(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4536497号
(P4536497)

(45)発行日　平成22年9月1日(2010.9.1)　(24)登録日　平成22年6月25日(2010.6.25)

(51)Int.Cl.		F I		
G11B 27/10	(2006.01)	G11B	27/10	A
G11B 20/10	(2006.01)	G11B	20/10	321Z
G11B 20/12	(2006.01)	G11B	20/12	
H04N 5/91	(2006.01)	H04N	5/91	Z

請求項の数 11　(全 129 頁)

(21)出願番号　特願2004-350486 (P2004-350486)
(22)出願日　平成16年12月2日 (2004.12.2)
(65)公開番号　特開2006-164336 (P2006-164336A)
(43)公開日　平成18年6月22日 (2006.6.22)
審査請求日　平成19年11月20日 (2007.11.20)

前置審査

(73)特許権者　000002185
ソニー株式会社
東京都港区港南１丁目７番１号
(73)特許権者　395015319
株式会社ソニー・コンピュータエンタテインメント
東京都港区南青山二丁目６番２１号
(74)代理人　100082131
弁理士　稲本　義雄
(72)発明者　藤波　靖
東京都品川区北品川６丁目７番３５号　ソニー株式会社内
(72)発明者　浜田　俊也
東京都品川区北品川６丁目７番３５号　ソニー株式会社内

最終頁に続く

(54)【発明の名称】データ処理装置およびデータ処理方法、プログラムおよびプログラム記録媒体、並びにデータ記録媒体

(57)【特許請求の範囲】

【請求項１】

所定の変数を用いて時刻を計時する計時手段と、

アクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を加算して、前記変数を設定する設定手段とを備え、

前記アクセスユニット単位の表示時間の情報は、pic_structである

データ処理装置。

【請求項２】

前記アクセスユニットは、ビデオデータまたは字幕のアクセスユニットである

請求項１に記載のデータ処理装置。

【請求項３】

前記設定手段は、前記ビデオデータのアクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を、前記アクセスユニットの直後のアクセスユニットの表示を開始するタイミングで加算して、前記変数を設定する

請求項１に記載のデータ処理装置。

【請求項４】

前記設定手段は、さらに、前記アクセスユニット毎にタイムスタンプの有無を検出し、前記前記アクセスユニット毎にタイムスタンプがある場合、前記タイムスタンプの値を前

記変数に代入して設定し、前記アクセスユニット毎にタイムスタンプがない場合、アクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を加算して、前記変数を設定する
請求項１に記載のデータ処理装置。
【請求項５】
前記所定のアクセスユニットに対して所定のマークタイムスタンプの時刻で所定の処理を施すマーク処理手段をさらに備え、
前記設定手段は、前記所定のアクセスユニットに対して所定の処理を施すマークタイムスタンプの時刻が、前記計時手段により計時されている時刻から、前記アクセスユニット単位の表示時間内の時刻であるとき、前記計時手段により計時されている時刻が前記マークタイムスタンプの時刻となっていることを前記マーク処理手段に対して通知し、前記マーク処理手段は、前記通知に基づいて、前記所定のアクセスユニットに対して所定の処理を施す
請求項１に記載のデータ処理装置。
【請求項６】
前記マークタイムスタンプの時刻の最小間隔は、前記アクセスユニット単位の最大表示時間である
請求項５に記載のデータ処理装置。
【請求項７】
所定の変数を用いて時刻を計時する計時ステップと、
アクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を加算して、前記変数を設定する設定ステップとを含み、
前記アクセスユニット単位の表示時間の情報は、pic_structである
データ処理方法。
【請求項８】
所定の変数を用いて時刻を計時する計時ステップと、
アクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を加算して、前記変数を設定する設定ステップとを含む処理をコンピュータに実行させ、
前記アクセスユニット単位の表示時間の情報は、pic_structである
プログラムが記録されたプログラム記録媒体。
【請求項９】
所定の変数を用いて時刻を計時する計時ステップと、
アクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を加算して、前記変数を設定する設定ステップとを含む処理をコンピュータに実行させ、
前記アクセスユニット単位の表示時間の情報は、pic_structである
プログラム。
【請求項１０】
所定の変数を用いて時刻を計時する計時手段と、
アクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を加算して、前記変数を設定する設定手段と、
前記所定のアクセスユニットに対して所定のマークタイムスタンプの時刻で所定の処理を施すマーク処理手段と
を備えるデータ処理装置により読み出されるデータを記録するデータ記録媒体であって、
前記所定のアクセスユニットに対して所定の時刻で所定の処理を施すマークタイムスタンプを含むデータが記録され、
前記マークタイムスタンプの時刻の最小間隔が、前記アクセスユニット単位の最大表示

時間である
データ記録媒体。
【請求項１１】
所定の変数を用いて時刻を計時する計時手段と、
アクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を加算して、前記変数を設定する設定手段と、
前記所定のアクセスユニットに対して所定のマークタイムスタンプの時刻で所定の処理を施すマーク処理手段と
を備えるデータ処理装置により読み出されるデータのデータ構造であって、
前記所定のアクセスユニットに対して所定の時刻で所定の処理を施すマークタイムスタンプを含み、
前記マークタイムスタンプの時刻の最小間隔が、前記アクセスユニット単位の最大表示時間である
データ構造。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、データ処理装置およびデータ処理方法、プログラムおよびプログラム記録媒体、並びにデータ記録媒体に関し、特に、例えば、利便性等の高いデータ処理を可能とするデータ処理装置およびデータ処理方法、プログラムおよびプログラム記録媒体、並びにデータ記録媒体に関する。
【背景技術】
【０００２】
近年、大容量で、ランダムアクセスが可能な記録メディアとして、例えば、DVD(Digital Versatile Disc)が普及し、さらに、DVDを利用して各種の処理を行うDVD装置も広く普及している。
【０００３】
DVD装置としては、例えば、DVDに対して、テレビジョン放送番組のデータ等の記録再生を行うDVDレコーダや、DVDに地図情報等を記録し、その地図情報の表示を行うカーナビゲーションシステム、DVDにゲームのプログラム等を記録し、そのプログラムを実行するゲーム装置などがある。
【０００４】
なお、DVDについては、例えば、非特許文献１に、その詳細が記載されている。
【０００５】
【非特許文献１】DVD Specifications for Read-Only Disc Part 3; Version 1.1 Dcecmber 1997
【発明の開示】
【発明が解決しようとする課題】
【０００６】
DVDには、上述したように、大量のデータを記録することができる。従って、DVD装置には、そのような大量のデータについて、利便性等の高いデータ処理を行うことが要請される。
【０００７】
本発明は、このような状況に鑑みてなされたものであり、利便性等の高いデータ処理を行うことができるようにするものである。
【課題を解決するための手段】
【０００８】
本発明のデータ処理装置は、所定の変数を用いて時刻を計時する計時手段と、アクセスユニット単位の表示時間の情報を取得し、取得されたアクセスユニット単位の表示時間の情報により得られる値を加算して、変数を設定する設定手段とを備え、アクセスユニット

単位の表示時間の情報は、pic_structである。

【０００９】

前記アクセスユニット単位の表示時間の情報は、pic_structとするようにすることができる。

【００１０】

前記アクセスユニットは、ビデオデータまたは字幕のアクセスユニットとするようにすることができる。

【００１１】

前記設定手段には、ビデオデータのアクセスユニット単位の表示時間の情報を取得し、取得されたアクセスユニット単位の表示時間の情報により得られる値を、アクセスユニットの直後のアクセスユニットの表示を開始するタイミングで加算して、変数を設定させるようにすることができる。

【００１２】

前記設定手段には、さらに、アクセスユニット毎にタイムスタンプの有無を検出させ、アクセスユニット毎にタイムスタンプがある場合、タイムスタンプの値を変数に代入して設定させ、アクセスユニット毎にタイムスタンプがない場合、アクセスユニット単位の表示時間の情報を取得し、取得されたアクセスユニット単位の表示時間の情報により得られる値を加算して、変数を設定させるようにすることができる。

【００１３】

前記所定のアクセスユニットに対して所定のマークタイムスタンプの時刻で所定の処理を施すマーク処理手段をさらに設けるようにさせることができ、設定手段には、所定のアクセスユニットに対して所定の処理を施すマークタイムスタンプの時刻が、計時手段により計時されている時刻から、アクセスユニット単位の表示時間内の時刻であるとき、計時手段により計時されている時刻がマークタイムスタンプの時刻となっていることをマーク処理手段に対して通知させるようにすることができ、マーク処理手段には、通知に基づいて、所定のアクセスユニットに対して所定の処理を施すようにさせることができる。

【００１４】

前記マークタイムスタンプの時刻の最小間隔は、アクセスユニット単位の最大表示時間とするようにすることができる。

【００１５】

本発明のデータ処理方法は、所定の変数を用いて時刻を計時する計時ステップと、アクセスユニット単位の表示時間の情報を取得し、取得されたアクセスユニット単位の表示時間の情報により得られる値を加算して、変数を設定する設定ステップとを含み、アクセスユニット単位の表示時間の情報は、pic_structである。

【００１６】

本発明のプログラム記録媒体のプログラムは、所定の変数を用いて時刻を計時する計時ステップと、アクセスユニット単位の表示時間の情報を取得し、取得されたアクセスユニット単位の表示時間の情報により得られる値を加算して、変数を設定する設定ステップとを含み、アクセスユニット単位の表示時間の情報は、pic_structである。

【００１７】

本発明のプログラムは、所定の変数を用いて時刻を計時する計時ステップと、アクセスユニット単位の表示時間の情報を取得し、取得されたアクセスユニット単位の表示時間の情報により得られる値を加算して、変数を設定する設定ステップとを含む処理をコンピュータに実行させ、アクセスユニット単位の表示時間の情報は、pic_structである。

【００１８】

本発明のデータ記録媒体は、所定の変数を用いて時刻を計時する計時手段と、アクセスユニット単位の表示時間の情報を取得し、取得されたアクセスユニット単位の表示時間の情報により得られる値を加算して、変数を設定する設定手段と、所定のアクセスユニットに対して所定のマークタイムスタンプの時刻で所定の処理を施すマーク処理手段とを備えるデータ処理装置により読み出されるデータを記録するデータ記録媒体であって、所定の

アクセスユニットに対して所定の時刻で所定の処理を施すマークタイムスタンプを含むデータが記録され、マークタイムスタンプの時刻の最小間隔が、アクセスユニット単位の最大表示時間である。

【００１９】

本発明のデータ構造は、所定の変数を用いて時刻を計時する計時手段と、アクセスユニット単位の表示時間の情報を取得し、取得されたアクセスユニット単位の表示時間の情報により得られる値を加算して、変数を設定する設定手段と、所定のアクセスユニットに対して所定のマークタイムスタンプの時刻で所定の処理を施すマーク処理手段とを備えるデータ処理装置により読み出されるデータのデータ構造であって、所定のアクセスユニットに対して所定の時刻で所定の処理を施すマークタイムスタンプを含み、マークタイムスタンプの時刻の最小間隔が、アクセスユニット単位の最大表示時間である。

【００２０】

本発明のデータ処理装置および方法、並びに、プログラムおよびプログラム記録媒体においては、所定の変数を用いて時刻が計時され、アクセスユニット単位の表示時間の情報が取得され、取得されたアクセスユニット単位の表示時間の情報により得られる値が加算されて、変数が設定され、アクセスユニット単位の表示時間の情報が、pic_structとされる。

【００２１】

本発明のデータ記録媒体およびデータ構造においては、所定のアクセスユニットに対して所定の時刻で所定の処理を施すマークタイムスタンプを含むデータが記録され、マークタイムスタンプの時刻の最小間隔が、アクセスユニット単位の最大表示時間とされる。

【００２２】

本発明のデータ処理装置は、独立した装置であっても良いし、データ処理を行うブロックであっても良い。

【発明の効果】

【００２３】

本発明によれば、利便性等の高いデータ処理が可能となる。特に、効率的な再生を可能にすると共に、ストリームデータを再生する装置が単独で計時する時計の機能がなくてもタイムスタンプに従って正確にストリームデータを再生することが可能になる。

【発明を実施するための最良の形態】

【００２４】

以下に本発明の実施の形態を説明するが、請求項に記載の構成要件と、発明の実施の形態における具体例との対応関係を例示すると、次のようになる。この記載は、請求項に記載されている発明をサポートする具体例が、発明の実施の形態に記載されていることを確認するためのものである。従って、発明の実施の形態中には記載されているが、構成要件に対応するものとして、ここには記載されていない具体例があったとしても、そのことは、その具体例が、その構成要件に対応するものではないことを意味するものではない。逆に、具体例が構成要件に対応するものとしてここに記載されていたとしても、そのことは、その具体例が、その構成要件以外の構成要件には対応しないものであることを意味するものでもない。

【００２５】

さらに、この記載は、発明の実施の形態に記載されている具体例に対応する発明が、請求項に全て記載されていることを意味するものではない。換言すれば、この記載は、発明の実施の形態に記載されている具体例に対応する発明であって、この出願の請求項には記載されていない発明の存在、すなわち、将来、分割出願されたり、補正により追加される発明の存在を否定するものではない。

【００２６】

すなわち、本発明のデータ処理装置は、所定の変数を用いて時刻を計時する計時手段（例えば、図２の計時部２１４Ａ）と、アクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を加算して、前

記変数を設定する設定手段（例えば、図２のデコード制御モジュール２１４）とを備え、アクセスユニット単位の表示時間の情報は、pic_structである。

【００２７】

また、本発明のデータ処理方法は、所定の変数を用いて時刻を計時する計時ステップ（例えば、図３８のフローチャートにおけるステップＳ１４３の処理）と、アクセスユニット単位の表示時間の情報を取得し、取得された前記アクセスユニット単位の表示時間の情報により得られる値を加算して、前記変数を設定する設定ステップ（例えば、図３８のフローチャートにおけるステップＳ１４４の処理）とを含み、アクセスユニット単位の表示時間の情報は、pic_structである。

【００２８】

尚、プログラム記録媒体およびプログラムについては、データ処理方法と同様であるので、その説明は省略する。

【００２９】

以下、図面を参照して、本発明の実施の形態について説明する。

【００３０】

［ハードウェア構成］

図１は、本発明を適用したディスク再生装置の一実施の形態のハードウェアの構成例を示すブロック図である。

【００３１】

図１のディスク再生装置は、例えば、ディスクプレーヤや、ゲーム装置、カーナビゲーションシステムその他に適用することができる。

【００３２】

図１のディスク再生装置において、ディスク１０１は、例えば、DVDなどの光ディスク、あるいは光磁気ディスク、磁気ディスクなどであり、ビデオデータや、オーディオデータ、字幕データなどのコンテンツデータ、さらには、コンテンツデータを再生するのに必要なデータが記録されている。

【００３３】

なお、ディスク１０１に記録されるデータ（記録データ）には、必要に応じて、コンピュータが実行可能なプログラムも含まれる。また、本実施の形態では、記録媒体として、ディスク状の記録媒体であるディスク１０１を採用するが、その他、記録媒体としては、例えば、半導体メモリや、テープ状の記録媒体であってもよい。さらに、図１のディスク再生装置には、遠方にあるディスク１０１から読み出されて送信されてくるデータを入力することができる。即ち、ディスク１０１からのデータの読み出しは、ディスク再生装置に接続した別の装置で行い、その別の装置で読み出されたデータを、ディスク再生装置で受信して処理することができる。また、ディスク再生装置では、ディスク１０１に記録されたデータと同様のデータをストレージに記憶しているサーバ等から、インターネット等のネットワークを介して、データの配信を受けて処理することも可能である。さらに、ディスク再生装置では、サーバその他の装置からのデータを受信し、一旦、ディスク１０１に記録してから、そのディスク１０１に記録されたデータを処理することも可能である。

【００３４】

ディスクドライブ１０２には、ディスク１０１が着脱可能になっている。ディスクドライブ１０２は、図示せぬインターフェースを内蔵し、そのインターフェースを通じて、ドライブインターフェース１１４に接続されている。ディスクドライブ１０２は、そこに装着されたディスク１０１を駆動し、ドライブインターフェース１１４からの読み出し等の命令にしたがって、ディスク１０１からデータを読み出して、ドライブインターフェース１１４に供給する等の処理を行う。

【００３５】

バス１１１には、CPU(Central Processing Unit)１１２、メモリ１１３、ドライブインターフェース１１４、入力インターフェース１１５、ビデオデコーダ１１６、オーディオデコーダ１１７、ビデオ出力インターフェース１１８、オーディオ出力インターフェース

１１９が接続されている。
【００３６】
CPU１１２およびメモリ１１３は、コンピュータシステムを形成している。即ち、CPU１１２は、メモリ１１３に記憶されたプログラムである、後述するソフトウェアモジュール群を実行し、ディスク再生装置全体を制御するとともに、後述する各種の処理を行う。メモリ１１３は、CPU１１２が実行するソフトウェアモジュール群を記憶している。また、メモリ１１３は、CPU１１２の動作上必要なデータを一時記憶する。なお、メモリ１１３は、不揮発性メモリのみ、または揮発性メモリと不揮発性メモリとの組み合わせで構成することが可能である。また、図１のディスク再生装置に、ハードディスクを設け、そのハードディスクに、CPU１１２が実行するソフトウェアモジュール群を記録（インストール）しておく場合には、メモリ１１３は、揮発性メモリのみで構成することが可能である。
【００３７】
ここで、CPU１１２が実行するプログラム（ソフトウェアモジュール群）は、ディスク再生装置に内蔵されている記録媒体としてのメモリ１１３に予め記録しておく（記憶させておく）ことができる。
【００３８】
あるいはまた、プログラムは、ディスク１０１、さらには、ディスク１０１以外のフレキシブルディスク、CD-ROM(Compact Disc Read Only Memory), MO(Magneto Optical)ディスク、磁気ディスク、メモリカードなどのリムーバブル記録媒体に、一時的あるいは永続的に格納（記録）しておくことができる。このようなリムーバブル記録媒体は、いわゆるパッケージソフトウエアとして提供することができる。
【００３９】
なお、プログラムは、メモリ１１３にあらかじめ記憶させておくこと、あるいは、上述したようなリムーバブル記録媒体からディスク再生装置にインストールすることができる。また、プログラムは、ダウンロードサイトから、ディジタル衛星放送用の人工衛星を介して、ディスク再生装置に無線で転送したり、LAN(Local Area Network)、インターネットといったネットワークを介して、ディスク再生装置に有線で転送し、ディスク再生装置では、そのようにして転送されてくるプログラムを、入力インターフェース１１５で受信し、内蔵するメモリ１１３にインストールすることができる。
【００４０】
さらに、プログラムは、１のCPUにより処理されるものであっても良いし、複数のCPUによって分散処理されるものであっても良い。
【００４１】
ドライブインターフェース１１４は、CPU１１２の制御の下、ディスクドライブ１０２を制御し、これにより、ディスクドライブ１０２がディスク１０１から読み出したデータを、バス１１１を介して、CPU１１２や、メモリ１１３、ビデオデコーダ１１６、オーディオデコーダ１１７などに供給する。
【００４２】
入力インターフェース１１５は、図示せぬキー（ボタン）や、リモコン（リモートコントローラ）がユーザに操作されることによって供給される信号を受信し、バス１１１を介して、CPU１１２に供給する。なお、入力インターフェース１１５は、その他、例えば、モデム（ADSL(Asymmetric Digital Subscriber Line)モデムを含む）や、NIC(Network Interface Card)などの通信インターフェースとしても機能する。
【００４３】
ビデオデコーダ１１６は、ディスクドライブ１０２によってディスク１０１から読み出され、ドライブインターフェース１１４およびバス１１１を介して供給される、ビデオデータの符号化データ（符号化オーディオデータ）をデコードし、その結果得られるビデオデータを、バス１１１を介して、CPU１１２やビデオ出力インターフェース１１８に供給する。
【００４４】

