[0993] 한편, 스텝 S371에서, 비디오 스트림을 재생 중이라고 판정된 경우, 스텝 S372로 진행하여, 플레이어 제어 모듈(212)은, 재생 중인 비디오 스트림에 대응하는 PlayList()(도 5)로부터, capture_enable_flag_PlayList를 취득함과 함께, 재생 중인 비디오 스트림에 대응하는 클립 정보 파일 Clip()(도 10)으로부터, capture_enable_flag_Clip을 취득한다.

[0994] 여기에서, PlayList()에서의 capture_enable_flag_PlayList는, 도 5에서 설명한 바와 같이, 그 PlayList()에 의해 재생되는 비디오 스트림에 대응하는 비디오 데이터(PlayList()에 속하는 비디오 데이터)의 2차 이용을 허가하는지의 여부를 나타낸다. 또한, 클립 정보 파일 Clip()에서의 capture_enable_flag_Clip은, 도 10에서 설명한 바와 같이, 그 클립 정보 파일 Clip()에 대응하는 클립 스트림 파일에 저장되어 있는 비디오 스트림에 대응하는 비디오 데이터의 2차 이용을 허가하는지의 여부를 나타낸다.

[0995] 스텝 S372의 처리 후는, 스텝 S373으로 진행하여, 플레이어 제어 모듈(212)은, 직전의 스텝 S372에서 취득된 capture_enable_flag_PlayList와 capture_enable_flag_Clip에 기초하여, 캡쳐 지시가 입력 인터페이스(115)(도 1)로부터 입력되었을 때에 재생되고 있던 비디오 데이터의 픽처의 캡쳐의 가부를 판정한다.

[0996] 스텝 S373에서, 캡쳐 지시가 입력 인터페이스(115)로부터 입력되었을 때에 재생되고 있던 비디오 데이터의 픽처의 캡쳐가 불가능하다고 판정된 경우, 즉, 직전의 스텝 S373에서 취득된 capture_enable_flag_PlayList 또는 capture_enable_flag_Clip 중 적어도 한쪽이, 비디오 데이터의 2차 이용을 허가하지 않는다는 취지의 0으로 되어 있는 경우, 스텝 S374로 진행하여, 플레이어 제어 모듈(212)은, 그래픽 처리 모듈(219)을 제어함으로써, 비디오 데이터의 캡쳐가 불가능하다는 취지의 에러 메시지를 오버레이시키고, 캡쳐 제어 처리를 종료한다. 이에 의해, 에러 메시지가 표시된다.

[0997] 한편, 스텝 S373에서, 캡쳐 지시가 입력 인터페이스(115)로부터 입력되었을 때에 재생되고 있던 비디오 데이터의 픽쳐의 캡쳐가 가능하다고 판정된 경우, 즉, 직전의 스텝 S373에서 취득된 capture_enable_flag_PlayList 및 capture_enable_flag_Clip의 양쪽이, 비디오 데이터의 2차 이용을 허가한다는 취지의 1로 되어 있는 경우, 스텝 S375로 진행하여, 플레이어 제어 모듈(212)은, 캡쳐지시가 입력 인터페이스(115)로부터 입력되었을 때에 재생되고 있던 비디오 데이터의 픽쳐의 캡쳐의 지시를, 그래픽 처리 모듈(219)에 공급하고, 스텝 S376으로 진행한다.

[0998] 스텝 S376에서는, 그래픽 처리 모듈(219)은, 플레이어 제어 모듈(212)로부터의 캡쳐의 지시에 따라, 비디오 디코더 제어 모듈(216)로부터의 비디오 데이터의 픽쳐를 캡쳐하고, 메모리(113)(도 1)에 기억시켜서, 캡쳐 제어 처리를 종료한다. 또한, capture_enable_flag가 복수 비트 구성으로 되어 있으며, 사용 조건의 제약이 행하여지고 있는 경우에는 이 시점에서 대응이 행하여진다. 즉 캡쳐한 화상의 크기에 제한이 있는 경우에는, 이 시점에서 축소한 화상이 캡쳐된다. 또한 사용하는 어플리케이션에 제약이 있는 경우에는 그 취지를 알리는 플래그가 동시에 기록된다.

[0999] 이상과 같이, 캡쳐 제어 처리에서는, 유저로부터의 캡쳐 지시가 있었을 때에 재생되고 있는 비디오 스트림에 대응하는 PlayList()(도 5)와 클립 정보 파일 Clip()(도 10) 각각의 capture_enable_flag_PlayList와 capture_enable_flag_Clip과의 논리곱을 취하고, 그 논리곱이 1인 경우, 즉, capture_enable_flag_PlayList와 capture_enable_flag_Clip이, 모두, 2차 이용을 허가하는 1로 되어 있는 경우에만, 비디오 데이터의 2차 이용이 가능하다고 판단되어, 캡쳐가 행해진다.

[1000] 따라서, 예를 들면, 도 25에서의 1번째의 PlayList#0의 1번째의 PlayItem#0에 따라서, 비디오 스트림의 재생, 즉, 클립 스트림 파일 "00001.PS"에 다중화된 비디오 스트림의 재생이 행하여지고 있는 경우에, 유저로부터의 캡쳐 지시가 있었을 때에는, 1번째의 PlayList#0에서의 capture_enable_flag_PlayList는 1이며, 그 1번째의 PlayItem#0에 의해 재생되는 클립 스트림 파일 "00001.PS"에 대응하는 도 26A 및 도 26B의 클립 정보 파일 "00001.CLP"에서의 capture_enable_flag_Clip은 1이기 때문에, 재생 중인 비디오 데이터(클립 스트림 파일 "00001.PS"에 다중화된 비디오 스트림에 대응하는 비디오 데이터)의 2차 이용은 가능하다고 판단되어, 캡쳐가 행해진다.

[1001] 또한, 예를 들면, 도 25에서의 1번째의 PlayList#0의 2번째의 PlayItem#1에 따라서, 비디오 스트림의 재생, 즉, 클립 스트림 파일 "00002.PS"에 다중화된 비디오 스트림의 재생이 행하여지고 있는 경우에, 유저로부터의 캡쳐 지시가 있었을 때에는, 1번째의 PlayList#0에서의 capture_enable_flag_PlayList는 1이며, 그 2번째의 PlayItem#1에 의해 재생되는 클립 스트림 파일 "00002.PS"에 대응하는 도 26A 및 도 26B의 클립 정보 파일 "00002.CLP"에서의 capture_enable_flag_Clip은 0이기 때문에, 재생 중인 비디오 데이터(클립 스트림 파일 "00002.PS"에 다중화된 비디오 스트림에 대응하는 비디오 데이터)의 2차 이용은 불가능하다고 판단되어, 캡쳐가 행하여지지 않는다.

[1002] 또한, 예를 들면, 도 25에서의 2번째의 PlayList#1의 PlayItem#0에 따라서, 비디오 스트림의 재생, 즉, 클립 스트림 파일 "00003.PS"에 다중화된 비디오 스트림의 재생이 행하여지고 있는 경우에, 유저로부터의 캡쳐 지시가 있었을 때에는, 2번째의 PiayList#1에서의 capture_enable_flag_PlayList는 0이며, 2번째의 PlayList#1의 PlayItem#0에 의해 재생되는 클립 스트림 파일 "00003.PS"에 대응하는 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"에서의 capture_enable_flag_Clip은 1이기 때문에, 재생 중인 비디오 데이터(클립 스트림 파일 "00003.PS"에 다중화된 비디오 스트림에 대응하는 비디오 데이터)의 2차 이용은 불가능하다고 판단되어, 캡쳐는 행하여지지 않는다.

[1003] 또한, 이 경우, 2번째의 PIayList#1에서의 capture_enable_flag_PlayList가 0인 것이 확인된 시점에서, 비디오 데이터의 2차 이용은 불가능하다고 판단할 수 있기 때문에, 2번째의 PlayList#1의 PlayItem#0에 의해 재생되는 클립 스트림 파일 "00003.PS"에 대응하는 도 26A 및 도 26B의 클립 정보 파일 "00003.CLP"에서의 capture_enable_flag_Clip의 확인은 생략할 수 있다.

[1004] 캡쳐 제어 처리에 의해 캡쳐되어, 메모리(113)에 기억된 픽쳐는, 백그라운드/스크린 세이버 처리에서 2차 이용할 수 있다.

[1005] 백그라운드/스크린 세이버 처리는, 예를 들면, 플레이어 제어 모듈(212)이 동작하고 있지만, 엘리멘터리 스트림의 재생이 행하여지지고 있지 않은 상태, 즉, 디스크 드라이브(102)(도 1)에 디스크(101)가 삽입되어 있지 않은 상태, 혹은 엘리멘터리 스트림의 재생이 종료한 상태로 되었을 때 등에 행해진다.

[1006] 즉, 백그라운드/스크린 세이버 처리에서는, 스텝 S381에서, 플레이어 제어 모듈(212)은, 캡쳐 제어 처리에 의해 메모리(113)에 기억된 픽쳐를 표시하도록, 그래픽 처리 모듈(219)을 제어한다. 그래픽 처리 모듈(219)은, 플레이어 제어 모듈(212)로부터의 제어에 따라, 캡쳐 제어 처리에 의해 메모리(113)에 기억된 픽쳐를 표시시킨다.

[1007] 여기에서, 그래픽 처리 모듈(219)에서, 메모리(113)에 기억된 픽쳐를 정지 화상에서 표시시키면, 소위 벽지(백그라운드)가 실현되어, 일정 주기로 확대나 축소, 이동 등을 하면서 표시시키면, 스크린 세이버가 실현된다. 또한, 캡쳐 제어 처리에 의해 메모리(113)에 기억된 픽쳐의 표시를 행하는 백그라운드/스크린 세이버 처리는, 플레이어 제어 모듈(212)이 아닌, 다른 독립된 어플리케이션에 의해 행하는 것이 가능하다.

[1008] 또한, 이 때 메모리(113)에 기억된 픽쳐에 사용 제한을 나타내는 플래그가 부가되어 있는 경우에는 그 제한을 따른다.

[1009] 이상과 같이, 비디오 액세스 유닛 단위보다 큰 단위의, 예를 들면, PlayList()나 PlayItem()에 대응하는 비디오 데이터의 2차 이용을 허가하는지의 여부를 나타내는, 재생 중인 비디오 데이터에 대한 capture_enable_flag_PlayList나 capture_enable_flag_Clip이 취득되고, 그 capture_enable_flag_PlayList나 capture_enable_flag_Clip에 기초하여, 재생 중인 비디오 데이터의 2차 이용이 허가 되어 있는지의 여부가 판정된다. 그리고, 재생 중인 비디오 데이터의 2차 이용이 허가되어 있다고 판정된 경우, 재생 중인 비디오 데이터가 캡쳐되고, 그 캡쳐 된 비디오 데이터를 이용한 백그라운드/스크린 세이버 처리가 실행된다. 따라서, 비디오 데이터의 2차 이용의 제어가 가능해진다.

[1010] 또한, 도 44의 캡쳐 제어 처리에서는, PlayList()(도 5)에서, capture_enable_flag_PlayList를 설정함과 함께, PlayItem()에 의해 재생되는 클립 스트림 파일에 대응하는 클립 정보 파일 Clip()(도 10)에서, capture_enable_flag_Clip을 설정하고, 그 capture_enable_flag_PlayList와 capture_enable_flag_Clip의 양방을 이용하여, 2차 이용의 허가(가부)를 판정하도록 했지만, PlayList()(도 5)에서, capture_enable_flag_PlayList를 설정하는 것만으로, 또는 PlayItem()에 의해 재생되는 클립 스트림 파일에 대응하는 클립 정보 파일 Clip()(도 10)에서, capture_enable_flag_Clip을 설정하는 것만으로 하여, capture_enable_flag_PlayList 또는 capture_enable_flag_Clip의 한쪽만을 이용하여, 2차 이용의 가부를 판정하도록 하는 것도 가능하다.

[1011] 또한, 도 44의 캡쳐 제어 처리에서는, 스텝 S376에서, 그래픽 처리 모듈(219)이, 플레이어 제어 모듈(212)로부터의 캡쳐의 지시에 따라, 비디오 디코더 제어 모듈(216)로부터의 비디오 데이터의 픽쳐, 즉, 1개의 픽쳐만을 캡쳐하도록 했지만, 그 외에, 복수의 픽쳐를 캡쳐하는 것도 가능하다. 즉, 비디오 디코더 제어 모듈(216)이 출력하는 시계열의 복수의 픽쳐(동화상으로서의 복수의 픽쳐의 시퀀스)를 캡쳐하는 것이 가능하다. 이 경우, 한번에 캡쳐되는 픽쳐의 매수는, 예를 들면, 미리 결정해 둘 수 있다. 혹은, capture_enable_flag_PlayList나 capture_enable_flag_Clip의 비트를 확장하고, 그 capture_enable_flag_PlayList나 capture_enable_flag_Clip에, 한번에 캡쳐 가능한 픽쳐의 매수의 정보를 포함시키도록 하여도 된다.

[1012] 또한, 전술한 경우에는, 비디오 데이터의 2차 이용을 허가하는지의 여부의 이용 허가 정보(capture_enable_flag_PlayList, capture_enable_flag_Clip)를, PlayList()나, 클립 정보 파일 Clip()에 기술하여, 그 이용 허가 정보에 의해, PlayList()에 의해 재생되는 비디오 데이터 전체나, 클립 정보 파일 Clip()에 대응하는 클립 스트림 파일에 다중화된 비디오 스트림에 대응하는 비디오 데이터 전체에 대한 2차 이용의 가부를 판정하도록 했지만, 이용 허가 정보는, 그 밖의 임의의 단위의 비디오 데이터에 대하여 기술하고, 그 이용 허가 정보에 의해, 임의의 단위의 비데 데이터에 대한 2차 이용의 가부를 판정하는 것이 가능하다.

[1013] 즉, 도 45는, 이용 허가 정보가 배치된 private_stream2_PES_payload()의 신택스를 나타내고 있고, 도 46은, 이용 허가 정보가 배치된 au_information()의 신택스를 나타내고 있다.

[1014] 또한, 도 45의 private_stream2_PES_payload()는, video_stream_id의 직전에, 이용 허가 정보로서의 capture_enable_flag_ps2가 배치되어 있는 것 외에는, 도 23에서의 경우와 마찬가지로 구성되어 있다. 도 46의 au_information()도, pic_struct_copy의 직전에, 이용 허가 정보로서의 capture_enable_flag_AU가 배치되어 있는 것 외에는, 도 24에서의 경우와 마찬가지로 구성되어 있다.

[1015] 도 45의 private_stream2_PES_payload()에 배치된 capture_enable_flag_ps2는, 그 private_stream2_PES_payload()를 포함하는 private_stream_2의 PES_packet()으로부터, 다음의

private_stream_2의 PES_packet()의 직전까지 배치되는 비디오 스트림에 대응하는 비디오 데이터의 2차 이용을 허가하는지의 여부를 나타낸다. 따라서, 도 45의 private_stream2_PES_payload()에 배치된 capture_enable_flag_ps2에 따르면, 임의의 디코드 개시 가능점으로부터 다음의 디코드 개시 가능점까지의 사이의 비디오 데이터에 대해서, 그 2차 이용을 허가하는지의 여부를 판정할 수 있다.

[1016] 또한, 도 46의 au_information()에 배치된 capture_enable_flag_AU는, 그 capture_enable_flag_AU에 대응하는 비디오 액세스 유닛의 비디오 데이터의 2차 이용을 허가하는지의 여부를 나타낸다. 따라서, 도 46의 au_information()에 배치된 capture_enable_flag_AU에 따르면, 비디오 액세스 유닛 단위의 비디오 데이터에 대해서, 즉 픽쳐 단위로, 그 2차 이용을 허가하는지의 여부를 판정할 수 있다.

[1017] 여기에서, PlayList()(도 5)에서의 이용 허가 정보로서의 capture_enable_flag_PlayList, 클립 정보 파일 Clip()(도 10)에서의 이용 허가 정보로서의 capture_enable_flag_Clip, private_stream2_PES_payload()(도 45)에서의 이용 허가 정보로서의 capture_enable_flag_ps2, au_information()(도 46)에서의 이용 허가 정보로서의 capture_enable_flag_AU는, 그 중의 2 이상을 중복하여 채용하는 것이 가능하며, 이 경우, 임의의 비디오 데이터의 픽쳐의 2차 이용의 가부는, 그 중복하여 채용되는 2 이상의 이용 허가 정보의 논리곱 등에 기초하여 판정할 수 있다.

[1018] 또한, 도 46의 au_information()이 배치되는 도 23 또는 도 45의 private_stream2_PES_payload()를 포함하는 private_stream_2의 PES_packet()은, 도 36의 스텝 S211에서 설명한 바와 같이, 버퍼 제어 모듈(215)(도 3)의 비디오 읽어내기 기능부(233)가, 버퍼(215A)에 기억된 프로그램 스트림 내에서 탐색한다. 따라서, capture_enable_flag_ps2가 배치된 도 45의 private_stream2_PES_payload()나, capture_enable_flag_AU가 배치된 도 46의 au_information()을 채용하는 경우에는, 플레이어 제어 모듈(212)은, 비디오 데이터의 2차 이용의 가부를 판정하는 것에 있어서, capture_enable_flag_ps2나 capture_enable_flag_AU를, 비디오 읽어내기 기능부(233)에 조회할 필요가 있다.

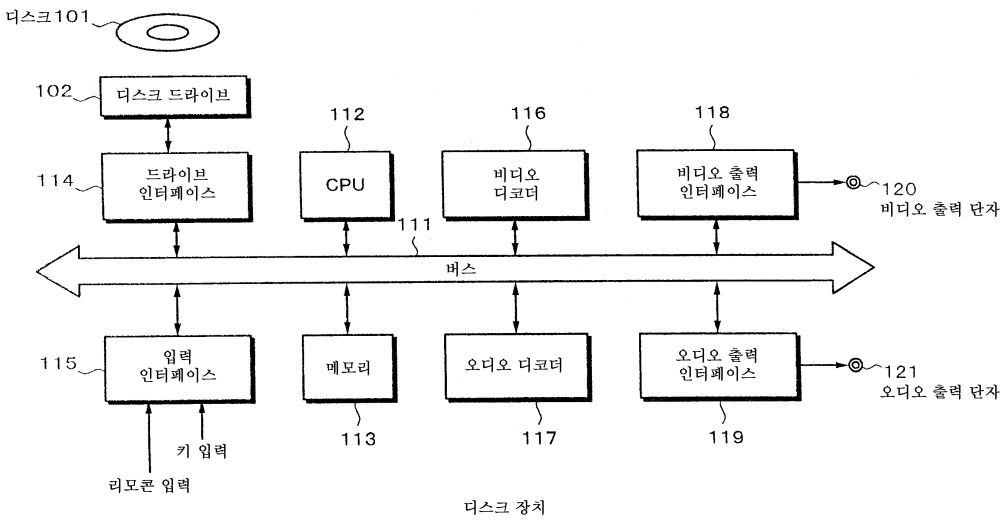
[1019] 또한, 본 실시 형태에서는, 전술한 일련의 처리를, 소프트웨어에 의해 행하는 것이라고 했지만, 전술한 일련의 처리는, 전용의 하드웨어에 의해 행할 수도 있다.