オーディオデコーダ１１７は、ディスクドライブ１０２によってディスク１０１から読み出され、ドライブインターフェース１１４およびバス１１１を介して供給される、オーディオデータの符号化データ（符号化オーディオデータ）をデコードし、その結果得られるオーディオデータを、バス１１１を介して、CPU１１２やオーディオ出力インターフェース１１９に供給する。

【００４５】

ビデオ出力インターフェース１１８は、バス１１１を介して供給されるビデオデータに必要な処理を施し、ビデオ出力端子１２０から出力する。オーディオ出力インターフェース１１９は、バス１１１を介して供給されるオーディオデータに必要な処理を施し、オーディオ出力端子１２１から出力する。

【００４６】

ビデオ出力端子１２０は、図示せぬCRT(Cathode Ray Tube)や、液晶パネル等のビデオ出力装置に接続されており、従って、ビデオ出力端子１２０から出力されるビデオデータは、ビデオ出力装置に供給されて表示される。オーディオ出力端子１２１は、図示せぬスピーカやアンプなどのオーディオ出力装置に接続されており、従って、オーディオ出力端子１２１から出力されるオーディオデータは、オーディオ出力装置に供給されて出力される。

【００４７】

なお、ディスク再生装置から、ビデオ出力装置とオーディオ出力装置へのビデオデータとオーディオデータの供給は、有線または無線のいずれによって行うことも可能である。

【００４８】

［ソフトウェアモジュール群の構成］

次に、図２は、図１のCPU１１２が実行するソフトウェアモジュール群の構成例を示している。

【００４９】

CPU１１２が実行するソフトウェアモジュール群は、オペレーティングシステム（OS）２０１と、アプリケーションプログラムとしてのビデオコンテンツ再生プログラム２１０に大別される。

【００５０】

「オペレーティングシステム２０１」

オペレーティングシステム２０１は、ディスク再生装置の電源が投入されると最初に起動し（CPU１１２がオペレーティングシステム２０１を実行し）、初期設定等の必要な処理を行い、アプリケーションプログラムであるビデオコンテンツ再生プログラム２１０を呼び出す。

【００５１】

オペレーティングシステム２０１は、ビデオコンテンツ再生プログラム２１０に対して、ファイルの読み出し等のインフラ（インフラストラクチャ(infrastructure)）的なサービスを提供する。即ち、オペレーティングシステム２０１は、例えば、ファイルの読み出しに関しては、ビデオコンテンツ再生プログラム２１０からのファイルの読み出しのリクエストに対して、ドライブインターフェース１１４を介してディスクドライブ１０２を操作して、ディスク１０１のデータを読み出し、ビデオコンテンツ再生プログラム２１０に渡すサービスを提供する。また、オペレーティングシステム２０１は、ファイルシステムの解釈等も行う。

【００５２】

なお、オペレーティングシステム２０１は、マルチタスク処理の機能を備えており、複数のソフトウェアモジュールを、時分割で（見かけ上）同時に動作させることができる。即ち、ビデオコンテンツ再生プログラム２１０は、幾つかのソフトウェアモジュールで構成されるが、各ソフトウェアモジュールは、並列で動作することができる。

【００５３】

「ビデオコンテンツ再生プログラム２１０」

ビデオコンテンツ再生プログラム２１０は、スクリプト制御モジュール２１１、プレイヤ制御モジュール２１２、コンテンツデータ供給モジュール２１３、デコード制御モジュール２１４、バッファ制御モジュール２１５、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、字幕デコーダ制御モジュール２１８、グラフィックス処理モジュール２１９、ビデオ出力モジュール２２０、およびオーディオ出力モジュール２２１で構成されている。

【００５４】

ビデオコンテンツ再生プログラム２１０は、ディスク１０１の再生にあたって中心的な役割を果たすソフトウェアであり、ディスク１０１がディスクドライブ１０２に装着（挿入）されると、そのディスク１０１が、コンテンツが記録された後述するフォーマットのディスクであるかを確認する。さらに、ビデオコンテンツ再生プログラム２１０は、ディスク１０１から、後述するスクリプトファイルを読み出して実行し、また、ディスク１０１から、そのディスク１０１に記録されたコンテンツを再生するのに必要なメタデータ（データベース情報）のファイルを読み出し、そのメタデータに基づいて、コンテンツの再生を制御する。

【００５５】

以下、図２のビデオコンテンツ再生プログラム２１０を構成するソフトウェアモジュールについて説明する。なお、図２においては、原則として、実線の矢印は、コンテンツのデータを表し、点線の矢印は、制御のデータを表す。

【００５６】

「スクリプト制御モジュール２１１」

スクリプト制御モジュール２１１は、ディスク１０１に記録されたスクリプトファイルに記述されているスクリプトプログラム（スクリプト）を解釈して実行する。スクリプトプログラムでは、例えば、「グラフィクス処理モジュール２１９を操作し、メニュー等の画像を作成して表示する」、「リモコン等のUI(User Interface)からの信号に従いメニューの表示を変更する（例えば、メニュー上のカーソルを移動する等）」、「プレイヤ制御モジュール２１２を制御する」等の動作を記述することができる。

【００５７】

「プレイヤ制御モジュール２１２」

プレイヤ制御モジュール２１２は、ディスク１０１に記録されているメタデータ（データベース情報）等を参照し、コンテンツの再生に関する制御を行う。即ち、プレイヤ制御モジュール２１２は、例えば、ディスク１０１に記録されている、後述するPlayList()やClip()を解析し、その解析結果にしたがって、コンテンツデータ供給モジュール２１３や、デコード制御モジュール２１４、バッファ制御モジュール２１５を制御する。また、プレイヤ制御モジュール２１２は、スクリプト制御モジュール２１１や入力インターフェース１１５からの指示にしたがい、再生対象のストリームを切り替える、後述するストリーム切り替え等の制御を行う。さらに、プレイヤ制御モジュール２１２は、デコード制御モジュール２１４から時刻を取得し、時刻表示や、後述するマーク（Mark()）の処理等を行う。

【００５８】

「コンテンツデータ供給モジュール２１３」

コンテンツデータ供給モジュール２１３は、プレイヤ制御モジュール２１２の制御にしたがい、あるいは、バッファ制御モジュール２１５に蓄積されたデータの量に基づき、ディスク１０１からのコンテンツのデータやメタデータ等の読み出しを、オペレーティングシステム２０１に要求する。

【００５９】

なお、オペレーティングシステム２０１が、コンテンツデータ供給モジュール２１３からの要求に応じてディスク１０１から読み出したメタデータ等は、必要なモジュールに供給される。また、オペレーティングシステム２０１が、コンテンツデータ供給モジュール２１３からの要求に応じてディスク１０１から読み出したコンテンツのデータは、バッフ

ァ制御モジュール２１５に供給される。

【００６０】

「デコード制御モジュール２１４」

デコード制御モジュール２１４は、プレイヤ制御モジュール２１２からの制御にしたがい、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、および字幕デコーダ制御モジュール２１８の動作を制御する。また、デコード制御モジュール２１４は、時刻を計時する計時部２１４Ａを内蔵し、ビデオデコーダ制御モジュール２１６の制御によって出力されるビデオデータの出力と、そのビデオデータと同期して出力されるべきデータ（出力データ）の出力、即ち、ここでは、オーディオデコーダ制御モジュール２１７の制御によって出力されるオーディオデータの出力との同期を管理する。

【００６１】

計時部２１４Ａは、外部から供給された基準クロックあるいは、デコーダ等に同期する内部クロックを計数することにより自律的に計時するようにしてもよい。

【００６２】

しかしながら、図２で示される各種のモジュールを制御することにより実現される、ソフトウェアベースのデコーダでは、これらの処理とは別に、独立した計時処理をソフトウェアにより実行させることは、CPU１１２の処理負荷を増大させることになる。このため、計時部２１４Ａは、エレメンタリストリームのデコーダ（例えばビデオデコーダ）からのビデオデータの出力に応じて時刻を更新するという方式（上述の内部クロックを自律的に計時する方式）が主流になりつつある。

【００６３】

図３は、計時部２１４Ａが独立した時計を示す場合の時刻と実際の時間経過の関係を図に示している。図３においては、計時部２１４Ａは90kHzのクロックでカウントアップするので、時刻0から右上がりの直線で増加し、時刻33.3ミリ秒の瞬間に時計が示す時刻は3003となっている。

【００６４】

図４は、計時部２１４Ａがビデオデコーダのビデオデータの出力に応じて時刻を更新する時計を示す場合の時刻の一例を示している。図４の場合、時刻は、例えば、33.3ミリ秒が経過すると時計の出力時刻が3003に更新され、83.3ミリ秒が経過すると出力時刻は7507に更新され、また、116ミリ秒が経過すると出力時刻は10510に更新される。ここで、1フレームの出力時間は、16.66ミリ秒である。

【００６５】

尚、図３においては、実際には1/90kHzの分解能であらわされる階段状に時刻が推移することになるが、図４との対比のため、図３では直線で表現されている。また、以降においては、計時部２１４Ａは、図４を参照して説明した、ビデオデータの出力に応じて時刻を更新するという方式であるものとして説明する。

【００６６】

「バッファ制御モジュール２１５」

バッファ制御モジュール２１５は、図１のメモリ１１３の記憶領域の一部であるバッファ２１５Ａを内蔵しており、そのバッファ２１５Ａに、コンテンツデータ供給モジュール２１３がオペレーティングシステム２０１に要求を行うことによってディスク１０１から読み出されたコンテンツのデータを一時記憶する。

【００６７】

また、バッファ制御モジュール２１５は、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、または字幕デコーダ制御モジュール２１８の要求にしたがって、バッファ２１５Ａに記憶されたデータを、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、または字幕デコーダ制御モジュール２１８に供給する。

【００６８】

即ち、バッファ制御モジュール２１５は、後述する図５で説明するビデオ読み出し機能

部２３３、オーディオ読み出し機能部２３４、および字幕読み出し機能部２３５を内蔵している。そして、バッファ制御モジュール２１５は、ビデオデコーダ制御モジュール２１６からのデータの要求を、ビデオ読み出し機能部２３３で処理することにより、バッファ２１５Ａに記憶されたデータを、ビデオデコーダ制御モジュール２１６に供給する。同様に、バッファ制御モジュール２１５は、オーディオデコーダ制御モジュール２１７からのデータの要求を、オーディオ読み出し機能部２３４で処理することにより、バッファ２１５Ａに記憶されたデータを、オーディオデコーダ制御モジュール２１７に供給するとともに、字幕デコーダ制御モジュール２１８からのデータの要求を、字幕読み出し機能部２３５で処理することにより、バッファ２１５Ａに記憶されたデータを、字幕デコーダ制御モジュール２１８に供給する。

【００６９】

「ビデオデコーダ制御モジュール２１６」

ビデオデコーダ制御モジュール２１６は、バッファ制御モジュール２１５内のビデオ読み出し機能部２３３（図５）を操作して、ビデオデータを符号化したデータ（ビデオ符号化データ）を、ビデオアクセスユニット単位で、バッファ制御モジュール２１５のバッファ２１５Ａから読み出し、図１のビデオデコーダ１１６に供給する。また、ビデオデコーダ制御モジュール２１６は、ビデオデコーダ１１６を制御し、ビデオアクセスユニット単位のデータをデコードさせる。さらに、ビデオデコーダ制御モジュール２１６は、ビデオデコーダ１１６でのデコードの結果得られるビデオデータを、グラフィクス処理モジュール２１９に供給する。

【００７０】

ここで、ビデオアクセスユニットとは、例えば、ビデオデータの１ピクチャ（１フレームまたは１フィールド）分である。

【００７１】

「オーディオデコーダ制御モジュール２１７」

オーディオデコーダ制御モジュール２１７は、バッファ制御モジュール２１５内のオーディオ読み出し機能部２３４（図５）を操作して、オーディオデータを符号化したデータ（オーディオ符号化データ）を、オーディオアクセスユニット単位で、バッファ制御モジュール２１５のバッファ２１５Ａから読み出し、図１のオーディオデコーダ１１７に供給する。また、オーディオデコーダ制御モジュール２１７は、オーディオデコーダ１１７を制御し、オーディオアクセスユニット単位のデータをデコードさせる。さらに、オーディオデコーダ制御モジュール２１７は、オーディオデコーダ１１７でのデコードの結果得られるオーディオデータを、オーディオ出力モジュール２２１に供給する。

【００７２】

ここで、オーディオアクセスユニットとは、オーディオデータの所定のデータ量分（例えば、１ピクチャに同期して出力される分）である。本実施の形態では、オーディオアクセスユニットは、例えば、既知の固定長であるとする。

【００７３】

「字幕デコーダ制御モジュール２１８」

字幕デコーダ制御モジュール２１８は、バッファ制御モジュール２１５内の字幕読み出し機能部２３５（図５）を操作して、字幕データを符号化したデータ（字幕符号化データ）を、字幕アクセスユニット単位で、バッファ制御モジュール２１５のバッファ２１５Ａから読み出す。また、字幕デコーダ制御モジュール２１８は、内部に、図示せぬ字幕デコードソフトウェアを備えており、バッファ２１５Ａから読み出したデータをデコードする。さらに、字幕デコーダ制御モジュール２１８は、そのデコードの結果得られる字幕データ（字幕の画像データ）を、グラフィクス処理モジュール２１９に供給する。

【００７４】

ここで、字幕アクセスユニットとは、字幕データの所定のデータ量分（例えば、１ピクチャに同期して出力される分）である。本実施の形態では、字幕アクセスユニットのサイズは、例えば、その字幕アクセスユニットの先頭に記述されていることとする。

【００７５】
「グラフィクス処理モジュール２１９」
グラフィクス処理モジュール２１９は、プレイヤ制御モジュール２１２の制御（指示）にしたがい、字幕デコーダ制御モジュール２１８からの字幕データの拡大や縮小を行い、ビデオデコーダ制御モジュール２１６からのビデオデータと加算（オーバーレイ）する。さらに、グラフィクス処理モジュール２１９は、字幕データとの加算後のビデオデータのサイズ（画枠）を、図１のビデオ出力端子１２０に接続されたビデオ出力装置の表示画面にあわせるための拡大または縮小等を行い、その結果得られるビデオデータを、ビデオ出力モジュール２２０に出力する。
【００７６】
また、グラフィクス処理モジュール２１９は、スクリプト制御モジュール２１１やプレイヤ制御モジュール２１２の指示（制御）に従い、メニューやメッセージ等を生成し、出力ビデオデータにオーバーレイする。
【００７７】
さらに、グラフィクス処理モジュール２１９は、図１のビデオ出力端子１２０に接続されたビデオ出力装置のアスペクト比と、ディスク１０１に記録されたビデオデータのアスペクト比を指示する情報等とに基づいて、ビデオ出力モジュール２２０に出力するビデオデータのアスペクト比の変換を行う。
【００７８】
即ち、例えば、ビデオ出力装置のアスペクト比が16:9である場合において、ビデオデータのアスペクト比を指示する情報が4:3のアスペクト比を表しているときには、グラフィクス処理モジュール２１９は、ビデオ出力モジュール２２０に出力するビデオデータを、横方向（水平方向）にスクイーズ（縮小）処理し、左右に黒味を入れて出力する。また、例えば、ビデオ出力装置のアスペクト比が4:3である場合において、ビデオデータのアスペクト比を指示する情報が16:9のアスペクト比を表しているときには、グラフィクス処理モジュール２１９は、ビデオ出力モジュール２２０に出力するビデオデータを、縦方向（垂直方向）にスクイーズ（縮小）処理し、上下に黒味を入れて出力する。
【００７９】
なお、ビデオ出力装置のアスペクト比と、ビデオデータのアスペクト比を指示する情報が表すアスペクト比とが、いずれも、4:3や16:9で、同一である場合、グラフィクス処理モジュール２１９は、ビデオ出力モジュール２２０に出力するビデオデータを、スクイーズ処理することなく、そのまま出力する。
【００８０】
その他、グラフィクス処理モジュール２１９は、例えば、プレイヤ制御モジュール２１２からの要求に応じて、現在処理中のビデオデータをキャプチャする。さらに、グラフィクス処理モジュール２１９は、そのキャプチャしたビデオデータを記憶し、あるいは、プレイヤ制御モジュール２１２に供給する。
【００８１】
「ビデオ出力モジュール２２０」
ビデオ出力モジュール２２０は、図１のメモリ１１３の一部を排他的に占有してFIFO(First In First Out)２２０Ａ（バッファ）として使用し、グラフィクス処理モジュール２１９からのビデオデータを一時的に記憶し、また、そのFIFO２２０Ａに記憶されたビデオデータを適宜読み出して、ビデオ出力端子１２０（図１）に出力する。
【００８２】
「オーディオ出力モジュール２２１」
オーディオ出力モジュール２２１は、図１のメモリ１１３の一部を排他的に占有してFIFO２２１Ａ（バッファ）として使用し、オーディオデコーダ制御モジュール２１７（オーディオデコーダ１１７）からのオーディオデータを一時的に記憶し、また、そのFIFO２２１Ａに記憶されたオーディオデータを適宜読み出して、オーディオ出力端子１２１（図１）に出力する。