[1020] 또한, 본 실시 형태에서는, 비디오 디코더(116)(도 1)로서, 하드웨어 디코더를 채용하는 것이라고 했지만, 비디오 디코더(116)로서는, 소프트웨어 디코더를 채용하는 것도 가능하다. 오디오 디코더(117)(도 1)에 대해서도, 마찬가지이다.

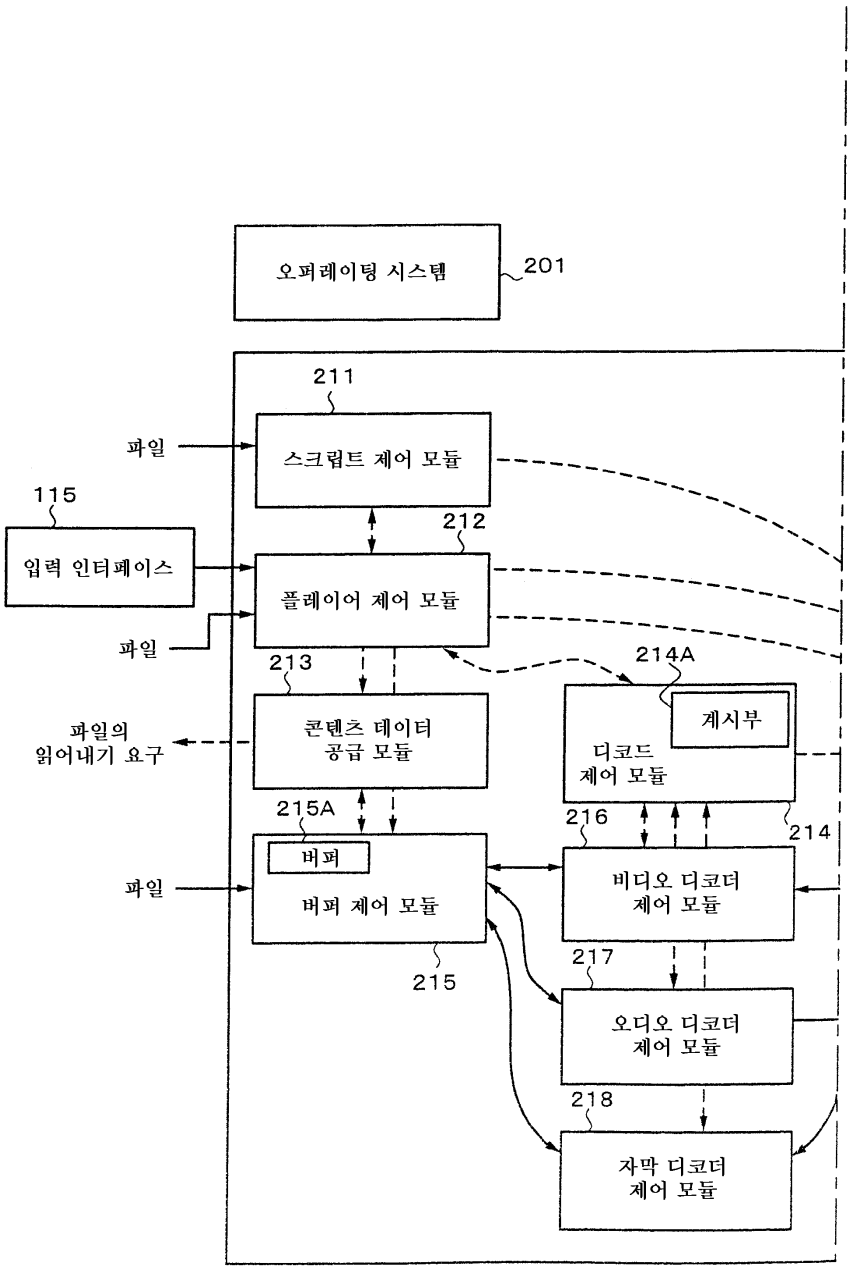
[1021] 또한, 본 실시 형태에서는, 자막 디코더로서, 소프트웨어 디코더를 채용하는 것이라고 했지만, 자막 디코더로서는, 하드웨어 디코더를 채용하는 것도 가능하다.

도면

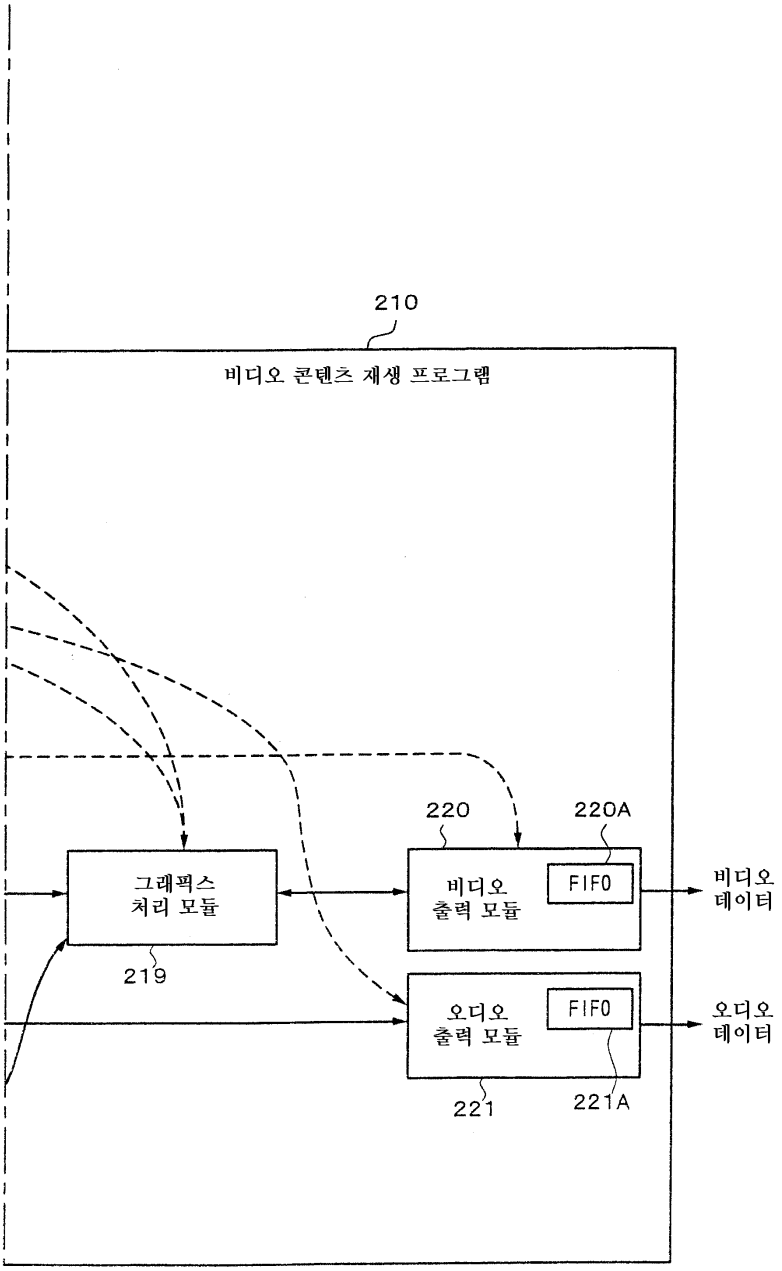
도면1



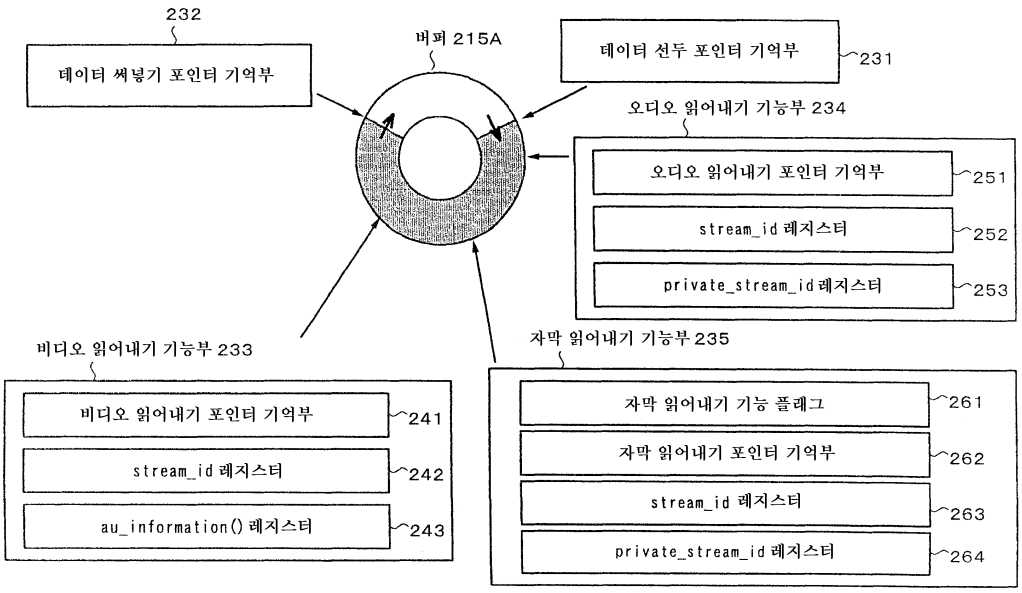
도면2A


오퍼레이팅 시스템
201
211
스크립트 제어 모듈
파일
115
입력 인터페이스
212
플레이어 제어 모듈
파일
213
214A
콘텐츠 데이터 공급 모듈
계시부
디코드 제어 모듈
파일의 읽어내기 요구
215A
216
214
버퍼
버퍼 제어 모듈
파일
비디오 디코더 제어 모듈
215
217
오디오 디코더 제어 모듈
218
자막 디코더 제어 모듈