【００８３】
さらに、オーディオ出力モジュール２２１は、オーディオデコーダ制御モジュール２１７からのオーディオデータが、左チャネルが「主音声」のオーディオデータで、右チャネルの「副音声」のオーディオデータであるデュアル(Dual)（二ヶ国語）モードのオーディオデータである場合、あらかじめ指定された音声出力モードに従って、オーディオデコーダ制御モジュール２１７からのオーディオデータを、オーディオ出力端子１２１に出力する。
【００８４】
即ち、音声出力モードとして、例えば、「主音声」が指定されているときには、オーディオ出力モジュール２２１は、オーディオデコーダ制御モジュール２１７からのオーディオデータのうちの左チャネルのオーディオデータを、右チャネルのオーディオデータとしてコピーし、その左チャネルと右チャネルのオーディオデータ（「主音声」のオーディオデータ）を、オーディオ出力端子１２１に出力する。また、音声出力モードとして、「副音声」が指定されているときには、オーディオ出力モジュール２２１は、オーディオデコーダ制御モジュール２１７からのオーディオデータのうちの右チャネルのオーディオデータを、左チャネルのオーディオデータとしてコピーし、その左チャネルと右チャネルのオーディオデータ（「副音声」のオーディオデータ）を、オーディオ出力端子１２１に出力する。さらに、音声出力モードとして、「主・副」が指定されているときには、オーディオ出力モジュール２２１は、オーディオデコーダ制御モジュール２１７からのオーディオデータを、そのまま、オーディオ出力端子１２１に出力する。
【００８５】
なお、オーディオデコーダ制御モジュール２１７からのオーディオデータが、ステレオ(Stereo)モードのオーディオデータである場合、オーディオ出力モジュール２２１は、音声出力モードの指定にかかわらず、オーディオデコーダ制御モジュール２１７からのオーディオデータを、そのまま、オーディオ出力端子１２１に出力する。
【００８６】
ここで、音声出力モードの指定は、例えば、ビデオコンテンツ再生プログラム２１０が生成するメニューが表示された画面等において、ユーザがリモコン等を操作することにより対話的に行うことができる。
【００８７】
［バッファ制御モジュール２１５の構成］
次に、図５は、図２のバッファ制御モジュール２１５の構成例を示している。
【００８８】
バッファ制御モジュール２１５は、図１のメモリ１１３の一部を、バッファ２１５Ａとして排他的に使用し、そのバッファ２１５Ａに、ディスク１０１から読み出されたデータを一時記憶させる。また、バッファ制御モジュール２１５は、バッファ２１５Ａに記憶されたデータを読み出して、図２のビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、または字幕デコーダ制御モジュール２１８に供給する。
【００８９】
即ち、バッファ制御モジュール２１５は、バッファ２１５Ａの他、メモリ１１３の一部であるデータ先頭ポインタ記憶部２３１、およびデータ書き込みポインタ記憶部２３２を有するとともに、内部モジュールとして、ビデオ読み出し機能部２３３、オーディオ読み出し機能部２３４、字幕読み出し機能部２３５を有する。
【００９０】
バッファ２１５Ａは、例えば、リングバッファであり、ディスク１０１から読み出されたデータを順次記憶し、その記憶容量分のデータを記憶した後は、最も古いデータに上書きする形で、最新のデータを、いわば無限ループ状に記憶していく。
【００９１】
データ先頭ポインタ記憶部２３１は、バッファ２１５Ａに記憶されたデータのうち、まだ、バッファ２１５Ａから読み出されていない最も古いデータが記憶されている位置（ア

ドレス）を指すデータ先頭ポインタを記憶する。

【００９２】

データ書き込みポインタ記憶部２３２は、ディスク１０１から読み出された最新のデータが書き込まれるバッファ２１５Ａの位置（アドレス）を指す書き込みポインタを記憶する。

【００９３】

ここで、データ書き込みポインタが指す位置は、バッファ２１５Ａに、ディスク１０１から読み出されたデータが記憶されるごとに、図中、右回り（時計回り）に更新されていき、データ先頭ポインタが指す位置は、バッファ２１５Ａからのデータの読み出しに応じて、図中、右回りに更新されていく。したがって、バッファ２１５Ａに記憶されたデータのうち、いわば有効なデータは、データ先頭ポインタが指す位置から、右回りに、データ書き込みポインタが指す位置までに記憶されているデータである。

【００９４】

ビデオ読み出し機能部２３３は、図２のビデオデコーダ制御モジュール２１６からの要求に応じて、バッファ２１５Ａからビデオストリーム（ビデオデータに関するエレメンタリストリーム）を読み出し、ビデオデコーダ制御モジュール２１６に供給する。オーディオ読み出し機能部２３４も、図２のオーディオデコーダ制御モジュール２１７からの要求に応じて、バッファ２１５Ａからオーディオストリーム（オーディオデータに関するエレメンタリストリーム）を読み出し、オーディオデコーダ制御モジュール２１７に供給する。字幕読み出し機能部２３５も、図２の字幕デコーダ制御モジュール２１８からの要求に応じて、バッファ２１５Ａから字幕ストリーム（字幕データに関するエレメンタリストリーム）を読み出し、字幕デコーダ制御モジュール２１８に供給する。

【００９５】

即ち、光ディスク１０１には、例えば、MPEG(Moving Picture Experts Group)2の規格に準拠したプログラムストリーム（MPEG2-System Program Stream）が記録されており、バッファ２１５Ａには、光ディスク１０１から読み出されたプログラムストリームが記憶される。このプログラムストリームは、ビデオストリームや、オーディオストリーム、字幕ストリーム等の１以上のエレメンタリストリームが時分割多重されている。ビデオ読み出し機能部２３３は、プログラムストリームのデマルチプレクスの機能を有し、バッファ２１５Ａに記憶されたプログラムストリームから、ビデオストリームを分離して読み出す。

【００９６】

同様に、オーディオ読み出し機能部２３４も、プログラムストリームのデマルチプレクスの機能を有し、バッファ２１５Ａに記憶されたプログラムストリームから、オーディオストリームを分離して読み出す。字幕読み出し機能部２３５も、プログラムストリームのデマルチプレクスの機能を有し、バッファ２１５Ａに記憶されたプログラムストリームから、字幕ストリームを分離して読み出す。

【００９７】

ここで、ビデオ読み出し機能部２３３は、図１のメモリ１１３の一部であるビデオ読み出しポインタ記憶部２４１、stream_idレジスタ２４２、およびau_information()レジスタ２４３を有している。

【００９８】

ビデオ読み出しポインタ記憶部２４１は、バッファ２１５Ａの、ビデオストリームが記憶された位置（アドレス）を指すビデオ読み出しポインタを記憶し、ビデオ読み出し機能部２３３は、バッファ２１５Ａの、ビデオ読み出しポインタが指す位置に記憶されているデータを、ビデオストリームとして読み出す。stream_idレジスタ２４２は、バッファ２１５Ａに記憶されたプログラムストリームを解析し、そのプログラムストリームの中から読み出すビデオストリームを識別（特定）するための後述するstream_idを記憶する。au_information()レジスタ２４３は、バッファ２１５Ａからビデオストリームを読み出すために必要な（ビデオストリームの読み出しに利用される）データである後述するau_infor

mation()を記憶する。

【００９９】

オーディオ読み出し機能部２３４は、図１のメモリ１１３の一部であるオーディオ読み出しポインタ記憶部２５１、stream_idレジスタ２５２、およびprivate_stream_idレジスタ２５３を有している。

【０１００】

オーディオ読み出しポインタ記憶部２５１は、バッファ２１５Ａの、オーディオストリームが記憶された位置（アドレス）を指すオーディオ読み出しポインタを記憶し、オーディオ読み出し機能部２３４は、バッファ２１５Ａの、オーディオ読み出しポインタが指す位置に記憶されているデータを、オーディオストリームとして読み出す。stream_idレジスタ２５２とprivate_stream_idレジスタ２５３は、バッファ２１５Ａに記憶されたプログラムストリームを解析し、そのプログラムストリームの中から読み出すオーディオストリームを識別するための後述するstream_idとprivate_stream_idを、それぞれ記憶する。

【０１０１】

字幕読み出し機能部２３５は、図１のメモリ１１３の一部である字幕読み出し機能フラグ記憶部２６１、字幕読み出しポインタ記憶部２６２、stream_idレジスタ２６３、およびprivate_stream_idレジスタ２６４を有している。

【０１０２】

字幕読み出し機能フラグ記憶部２６１は、字幕読み出し機能フラグを記憶する。字幕読み出し機能フラグ記憶部２６１に記憶された字幕読み出し機能フラグが、例えば０である場合、字幕読み出し機能部２３５は機能動作せず、字幕読み出し機能フラグ記憶部２６１に記憶された字幕読み出し機能フラグが、例えば１である場合、字幕読み出し機能部２３５は機能する。

【０１０３】

字幕読み出しポインタ記憶部２６２は、バッファ２１５Ａの、字幕ストリームが記憶された位置（アドレス）を指す字幕読み出しポインタを記憶し、字幕読み出し機能部２３５は、バッファ２１５Ａの、字幕読み出しポインタが指す位置に記憶されているデータを、字幕ストリームとして読み出す。stream_idレジスタ２６３とprivate_stream_idレジスタ２６４は、バッファ２１５Ａに記憶されたプログラムストリームを解析し、そのプログラムストリームの中から読み出す字幕ストリームを識別するための後述するstream_idとprivate_stream_idを、それぞれ記憶する。

【０１０４】

［ディスク１０１に記録されたデータのデータフォーマットの説明］

次に、ディスク１０１に記録されたデータのデータフォーマットについて説明する。

【０１０５】

図６は、ディスク１０１のディレクトリ構造を模式的に示している。

【０１０６】

ディスク１０１のファイルシステムとしては、例えば、ISO(International Organization for Standardization)-9660や、UDF(Universal Disk Format:http://www.osta.org/specs/)などで規定されたファイルシステムが用いられており、ディスク１０１に記録されたデータのファイルはディレクトリ構造により階層的に管理されている。ここで、ファイルシステムは、上述したファイルシステムに限定されるものではない。

【０１０７】

図６では、ファイルシステムの基点を示すルート(root)ディレクトリに、”VIDEO”ディレクトリが置かれ、”VIDEO”ディレクトリには、”CLIP”ディレクトリと、”STREAM”ディレクトリとの２つのディレクトリが置かれている。

【０１０８】

”VIDEO”ディレクトリには、”CLIP”ディレクトリと”STREAM”ディレクトリとの２つのディレクトリの他に、”SCRIPT.DAT”ファイルと、”PLAYLIST.DAT”ファイルの２つのデータファイルが置かれている。

【０１０９】
”SCRIPT.DAT”ファイルは、スクリプトプログラムが記述されたスクリプトファイルである。即ち、”SCRIPT.DAT”ファイルには、ディスク１０１の再生形態をインタラクティブなものとするために使用するスクリプトプログラムが記述されている。この”SCRIPT.DAT”ファイルに記述されたスクリプトプログラムは、図２のスクリプト制御モジュール２１１によって解釈、実行される。
【０１１０】
”PLAYLIST.DAT”ファイルには、ディスク１０１に記録されたビデオデータ等のコンテンツの再生手順が記述されたプレイリスト（後述する図７のPlayList()）が１以上格納されている。
【０１１１】
”CLIP”ディレクトリには、１以上のクリップ情報ファイルが置かれ、”STREAM”ディレクトリには、１以上のクリップストリームファイルが置かれる。即ち、図６では、”CLIP”ディレクトリには、３つのクリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”が置かれており、”STREAM”ディレクトリには、３つのクリップストリームファイル”00001.PS”，”00002.PS”，”00003.PS”が置かれている。
【０１１２】
クリップストリームファイルには、ビデオデータ、オーディオデータ、字幕データなどの１以上のデータ（ストリーム）を圧縮、符号化して得られる１以上のエレメンタリストリームを時分割多重化したプログラムストリームが格納されている。
【０１１３】
クリップ情報ファイルには、対応するクリップストリームファイルの性質等の、クリップストリームに関する（ファイル）メタデータが記述されている。
【０１１４】
即ち、クリップストリームファイルとクリップ情報ファイルとは、１対１に対応している。図６では、クリップストリームファイルには、5文字の数字＋ピリオド＋”PS”という命名規則にしたがって、ファイル名が付されており、クリップ情報ファイルには、対応するクリップストリームファイルと同一の5文字の数字＋ピリオド＋”CLP”という命名規則にしたがって、ファイル名が付されている。
【０１１５】
従って、ファイルが、クリップストリームファイルまたはクリップ情報ファイルのうちのいずれであるかは、ファイル名の拡張子（ピリオドより右側の部分）によって識別することができ、さらに、対応するクリップストリームファイルとクリップ情報ファイルとは、ファイル名の拡張子以外の部分（ピリオドより左側の部分）が一致するかどうかによって識別することができる。
【０１１６】
以下、ディスク１０１に記録された各ファイルの詳細について説明する。
【０１１７】
「PLAYLIST.DAT」
図７は、図６の”VIDEO”ディレクトリ下の”PLAYLIST.DAT”ファイルの内部構造（シンタクス(syntax)）を示している。
【０１１８】
ここで、図７において、”Syntax”の欄の記載がデータ構造を表し、”No. of bits”の欄の記載は、対応する行の”Syntax”の欄のデータのビット長を表す。さらに、”Mnemonic”の欄の記載のうちの”bslbf”(bit string left bit first)は、対応する行の”Syntax”の欄のデータが左のビットから送り出されることを意味し、”uimsbf”(unsigned integer most significant bit first)は、対応する行の”Syntax”の欄のデータが、符号なし整数値であり、最上位ビットから送り出されることを意味する。以下説明する、図７と同様の図についても、同様である。
【０１１９】

”PLAYLIST.DAT”ファイルにおいては、その先頭から、その名称（ファイル名）等の情報を記述するためのname_length（8ビット）とname_string（255バイト）が順次配置される。

【０１２０】

即ち、name_lengthは、その後に配置されるname_stringのサイズを、バイト数で表す。name_stringは、”PLAYLIST.DAT”ファイルの名称（ファイル名）を表す。

【０１２１】

なお、name_stringについては、その先頭から、name_lengthで表されるバイト数までが有効な名称として使用される。たとえばname_lengthが値10である場合には、name_stringの先頭から10バイト分が有効な名称として解釈される。

【０１２２】

name_stringの後には、number_of_PlayLists（16ビット）が配置される。number_of_PlayListsは、続くPlayList()の個数を表す。number_of_PlayListsの後に、そのnumber_of_PlayListsの数だけのPlayList()が配置される。

【０１２３】

PlayList()は、ディスク１０１に記録されたクリップストリームファイルの再生手順が記述されたプレイリストであり、以下のような内部構造を有する。

【０１２４】

即ち、PlayList()の先頭には、PlayList_data_length（32ビット）が配置される。PlayList_data_lengthは、そのPlayList()のサイズを表す。

【０１２５】

PlayList_data_lengthの後には、reserved_for_word_alignment（15ビット）とcapture_enable_flag_PlayList（1ビット）が順次配置される。15ビットのreserved_for_word_alignmentは、その後に配置される１ビットのcapture_enable_flag_PlayListの位置で、いわゆるワードアライン(word alignment)をとるため（１６ビットの位置に揃えるため）に配置される。capture_enable_flag_PlayListは、PlayList()によって再生されるビデオストリームに対応するビデオデータ（PlayList()に属するビデオデータ）の、光ディスク１０１が再生される図１のディスク再生装置内での２次利用を許可するか否かを表す１ビットのフラグである。capture_enable_flag_PlayListが、０または１のうちの、例えば１である場合、PlayList()に属するビデオデータの２次利用が許可されていることを表し、capture_enable_flag_PlayListが、０または１のうちの、例えば０である場合、PlayList()に属するビデオデータの２次利用が許可されていない（禁止されている）ことを表す。

【０１２６】

なお、図７では、capture_enable_flag_PlayListを１ビットとしたが、その他、capture_enable_flag_PlayListは、複数ビットで構成し、PlayList()に属するビデオデータの２次利用を、いわば段階的に許可するようにすることが可能である。即ち、capture_enable_flag_PlayListは、例えば、２ビットで構成することができる。そして、capture_enable_flag_PlayListの値が00B（Bは、その前の数字が２進数であることを表す）である場合には、ビデオデータの２次利用を禁止し、capture_enable_flag_PlayListの値が01Bである場合には、ビデオデータを、64×64ピクセル以下のサイズに縮小して利用する２次利用のみを許可することができる。また、capture_enable_flag_PlayListの値が10Bである場合には、サイズの制限なしで、ビデオデータの２次利用を許可することができる。

【０１２７】

さらに、上述のように、ビデオデータの２次利用にあたって、サイズに制限を設けるのではなく、用途に制限を設けるようにすることも可能である。即ち、capture_enable_flag_PlayListの値が01Bである場合には、ビデオコンテンツ再生アプリケーション２１０（図２）のみでの２次利用を許可し、capture_enable_flag_PlayListの値が10Bである場合には、図１のディスク再生装置内の、ビデオコンテンツ再生アプリケーション２１０を含む任意のアプリケーションによる２次利用を許可することができる。ここで、図１のディスク再生装置内のビデオコンテンツ再生アプリケーション２１０以外のアプリケーション

としては、例えば、壁紙(バックグラウンド)やスクリーンセーバーの表示の処理を行うアプリケーションなどがある。

【０１２８】

なお、capture_enable_flag_PlayListを、上述のように、２ビットとした場合、その前に配置されるreserved_for_word_alignmentは、ワードアラインをとるために、１４ビットとなる。

【０１２９】

また、capture_enable_flag_PlayListにより、ビデオデータのディスク再生装置内での２次利用を許可する他、ディスク再生装置外での２次利用を許可するようにすることも可能である。ここで、ビデオデータの、ディスク再生装置外での２次利用を許可する場合には、ビデオデータは、例えば、ディスク再生装置に着脱可能な記録媒体やディスク再生装置に接続可能な他の装置に着脱可能な記録媒体に記録され、あるいはインターネット等のネットワークを介して、他の装置に送信（配信）される。この場合、ビデオデータには、そのビデオデータを記録媒体に記録する回数や配信する回数を制限する情報を付加するようにすることができる。

【０１３０】

capture_enable_flag_PlayListに続いては、PlayList_name_length（8ビット）とPlayList_name_string（255バイト）とが順次配置される。PlayList_name_lengthは、その後に配置されるPlayList_name_stringのサイズを、バイト数で表し、PlayList_name_stringは、PlayList()の名称を表す。

【０１３１】

PlayList_name_stringの後には、number_of_PlayItems（16ビット）が配置される。number_of_PlayItemsは、続くPlayItem()の個数を表す。

【０１３２】

number_of_PlayItemsの後には、そのnumber_of_PlayItemsの数だけのPlayItem()の構造が記述される。

【０１３３】

ここで、１つのPlayList()では、PlayItem()単位で、コンテンツの再生手順を記述することができる。

【０１３４】

また、PlayList()の中の、number_of_PlayItemsの数だけのPlayItem()それぞれに対しては、そのPlayList()の中でユニークなID(Identification)が付される。即ち、PlayList()中の最初のPlayItem()には、IDとして0番が付され、以下、続くPlayItem()に対して、その出現順に、1番、2番、・・・と通し番号のIDが付される。

【０１３５】

number_of_PlayItemsの数だけのPlayItem()の後には、１つのPlayListMark()が配置される。PlayListMark()は、PlayList()にしたがって行われる再生の時間軸上の印となる後述するMark()の集合で、その詳細については、図９を参照して後述する。

【０１３６】

「PlayItem()の説明」

次に、図８は、図７のPlayList()に含まれるPlayItem()の内部構造を示している。

【０１３７】

PlayItem()の先頭には、length（16ビット）が配置され、lengthは、それを含むPlayItem()のサイズを表す。

【０１３８】

lengthに続いては、Clip_Information_file_name_length（16ビット）とClip_Information_file_name（可変長）が順次配置される。Clip_Information_file_name_lengthは、その後に配置されるClip_Information_file_nameのサイズを、バイト数で表す。Clip_Information_file_nameは、PlayItem()によって再生するクリップストリームファイル（図６の拡張子がPSのファイル）に対応するクリップ情報ファイル（図６の拡張子がCLPのファイ

ル）のファイル名を表す。なお、クリップストリームファイルおよびクリップ情報ファイルのファイル名の、上述した命名規則により、Clip_Information_file_nameから、PlayItem()によって再生するクリップ情報ファイルのファイル名を認識し、そのクリップストリームファイルを特定することができる。

【０１３９】

Clip_Information_file_nameに続いては、IN_time（32ビット）とOUT_time（32ビット）が順次配置される。

【０１４０】

IN_timeとOUT_timeは、それぞれ、Clip_Information_file_nameから特定されるクリップストリームファイルの再生開始位置と再生終了位置を指定する時刻情報である。