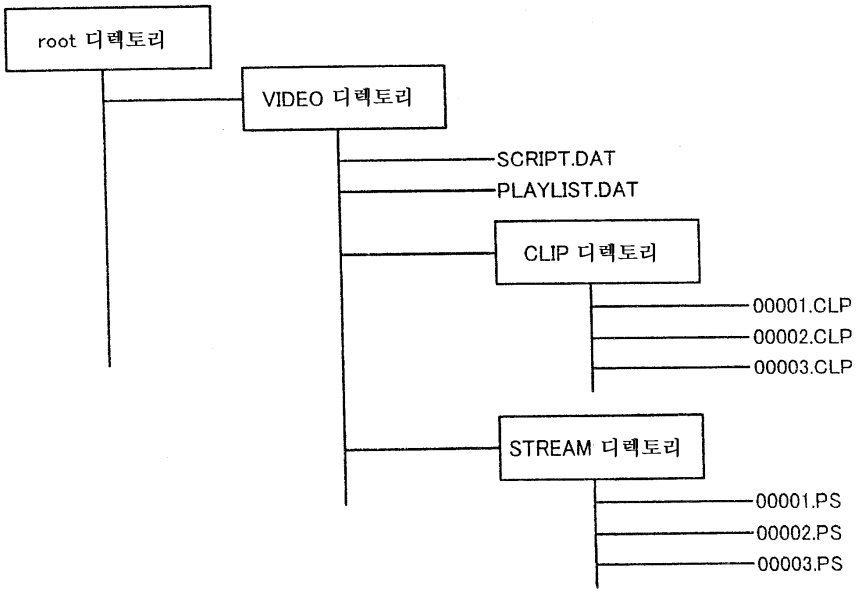
도면2B


210
비디오 콘텐츠 재생 프로그램
220
220A
그래픽스
처리 모듈
219
비디오
출력 모듈
FIFO
비디오
데이터
오디오
출력 모듈
FIFO
오디오
데이터
221
221A

도면3


232
데이터 써넣기 포인터 기억부
버퍼 215A
데이터 선두 포인터 기억부
231
오디오 읽어내기 기능부 234
오디오 읽어내기 포인터 기억부
251
stream_id 레지스터
252
private_stream_id 레지스터
253
비디오 읽어내기 기능부 233
비디오 읽어내기 포인터 기억부
241
stream_id 레지스터
242
au_information() 레지스터
243
자막 읽어내기 기능부 235
자막 읽어내기 기능 플래그
261
자막 읽어내기 포인터 기억부
262
stream_id 레지스터
263
private_stream_id 레지스터
264

도면4


root 디렉토리
VIDEO 디렉토리
SCRIPT.DAT
PLAYLIST.DAT
CLIP 디렉토리
00001.CLP
00002.CLP
00003.CLP
STREAM 디렉토리
00001.PS
00002.PS
00003.PS

도면5

신택스	비트수	니모닉
"PLAYLIST.DAT"{		
name_length	8	uimsbf
name_string	8x255	bslbf
number_of_PlayLists	16	uimsbf
for(i=0;i<number_of_PlayLists;i++){		
PlayList(){ // A PlayList()		
PlayList_data_length	32	uimsbf
//속성 정보		
reserved_for_word_alignment	15	bslbf
capture_enable_flag_PlayList	1	bslbf
PlayList_name_length	8	uimsbf
PlayList_name_string	8*255	bslbf
//		
number_of_PlayItems	16	uimsbf
for(i=0;i<number_of_PlayItems;i++){		
PlayItem()		
}		
PlayListMark()		
}		
}		
}		

도면6

신택스	비트수	니모닉
PlayItem(){		
length	16	uimsbf
Clip_Information_file_name_length	16	uimsbf
Clip_Information_file_name	8*Clip_Information_file_name_length	bslbf uimsbf uimsbf
IN_time	32	
OUT_time	32	
}		

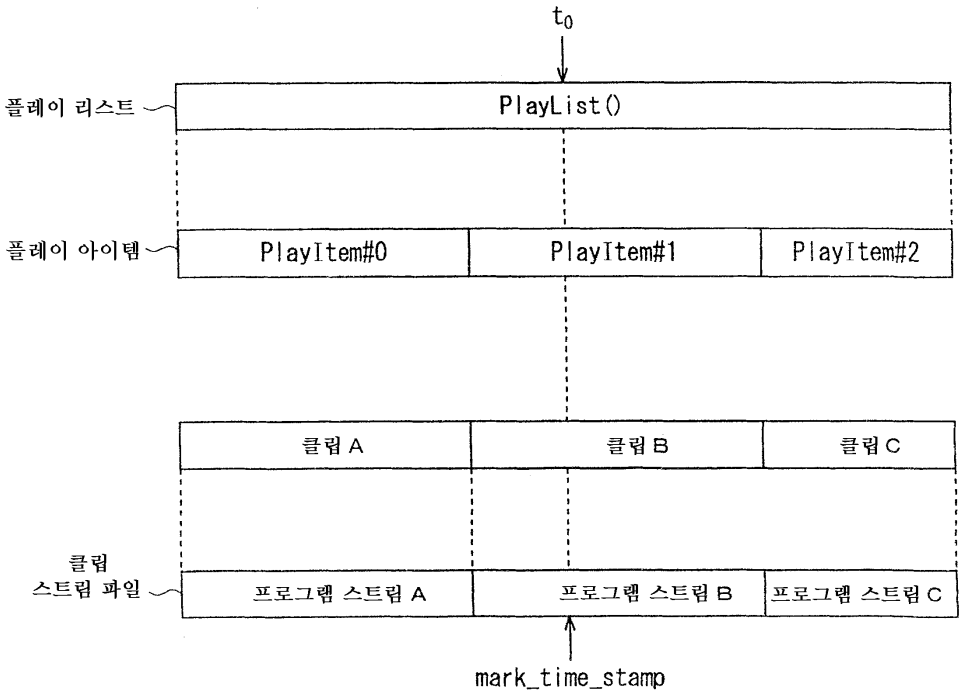
도면7

신택스	비트수	니모닉
PlayListMark(){		
length	32	uimsbf
number_of_PlayList_marks	16	uimsbf
for(i=0;i < number_of_PlayList_marks;i++){		
Mark(){		
mark_type	8	uimsbf
mark_name_length	8	uimsbf
ref_to_PlayItem_id	16	uimsbf
mark_time_stamp	32	uimsbf
entry_ES_stream_id	8	uimsbf
entry_ES_private_stream_id	8	uimsbf
mark_data	32	bslbf
mark_name_string	8*24	bslbf
}		
}		
}		

도면8

mark_type	스트림 코딩
0	예약
1	"Chapter" 마크
2	"Index" 마크
3	"Event" 마크
4~255	예약

도면9



도면10

신택스	비트수	니모닉
XXXXX.CLP{		
presentation_start_time	32	uimsbf
presentation_end_time	32	uimsbf
reserved_for_word_alignment	7	bslbf
capture_enable_flag_Clip	1	bslbf
number_of_streams	8	uimsbf
for(i=0;i < number_of_streams;i++){		
StreamInfo(){		
length	16	uimsbf
stream_id	8	uimsbf
private_stream_id	8	uimsbf
StaticInfo()		
reserved_for_word_alignment	8	bslbf
number_of_DynamicInfo	8	uimsbf
for(j=0;j < number_of_DynamicInfo;j++){		
pts_change_point	32	uimsbf
DynamicInfo()		
}		
}		
}		
EP_map()		
}		

도면11

엘리멘터리 스트림 종류	stream_id	private_stream_id
비디오	0xE0～0xEF	(없음)
ATRAC 오디오	0xBD	0x00～0x0F
LPCM 오디오	0xBD	0x10～0x1F
자막	0xBD	0x80～0x9F

도면12

신택스	비트수	니모닉
StaticInfo() {		
if(stream == VIDEO){		
reserved_for_word_alignment	16	bslbf
picture_size	4	uimsbf
frame_rate	4	uimsbf
reserved_for_word_alignment	7	bslbf
cc_flag	1	bslbf
}else if (stream == AUDIO) {		
audio_language_code	16	bslbf
channel_configuration	8	uimsbf
reserved_for_word_alignment	3	bslbf
lfe_existence	1	bslbf
sampling_frequency	4	uimsbf
}else if (stream == SUBTITLE) {		
subtitle_language_code	16	bslbf
reserved_for_word_alignment	15	bslbf
configurable_flag	1	uimsbf
}		
}		

도면13

신택스	비트수	니모닉
DynamicInfo(i,j){		
reserved_for_word_alignment	8	bslbf
if(stream == VIDEO){		
reserved_for_word_alignment	4	bslbf
display_aspect_ratio	4	uimsbf
}else if(stream == AUDIO){		
reserved_for_word_alignment	4	bslbf
channel_assignment	4	uimsbf
}else if(stream == SUBTITLE){		
reserved_for_word_alignment	8	bslbf
}		
}		