【０１４１】

IN_timeによれば、クリップストリームファイルの（先頭を含む）途中の位置を再生開始位置として指定することができ、OUT_timeによれば、クリップストリームファイルの（最後を含む）途中の位置を再生終了位置として指定することができる。

【０１４２】

ここで、PlayItem()によれば、Clip_Information_file_nameから特定されるクリップストリームファイルの、IN_timeからOUT_timeまでの間のコンテンツが再生される。このPlayItem()によって再生されるコンテンツを、以下、適宜、クリップという。

【０１４３】

「PlayListMark()の説明」

次に、図９は、図７のPlayList()に含まれるPlayListMark()の内部構造を示している。

【０１４４】

PlayListMark()は、上述したように、そのPlayListMark()を含むPlayList()（図７）にしたがって行われる再生の時間軸上の印となる、０以上のMark()の集合である。１つのMark()は、PlayList()にしたがって行われる再生の時間軸上の１つの時刻（位置）を表す時刻情報、Mark()のタイプを表すタイプ情報、およびタイプ情報がイベントを発生させるタイプを表しているときの、そのイベントの引数となる引数情報を、少なくとも有する。

【０１４５】

即ち、PlayListMark()の先頭には、length（32ビット）が配置される。lengthは、それを含むPlayListMark()のサイズを表す。

【０１４６】

lengthの後には、number_of_PlayList_marks（16ビット）が配置され、number_of_PlayList_marksは、それに続いて配置されるMark()の個数を表す。number_of_PlayList_marksの後には、そのnumber_of_PlayList_marksの数だけMark()の構造が記述される。

【０１４７】

Mark()の先頭には、mark_type（8ビット）が配置される。mark_typeは、上述のタイプ情報であり、それを含むMark()のタイプを表す。

【０１４８】

本実施の形態では、Mark()のタイプとして、例えば、チャプタ(Chapter)、インデクス(Index)、イベント(Event)の３種類が用意されている。

【０１４９】

タイプがチャプタのMark()（以下、適宜、チャプタマークという）は、PlayList()を分割する頭出しの単位であるチャプタの先頭位置の印である。また、タイプがインデクスのMark()（以下、適宜、インデクスマークという）は、チャプタを細分化した単位であるインデクスの先頭位置の印である。タイプがイベントのMark()（以下、適宜、イベントマークという）は、PlayList()にしたがったコンテンツの再生中においてイベントを発生させる位置の印である。イベントマークによるイベントの発生は、後述するように、スクリプト制御モジュール２１１に通知される。

【０１５０】

ここで、mark_typeの値と、Mark()のタイプとの関係を、図１０に示す。図１０によれ

ば、チャプタマークのmark_typeには、１がセットされる。また、インデクスマークのmark_typeには、２がセットされ、イベントマークのmark_typeには、３がセットされる。なお、図１０では、mark_typeで表される８ビットの値のうちの、０と、４乃至２５５は、将来の拡張のための予約(reserved)とされている。

【０１５１】

図９に戻り、mark_typeの後には、mark_name_length（8ビット）が配置される。また、Mark()の最後には、mark_name_string（24バイト）が配置される。mark_name_lengthとmark_name_stringは、Mark()の名称を記述するためのものであり、mark_name_lengthは、mark_name_stringの有効なサイズを、mark_name_stringは、Mark()の名称を、それぞれ表す。従って、mark_name_stringの先頭からmark_name_lengthが表すバイト数までが、Mark()の有効な名称を表す。

【０１５２】

mark_name_lengthに続いては、PlayList()上で定義されるMark()をクリップストリームファイルと対応付ける４つの要素ref_to_PlayItem_id（16ビット）、mark_time_stamp（32ビット）、entry_ES_stream_id（8ビット）、entry_ES_private_stream_id（8ビット）が順次配置される。

【０１５３】

ref_to_PlayItem_idには、Mark()が属するPlayItem()に対して通し番号で付されたIDが記述される。ref_to_PlayItem_idによって、Mark（）が属するPlayItem()（図８）が特定され、ひいては、図８で説明したように、クリップ情報ファイルとクリップストリームファイルが特定される。

【０１５４】

mark_time_stampは、ref_to_PlayItem_idによって特定されるクリップストリームファイル内でのMark()が表す位置（時刻）を表す。

【０１５５】

ここで、図１１は、PlayList(), PlayItem()、クリップ、およびクリップストリームファイルに格納されたプログラムストリームの関係を示している。

【０１５６】

図１１では、PlayList()は、３つのPlayItem()から構成されており、その３つのPlayItem()それぞれには、通し番号で付されるID#0, #1, #2が付されている。ここで、以下、適宜、ID#iが付されたPlayItem()を、PlayItem#iと記述する。

【０１５７】

また、図１１では、PlayItem#0, PlayItem#1, PlayItem#2によって再生されるコンテンツであるクリップが、それぞれ、クリップＡ、クリップＢ、クリップＣとして示されている。

【０１５８】

クリップの実体は、図８のPlayItem()におけるClip_Information_file_nameから特定される（クリップ情報ファイルから、さらに特定される）クリップストリームファイルに格納されたプログラムストリームのうちの、IN_timeからOUT_timeまでのプログラムストリームである。図１１では、クリップＡ、クリップＢ、クリップＣの実体としてのプログラムストリームが、プログラムストリームＡ、プログラムストリームＢ、プログラムストリームＣとして、それぞれ示されている。

【０１５９】

例えば、図１１において、PlayList()にしたがって行われる再生の時間軸上の位置（時刻）t0の印となるMark()においては、そのref_to_PlayItem_idとmark_time_stampは、次のように記述される。

【０１６０】

即ち、時刻t0は、PlayItem#1の再生が行われる時刻であるため、ref_to_PlayItem_idには、そのPlayItem#1のIDである１が記述される。さらに、時刻t0では、PlayItem#1によって、クリップＢの実体であるプログラムストリームＢが再生されるため、mark_time_stam

pには、プログラムストリームＢが格納されたクリップストリームファイルにおける時刻t0に相当する時刻が記述される。

【０１６１】

再び、図９に戻り、entry_ES_stream_idと、entry_ES_private_stream_idは、Mark()を、特定のエレメンタリストリームに関連付ける場合に、そのエレメンタリストリームを特定するために使用される。即ち、entry_ES_stream_idには、Mark()を関連付けるエレメンタリストリーム（が配置される、後述する図１６乃至図１８に示すPES_packet()）の後述するstream_idが記述される。また、entry_ES_private_stream_idには、必要に応じて、Mark()を関連付けるエレメンタリストリーム（が配置される、後述する図２１に示すprivate_stream1_PES_payload()におけるprivate_header()）の後述するprivate_stream_idが記述される。

【０１６２】

例えば、ビデオストリーム＃１とビデオストリーム＃２が多重化されているクリップにおいて、ビデオストリーム＃１を再生している場合と、ビデオストリーム＃２を再生している場合でチャプタの発生時刻を変更したいときには、ビデオストリーム＃１再生時のチャプタマーク発生時刻のMark()のentry_ES_stream_idとentry_ES_private_stream_idに、ビデオストリーム＃１のstream_idとprivate_stream_idが記述され、また、ビデオストリーム＃２再生時のチャプタマーク発生時刻のMark（）のentry_ES_stream_idとentry_ES_private_stream_idに、ビデオストリーム＃２のstream_idとprivate_stream_idが記述される。

【０１６３】

なお、特定のエレメンタリストリームに関連付けないMark()のentry_ES_stream_idと、entry_ES_private_stream_idには、例えば、いずれも０が記述される。

【０１６４】

entry_ES_private_stream_idの後には、mark_data（３２ビット）が配置される。mark_dataは、Mark()がイベントマークである場合に、そのイベントマークによって発生されるイベントの引数となる引数情報である。なお、mark_dataは、Mark()がチャプタマークやインデクスマークである場合に、そのチャプタマークやインデクスマークが表すチャプタやインデクスの番号として使用することも可能である。

【０１６５】

「Clip()の説明」

次に、図６の”CLIP”ディレクトリに置かれる、拡張子がCLPのクリップ情報ファイルの内部構造について説明する。

【０１６６】

図６では、”CLIP”ディレクトリに、３つのクリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”が置かれており、それぞれには、”STREAM”ディレクトリに置かれたクリップストリームファイル”00001.PS”，”00002.PS”，”00003.PS”の性質等を示すメタデータが格納されている。

【０１６７】

図１２は、そのようなクリップ情報ファイルClip()の内部構造を示している。

【０１６８】

クリップ情報ファイルClip()の先頭には、presentation_start_timeとpresentation_end_time（いずれも３２ビット）が、順次配置される。presentation_start_timeとpresentation_end_timeは、クリップ情報ファイルClip()に対応するクリップストリームファイル（に格納されているプログラムストリーム）の先頭と最後の時刻を表す。なお、クリップストリームファイルの時刻は、MPEG2-Systemの時刻で使われている90kHzの倍数で記述される。

【０１６９】

presentation_end_timeに続いては、reserved_for_word_alignment（7ビット）とcapture_enable_flag_Clip（1ビット）が順次配置される。7ビットのreserved_for_word_align

mentは、ワードアラインをとるためのもので、capture_enable_flag_Clipは、上述した図７のcapture_enable_flag_PlayListと同様に、ビデオデータの２次利用を許可するか否かを表すフラグである。

【０１７０】

但し、図７のcapture_enable_flag_PlayListは、PlayList()によって再生されるビデオストリームに対応するビデオデータ（PlayList()に属するビデオデータ）の２次利用を許可するか否かを表すのに対して、図１２のcapture_enable_flag_Clipは、クリップ情報ファイルClip()に対応するクリップストリームファイルに格納されているビデオストリーム（ビデオのエレメンタリストリーム）に対応するビデオデータの２次利用を許可するか否かを表す。従って、図７のcapture_enable_flag_PlayListと、図１２のcapture_enable_flag_Clipとでは、２次利用を許可するビデオデータの単位（範囲）が異なる。

【０１７１】

なお、図１２のcapture_enable_flag_Clipも、図７のcapture_enable_flag_PlayListで説明したように、１ビットではなく、複数ビットとすることが可能である。

【０１７２】

capture_enable_flag_Clipの後には、number_of_streams（8ビット）が配置され、このnumber_of_streamsには、それに続くStreamInfo()構造の個数が記述される。従って、number_of_streamsに続いては、そのnumber_of_Streamsの数だけ、StreamInfo()の構造が記述される。

【０１７３】

StreamInfo()の先頭には、length（16ビット）が配置され、このlengthは、それを含むStreamInfo()のサイズを表す。lengthに続いては、stream_id（8ビット）とprivate_stream_id（8ビット）が配置されており、このstream_idとprivate_stream_idによって、StreamInfo()に関連付けるエレメンタリストリームが特定（識別）される。

【０１７４】

ここで、図１３は、エレメンタリストリームを識別するstream_idおよびprivate_stream_idと、エレメンタリストリームとの関係を示している。

【０１７５】

stream_idは、MPEG2-System規格において規定されているのと同一のものであり、その値は、MPEG2-System規格において、エレメンタリストリーム（データ）の属性（種類）ごとに、あらかじめ決められている。従って、MPEG2-System規格で定義されている属性のエレメンタリストリームは、stream_idだけで特定することができる。

【０１７６】

本実施の形態では、MPEG2-System規格において規定されていない属性のエレメンタリストリームも扱うことが可能になっており、private_stream_idは、MPEG2-System規格において規定されていない属性のエレメンタリストリームを識別するための情報である。

【０１７７】

図１３では、MPEGで規定されている符号化（復号）方式でエンコードされたビデオのエレメンタリストリーム、ATRAC(Adaptive TRansform Acoustic Coding)方式でエンコードされたオーディオのエレメンタリストリーム（以下、適宜、ATRACオーディオストリームという）、LPCM(Linear Pulse Code Modulation)方式でエンコードされたオーディオのエレメンタリストリーム（以下、適宜、LPCMオーディオストリームという）、字幕のエレメンタリストリーム（以下、適宜、字幕ストリームという）の４つの属性のエレメンタリストリームについて、stream_idおよびprivate_stream_idとの関係を示している。

【０１７８】

MPEG2-System規格では、MPEGで規定されている符号化方式でエンコードされたビデオのエレメンタリストリームは、0xE0乃至0xEFの範囲の値を（0xは、その後に続く文字列が１６進数であることを表す）、エレメンタリストリームを識別するstream_idとして用いて多重化することが規定されている。従って、MPEGで規定されている符号化方式でエンコードされたビデオのエレメンタリストリームについては、0xE0乃至0xEFの範囲の値のstream

_idで識別することができる16本のビデオのエレメンタリストリームを、プログラムストリームに多重化することができる。

【０１７９】

なお、MPEGで規定されている符号化方式でエンコードされたビデオのエレメンタリストリームの識別は、0xE0乃至0xEFの範囲の値のstream_idで行うことができるので、private_stream_idは不要である（無視することができる）。

【０１８０】

一方、MPEG2-Systemでは、ATRACオーディオストリーム、LPCMオーディオストリーム、字幕ストリームについては、stream_idは定義されていない。

【０１８１】

そこで、本実施の形態では、MPEG2-Systemでstream_idが定義されていないエレメンタリストリームについては、そのstream_idに、MPEG2-Systemにおいてprivate_stream_1という属性を表す値である0xBDを採用し、さらに、図１３に示すように、private_stream_idを用いて識別（特定）を行うこととしている。

【０１８２】

即ち、ATRACオーディオストリームの識別には、0x00乃至0x0Fの範囲の値のprivate_stream_idが使用される。従って、プログラムストリームには、16本のATRACオーディオストリームを多重化することができる。また、LPCMオーディオストリームの識別には、0x10乃至0x1Fの範囲の値private_stream_idが使用される。従って、プログラムストリームには、16本のLPCMオーディオストリームを多重化することができる。さらに、字幕ストリームの識別には、0x80乃至0x9Fの範囲の値のprivate_stream_idが使用される。従って、プログラムストリームには、32本の字幕ストリームを多重化することができる。

【０１８３】

なお、stream_idおよびprivate_stream_idについては、さらに後述する。

【０１８４】

図１２に戻り、private_stream_idの後には、StaticInfo(), reserved_for_word_alignment（8ビット）が順次配置される。StaticInfo()には、（そのStaticInfo()を含むStreamInfo()に記述された）stream_idおよびprivate_stream_idによって特定されるエレメンタリストリームの再生中に変化しない情報が記述される。StaticInfo()の詳細については、図１４を参照して後述する。

【０１８５】

reserved_for_word_alignmentは、ワードアラインをとるために使用される。

【０１８６】

reserved_for_word_alignmentに続いては、number_of_DynamicInfo（8ビット）が配置され、number_of_DynamicInfoは、その後に続いて配置されるpts_change_point（32ビット）とDynamicInfo()のセットの数を表す。

【０１８７】

従って、number_of_DynamicInfoに続いては、そのnumber_of_DynamicInfoの数だけのセット数のpts_change_pointとDynamicInfo()の構造が記述される。

【０１８８】

pts_change_pointは、それとセットになっているDynamicInfo()の情報が有効になる時刻を表す。ここで、エレメンタリストリームの先頭の時刻を表すpts_change_pointは、そのエレメンタリストリームが格納されたクリップストリームファイルに対応するクリップ情報ファイルClip()の最初に記述されるpresentation_start_timeに等しい。

【０１８９】

DynamicInfo()には、stream_idおよびprivate_stream_idによって特定されるエレメンタリストリームの再生中に変化する、いわば動的な情報が記述される。DynamicInfo()に記述された情報は、それとセットになっているpts_change_pointが表す再生時刻となったときに有効になる。なお、DynamicInfo()の詳細については、図１５を参照して後述する。

【０１９０】
number_of_DynamicInfoの数だけのセットのpts_change_pointとDynamicInfo()の後には、EP_map()が配置される。なお、EP_map()については、図１６を参照して後述する。
【０１９１】
「StaticInfo()の説明」
次に、図１４を参照して、図１２のStaticInfo()の詳細について説明する。
【０１９２】
図１４は、StaticInfo()のシンタクスを示している。
【０１９３】
StaticInfo()は、対応するエレメンタリストリームの属性（種類）により内容が異なっている。StaticInfo()に対応するエレメンタリストリームの属性は、そのStaticInfo()を含む図１２のStreamInfo()に含まれるstream_idとprivate_stream_idにより判断される。
【０１９４】
StaticInfo()に対応するエレメンタリストリームがビデオストリームである場合(stream==VIDEO)、StaticInfo()は、picture_size（4ビット），frame_rate（4ビット），およびcc_flag（1ビット）と、ワードアラインをとるためのreserved_for_word_alingmentとで構成される。
【０１９５】
picture_sizeは、ビデオストリームに対応するビデオデータ（によって表示される画像）の大きさを表す。frame_rateは、ビデオストリームに対応するビデオデータのフレーム周波数を表す。cc_flagは、ビデオストリームにクローズドキャプション(Closed Caption)データが含まれているか否かを表す。即ち、例えば、ビデオストリームにクローズドキャプションデータが含まれている場合には、cc_flagは１とされ、ビデオストリームにクローズドキャプションデータが含まれていない場合には、cc_flagは０とされる。
【０１９６】
StaticInfo()に対応するエレメンタリストリームがオーディオストリームである場合(stream==AUDIO)、StaticInfo()は、audio_language_code（16ビット），channel_configuration（8ビット），lfe_existence（1ビット）、およびsampling_frequency（4ビット）と、ワードアラインをとるためのreserved_for_word_alingmentとで構成される。
【０１９７】
audio_language_codeには、オーディオストリームに含まれているオーディオデータの言語を表すコードが記述される。channel_configurationは、モノラル(mono)/ステレオ(stereo)/マルチチャネル等の、オーディオストリームに含まれているオーディオデータの属性を表す。lfe_existenceは、オーディオストリームに低域強調チャネルが含まれているかどうかを表し、含まれていれば１となり、含まれていなければ０となる。sampling_frequencyは、オーディオストリームに含まれているオーディオデータのサンプリング周波数を示す情報である。
【０１９８】
StaticInfo()に対応するエレメンタリストリームが字幕ストリームである場合(stream==SUBTITLE)、StaticInfo()は、subtitle_language_code（16ビット）およびconfigurable_flag（1ビット）と、ワードアラインをとるためのreserved_for_word_alingmentとで構成される。
【０１９９】
subtitle_language_codeには、字幕ストリームに含まれている字幕データの言語を表すコードが記述される。configurable_flagは、字幕ストリームに含まれている字幕データの表示をデフォルトの表示方式から変更することを許可するか否かを表す情報で、例えば、表示方式の変更が許可されている場合には１が記述され、許可されていない場合には０が記述される。なお、字幕データの表示方式としては、字幕データの表示サイズや、表示位置、表示色、表示パターン（例えば、点滅表示など）、表示方向（例えば、垂直方向や水平方向）などがある。