도면14

신택스	비트수	니모닉
EP_map() {		
reserved_for_word_alignment	8	bslbf
number_of_stream_id_entries	8	uimsbf
for (k=0;k<number_of_stream_id_entries;k++) {		
stream_id	8	bslbf
private_stream_id	8	bslbf
number_of_EP_entries	32	uimsbf
for (i=0;i< number_of_EP_entries;i++) {		
PTS_EP_start	32	uimsbf
RPN_EP_start	32	uimsbf
}		
}		
}		

도면15A

테이블 2-31, 프로그램 스트림		
신택스	비트수	니모닉

MPEG2_program_stream() {
do {
pack()
}while(nextbits()== pack_start_code)
MPEG_program_end_code
}
	32	bslbf

테이블 2-32, 프로그램 스트림 팩		
신택스	비트수	니모닉

pack() {
pack_header()
while(nextbits()==packet_start_code_prefix) {
PES_packet()
}
}		

도면15B

테이블 33, 프로그램 스트림 팩 헤더

신택스	비트수	니모닉
pack_header() {		
pack_start_code	32	bslbf
'01'	2	bslbf
system_clock_reference_base[32..30]	3	bslbf
marker_bit	1	bslbf
system_clock_reference_base[29..15]	15	bslbf
marker_bit	1	bslbf
system_clock_reference_base[14..0]	15	bslbf
marker_bit	1	bslbf
system_clock_reference_extension	9	uimsbf
marker_bit	1	bslbf
program_mux_rate	22	uimsbf
marker_bit	1	bslbf
marker_bit	1	bslbf
reserved	5	bslbf
pack_stuffing_length	3	uimsbf
for(i=0;i<pack_stuffing_length;i++) {		
stuffing_byte	8	bslbf
}		
if(nextbits()==system_header_start_code) {		
system_header()		
}		
}		

도면16A

테이블 2-17, PES 패킷

신택스	비트수	니모닉
PES_packet() {		
packet_start_code_prefix	24	bslbf
stream_id	8	uimsbf
PES_packet_length	16	uimsbf
if(stream_id !=program_stream_map		
&& stream_id !=padding_stream		
&& stream_id !=private_stream_2		
&& stream_id !=ECM		
&& stream_id !=EMM		
&& stream_id !=program_stream_directory		
&& stream_id !=DSMCC_stream		

도면16B

&& stream_id !=ITU-T Rec. H.222.1 type E stream){		
'10'	2	bslbf
PES_scrambling_control	2	bslbf
PES_priority	1	bslbf
data_alignment_indicator	1	bslbf
copyright	1	bslbf
original_or_copy	1	bslbf
PTS_DTS_flags	2	bslbf
ESCR_flag	1	bslbf
ES_rate_flag	1	bslbf
DSM_trick_mode_flag	1	bslbf
additional_copy_info_flag	1	bslbf
PES_CRC_flag	1	bslbf
PES_extension_flag	1	bslbf
PES_header_data_length	8	uimsbf
if (PTS_DTS_flags == '10') {		
'0010' 4 bslbf	4	bslbf
PTS [32..30]	3	bslbf
marker_bit	1	bslbf
PTS [29..15]	15	bslbf
marker_bit	1	bslbf
PTS [14..0]	15	bslbf
marker_bit	1	bslbf
}		

도면17A

테이블 2-17. PES 패킷(계속)

신택스	비트수	니모닉
if(PTS_DTS_flags=='11'){		
'0011'	4	bslbf
PTS[32..30]	3	bslbf
marker_bit	1	bslbf
PTS [29..15]	15	bslbf
marker_bit	1	bslbf
PTS [14..0]	15	bslbf
marker_bit	1	bslbf
'0001'	4	bslbf
DTS [32..30]	3	bslbf
marker_bit	1	bslbf
DTS [29..15]	15	bslbf
marker_bit	1	bslbf
DTS [14..0]	15	bslbf
marker_bit	1	bslbf
}		

도면17B

if (ESCR_flag=='1'){		
reserved	2	bslbf
ESCR_base[32..30]	3	bslbf
marker_bit	1	bslbf
ESCR_base[29..15]	15	bslbf
marker_bit	1	bslbf
ESCR_base[14..0]	15	bslbf
marker_bit	1	bslbf
ESCR_extension	9	uimsbf
marker_bit	1	bslbf
}		
if(ES_rate_flag=='1'){		
marker_bit	1	bslbf
ES_rate	22	uimsbf
marker_bit	1	bslbf
}		

도면17C

if(DSM_trick_mode_flag=='1'){		
trick_mode_control	3	uimsbf
if(trick_mode_control==fast_forward){		
field_id	2	bslbf
intra_slice_refresh	1	bslbf
frequency_truncation	2	bslbf
}		
else if(trick_mode_control==slow_motion){		
rep_cntrl	5	uimsbf
}		
else if(trick_mode_control==freeze_frame){		
field_id	2	uimsbf
reserved	3	bslbf
}		
else if(trick_mode_control==fast_reverse){		
field_id	2	bslbf
intra_slice_refresh	1	bslbf
frequency_truncation	2	bslbf
else if(trick_mode_control==slow_reverse){		
rep_cntrl	5	uimsbf
}		
else		
reserved	5	bslbf
}		
if(additional_copy_info_flag=='1'){		
marker_bit	1	bslbf
additional_copy_info	7	bslbf
}		
if(PES_CRC_flag=='1'){		
previous_PES_packet_CRC	16	bslbf
}		

도면18A

테이블 2-17, PES 패킷(종료)

신택스	비트수	니모닉
if(PES_extension_flag=='1'){		
PES_private_data_flag	1	bslbf
pack_header_field_flag	1	bslbf
program_packet_sequence_counter_flag	1	bslbf
P-STD_buffer_flag	1	bslbf
reserved	3	bslbf
PES_extension_flag_2	1	bslbf
if(PES_private_data_flag=='1'){		
PES_private_data	128	bslbf
}		
if (pack_header_field_flag == '1') {		
pack_field_length	8	uimsbf
pack_header()		
}		
if (program_packet_sequence_counter_flag == '1') {		
marker_bit	1	bslbf
program_packet_sequence_counter	7	uimsbf
marker_bit	1	bslbf
MPEG1_MPEG2_identifier	1	bslbf
original_stuff_length	6	uimsbf
}		
if(P-STD_buffer_flag == '1') {		
'01'	2	bslbf
P-STD_buffer_scale	1	bslbf
P-STD_buffer_size	13	uimsbf
}		

도면18B

```
                    if( PES_extension_flag_2 == '1') {
                            marker_bit                                  1       bslbf
                            PES_extension_field_length                  7       uimsbf
                            for (i=0;i<PES_extension_field_length;i++) {
                                    reserved                            8       bslbf
                            }
                    }
            }
            for (i=0;i<N1;i++) {
                    stuffing_byte                                       8       bslbf
            }
            for (i=0;i<N2;i++) {
                    PES_packet_data_byte                                8       bslbf
            }
        }
        else if(stream_id == program_stream_map
        ||stream_id == private_stream_2
        ||stream_id == ECM
        ||stream_id == EMM
        ||stream_id == program_stream_directory
        ||stream_id == DSMCC_stream
        ||stream_id == ITU-T Rec.H.222.1 type E stream) {
                for (i=0;i<PES_packet_length;i++) {
                    PES_packet_data_byte                                8       bslbf
                }
        }
        else if(stream_id == padding_stream) {
                for (i=0;i<PES_packet_length;i++) {
                    padding_byte                                        8       bslbf
                }
        }
}
```

도면19A

테이블 2-18, stream_id 어사인먼트

stream_id	주역	스트림 코딩
1011 1100	1	program_stream_map
1011 1101	2	private_stream_1
1011 1110		padding_stream
1011 1111	3	private_stream_2
110x xxxx		ISO/IEC 13818-3 or ISO/IEC 11172-3 or ISO/IEC 13818-7 or ISO/IEC 14496-3 audio stream number x xxxx
1110 xxxx		ITU-T Rec. H.262 | ISO/IEC 13818-2 or ISO/IEC 11172-2 or ISO/IEC 14496-2 video stream number xxxx
1111 0000	3	ECM_stream
1111 0001	3	EMM_stream
1111 0010	5	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 Annex A or ISO/IEC 13818-6_DSMCC_stream
1111 0011	2	ISO/IEC_13522_stream
1111 0100	6	ITU-T Rec. H.222.1 type A
1111 0101	6	ITU-T Rec. H.222.1 type B
1111 0110	6	ITU-T Rec. H.222.1 type C
1111 0111	6	ITU-T Rec. H.222.1 type D
1111 1000	6	ITU-T Rec. H.222.1 type E
1111 1001	7	ancillary_stream
1111 1010		ISO/IEC14496-1_SL-packetized_stream
1111 1011		ISO/IEC14496-1_FlexMux_stream
1111 1100 … 1111 1110		reserved data stream
1111 1111	4	program_stream_directory

도면19B

기호 X는 값「0」 또는「1」모두 가능하며, 스트림의 형은 동일. 스트림 번호는 X로 주어진다.
주1: program__stream__map의 PES 패킷은 2. 5. 4. 1로 규정된 독자적인 신택스를 사용한다.
주2: private__stream__1 및 ISO/IEC__13552__stream의 PES 패킷은 ITU-T Rec. H.262인 ISO/IEC13818-2 비디오·스트림 및 ISO/IEC13818-3 오디오·스트림으로 규정된 것과 동일한 PES 패킷의 신택스에 따른다.
주3: private__stream__2、ECM__STREAM、및 EMM__stream의 PES 패킷은 PES__packet__length 필드 후에 몇몇 신택스가 지정되어 있지 않은 것을 제외하고 private__stream__1과 유사.
주4: program__stream__directory의 PES 패킷은 2. 5. 5로 규정된 독자적은 신택스를 사용한다.
주5: DSM-CC__stream은 ISO/IEC13818-6으로 규정된 독자적인 신택스를 사용한다.
주6: 이 stream__id는 표2-29 stream__type의 0x09와 조합하여 사용한다.
주7: 이 stream__id 는 프로그램·스트림 또는 ISO/IEC11172-1 시스템·스트림의 데이터용의 트랜스포트·스트림의 PES 패킷에만 사용한다 (2. 4. 3. 7참조)

도면20

stream_id	스트림 코딩 (스트림 종류)
1011 1101 1011 1110 1011 1111 110x xxxx 1110 xxxx	private_stream_1 padding_stream private_stream_2 오디오 스트림 넘버 x xxxx 비디오 스트림 넘버 xxxx

도면21

신택스	비트수	니모닉
private_stream1_PES_payload(){		
private_header(){		
private_stream_id	8	uimsbf
if(stream == ATRAC){		
reserved_for_future_use	8	bslbf
AU_locator	16	uimsbf
if(stream == LPCM){		
fs_flag	1	uimsbf
reserved_for_future_use	3	bslbf
ch_flag	4	uimsbf
AU_locator	16	uimsbf
}else if (stream == SUBTITLE){		
reserved_for_future_use	8	bslbf
AU_locator	16	uimsbf
}		
}		
private_payload()		
}		

도면22

private_stream_id	스트림 코딩 (스트림 종류)
0000 xxxx 0001 xxxx 100x xxxx	ATRAC 오디오 스트림 넘버 xxxx LPCM 스트림 넘버 xxxx 서브 타이틀 스트림 넘버 x xxxx

도면23

신택스	비트수	니모닉
private_stream2_PES_payload() {		
reserved_for_future_use	8	bslbf
video_stream_id	8	uimsbf
1stRef_picture	16	uimsbf
2ndRef_picture	16	uimsbf
3rdRef_picture	16	uimsbf
4thRef_picture	16	uimsbf
au_information()		
VBI()		
for (i=0; i<N1; i++) {		
padding_word	16	bslbf
}		
}		

도면24

신택스	비트수	니모닉
au_information() {		
length	16	uimsbf
reserved_for_word_alignment	8	bslbf
number_of_access_unit	8	uimsbf
for (i=0; i<number_of_access_unit; i++) {		
reserved	4	bslbf
pic_struct_copy	4	uimsbf
au_ref_flag	1	uimsbf
reserved	2	bslbf
AU_length	21	uimsbf
}		uimsbf
}		

도면25

// "PLAYLIST.DAT"		
number_of_PlayLists	2	
// PlayList()	PlayList #0	PlayList #0
capture_enable_flag_PlayList	1	0
number_of_PlayItems	2	1
// PlayItem #0		
Clip_Information_file_name	"00001.CLP"	"00003.CLP"
IN_time	180,090	90,000
OUT_time	27,180,090	81,090,000
// PlayItem #1		
Clip_Information_file_name	"00002.CLP"	
IN_time	90,000	
OUT_time	27,090,000	

도면26A

// Clip()	"00001.CLP"	"00002.CLP"	"00003.CLP"
presentation_start_time	90,000	90,000	90,000
presentation_end_time	27,990,000	27,090,000	81,090,000
capture_enable_flag_Clip	1	0	1
number_of_streams	4	4	3
// stream #0	StreamInfo()	StreamInfo()	StreamInfo()
stream_id	0xE0(video stream)	0xE0(video stream)	0xE0(video stream)
private_stream_id	0x00	0x00	0x00
	picture_size='720x480'	picture_size='720x480'	picture_size='720x480'
	frame_rate='29.97Hz'	frame_rate='29.97Hz'	frame_rate='29.97Hz'
	cc_flag='Yes'	cc_flag='Yes'	cc_flag='No'
number_of_DynamicInfo	0	0	2
// stream #1	StreamInfo()	StreamInfo()	StreamInfo()
stream_id	0xBD(private_stream_1)	0xBD(private_stream_1)	0xE1(video stream)
private_stream_id	0x00(ATRAC)	0x00(ATRAC)	0x00
	audio_language_code=' 일본어 '	audio_language_code=' 일본어 '	picture_size='720x480'
	channel_configuration='STEREO'	channel_configuration='STEREO'	frame_rate='29.97Hz'
	lfe_existence='NO'	lfe_existence='NO'	cc_flag='No'
	sampling_frequency='48KHz'	sampling_frequency='48KHz'	
number_of_DynamicInfo	0	0	0

도면26B

// stream #2	StreamInfo()	StreamInfo()	StreamInfo()
stream_id	0xBD(private_stream_1)	0xBD(private_stream_1)	0xBD(private_stream_1)
private_stream_id	0x80(Subtitle)	0x80(Subtitle)	0x00(ATRAC)
	subtitle_language_code='일본어'	subtitle_language_code='일본어'	audio_language_code= '일본어'
	configurable_flag=0	configurable_flag=0	channel_configuration='STEREO'
			lfe_existence='NO'
			sampling_frequency='48KHz'
number_of_DynamicInfo	0	0	3
// stream #3	StreamInfo()	StreamInfo()	
stream_id	0xBD(private_stream_1)	0xBD(private_stream_1)	
private_stream_id	0x81(Subtitle)	0x81(Subtitle)	
	subtitle_language_code='일본어'	subtitle_language_code='일본어'	
	configurable_flag=1	configurable_flag=1	
number_of_DynamicInfo	0	0	

도면27

EP_map() for "00001.CLP"		
number_of_stream_id_entries	1	
stream_id	0xE0(video stream)	
private_stream_id	0x00	
	PTS_EP_start	RPN_EP_start
	90000	0
	135045	244
	180090	305
	225135	427
	270180	701

도면28

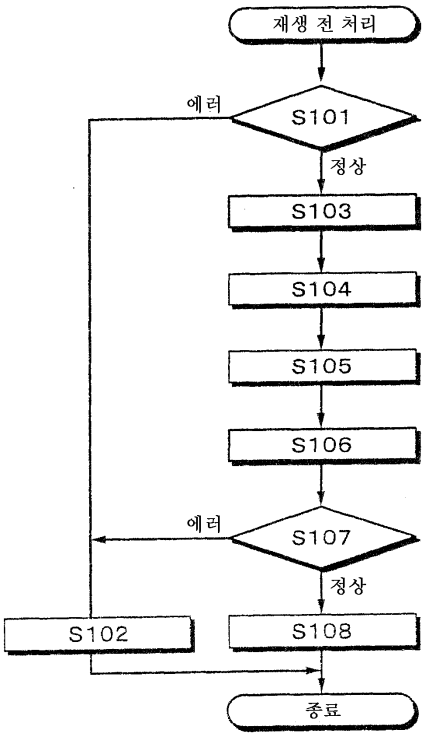
PlayList #0のPlayListMark()
number_of_PlayList_marks==7

Mark()	mark_type	ref_to_PlayItem_id	mark_time_stamp	entry_ES_stream_id	entry_ES_private_stream_id	mark_data		
#0	Chapter	0	180,090	0	0	1	선두	선두
#1	Index	0	5,580,090	0	0	1	1분	1분
#2	Index	0	10,980,090	0	0	2	2분	2분
#3	Event	0	16,380,090	0	0	0	3분	3분
#4	Chpater	1	90,000	0	0	2	선두	5분
#5	Index	1	5,490,000	0	0	1	1분	6분
#6	Index	1	10,890,000	0	0	2	2분	7분

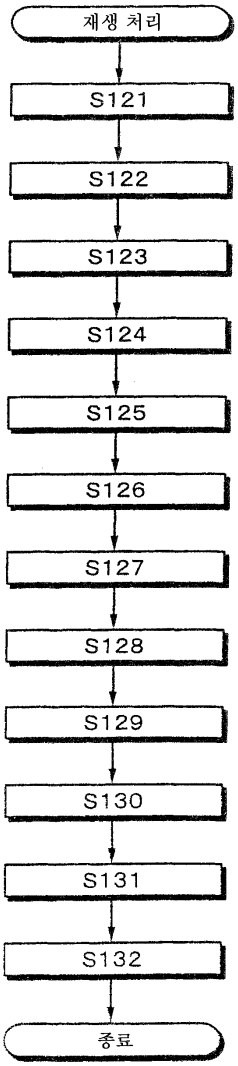
PlayList #1のPlayListMark()
number_of_PlayList_marks==3

Mark()	mark_type	ref_to_PlayItem_id	mark_time_stamp	entry_ES_stream_id	entry_ES_private_stream_id	mark_data	
#0	Chapter	0	90,000	0	0	0	선두
#1	Event	0	27,090,000	0xE0	0	1	5분
#2	Event	0	27,540,000	0xE1	0	2	5분5초(305초)