【０２００】
「DynamicInfo()の説明」
次に、図１５を参照して、図１２のDynamicInfo()の詳細について説明する。
【０２０１】
図１５は、DynamicInfo()のシンタクスを示している。
【０２０２】
DynamicInfo()の先頭には、ワードアラインのためのreserved_for_word_alignment（8ビット）が配置されており、その後に続く要素は、DynamicInfo()に対応するエレメンタリストリームの属性により内容が異なっている。DynamicInfo()に対応するエレメンタリストリームの属性は、図１４で説明したStaticInfo()における場合と同様に、DynamicInfo()を含む図１２のStreamInfo()に含まれるstream_idとprivate_stream_idにより判断される。
【０２０３】
DynamicInfo()には、図１２で説明したように、エレメンタリストリームの再生中に変化する動的な情報が記述される。この動的な情報は、特に限定されるものではないが、図１５の実施の形態では、DynamicInfo()に対応するエレメンタリストリームに対応するデータ、即ち、エレメンタリストリームが処理されることによって出力されるデータの出力属性（エレメンタリストリームから得られるデータの出力属性）が、DynamicInfo()に記述される。
【０２０４】
具体的には、DynamicInfo()に対応するエレメンタリストリームがビデオストリームである場合(stream==VIDEO)、DynamicInfo()は、display_aspect_ratio（4ビット）と、ワードアラインのためのreserved_for_word_alingmentとで構成される。diplay_aspect_ratioには、ビデオストリームに対応するビデオデータの出力属性（表示方式）としての、例えば、そのビデオデータのアスペクト比が記述される。即ち、diplay_aspect_ratioには、例えば、アスペクト比が、16:9または4:3のうちのいずれであるかを表す情報が記述される。なお、ビデオストリームのDynamicInfo()には、アスペクト比の他、例えば、ビデオデータによって表示される画像のサイズ（Ｘ画素×Ｙ画素）などを記述することが可能である。
【０２０５】
DynamicInfo()に対応するエレメンタリストリームがオーディオストリームである場合(stream==AUDIO)、DynamicInfo()は、channel_assignment（4ビット）と、ワードアラインをとるためのreserved_for_word_alingmentとで構成される。channel_assignmentには、オーディオストリームに2チャネルのオーディオデータが含まれている場合に、その２チャネルの出力属性（出力方式）が記述される。即ち、channel_assignmentには、オーディオデータが、ステレオまたはデュアル（二ヶ国語）のうちのいずれのチャネル割り当てがされているものであるかを表す情報が記述される。
【０２０６】
DynamicInfo()に対応するエレメンタリストリームが字幕ストリームである場合(stream==SUBTITLE)、DynamicInfo()は、ワードアラインをとるためのreserved_for_word_alingmentで構成される。即ち、図１５の実施の形態では、字幕ストリームに関しては、動的な情報としての出力属性は定義されていない。
【０２０７】
「EP_map()の説明」
次に、図１６を参照して、図１２のEP_map()の詳細について説明する。
【０２０８】
図１６は、EP_map()のシンタクスを示している。
【０２０９】
EP_map()には、そのEP_map()を含む図１２のクリップ情報ファイルClip()に対応するクリップストリームファイルに格納されたプログラムストリームに多重化されているエレメ

ンタリストリーム毎に、各エレメンタリストリームの、デコードを開始することができるデコード開始可能点（エントリポイント）の情報が記述される。

【０２１０】

ここで、固定レートのストリームについては、デコード開始可能点は、計算によって求めることができるが、可変レートのストリーム、あるいはMPEG規格にしたがって符号化されたビデオストリームのように、ビデオアクセスアクセスユニットごとにサイズが異なるストリームについては、デコード開始可能点は、計算によって求めることができず、実際にストリームを解析しないと見つけることができない。デコード開始可能点を迅速に認識することは、ランダムアクセスを行うために重要であり、EP_map()によれば、デコード開始可能点を迅速に認識することができる。

【０２１１】

なお、MPEG2-Videoでは、Sequence_header()（シーケンスヘッダ）等を含めたイントラピクチャの先頭が、デコード開始可能点である。

【０２１２】

EP_map()の先頭には、ワードアラインのためのreserved_for_word_alignment（8ビット）が配置されており、続いてnumber_of_stream_id_entries（8ビット）が配置されている。number_of_stream_id_entriesは、EP_map()にデコード開始可能点の情報が記述されているエレメンタリストリームの本数を表す。

【０２１３】

number_of_stream_id_entriesの後には、エレメンタリストリームを識別するための情報と、そのエレメンタリストリームのデコード開始可能点の情報とが、number_of_stream_id_entriesが表す数だけ繰り返し配置される。

【０２１４】

即ち、number_of_stream_id_entriesの直後には、エレメンタリストリームを識別する情報としてのstream_id（8ビット）とprivate_stream_id（8ビット）が配置され、それに続けて、number_of_EP_entries（32ビット）が配置される。number_of_EP_entriesは、その直前のstream_idとprivate_stream_idで識別（特定）されるエレメンタリストリームのデコード開始可能点の数を表す。

【０２１５】

number_of_EP_entriesの後には、その直前のstream_idとprivate_stream_idで特定されるエレメンタリストリームのデコード開始可能点の情報がnumber_of_EP_entriesが表す数だけ繰り返し配置される。

【０２１６】

すなわち、該当エレメンタリストリームがビデオストリームである場合には、まずindex_N_minus1とN-th_Ref_picture_copyが置かれ、その後ろにデコード開始可能点の情報としてのPTS_EP_start（32ビット）とRPN_EP_start（32ビット）が置かれ、これらが繰り返し配置される。index_N_minus1とN-th_Ref_picture_copyについては後述する。

【０２１７】

また、該当エレメンタリストリームがビデオストリーム以外である場合には、16ビットのreserved_for_future_useの後にデコード開始可能点の情報としてのPTS_EP_start（32ビット）とRPN_EP_start（32ビット）が置かれ、これらが繰り返し配置される。

【０２１８】

N-th_Ref_picture_copyには、後述するRPN_EP_startで示されるprivate_stream_2に記録されている、1st_Ref_picture、2ndRef_picture、3rdRef_picture、4thRef_pictureのいずれかがコピーされる。またindex_N_minus1にはどのフィールドをコピーしたかという情報が図１７に示す値を用いて記録される。すなわち、図１７で示されるように、index_N_minus1には、1st_Ref_pictureがコピーされた場合、０が、2ndRef_pictureがコピーされた場合、１が、3rdRef_pictureがコピーされた場合、２が、4thRef_pictureがコピーされた場合、４が、記録されることになる。

【０２１９】

デコード開始可能点の情報の１つであるPTS_EP_startは、上述のようにstream_idとprivate_stream_idで特定されるエレメンタリストリームが多重化されているプログラムストリームが格納されたクリップストリームファイル内での、デコード開始可能点の時刻（再生時刻）を表す。

【０２２０】

デコード開始可能点の情報の他の１つであるRPN_EP_startには、上述のようにstream_idとprivate_stream_idで特定されるエレメンタリストリームが多重化されているプログラムストリームが格納されたクリップストリームファイル内での、デコード開始可能点の位置を、プログラムストリームのpack()単位で数えたときの値が記述される。なお、本実施の形態では、pack()のサイズは2048バイトで固定であるとする。また、本実施の形態では、ディスク１０１（図１）の１セクタが、2048バイトであるとする。

【０２２１】

ここで、ビデオストリームについては、そのデコード開始可能点（エントリポイント）の直前に、後述するprivate_stream_2パケット（private_stream_2の属性のPES_packet()）が配置されている。private_stream_2パケットは、そのprivate_stream_2パケットから、次のprivate_stream_2パケットまでの間に配置されているビデオストリームをデコードするのに利用される情報が格納されている。このため、ビデオストリームについては、デコード開始可能点の情報としてのRPN_EP_startには、実際のデコード開始可能点そのものではなく、実際のデコード開始可能点の直前に配置されているprivate_stream_2パケットの先頭の位置が記述される。

【０２２２】

また、EP_map()において、デコード開始可能点の情報としてのPTS_EP_startとRPN_EP_startとのセットは、stream_idとprivate_stream_idで特定されるエレメンタリストリームごとに、あらかじめ、昇順にソートされている。これにより、デコード開始可能点の情報としてのPTS_EP_startとRPN_EP_startとのセットは、二分探索が可能となっている。

【０２２３】

なお、可変レートのストリームや、ビデオアクセスアクセスユニットごとにサイズが異なるストリームを対象としたランダムアクセスの方法は、例えば、特開2000-341640号公報（特願平11-317738号）などに記載されている。

【０２２４】

「クリップストリームファイルの説明」

次に、図６の”STREAM”ディレクトリに置かれる、拡張子がPSのクリップストリームファイル（図６では、”00001.PS”，”00002.PS”，”00003.PS”）の内部構造について説明する。

【０２２５】

クリップストリームファイルは、MPEG-2 System (ISO/IEC 13818-1)に定義されたMPEG2_Program_Stream()をベースに構成されている。

【０２２６】

即ち、図１８は、MPEG-2 System(ISO/IEC 13818-1:2000)規格に記述されているTable2-31, Table2-32, Table2-33を示している。

【０２２７】

クリップストリームファイルに格納されたプログラムストリームは、MPEG-2 System規格のTable2-31に定義されているMPEG2_Program_Stream()であり、１つ以上のpack()と、１つのMPEG_program_end_codeで構成される。なお、MPEG2_Program_Stream()の説明は、特許第2785220号などにも記載されている。

【０２２８】

１つのpack()は、MPEG-2 System規格のTable2-32に定義されているように、１つのPack_header()と、任意の数のPES_packet()とで構成される。Pack_header()の詳細は、MPEG-2 System規格のTable2-33に定義されている。

【０２２９】

ここで、MPEG-2 System規格では、pack()のサイズは、可変長として定義されているが、ここでは、図１６で説明したように、2048バイトで固定であるとする。さらに、ここでは、１つのpack()のPES_packet()の数は、１つ、２つ、または３つとする。Pack()が後述するprivate_stream_2パケットで始まる場合、その直後（同じPack()内）に対応するビデオストリームのPES_packet()が必ず存在する。またこれに加えて３つ目のPES_packet()としてpadding_packet（パディングパケット）を置くことができる。なおprivate_stream_2パケットは必ずPack()の先頭におかれる。

【０２３０】

Pack()がprivate_stream_2パケットで始まらない場合には、Pack()の先頭にはビデオ、オーディオ、字幕などのコンテンツデータの格納されたPES_packet()が置かれる。これに加えて２つ目のPES_packet()としてpadding_packet（パディングパケット）を置くことができる。

【０２３１】

図１９乃至図２１は、MPEG-2 System規格のTable2-17で定義されているPES_packet()を示している。

【０２３２】

PES_packet()は、packet_start_code_prefix, stream_id、およびPES_packet_length（図１９）と、stream_id等により構造の変化するヘッダ部分（stuffing_byteを含む）（図１９乃至図２１）と、PES_packet_data_byte（図２１）とに大別することができる。なお、PES_packet()が、padding_packetである場合(stream_id==padding_stream)、PES_packet_data_byteに代えて、padding_byte(0xFF)（図２１）が必要な数だけ繰り返し配置される。

【０２３３】

ここで、PES_packet()のヘッダ部分には、図１９および図２０に示すように、PTS(Presentation Time Stamp)と呼ばれる表示タイミングを示す情報と、DTS(Decoding Time Stamp)と呼ばれるデコードタイミングを示す情報とを配置することができる。本実施の形態では、すべてのアクセスユニット（MPEG2-Systemで定義された、エレメンタリストリームを構成するデコード単位）に対してPTSが付加され、MPEG2-Systemに定める場合にDTSが付加されるとする。

【０２３４】

プログラムストリームに多重化されるエレメンタリストリームは、PES_packet()のPES_packet_data_byte（図２１）に格納される。そして、PES_packet()のstream_idには、そのPES_packet_data_byteに格納されたエレメンタリストリームを識別するために、そのエレメンタリストリームの属性に応じた値が記述される。

【０２３５】

PES_packet()のstream_idに記述される値と、エレメンタリストリームの属性（種類）との関係は、MPEG-2 System規格のTable 2-18に定義されている。ここで、図２２に、MPEG-2 System規格のTable 2-18を示す。

【０２３６】

本実施の形態では、図２２に示したMPEG-2 System規格で定義されているstream_idのうちの、例えば、図２３に示す値を採用する。

【０２３７】

即ち、本実施の形態では、10111101B, 10111110B, 10111111B, 110xxxxxB, 1110xxxxBの５パターンを、stream_idの値として採用する。なお、”x”は、０または１のうちの任意の値を表す。

【０２３８】

そして、private_stream_1と呼ばれる属性のエレメンタリストリームのPES_packet()のstream_idは、図２３にしたがい、10111101Bとされる。また、padding_packetのPES_packet()のstream_idは、図２３にしたがい、10111110Bとされる。さらに、private_stream_2と呼ばれる属性のエレメンタリストリームのPES_packet()のstream_idは、図２３にした

がい、10111111Bとされる。

【０２３９】

また、MPEGで定義されたオーディオストリーム（オーディオのエレメンタリストリーム）のPES_packet()のstream_idは、110xxxxxBとされる。なお、110xxxxxBのうちの下位５ビットxxxxxは、オーディオストリームを区別するオーディオストリームナンバであり、プログラムストリーム（MPEG2_Program_Stream()）には、このオーディオストリームナンバで区別することのできる数である３２（＝２5）本のオーディオストリーム（MPEGで定義されたオーディオストリーム）を多重化することができる。

【０２４０】

さらに、MPEGで定義されたビデオストリーム（ビデオのエレメンタリストリーム）のPES_packet()のstream_idは、1110xxxxBとされる。なお、1110xxxxBのうちの下位４ビットxxxxは、ビデオストリームを区別するビデオストリームナンバであり、プログラムストリームには、このビデオストリームナンバで区別することのできる数である１６（＝２4）本のビデオストリーム（MPEGで定義されたビデオストリーム）を多重化することができる。

【０２４１】

ところで、stream_idが1110xxxxBのPES_packet()は、MPEGで定義されたビデオストリームを格納するために使用され、stream_idが110xxxxxBのPES_packet()は、MPEGで定義されたオーディオストリームを格納するために使用される。一方、MPEGで定義されていない符号化方式（たとえばATRAC方式）のエレメンタリストリームを格納するのに使用するPES_packet()のstream_idは、MPEGでは規定されておらず、従って、MPEGで定義されていない符号化方式のエレメンタリストリームは、MPEGで定義されたビデオストリームやオーディオストリームと同様に、単純に、stream_idを指定して、PES_packet()に格納することはできない。

【０２４２】

そこで、本実施の形態では、private_stream_1のPES_packet()のPES_packet_data_byteを拡張し、MPEGで定義されていない符号化方式のエレメンタリストリームを格納する。

【０２４３】

ここで、private_stream_1のPES_packet()の、拡張したPES_packet_data_byteを、private_stream1_PES_payload()と記述する。

【０２４４】

「private_stream1_PES_payload()の説明」

図２４は、private_stream1_PES_payload()のシンタクスを示している。

【０２４５】

private_stream1_PES_payload()は、private_header()とprivate_payload()とで構成される。private_payload()には、ATRACオーディオストリームや、LPCMオーディオストリーム、字幕ストリームなどの、MPEGで定義されていない符号化方式のエレメンタリストリームが格納される。

【０２４６】

private_header()の先頭には、private_stream_id（8ビット）が配置される。private_stream_idは、private_payload()に格納されるエレメンタリストリームを識別する識別情報で、その属性（種類）に応じて、例えば、以下のような値とされる。

【０２４７】

即ち、図２５は、private_stream_idの値と、private_payload()に格納されるエレメンタリストリームの属性との関係を示している。

【０２４８】

図２５では、0000xxxxB, 0001xxxxB, 100xxxxxBの３パターンが、private_stream_idの値として採用されている。なお、”x”は、図２３における場合と同様に、０または１のうちの任意の値を表す。

【０２４９】

図２５によれば、ATRACオーディオストリームがprivate_payload()に格納されるprivate_stream1_PES_payload()のprivate_stream_idは、0000xxxxBとされる。なお、0000xxxxBのうちの下位４ビットxxxxは、ATRACオーディオストリームを区別するオーディオストリームナンバであり、プログラムストリーム（MPEG2_Program_Stream()）には、このオーディオストリームナンバで区別することのできる数である１６（＝２４）本のATRACオーディオストリームを多重化することができる。

【０２５０】

さらに、図２５によれば、LPCMオーディオストリームがprivate_payload()に格納されるprivate_stream1_PES_payload()のprivate_stream_idは、0001xxxxBとされる。なお、0001xxxxBのうちの下位４ビットxxxxは、LPCMオーディオストリームを区別するオーディオストリームナンバであり、プログラムストリームには、このオーディオストリームナンバで区別することのできる数である１６（＝２４）本のLPCMオーディオストリームを多重化することができる。

【０２５１】

また、図２５によれば、字幕ストリームがprivate_payload()に格納されるprivate_stream1_PES_payload()のprivate_stream_idは、100xxxxxBとされる。なお、100xxxxxBのうちの下位５ビットxxxxxは、字幕ストリームを区別する字幕ストリームナンバであり、プログラムストリームには、この字幕ストリームナンバで区別することのできる数である３２（＝25）本の字幕ストリームを多重化することができる。

【０２５２】

ここで、図２３と図２５の関係をまとめたものが、上述した図１３である。

【０２５３】

図２４に戻り、private_stream1_PES_payload()において、private_stream_idに続く要素は、private_payload()に格納されるエレメンタリストリームの属性により内容が異なっている。private_payload()に格納されるエレメンタリストリームの属性は、private_header()の先頭のprivate_stream_idにより判断される。

【０２５４】

private_payload()に格納されるエレメンタリストリームがATRACオーディオストリームである場合(private_stream_id==ATRAC)、将来の拡張用のreserved_for_future_use（8ビット）が配置され、その後、AU_locator（16ビット）が配置される。AU_locatorは、そのAU_locatorの直後の位置を基準として、private_payload()に格納されたATRACオーディオストリームのオーディオアクセスユニット（ATRACオーディオアクセスユニット）（オーディオフレーム）の先頭位置を表す。private_payload()にオーディオアクセスユニットが存在しない場合、AU_locatorには、例えば0xFFFFが記述される。

【０２５５】

private_payload()に格納されるエレメンタリストリームがLPCMオーディオストリームである場合(private_stream_id==LPCM)、fs_flag（1ビット），reserved_for_future_use（3ビット），ch_flag（4ビット）、およびAU_locator（16ビット）が順次配置される。

【０２５６】

fs_flagは、private_payload()に格納されるLPCMオーディオストリームのサンプリング周波数を示す。即ち、例えば、サンプリング周波数が48KHzの場合、fs_flagは0とされ、サンプリング周波数が44.1KHzの場合、fs_flagは1とされる。

【０２５７】

ch_flagは、private_payload()に格納されるLPCMオーディオストリームのチャネル数を示す。例えば、LPCMオーディオストリームがモノラルの場合、ch_flagは1とされ、LPCMオーディオストリームがステレオの場合、ch_flagは2とされる。

【０２５８】

AU_locatorは、そのAU_locatorの直後の位置を基準として、private_payload()に格納されるLPCMオーディオストリームのオーディオアクセスユニット（LPCMオーディオアクセスユニット）（オーディオフレーム）の先頭位置を示す。private_payload()にオーディ

オアクセスユニットが存在しない場合、AU_locatorには、例えば0xFFFFが記述される。

【０２５９】

private_payload()に格納されるエレメンタリストリームが字幕ストリームである場合(private_stream_id==SUBTITLE)、将来の拡張のためのreserved_for_future_use（８ビット）が配置され、その後に、AU_locator（16ビット）が配置される。AU_locatorは、そのAU_locatorの直後の位置を基準として、private_payload()に格納される字幕ストリームの字幕アクセスユニットの先頭位置を示す。private_payload()に字幕アクセスユニットが存在しない場合、AU_locatorには、例えば0xFFFFが記述される。