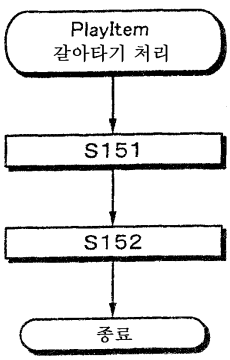
도면29



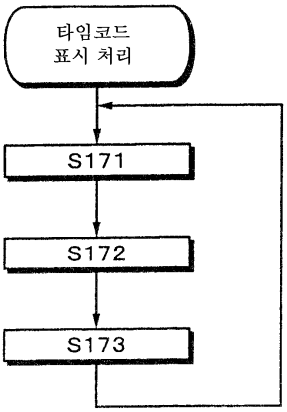
도면30


재생 처리
S121
S122
S123
S124
S125
S126
S127
S128
S129
S130
S131
S132
종료

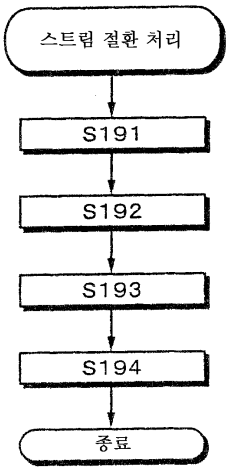
도면31


PlayItem
갈아타기 처리
S151
S152
종료

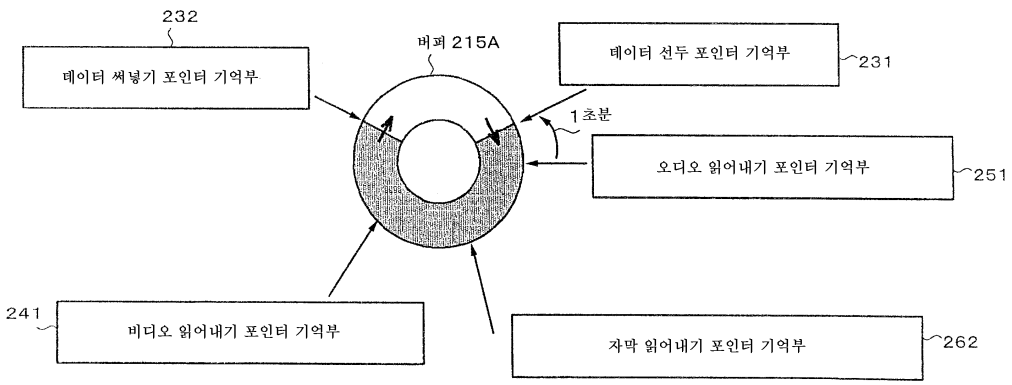
도면32



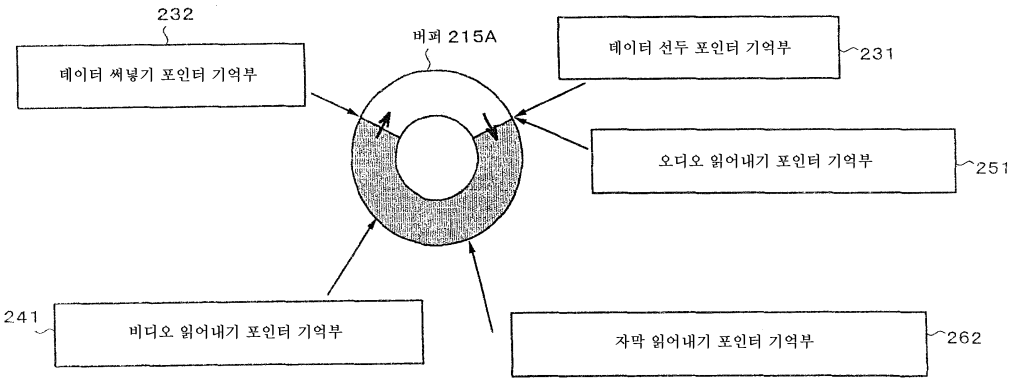
도면33



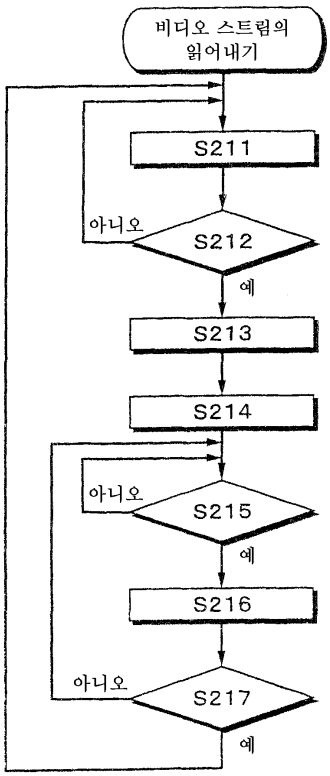
도면34



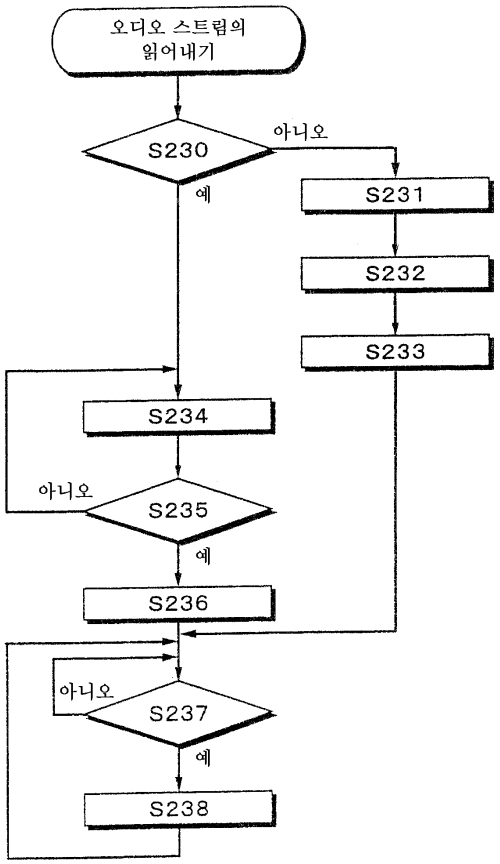
도면35


232
데이터 써넣기 포인터 기억부
버퍼 215A
데이터 선두 포인터 기억부
231
오디오 읽어내기 포인터 기억부
251
241
비디오 읽어내기 포인터 기억부
자막 읽어내기 포인터 기억부
262