【０２６０】

「private_stream2_PES_payload()の説明」

次に、図２６は、private_stream2_PES_payload()のシンタクスを示している。

【０２６１】

private_stream2_PES_payload()は、private_stream_2のPES_packet()のPES_packet_data_byte（図２１）を拡張したもの、即ち、private_stream_2のPES_packet()の、拡張したPES_packet_data_byteであり、ビデオストリームのデコードに利用される情報が記述される。

【０２６２】

本実施の形態では、private_stream_2のPES_packet()は、ビデオストリームにおけるデコード開始可能点の直前に配置される。従って、本実施の形態では、プログラムストリームからprivate_stream_2のPES_packet()を見つければ、その直後のビデオストリームからデコードを開始することができる。

【０２６３】

ここで、上述した図１６のEP_map()のRPN_EP_startは、ビデオストリームについては、private_stream_2のPES_packet()の先頭の位置を示す。

【０２６４】

private_stream2_PES_payload()の先頭には、将来の拡張用のreserved_for_future_use（8ビット）が配置され、続けて、video_stream_id（8ビット），1stRef_picture（16ビット），2ndRef_picture（16ビット），3rdRef_picture（16ビット），4thRef_picture（16ビット），au_information()、およびVBI()が、順次配置される。

【０２６５】

video_stream_idには、private_stream_2のPES_packet()の直後に配置されるビデオストリームのPES_packet()のstream_id（と同一の値）が記述される。このvideo_stream_idによって、private_stream_2のPES_packet()（のprivate_stream2_PES_payload()）に格納された情報を利用してデコードされるビデオストリーム（が格納されたPES_packet()）が特定される。

【０２６６】

1stRef_picture，2ndRef_picture，3rdRef_picture，4thRef_pictureは、video_stream_idによって特定されるビデオストリームの、private_stream_2のPES_packet()から次のprivate_stream_2のPES_packet()までの中の１，２，３，４番目の参照画像を含む最後のpack()の位置を相対値で、それぞれ表す。なお、1stRef_picture，2ndRef_picture，3rdRef_picture，4thRef_pictureについては、例えば、特開平09-46712号公報（特願平07-211420号）に、bytes_to_first_P_pic bytes_to_second_P_picとして、その詳細が開示されている。

【０２６７】

au_information()には、private_stream_2のPES_packet()から、次のprivate_stream_2のPES_packet()までのビデオストリームの中のビデオアクセスユニットに関する情報が記述される。au_information()の詳細については、図２７を参照して後述する。

【０２６８】

VBI()は、Closed Captionの情報を記述するために使用される。

【０２６９】

以上のようなprivate_stream2_PES_payload()を有するprivate_stream_2のPES_packet()は、ビデオストリームごとの、デコード開始可能点ごとに配置される。

【０２７０】

次に、図２７は、図２６のau_information()のシンタクスを示している。

【０２７１】

au_information()の先頭には、length（16ビット）が配置される。lengthは、それを含むau_information()のサイズを表す。lengthに続いては、reserved_for_word_alignment（8ビット）、およびnumber_of_access_unit（8ビット）が順次配置される。reserved_for_word_alignmentは、ワードアラインをとるために使用される。

【０２７２】

number_of_access_unitは、それを含むprivate_stream_2のPES_packet()から、次のprivate_stream_2のPES_packet()までの間に含まれるビデオアクセスユニット（ピクチャ）の数を表す。

【０２７３】

即ち、number_of_access_unitは、図２６のprivate_stream2_PES_payload()が同一のvideo_stream_idを有するprivate_stream_2のPES_packet()の中で、このau_informatnio()から次のau_information()の直前までに（このau_infromation()がクリップストリームファイルで最後のau_information()であれば、クリップストリームファイルの最後までに）、video_stream_idで示されるビデオストリームに含まれるアクセスユニット（ピクチャ）の数を示す。

【０２７４】

number_of_access_unitの後には、そのnumber_of_access_unitの数だけforループの内容が配置される。即ち、number_of_access_unitを含むprivate_stream_2のPES_packet()から、次のprivate_stream_2のPES_packet()までの間に含まれる１以上のビデオアクセスユニットそれぞれに関する情報が配置される。

【０２７５】

forループ内に配置される情報（ビデオアクセスユニットに関する情報）は、以下のようになっている。

【０２７６】

即ち、forループ内には、pic_struct_copy（4ビット），au_ref_flag（1ビット），AU_length（21ビット），reservedが配置される。

【０２７７】

pic_struct_copyには、ビデオストリームがMPEG4-AVC（ISO/IEC 14496-10）の場合に、対応するビデオアクセスユニットに対して設定されている、ISO/IEC 14496-10，D.2.2に定義されているpic_struct（）のコピーが記述される。なお、pic_struct（）は、例えば、ピクチャをフレームとして表示する、あるいは、ピクチャのトップフィールドを表示して、その後、ボトムフィールドを表示する、などといった情報である。

【０２７８】

ここで、図２８に、pic_structのテーブルを示す。

【０２７９】

pic_structは、ピクチャをどのように表示するかの表示方式を指示する表示方式指示情報として使用される。

【０２８０】

図２８のpic_structのテーブルによれば、ピクチャによって、１フレームを表示することを指示する場合には、そのピクチャのpic_structには、図２８の最も左のValueの欄に示されているように、0がセットされる。同様に、ピクチャによって、トップフィールドまたはボトムフィールドを表示することを指示する場合には、そのピクチャのpic_structには、それぞれ1または2がセットされる。さらに、ピクチャによって、トップフィールドとボトムフィールドをその順で表示することを指示する場合には、そのピクチャのpic_structには、3がセットされ、ボトムフィールドとトップフィールドをその順で表示するこ

とを指示する場合には、pic_structには、4がセットされる。また、ピクチャによって、トップフィールド、ボトムフィールド、繰り返しトップフィールドをその順で表示することを指示する場合には、そのピクチャのpic_structには、5がセットされ、ボトムフィールド、トップフィールド、繰り返しボトムフィールドをその順で表示することを指示する場合には、pic_structには、6がセットされる。さらに、ピクチャによって、１フレームを２回繰り返しまたは３回繰り返し表示することを指示する場合には、そのピクチャのpic_structには、それぞれ7または8がセットされる。

【０２８１】

au_ref_flagは、対応するアクセスユニットが、他のアクセスユニット（のピクチャ）のデコードにあたって参照される参照画像であるか否かを表し、参照画像である場合には１とされ、参照画像でない場合には０とされる。

【０２８２】

AU_lengthは、対応するアクセスユニットのサイズをバイト単位で表す。

【０２８３】

［ディスク１０１に記録されたデータの具体例］

次に、図２９乃至図３２は、図１のディスク１０１に記録された、上述したようなフォーマットのデータの具体例を示している。

【０２８４】

ここで、図２９乃至図３２では、ビデオストリームとしては、MPEG2-Videoを採用し、オーディオストリームとしては、ATRACオーディオストリームを採用している。但し、ビデオストリームやオーディオストリームは、これに限定されるものではない。即ち、ビデオストリームとしては、例えば、MPEG4-VisualやMPEG4-AVCなどを採用することができる。さらに、オーディオストリームとしては、例えば、MPEG1/2/4オーディオやLPCMオーディオストリームなどを採用することができる。

【０２８５】

なお、字幕ストリームは、ビデオストリームやオーディオストリームと異なり、同じ間隔で連続的なデコード・表示（出力）が行われるとは限らない。すなわち、字幕ストリームは、時折、図２のバッファ制御モジュール２１５から字幕デコーダ制御モジュール２１８に供給されてデコードされる。

【０２８６】

図２９乃至図３２は、ディスク１０１において、図６に示したように、”CLIP”ディレクトリに、３つのクリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”が記録され、”STREAM”ディレクトリに、その３つのクリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”それぞれに対応する３つのクリップストリームファイル”00001.PS”，”00002.PS”，”00003.PS”が記録されている場合の、”PLAYLIST.DAT”ファイル、クリップ情報ファイル”00001.CLP”，”00002.CLP”、および”00003.CLP”等の具体的な例を示している。但し、図２９乃至図３２では、”PLAYLIST.DAT”ファイル等のデータの一部を省略してある。

【０２８７】

即ち、図２９は、図７で説明した”PLAYLIST.DAT”ファイルの具体例を示している。

【０２８８】

図２９では、number_of_PlayListsは2となっており、従って、”PLAYLIST.DAT”ファイルに含まれるPlayList()の数は2である。図２９では、その２つのPlayList()のうちの１番目がPlayList #0と、２番目がPlayList #1と、それぞれ記載されている。

【０２８９】

１番目のPlayList()であるPlayList #0については、capture_enable_flag_PlayListが１となっており、従って、PlayList #0にしたがって再生されるビデオデータの２次利用が許可されている。また、PlayList #0については、number_of_PlayItemsが２となっており、従って、PlayList #0に含まれるPlayItem()の数は2である。図２９では、その２つのPlayItem()であるPlayItem#0とPlayItem#1の具体例が、「PlayList#0」の欄の下方に記載

されている。

【０２９０】

PlayList #0に含まれる１番目のPlayItem()であるPlayItem#0では、図８で説明したClip_Information_file_nameが”00001.CLP”に、In_timeが180,090に、OUT_timeが27,180,090に、それぞれなっている。従って、PlayList #0のPlayItem#0によって再生されるクリップは、クリップ情報ファイル”00001.CLP”に対応するクリップストリームファイル”00001.PS”の、時刻180,090から27,180,090までである。

【０２９１】

PlayList #0に含まれる２番目のPlayItem()であるPlayItem#1では、図８で説明したClip_Information_file_nameが”00002.CLP”に、In_timeが90,000に、OUT_timeが27,090,000に、それぞれなっている。従って、PlayList #0のPlayItem#1によって再生されるクリップは、クリップ情報ファイル”00002.CLP”に対応するクリップストリームファイル”00002.PS”の、時刻90,000から27,090,000までである。

【０２９２】

一方、図２９において、２番目のPlayList()であるPlayList #1については、capture_enable_flag_PlayListが0となっており、従って、PlayList #1にしたがって再生されるビデオデータの２次利用が許可されていない（禁止されている）。また、PlayList #1については、number_of_PlayItemsが１となっており、従って、PlayList #1に含まれるPlayItem()の数は１である。図２９では、その１つのPlayItem()であるPlayItem#0の具体例が、「PlayList#1」の欄の下方に記載されている。

【０２９３】

PlayList #1に含まれる１つのPlayItem()であるPlayItem#0では、図８で説明したClip_Information_file_nameが”00003.CLP”に、In_timeが90,000に、OUT_timeが81,090,000に、それぞれなっている。従って、PlayList #1のPlayItem#0によって再生されるクリップは、クリップ情報ファイル”00003.CLP”に対応するクリップストリームファイル”00003.PS”の、時刻90,000から81,090,000までである。

【０２９４】

次に、図３０は、図１２で説明したクリップ情報ファイルClip()の具体例を示している。即ち、図３０は、図６のクリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”の具体例を示している。

【０２９５】

クリップ情報ファイル”00001.CLP”においては、presentation_start_timeが90,000に、presentation_end_timeが27,990,000に、それぞれなっている。従って、クリップ情報ファイル”00001.CLP”に対応するクリップストリームファイル”00001.PS”に格納されたプログラムストリームによれば、310秒分（(27,990,000-90,000)/90kHz）のコンテンツが利用可能である。

【０２９６】

また、クリップ情報ファイル”00001.CLP”においては、capture_enable_flag_Clipが１になっており、従って、クリップ情報ファイル”00001.CLP”に対応するクリップストリームファイル”00001.PS”に格納されたプログラムストリームに多重化されたビデオストリーム（に対応するビデオデータ）については、その２次利用が許可されている。

【０２９７】

さらに、図３０において、クリップ情報ファイル”00001.CLP”のnumber_of_streamsは4となっており、従って、対応するクリップストリームファイル”00001.PS”に格納されたプログラムストリームには、４本のエレメンタリストリームが多重化されている。

【０２９８】

いま、その４本のエレメンタリストリームを、stream#0，stream#1，stream#2，stream#3とすると、図３０では、その４本のエレメンタリストリームstream#0，stream#1，stream#2，stream#3それぞれのStreamInfo()（図１２）の具体例が、「”00001.CLP”」の欄の下方に記載されている。

【０２９９】
クリップストリームファイル”00001.PS”の１本目のエレメンタリストリームstream#0については、stream_idが0xE0となっており、従って、このエレメンタリストリームstream#0は、図２３および図２５（あるいは図１３）で説明したように、ビデオストリームである。なお、本実施の形態では、ビデオストリームについては、上述したように、private_stream_idは無関係であるが、図３０では、0x00になっている。
【０３００】
また、クリップストリームファイル”00001.PS”の１本目のエレメンタリストリームであるビデオストリームstream#0については、そのStreamInfo()に含まれるStaticInfo()（図１４）のpicture_sizeが'720×480'に、frame_rateが'29.97Hz'に、cc_flagが'Yes'に、それぞれなっている。従って、このビデオストリームstream#0は、720×480ピクセルの、フレーム周期が29.97Hzのビデオデータであり、さらに、クローズドキャプションデータを含んでいる。
【０３０１】
さらに、クリップストリームファイル”00001.PS”の１本目のエレメンタリストリームであるビデオストリームstream#0については、StreamInfo()（図１２）のnumber_of_DynamicInfoが０になっており、pts_change_pointとDynamicInfo()とのセットは存在しない。
【０３０２】
次に、クリップストリームファイル”00001.PS”の２本目のエレメンタリストリームstream#1については、stream_idが0xBDとなっており、private_stream_idが0x00となっている。従って、このエレメンタリストリームstream#1は、図２３および図２５で説明したように、ATRACオーディオストリームである。
【０３０３】
また、クリップストリームファイル”00001.PS”の２本目のエレメンタリストリームであるATRACオーディオストリームstream#1については、そのStreamInfo()に含まれるStaticInfo()（図１４）のaudio_language_codeが'日本語'に、channel_configurationが'STEREO'に、lfe_existenceが'NO'に、sampling_frequencyが'48kHz'に、それぞれなっている。従って、このATRACオーディオストリームstream#1は、日本語で、かつステレオのオーディオデータである。また、低域強調チャネルは含まれておらず、サンプリング周波数は、48kHzである。
【０３０４】
さらに、クリップストリームファイル”00001.PS”の２本目のエレメンタリストリームであるATRACオーディオストリームstream#1については、StreamInfo()（図１２）のnumber_of_DynamicInfoが０になっており、pts_change_pointとDynamicInfo()とのセットは存在しない。
【０３０５】
次に、クリップストリームファイル”00001.PS”の３本目のエレメンタリストリームstream#2については、stream_idが0xBDとなっており、private_stream_idが0x80となっている。従って、このエレメンタリストリームstream#2は、図２３および図２５で説明したように、字幕ストリームである。
【０３０６】
また、クリップストリームファイル”00001.PS”の３本目のエレメンタリストリームである字幕ストリームstream#2については、そのStreamInfo()に含まれるStaticInfo()（図１４）のsubtitle_language_codeが'日本語'に、configurable_flagが０に、それぞれなっている。従って、この字幕ストリームstream#2は、日本語の字幕データであり、また、その表示方式を変更することは許可されていない（禁止されている）。
【０３０７】
さらに、クリップストリームファイル”00001.PS”の３本目のエレメンタリストリームである字幕ストリームstream#2については、StreamInfo()（図１２）のnumber_of_DynamicInfoが０になっており、pts_change_pointとDynamicInfo()とのセットは存在しない。

【０３０８】
次に、クリップストリームファイル”00001.PS”の４本目のエレメンタリストリームstream#3については、stream_idが0xBDとなっており、private_stream_idが0x81となっている。従って、このエレメンタリストリームstream#3は、図２３および図２５で説明したように、字幕ストリームである。
【０３０９】
なお、クリップストリームファイル”00001.PS”の３本目のエレメンタリストリームである字幕ストリームstream#2と、４本目のエレメンタリストリームである字幕ストリームstream#3とを区別するために、それぞれのprivate_stream_idは、0x80と0x81とになっている。
【０３１０】
また、クリップストリームファイル”00001.PS”の４本目のエレメンタリストリームである字幕ストリームstream#2については、そのStreamInfo()に含まれるStaticInfo()（図１４）のsubtitle_language_codeが'日本語'に、configurable_flagが１に、それぞれなっている。従って、この字幕ストリームstream#3は、日本語の字幕データであり、また、その表示方式を変更することが許可されている。
【０３１１】
さらに、クリップストリームファイル”00001.PS”の４本目のエレメンタリストリームである字幕ストリームstream#3については、StreamInfo()（図１２）のnumber_of_DynamicInfoが０になっており、pts_change_pointとDynamicInfo()とのセットは存在しない。
【０３１２】
次に、図３０において、クリップ情報ファイル”00002.CLP”については、presentation_start_timeが90,000に、presentation_end_timeが27,090,000に、それぞれなっている。従って、クリップ情報ファイル”00002.CLP”に対応するクリップストリームファイル”00002.PS”に格納されたプログラムストリームによれば、300秒分（(27,090,000-90,000)/90kHz）のコンテンツが利用可能である。
【０３１３】
また、クリップ情報ファイル”00002.CLP”においては、capture_enable_flag_Clipが０になっており、従って、クリップ情報ファイル”00002.CLP”に対応するクリップストリームファイル”00002.PS”に格納されたプログラムストリームに多重化されたビデオストリーム（に対応するビデオデータ）については、その２次利用が許可されていない（禁止されている）。
【０３１４】
さらに、図３０において、クリップ情報ファイル”00002.CLP”のnumber_of_streamsは4となっており、従って、対応するクリップストリームファイル”00002.PS”に格納されたプログラムストリームには、上述したクリップストリームファイル”00001.PS”における場合と同様に、４本のエレメンタリストリームが多重化されている。
【０３１５】
いま、その４本のエレメンタリストリームを、stream#0，stream#1，stream#2，stream#3とすると、図３０では、その４本のエレメンタリストリームstream#0，stream#1，stream#2，stream#3それぞれのStreamInfo()（図１２）の具体例が、「”00002.CLP”」の欄の下方に記載されている。
【０３１６】
ここで、図３０では、クリップストリームファイル”00002.PS”の１乃至４本目のエレメンタリストリームstream#0乃至#3それぞれのStreamInfo()の内容は、上述したクリップストリームファイル”00001.PS”の１乃至４本目のエレメンタリストリームstream#0乃至#3それぞれのStreamInfo()の内容と同一になっているので、その説明は省略する。
【０３１７】
なお、上述のように、クリップストリームファイル”00002.PS”の１乃至４本目のエレメンタリストリームstream#0乃至#3それぞれのStreamInfo()の内容は、クリップストリー

ムファイル”00001.PS”の１乃至４本目のエレメンタリストリームstream#0乃至#3それぞれのStreamInfo()の内容と同一であるので、クリップストリームファイル”00002.PS”の１本目のエレメンタリストリームstream#0はビデオストリームであり、２本目のエレメンタリストリームstream#1はATRACオーディオストリームである。また、その３本目のエレメンタリストリームstream#2と、４本目のエレメンタリストリームstream#3は、いずれも字幕ストリームである。

【０３１８】

次に、図３０において、クリップ情報ファイル”00003.CLP”については、presentation_start_timeが90,000に、presentation_end_timeが81,090,000に、それぞれなっている。従って、クリップ情報ファイル”00003.CLP”に対応するクリップストリームファイル”00003.PS”に格納されたプログラムストリームによれば、900秒分（(81,090,000-90,000)/90kHz）のコンテンツが利用可能である。

【０３１９】

また、クリップ情報ファイル”00003.CLP”においては、capture_enable_flag_Clipが１になっており、従って、クリップ情報ファイル”00003.CLP”に対応するクリップストリームファイル”00003.PS”に格納されたプログラムストリームに多重化されたビデオストリームについては、その２次利用が許可されている。

【０３２０】

さらに、図３０において、クリップ情報ファイル”00003.CLP”のnumber_of_streamsは３となっており、従って、対応するクリップストリームファイル”00003.PS”に格納されたプログラムストリームには、３本のエレメンタリストリームが多重化されている。

【０３２１】

いま、その３本のエレメンタリストリームを、stream#0，stream#1，stream#2とすると、図３０では、その３本のエレメンタリストリームstream#0，stream#1，stream#2それぞれのStreamInfo()（図１２）の具体例が、「”00003.CLP”」の欄の下方に記載されている。

【０３２２】

クリップストリームファイル”00003.PS”の１本目のエレメンタリストリームstream#0については、stream_idが0xE0となっており、従って、このエレメンタリストリームstream#0は、図２３および図２５（あるいは図１３）で説明したように、ビデオストリームである。なお、クリップストリームファイル”00001.PS”の１本目のエレメンタリストリームstream#0と同様に、private_stream_idは、0x00になっている。

【０３２３】

また、クリップストリームファイル”00003.PS”の１本目のエレメンタリストリームであるビデオストリームstream#0については、そのStreamInfo()に含まれるStaticInfo()（図１４）のpicture_sizeが'720×480'に、frame_rateが'29.97Hz'に、cc_flagが'No'に、それぞれなっている。従って、このビデオストリームstream#0は、720×480ピクセルの、フレーム周期が29.97Hzのビデオデータであり、さらに、クローズドキャプションデータを含んでいない。

【０３２４】

さらに、クリップストリームファイル”00003.PS”の１本目のエレメンタリストリームであるビデオストリームstream#0については、StreamInfo()（図１２）のnumber_of_DynamicInfoが２になっており、従って、そのStreamInfo()には、pts_change_pointとDynamicInfo()とのセットが２セット記述されている。

【０３２５】

次に、クリップストリームファイル”00003.PS”の２本目のエレメンタリストリームstream#1については、stream_idが0xE1となっており、従って、このエレメンタリストリームstream#1は、図２３および図２５（あるいは図１３）で説明したように、ビデオストリームである。なお、クリップストリームファイル”00003.PS”の１本目のエレメンタリストリームであるビデオストリームstream#0と、２本目のエレメンタリストリームであるビ

デオストリームstream#1とを区別するために、それぞれのstream_idは、0xE0と0xE1とになっている。また、クリップストリームファイル”00001.PS”の１本目のエレメンタリストリームstream#0と同様に、private_stream_idは、0x00になっている。

【０３２６】

また、クリップストリームファイル”00003.PS”の２本目のエレメンタリストリームであるビデオストリームstream#1については、そのStreamInfo()に含まれるStaticInfo()（図１４）のpicture_size, frame_rate, cc_flagが、１本目のエレメンタリストリームであるビデオストリームstream#0についてのものと同一になっている。従って、クリップストリームファイル”00003.PS”の２本目のエレメンタリストリームであるビデオストリームstream#1は、720×480ピクセルの、フレーム周期が29.97Hzのビデオデータであり、さらに、クローズドキャプションデータを含んでいない。

【０３２７】

さらに、クリップストリームファイル”00003.PS”の２本目のエレメンタリストリームであるビデオストリームstream#1については、StreamInfo()（図１２）のnumber_of_DynamicInfoが０になっており、pts_change_pointとDynamicInfo()とのセットは存在しない。

【０３２８】

次に、クリップストリームファイル”00003.PS”の３本目のエレメンタリストリームstream#2については、stream_idが0xBDとなっており、private_stream_idが0x00となっている。従って、このエレメンタリストリームstream#2は、図２３および図２５で説明したように、ATRACオーディオストリームである。

【０３２９】

また、クリップストリームファイル”00003.PS”の３本目のエレメンタリストリームであるATRACオーディオストリームstream#2については、そのStreamInfo()に含まれるStaticInfo()（図１４）のaudio_language_code, channel_configuration, lfe_existence, sampling_frequencyが、クリップストリームファイル”00001.PS”の２本目のエレメンタリストリームであるATRACオーディオストリームstream#1のものと同一になっている。従って、クリップストリームファイル”00003.PS”の３本目のエレメンタリストリームであるATRACオーディオストリームstream#2は、日本語で、かつステレオのオーディオデータである。また、低域強調チャネルは含まれておらず、サンプリング周波数は、48kHzである。

【０３３０】

さらに、クリップストリームファイル”00003.PS”の３本目のエレメンタリストリームであるATRACオーディオストリームstream#2については、StreamInfo()（図１２）のnumber_of_DynamicInfoが３になっており、従って、そのStreamInfo()には、pts_change_pointとDynamicInfo()とのセットが３セット記述されている。

【０３３１】

次に、図３１は、図１２で説明したクリップ情報ファイルClip()のうちのEP_map()具体例を示している。即ち、図３１は、図６のクリップ情報ファイル”00001.CLP”の中の、図１６のEP_map()の具体例を示している。

【０３３２】

図３１において、EP_map()のnumber_of_stream_id_entriesは1になっており、従って、このEP_map()には、１つのエレメンタリストリームについてのデコード開始可能点の情報が記述されている。

【０３３３】

また、図３１のEP_map()では、stream_idが0xE0になっている。従って、図２３および図２５で説明したことから、EP_map()には、0xE0となっているstream_idによって特定されるビデオストリームについてのデコード開始可能点の情報（RAPI（ランダムアクセスポイント情報））（PTS_EP_startとRPN_EP_start（図１６））が記述されている。即ち、図３１は、クリップ情報ファイル”00001.CLP”のEP_map()であり、クリップ情報ファイル”00001.CLP”に対応するクリップストリームファイル”00001.CLP”において、stream_i

dが0xE0のエレメンタリストリームは、図３０で説明したように、クリップストリームファイル”00001.CLP”の１本目のビデオストリームstream#0であるから、図３１のEP_map()に記述されている情報は、そのビデオストリームstream#0のデコード開始可能点のPTS_EP_startとRPN_EP_startである。

【０３３４】

図３１では、クリップストリームファイル”00001.CLP”の１本目のビデオストリームstream#0のデコード開始可能点のうちの、先頭から５点のPTS_EP_startとRPN_EP_startが記載されており、６点目以降のPTS_EP_startとRPN_EP_startの記載は省略してある。

【０３３５】

RPN_EP_start、PTS_EP_start、1stRef_Picture、2ndRef_Picture、3rdRef_Picture、4thRef_Pictureは、多重化ストリームに多重化されたすべてのRAPIの開始位置に加えて、RAPI直後に置かれているイントラピクチャ及びそれにつづく2枚目、3枚目、4枚目の参照画像の終了位置が示されている。

【０３３６】

先頭のRAPIの位置は0（セクタ）であり、直後のイントラピクチャのPTSは90,000である。また、該当イントラピクチャ、および2枚目、3枚目、4枚目の参照画像の終了位置は、RAPIの先頭からの相対セクタ数でそれぞれ、28，37，48，58である。

【０３３７】

また、２番目のRAPIの位置は244（セクタ）であり、直後のイントラピクチャのPTSは135,045である。また、該当イントラピクチャ、および2枚目、3枚目、4枚目の参照画像の終了位置は、RAPIの先頭からの相対セクタ数でそれぞれ10，18，25，31である。

【０３３８】

さらに、３番目のRAPIの位置は305（セクタ）であり、直後のイントラピクチャのPTSは180,090である。また、該当イントラピクチャ、および2枚目、3枚目、4枚目の参照画像の終了位置は、RAPIの先頭からの相対セクタ数でそれぞれ25，44，50，54である。

【０３３９】

また、４番目のRAPIの位置は427（セクタ）であり、直後のイントラピクチャのPTSは225,135である。また、該当イントラピクチャ、および2枚目、3枚目、4枚目の参照画像の終了位置は、RAPIの先頭からの相対セクタ数でそれぞれ8，15，22，29である。

【０３４０】

さらに、５番目のRAPIの位置は701（セクタ）であり、直後のイントラピクチャのPTSは270,180である。また、該当イントラピクチャ、および2枚目、3枚目、4枚目の参照画像の終了位置は、RAPIの先頭からの相対セクタ数でそれぞれ26，32，41，48である。

【０３４１】

N-th_Ref_picture_copyは、４点の参照画像終了位置（1stRef_Picture、2ndRef_Picture、3rdRef_Picture、4thRef_Picture）のうち、所定のセクタ数（エンコード処理において、まとめて読出可能なセクタ数）に近い値が格納される。図３１においては、セクタ数が”30”に一番近いものが選択されている。

【０３４２】

例えば、先頭のエントリに対してはPTS_EP_start＝90,000、RPN_EP_start＝0である。N-th_Ref_picture_copyは、28，37，48，58のなかから”30”に一番近い28が選択され、この結果index_N_minus1には1stRef_pictureを示す値”0”が格納されている。

【０３４３】

次に、２番目のエントリに対してはPTS_EP_start＝135,045、RPN_EP_start＝244である。N-th_Ref_picture_copyは、10，18，25，31のなかから”30”に一番近い31が選択され、この結果index_N_minus1には4thRef_pictureを示す値”3”が格納されている。

【０３４４】

これにより、図３１の例の5点のエントリに対するindex_N_minus1とN-th_Ref_picture_copyは（0，28），（3，31），（0，25），（3，29），（1，32）が格納される。

【０３４５】

この選択アルゴリズムは、再生機器における再生品質等、総合的に勘案して決定される。そこで、本実施例においては、比較的小さいセクタ数”30”に一番近いものを選ぶとしているが、これ以外のセクタ数であってもよい。また、index_N_minus1の値が小さいとき、RAPIの位置からN-th_Ref_picture_copyの大きさだけデータを読み込んだときに、そこに含まれる参照画像の数が少なく、逆にindex_N_minus1の値が大きいときは、含まれている参照画像の数が多いということになる。

【０３４６】

さらに、この例では5点のデータ全てに４枚の参照画像の終了位置が記述されているが、ビデオエンコードの方法、あるいはイントラピクチャの間隔によっては、参照画像が４枚未満となる可能性もある。そのような場合には、各エントリポイントに対して存在する、４枚以下の最大の参照画像の終了位置を記述するという方法も考えられる。

【０３４７】

なお、図３１のEP_map()において、private_stream_idは0x00になっているが、stream_idがビデオストリームを表している場合は、上述したように、private_stream_idは無関係である（無視される）。

【０３４８】

次に、図３２は、図２９で説明したPlayList #0とPlayList #1（図７のPlayList()）の中のPlayListMark()の具体例を示している。

【０３４９】

図３２上側は、PlayList #0のPlayListMark()（図９）を示している。

【０３５０】

図３２上側において、PlayList #0のPlayListMark()におけるnumber_of_PlayList_marksは7になっており、従って、PlayList #0（のPlayListMark()）に含まれるMark()の数は7である。

【０３５１】

また、図３２上側において、PlayList #0に含まれる７つのMark()のうちの１番目のMark()であるMark#0は、mark_type（図９）が'Chapter'になっているので、チャプタマークである。さらに、Mark#0は、ref_to_PlayItem_id（図９）が0になっているので、PlayList #0に含まれる図２９の２つのPlayItem#0と#1のうちの、PlayItem#0に属する。また、Mark#0は、mark_time_stampが180,090になっているので、PlayList #0に含まれるPlayItem#0によって再生されるクリップストリームファイルの時刻（再生時刻）180,090上の印である。さらに、Mark#0は、entry_ES_stream_idおよびentry_ES_private_stream_idがいずれも0になっているので、いずれのエレメンタリストリームにも関連付けられていない。また、Mark#0は、mark_dataが1になっているので、番号が１のチャプタを表す。

【０３５２】

なお、ここでは、PlayList #0に含まれるPlayItem#0によって再生されるクリップストリームファイルは、図２９で説明した、そのPlayItem#0のClip_Infomation_file_nameに記述されている”00001.CLP”から特定されるクリップストリームファイル”00001.PS”であり、従って、Mark#0のmark_time_stampが表す、上述した時刻180,090は、クリップストリームファイル”00001.PS”の時刻である。

【０３５３】

図３２上側において、PlayList #0に含まれる７つのMark()のうちの５番目のMark()であるMark#4も、１番目のMark#0と同様のチャプタマークである。

【０３５４】

即ち、５番目のMark()であるMark#4は、mark_type（図９）が'Chapter'になっているので、チャプタマークである。さらに、Mark#4は、ref_to_PlayItem_id（図９）が1になっているので、PlayList #0に含まれる図２９の２つのPlayItem#0と#1のうちの、PlayItem#1に属する。また、Mark#4は、mark_time_stampが90,000になっているので、PlayList #0に含まれるPlayItem#1によって再生されるクリップストリームファイルの時刻90,000上の印である。さらに、Mark#4は、entry_ES_stream_idおよびentry_ES_private_stream_idが

いずれも0になっているので、いずれのエレメンタリストリームにも関連付けられていない。また、Mark#4は、mark_dataが2になっているので、番号が２のチャプタを表す。

【０３５５】

なお、ここでは、PlayList #0に含まれるPlayItem#1によって再生されるクリップストリームファイルは、図２９で説明した、そのPlayItem#1のClip_Infomation_file_nameに記述されている”00002.CLP”から特定されるクリップストリームファイル”00002.PS”であり、従って、Mark#4のmark_time_stampが表す、上述した時刻90,000は、クリップストリームファイル”00002.PS”の時刻である。

【０３５６】

また、図３２上側において、PlayList #0に含まれる７つのMark()のうちの２番目のMark()であるMark#1は、mark_type（図９）が'Index'になっているので、インデクスマークである。さらに、Mark#1は、ref_to_PlayItem_id（図９）が0になっているので、PlayList #0に含まれる図２９の２つのPlayItem#0と#1のうちの、PlayItem#0に属する。また、Mark#1は、mark_time_stampが5,580,090になっているので、PlayList #0に含まれるPlayItem#0によって再生されるクリップストリームファイルの時刻5,580,090上の印である。さらに、Mark#1は、entry_ES_stream_idおよびentry_ES_private_stream_idがいずれも0になっているので、いずれのエレメンタリストリームにも関連付けられていない。また、Mark#1は、mark_dataが1になっているので、番号が１のインデクスを表す。

【０３５７】

なお、ここでは、PlayList #0に含まれるPlayItem#0によって再生されるクリップストリームファイルは、上述したように、クリップストリームファイル”00001.PS”であり、従って、Mark#1のmark_time_stampが表す、上述した時刻5,580,090は、クリップストリームファイル”00001.PS”の時刻である。

【０３５８】

図３２上側において、PlayList #0に含まれる７つのMark()のうちの３番目、６番目、７番目のMark()であるMark#2, Mark#5, Mark#6も、２番目のMark#1と同様のインデクスマークである。

【０３５９】

また、図３２上側において、PlayList #0に含まれる７つのMark()のうちの４番目のMark()であるMark#3は、mark_type（図９）が'Event'になっているので、イベントマークである。さらに、Mark#3は、ref_to_PlayItem_id（図９）が0になっているので、PlayList #0に含まれる図２９の２つのPlayItem#0と#1のうちの、PlayItem#0に属する。また、Mark#3は、mark_time_stampが16,380,090になっているので、PlayList #0に含まれるPlayItem#0によって再生されるクリップストリームファイルの時刻16,380,090上の印である。さらに、Mark#3は、entry_ES_stream_idおよびentry_ES_private_stream_idがいずれも0になっているので、いずれのエレメンタリストリームにも関連付けられていない。また、Mark#3は、mark_dataが0になっているので、引数として0を伴うイベントを発生させる。

【０３６０】

なお、ここでは、PlayList #0に含まれるPlayItem#0によって再生されるクリップストリームファイルは、上述したように、クリップストリームファイル”00001.PS”であり、従って、Mark#3のmark_time_stampが表す、上述した時刻16,380,090は、クリップストリームファイル”00001.PS”の時刻である。

【０３６１】

ここで、図３２上側では、PlayList #0のPlayListMark()の一覧表の欄外の右側に、Mark()が属するPlayItem()の先頭からの時間を示してあり、さらにその右側に、PlayList #0の先頭からの時刻を示してある。

【０３６２】

次に、図３２下側は、PlayList#1のPlayListMark()（図９）を示している。

【０３６３】

図３２下側において、PlayList#1のPlayListMark()におけるnumber_of_PlayList_marks

は3になっており、従って、PlayList#1（のPlayListMark()）に含まれるMark()の数は3である。

【０３６４】

また、図３２下側において、PlayList#1に含まれる３つのMark()のうちの１番目のMark()であるMark#0は、mark_type（図９）が'Chapter'になっているので、チャプタマークである。さらに、Mark#0は、ref_to_PlayItem_id（図９）が0になっているので、PlayList#1に含まれる図２９の１つのPlayItem#0に属する。また、Mark#0は、mark_time_stampが90,000になっているので、PlayList#1に含まれるPlayItem#0によって再生されるクリップストリームファイルの時刻90,000上の印である。さらに、Mark#0は、entry_ES_stream_idおよびentry_ES_private_stream_idがいずれも0になっているので、いずれのエレメンタリストリームにも関連付けられていない。また、Mark#0は、mark_dataが０になっているので、番号が０のチャプタを表す。

【０３６５】

なお、ここでは、PlayList#1に含まれるPlayItem#0によって再生されるクリップストリームファイルは、図２９で説明した、そのPlayItem#0のClip_Infomation_file_nameに記述されている”00003.CLP”から特定されるクリップストリームファイル”00003.PS”であり、従って、Mark#0のmark_time_stampが表す、上述した時刻90,000は、クリップストリームファイル”00003.PS”の時刻である。

【０３６６】

図３２下側において、PlayList#1に含まれる３つのMark()のうちの２番目のMark()であるMark#1は、mark_type（図９）が'Event'になっているので、イベントマークである。さらに、Mark#1は、ref_to_PlayItem_id（図９）が0になっているので、PlayList#1に含まれる図２９の１つのPlayItem#0に属する。また、Mark#1は、mark_time_stampが27,090,000になっているので、PlayList#1に含まれるPlayItem#0によって再生されるクリップストリームファイルの時刻27,090,000上の印である。さらに、Mark#1は、entry_ES_stream_idが0xE0で、entry_ES_private_stream_idが0になっているので、stream_idが0xE0で特定（識別）されるエレメンタリストリーム、即ち、図２３および図２５で説明したように、ビデオストリームに関連付けられている。また、Mark#1は、mark_dataが1になっているので、引数として1を伴うイベントを発生させる。

【０３６７】

なお、ここでは、PlayList#1に含まれるPlayItem#0によって再生されるクリップストリームファイルは、上述したように、クリップストリームファイル”00003.PS”であり、従って、Mark#1のmark_time_stampが表す、上述した時刻27,090,000は、クリップストリームファイル”00003.PS”の時刻である。