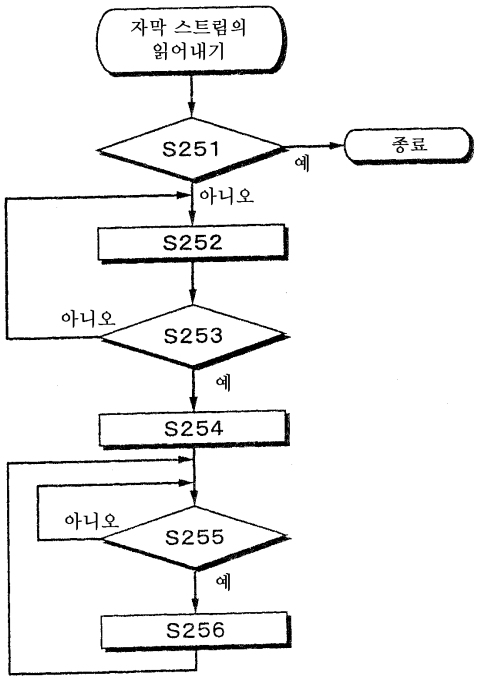
도면36


비디오 스트림의 읽어내기
S211
아니오
S212
예
S213
S214
아니오
S215
예
S216
아니오
S217
예

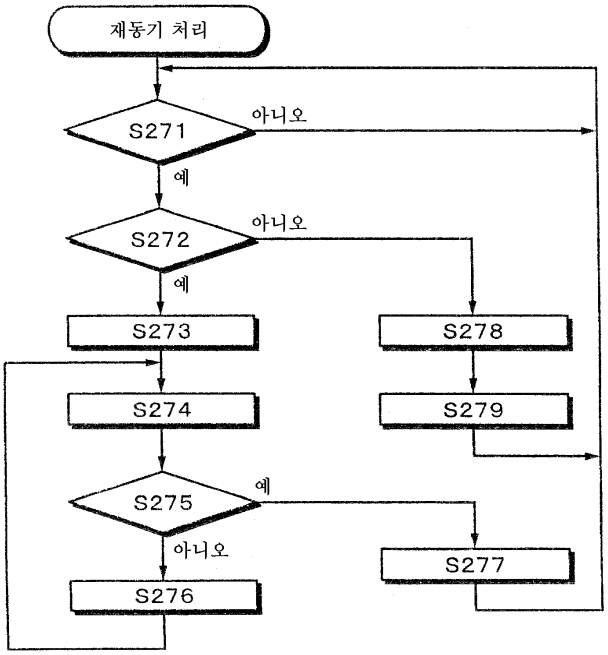
도면37


오디오 스트림의
읽어내기
S230
아니오
예
S231
S232
S233
S234
아니오
S235
예
S236
아니오
S237
예
S238

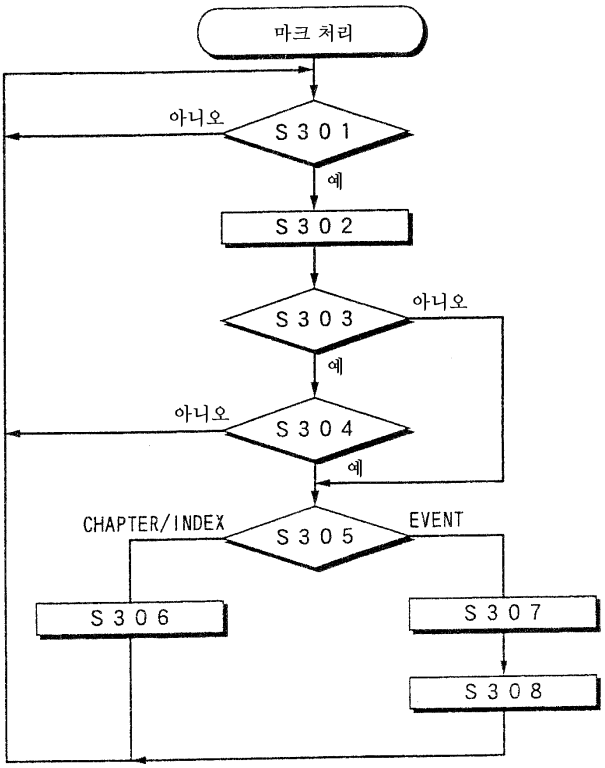
도면38


자막 스트림의
읽어내기
S251
예
종료
아니오
S252
아니오
S253
예
S254
아니오
S255
예
S256

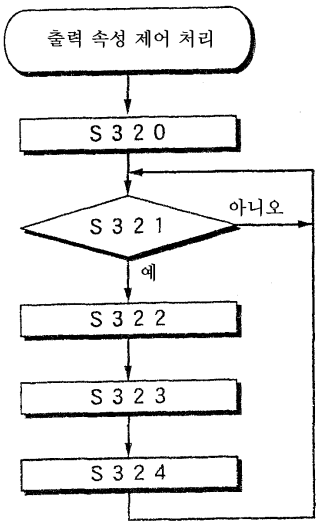
도면39


재동기 처리
S271
아니오
예
S272
아니오
예
S273
S278
S274
S279
S275
예
아니오
S277
S276

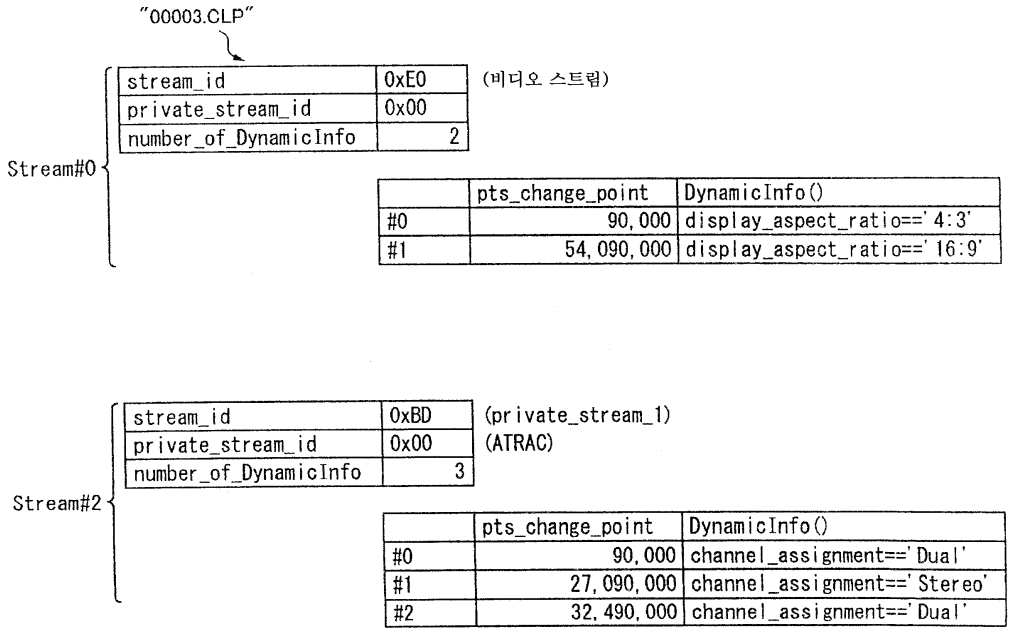
도면40


마크 처리
아니오
S 3 0 1
예
S 3 0 2
S 3 0 3
아니오
예
아니오
S 3 0 4
예
CHAPTER/INDEX
S 3 0 5
EVENT
S 3 0 6
S 3 0 7
S 3 0 8

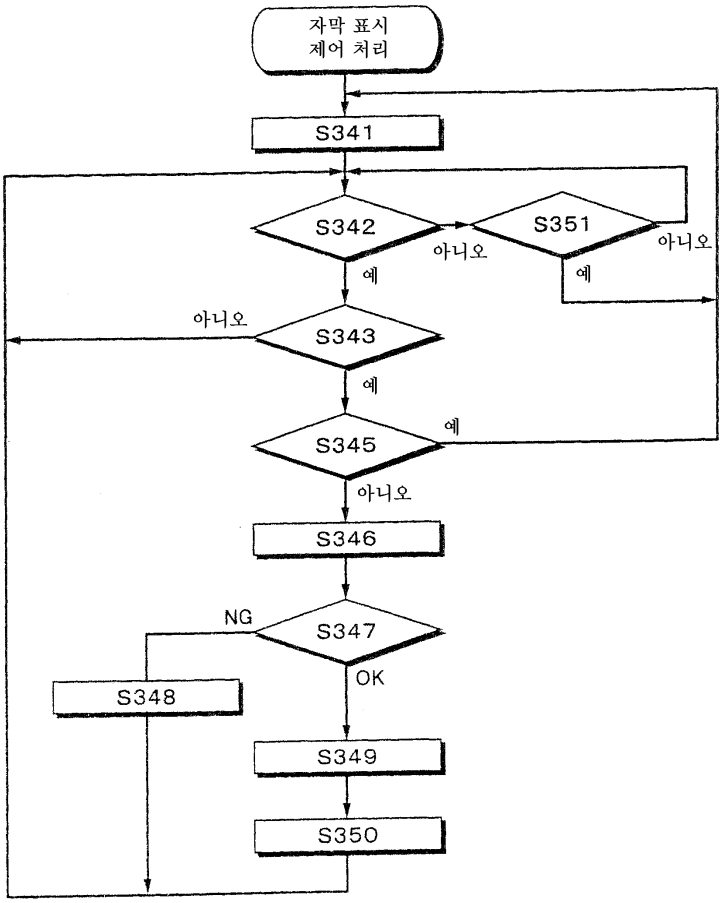
도면41


출력 속성 제어 처리
S320
S321
아니오
예
S322
S323
S324

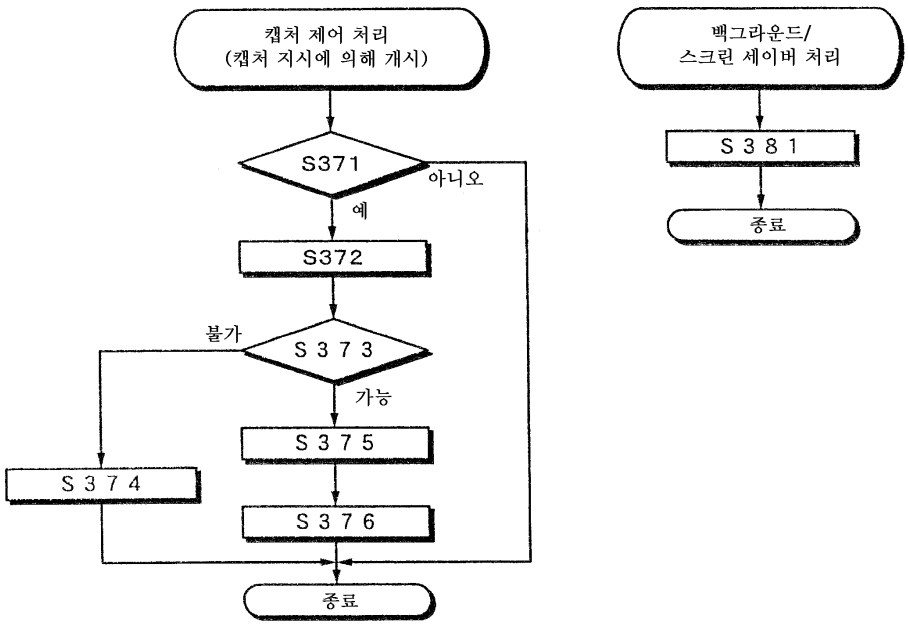
도면42


"00003.CLP"
Stream#0
stream_id 0xE0 (비디오 스트림)
private_stream_id 0x00
number_of_DynamicInfo 2
#0 90,000 display_aspect_ratio=='4:3'
#1 54,090,000 display_aspect_ratio=='16:9'
Stream#2
stream_id 0xBD (private_stream_1)
private_stream_id 0x00 (ATRAC)
number_of_DynamicInfo 3
#0 90,000 channel_assignment=='Dual'
#1 27,090,000 channel_assignment=='Stereo'
#2 32,490,000 channel_assignment=='Dual'

도면43


자막 표시
제어 처리
S341
S342
아니오
S351
아니오
예
예
아니오
S343
예
S345
예
아니오
S346
NG
S347
OK
S348
S349
S350

도면44


캡처 제어 처리
(캡처 지시에 의해 개시)
S371
아니오
예
S372
불가
S 3 7 3
가능
S 3 7 4
S 3 7 5
S 3 7 6
종료
백그라운드/
스크린 세이버 처리
S 3 8 1
종료

도면45

신택스	비트수	니모닉
private_stream2_PES_payload(){		
reserved_for_future_use	7	bslbf
capture_enable_ps2	1	uimsbf
video_stream_id	8	uimsbf
1stRef_picture	16	uimsbf
2ndRef_picture	16	uimsbf
3rdRef_picture	16	uimsbf
4thRef_picture	16	uimsbf
au_information()		
grain_variance_information()		
VBI()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
}		

도면46

신택스	비트수	니모닉
au_information(){		
length	16	uimsbf
reserved_for_word_align	8	bslbf
number_of_access_unit	8	uimsbf
for(i=0;i<number_of_access_unit;i++){		
reserved	3	bslbf
capture_enable_flag_AU	1	uimsbf
pic_struct_copy	4	uimsbf
au_ref_flag	1	uimsbf
reserved	2	bslbf
AU_length	21	uimsbf
}		uimsbf
}		