【０３６８】

また、Mark#1が関連付けられている、stream_idが0xE0のビデオストリームは、そのMark#1が属する、図２９のPlayList#1に含まれるPlayItem#0のClip_Infomation_file_nameに記述されている”00003.CLP”に記述されているstream_idが0xE0のビデオストリーム、即ち、図３０のクリップ情報ファイル”00003.CLP”から特定される、クリップストリームファイル”00003.PS”に多重化されている３本のエレメンタリストリームstream#0乃至#2のうちの、１本目のエレメンタリストリーム（ビデオストリーム）stream#0である。

【０３６９】

次に、図３２下側において、PlayList#1に含まれる３つのMark()のうちの３番目のMark()であるMark#2は、mark_type（図９）が'Event'になっているので、イベントマークである。さらに、Mark#2は、ref_to_PlayItem_id（図９）が0になっているので、PlayList#1に含まれる図２９の１つのPlayItem#0に属する。また、Mark#1は、mark_time_stampが27,540,000になっているので、PlayList#1に含まれるPlayItem#0によって再生されるクリップストリームファイルの時刻27,540,000上の印である。さらに、Mark#2は、entry_ES_stream_idが0xE1で、entry_ES_private_stream_idが0になっているので、stream_idが0xE1で特定（識別）されるエレメンタリストリーム、即ち、図２３および図２５で説明したよう

に、ビデオストリームに関連付けられている。また、Mark#2は、mark_dataが2になっているので、引数として2を伴うイベントを発生させる。

【０３７０】

なお、ここでは、PlayList#1に含まれるPlayItem#0によって再生されるクリップストリームファイルは、上述したように、クリップストリームファイル”00003.PS”であり、従って、Mark#2が表す、上述した時刻27,540,000は、クリップストリームファイル”00003.PS”の時刻である。

【０３７１】

また、Mark#2が関連付けられている、stream_idが0xE1のビデオストリームは、そのMark#2が属する、図２９のPlayList#1に含まれるPlayItem#0のClip_Infomation_file_nameに記述されている”00003.CLP”に記述されているstream_idが0xE1のビデオストリーム、即ち、図３０のクリップ情報ファイル”00003.CLP”から認識される、クリップストリームファイル”00003.PS”に多重化されている３本のエレメンタリストリームstream#0乃至#2のうちの、２本目のエレメンタリストリーム（ビデオストリーム）stream#1である。

【０３７２】

ここで、図３２下側では、PlayList#1のPlayListMark()の一覧表の欄外の右側に、Mark()が属するPlayItem()の先頭からの時刻を示してある。

【０３７３】

なお、図３２においては、チャプタマークやインデクスマークが表すチャプタやインデクスの番号が、mark_dataに記述されているが、チャプタマークやインデクスマークが表すチャプタやインデクスの番号は、mark_dataに記述しなくても、例えば、PlayListMark()におけるチャプタマークやインデクスマークの数をカウントすることによって認識することができる。

【０３７４】

［ディスク再生装置の動作説明］

次に、図１のディスク１０１に、図２９乃至図３２で説明したようなデータ（ファイル）が記録されているとして、図１のディスク再生装置の動作について説明する。

【０３７５】

ここで、多重方式としているMPEG2-systemの規定に拠れば、タイムスタンプは全てのアクセスユニットに付加する必要は無く、その間隔が0.7秒以下で良いとされている。つまり、タイムスタンプを持つアクセスユニットと持たないアクセスユニットが存在する。

【０３７６】

尚、ここで説明している例では、デコード開始位置のビデオストリームのアクセスユニットには必ずタイムスタンプが付加されていると仮定している。すなわち「再生準備処理」で後述するように、デコード開始位置はEP_map()を使用して式PTS_EP_start≦IN_timeを満たす最大のPTS_EP_startを、二分探索（バイナリサーチ）等を用いて検索している。EP_map()に登録されているビデオの再生開始位置直後のアクセスユニットには必ずタイムスタンプが付加されている。

【０３７７】

さらに、孤立フィールド(non-paired field)は存在しないとしている。すなわち、pic_struct=1のアクセスユニットの直後には必ずpic_struct=2のアクセスユニットが置かれる。またpic_struct=2のアクセスユニットの直後には必ずpic_struct=1のアクセスユニットが置かれる。

【０３７８】

さらにこの例ではpic_struct=７と８は発生しないとする。

【０３７９】

ディスク１０１がディスクドライブ１０２に挿入されると、その旨を示すメッセージがドライブインターフェース１１４、さらには、図２のオペレーティングシステム２０１を経由して、ビデオコンテンツ再生プログラム２１０に伝えられる。ビデオコンテンツ再生プログラム２１０は、ディスク１０１がディスクドライブ１０２に挿入された旨のメッセ

ージをオペレーティングシステム２０１から受信すると、図３３の再生前処理を開始する。

【０３８０】

「再生前処理」

即ち、図３３は、ビデオコンテンツ再生プログラム２１０が行う再生前処理を説明するフローチャートである。

【０３８１】

ここで、以下、フローチャートによって説明するディスク再生装置の動作または処理は、必ずしもフローチャートとして記載された順序に沿って時系列に行われる必要はなく、並列的あるいは個別に行われることもある。但し、本明細書では、便宜上、ディスク再生装置の動作または処理を、フローチャートに沿って説明する。

【０３８２】

再生前処理では、ビデオコンテンツ再生プログラム２１０は、ステップＳ１０１において、オペレーティングシステム２０１のファイルシステム機能を使用して、ディスク１０１をチェックし、ディスク１０１が、ビデオコンテンツ再生プログラム２１０用の正常なディスクであるか否かを判定する。

【０３８３】

ここで、上述したように、ディスク１０１へのアクセス（ファイルの読み出し）は、オペレーティングシステム２０１のファイルシステム機能を使用して行われるが、以下では、その説明は、適宜省略する。

【０３８４】

ステップＳ１０１において、ディスク１０１が正常なディスクでないと判定された場合、即ち、例えば、ディスク１０１に採用されているファイルシステムが、オペレーティングシステム２０１の対応していないタイプであった場合や、ルートディレクトリに”VIDEO”ディレクトリが置かれていない場合、ビデオコンテンツ再生プログラム２１０はディスク１０１に対応していないと判断して、ステップＳ１０２に進み、グラフィクス処理モジュール２１９が、エラー処理を行って、再生前処理を終了する。

【０３８５】

即ち、グラフィックス処理モジュール２１９は、エラー処理として、ディスク１０１が正常でない旨のエラーメッセージ（のビデオデータ）を生成し、ビデオ出力モジュール２２０から出力させ、これにより、エラーメッセージを表示させる。なお、エラー処理としては、その他、例えば、オーディオ出力モジュール２２１からの警告音の出力や、ディスクドライブ１０２からのディスク１０１の排出等を行うようにすることが可能である。

【０３８６】

一方、ステップＳ１０１において、ディスク１０１が正常なディスクであると判定された場合、ステップＳ１０３に進み、ビデオコンテンツ再生プログラム２１０は、コンテンツデータ供給モジュール２１３によって、オペレーティングシステム２０１に対し、ディスク１０１（図６）の”VIDEO”ディレクトリに置かれている”SCRIPT.DAT”ファイルと、”PLAYLIST.DAT”ファイルとの２つのデータファイルを要求して読み込み、ステップＳ１０４に進む。ステップＳ１０４では、”SCRIPT.DAT”ファイルが、スクリプト制御モジュール２１１に供給されるとともに、”PLAYLIST.DAT”ファイルが、プレイヤ制御モジュール２１２に供給される。

【０３８７】

その後、ステップＳ１０４からＳ１０５乃至Ｓ１０７に進み、プレイヤ制御モジュール２１２は、初期化処理を行う。なお、スクリプト制御モジュール２１１は、プレイヤ制御モジュール２１２の初期化処理が終了するまで待つ。

【０３８８】

「プレイヤ制御モジュール２１２の初期化処理」

初期化処理では、ステップＳ１０５において、プレイヤ制御モジュール２１２が、”PLAYLIST.DAT”ファイルの解析を行い、”PLAYLIST.DAT”ファイルの中で使われているクリ

ップ情報ファイルの数とそのファイル名を調査する。

【０３８９】

即ち、いまの場合、”PLAYLIST.DAT”ファイルは、図２９に示したものとなっており、プレイヤ制御モジュール２１２は、図２９の”PLAYLIST.DAT”ファイルにおいてnumber_of_PlayListsが2になっていることから、１番目のPlayList#0と２番目のPlayList#1との２つのPlayList()が存在することを認識する。さらに、プレイヤ制御モジュール２１２は、図２９の”PLAYLIST.DAT”ファイルにおける１番目のPlayList#0について、number_of_PlayItemsが2となっていることから、そのPlayList#0には、１番目のPlayItem#0と２番目のPlayItem#1との２つのPlayItem()が存在することを認識する。そして、プレイヤ制御モジュール２１２は、図２９の”PLAYLIST.DAT”ファイルにおける、PlayList#0に含まれる１番目のPlayItem#0と２番目のPlayItem#1それぞれのClip_Information_file_nameを参照することにより、PlayList#0に含まれる１番目のPlayItem#0のクリップ情報ファイル（のファイル名）が”00001.CLP”であり、２番目のPlayItem#1のクリップ情報ファイルが”00002.CLP”であることを認識する。

【０３９０】

プレイヤ制御モジュール２１２は、２番目のPlayList#1についても同様にして、そのnumber_of_PlayItemsが1であることから、１つのPlayItem()（PlayItem#0）が存在し、さらに、そのPlayItem#0の中のClip_Information_file_nameから、そのPlayItem#0のクリップ情報ファイルが”00003.CLP”であることを認識する。

【０３９１】

その後、ステップＳ１０５からＳ１０６に進み、プレイヤ制御モジュール２１２は、ディスク１０１から、ステップＳ１０５で認識したクリップ情報ファイル、即ち、ディスク１０１の”VIDEO”ディレクトリ内の”CLIP”ディレクトリから３つのクリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”を読み込む。

【０３９２】

ここで、ステップＳ１０６でのクリップ情報ファイルの読み込みは、最初に再生されるPlayList()のPlayItemのクリップ情報ファイルだけで十分であるが、本実施の形態では、上述したように、すべてのPlayList()のPlayItem()のクリップ情報ファイルを先読みしておくこととする。

【０３９３】

ステップＳ１０６の処理後は、ステップＳ１０７に進み、プレイヤ制御モジュール２１２は、ステップＳ１０５で認識したクリップ情報ファイルの読み込みに成功したかどうかを判定し、さらに、その読み込んだクリップ情報ファイルに対応するクリップストリームファイルが、ディスク１０１に存在するか否かを判定（チェック）する。即ち、ステップＳ１０７では、クリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”の読み込みに成功し、さらに、そのクリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”それぞれとファイル名の拡張子のみが異なるクリップストリームファイル”00001.PS”、”00002.PS”、”00003.PS”が、ディスク１０１の”VIDEO”ディレクトリの下にある”STREAM”ディレクトリに存在するかどうかを判定する。

【０３９４】

ステップＳ１０７において、ステップＳ１０５で認識したクリップ情報ファイルの読み込みに失敗したと判定されるか、または、クリップ情報ファイルに対応するクリップストリームファイルが、ディスク１０１に存在しないと判定された場合、即ち、例えば、”PLAYLIST.DAT”ファイルにしたがった再生に必要なクリップ情報ファイルやクリップストリームファイルが、ディスク１０１に記録されていない場合、ビデオコンテンツ再生プログラム２１０は、ディスク１０１に対応していないと判断して、ステップＳ１０２に進み、上述したエラー処理が行われ、再生前処理を終了する。

【０３９５】

一方、ステップＳ１０７において、ステップＳ１０５で認識したクリップ情報ファイルの読み込みに成功し、かつ、クリップ情報ファイルに対応するクリップストリームファイ

ルが、ディスク１０１に存在すると判定された場合、プレイヤ制御モジュール２１２は、初期化処理を終了し、ステップＳ１０８に進む。

【０３９６】

ステップＳ１０８では、スクリプト制御モジュール２１１が、”SCRIPT.DAT”ファイルの解釈と実行を開始する。

【０３９７】

例えば、いま、スクリプト制御モジュール２１１が、”SCRIPT.DAT”ファイルを実行することにより、１番目のPlayList()(PlayList#0）の再生が、プレイヤ制御モジュール２１２に指示されたとすると、図３４の再生処理が行われる。

【０３９８】

「再生処理」

即ち、図３４は、ビデオコンテンツ再生プログラム２１０が行う再生処理を説明するフローチャートである。

【０３９９】

「再生準備処理」

再生処理では、プレイヤ制御モジュール２１２は、まずステップＳ１２１とＳ１２２において、スクリプト制御モジュール２１１から再生を指示されたPlayList()、即ち、１番目のPlayList()(PlayList#0）の再生準備処理を行う。

【０４００】

即ち、プレイヤ制御モジュール２１２は、ステップＳ１２１において、１番目のPlayList#0に含まれる１番目のPlayItem#0のIN_time（図８）を確認して、ステップＳ１２２に進み、１番目のPlayList#0に含まれる１番目のPlayItem#0によって再生されるクリップストリームファイル”00001.PS”上の、そのPlayItem#0のIN_timeに相当する、再生を開始する再生開始位置を調査する。

【０４０１】

ここで、PlayItem()のIN_time（図８）が、クリップストリームファイルの先頭を指し示している場合には、クリップストリームファイルの先頭から、プログラムストリームを読み出せば良いが、IN_timeが、クリップストリームファイルの先頭以外を指し示している場合には、IN_timeに対応する位置を探し出し（調査し）、そこからプログラムストリームを読み出す必要がある。

【０４０２】

具体的には、図２９に示した場合、１番目のPlayList#0に含まれる１番目のPlayItem#0のIN_timeは、180,090である。プレイヤ制御モジュール２１２は、１番目のPlayList#0に含まれる１番目のPlayItem#0によって再生されるクリップストリームファイル”00001.CLP”の、図３１に示したEP_map()から、PlayItem#0のIN_timeである180,090に適した再生開始位置を探し出す。

【０４０３】

即ち、プレイヤ制御モジュール２１２は、例えば、EP_map()に記述されたデコード開始可能点を表すPTS_EP_startのうちの、式PTS_EP_start≦IN_timeを満たす最大のPTS_EP_startを、二分探索（バイナリサーチ）等を用いて検索する。ここで、IN_time以下のPTS_EP_startを検索するのは、IN_timeによって表される位置が、デコード開始可能点であるとは限らないからである。

【０４０４】

いまの場合、IN_timeは、上述したように、180,090である。また、１番目のPlayList#0に含まれる１番目のPlayItem#0によって再生されるクリップストリームファイル”00001.CLP”の、図３１に示したEP_map()においては、式PTS_EP_start≦IN_timeを満たす最大のPTS_EP_startとして、180,090が記述されている。従って、プレイヤ制御モジュール２１２では、図３１に示したEP_map()から、その180,090となっているPTS_EP_startが検索される。

【０４０５】

さらに、プレイヤ制御モジュール２１２では、その検索されたPTS_EP_startに対応するRPN_EP_startである305（セクタ）が読み出され、その305であるRPN_EP_startによって表されるクリップストリームファイル”00001.PS”上の位置が、再生開始位置として決定される。

【０４０６】

プレイヤ制御モジュール２１２は、以上のようにして再生開始位置を決定すると、ステップＳ１２２からＳ１２３に進み、タイムコードを表示するように、グラフィクス処理モジュール２１９を制御する。グラフィクス処理モジュール２１９は、プレイヤ制御モジュール２１２の制御にしたがい、タイムコード（のビデオデータ）を生成してビデオ出力モジュール２２０に出力する。これにより、タイムコードの表示が開始される。

【０４０７】

ここで、ステップＳ１２３で表示が開始されるタイムコードは、例えば、PlayList()の先頭を00:00:00（時間:分:秒）に換算した値とする。なお、タイムコードとともに、またはタイムコードではなく、チャプタやインデクスの番号を表示するようにしても良い。

【０４０８】

「PlaylistMark()の解析処理」

ステップＳ１２３でタイムコードの表示が開始された後は、ステップＳ１２４に進み、プレイヤ制御モジュール２１２は、スクリプト制御モジュール２１１から再生を指示されたPlayList()、即ち、１番目のPlayList()(PlayList#0）に記述されているPlayListMark()（図９）を解析する解析処理を行う。

【０４０９】

具体的には、プレイヤ制御モジュール２１２は、既に読み込んである”PLAYLIST.DAT”ファイルにおける１番目のPlayList#0の、図３２上側に示したPlayListMark()において、number_of_PlayList_marksが7になっていることから、そのPlayList#0に含まれるMark()の数が７であることを認識する。

【０４１０】

さらに、プレイヤ制御モジュール２１２は、PlayList#0に含まれる、図３２上側の７つのMark()を解析し、そのref_to_PlayItem_idから、７つのMark()のうちの１番目から４番目までの４つのMark()が、PlayList#0の１番目のPlayItem()（PlayItem#0）に属していることを認識する。

【０４１１】

その後、プレイヤ制御モジュール２１２は、PlayList#0の１番目のPlayItem#0に属している４つのMark()のmark_time_stampを取り出し、要素数が4の配列として、デコード制御モジュール２１４に渡す。即ち、これにより、図３２上側の７つのMark()のうちの１番目から４番目までの４つのMark()それぞれのmark_time_stampである{180,090}、{5,580,090}、{10,980,090}、{16,380,090}の４つの時刻が、プレイヤ制御モジュール２１２から、デコード制御モジュール２１４に渡される。このときこれら時刻の属性は「マーク処理」であることも、プレイヤ制御モジュール２１２から、デコード制御モジュール２１４に伝えられる。デコード制御モジュール２１４は、計時部２１４Ａで計時している時刻が、「マーク処理」の属性の時刻に一致したとき、その旨を示すメッセージ、「マーク処理」の属性の時刻に一致した時刻、および「マーク処理」の属性を、プレイヤ制御モジュール２１２に伝える。

【０４１２】

「再生するエレメンタリストリームの決定処理」

次に、ステップＳ１２４からＳ１２５に進み、プレイヤ制御モジュール２１２は、再生するエレメンタリストリームを決定する。

【０４１３】

即ち、プレイヤ制御モジュール２１２は、スクリプト制御モジュール２１１から再生を指示されたPlayList()である１番目のPlayList#0における１番目のPlayItem#0（図２９）のClip_Information_fime_nameにファイル名が記述されている、図３０のクリップ情報フ

ァイル”00001.CLP”において、number_of_streamsが4になっていることから、対応するクリップストリームファイル”00001.PS”に、４本のエレメンタリストリームが多重化されていることを認識する。さらに、プレイヤ制御モジュール２１２は、その４本のエレメンタリストリームに対する、図３０のクリップ情報ファイル”00001.CLP”のStaticInfo()のstream_idと必要なprivate_stream_idを順に調査し、その４本のエレメンタリストリームが、１本のビデオストリーム、１本のATRACオーディオストリーム、および２本の字幕ストリームであることを認識する。即ち、クリップストリームファイル”00001.PS”に多重化されている各属性のエレメンタリストリームの本数が認識される。

【０４１４】

なお、クリップストリームファイルに多重化されている各属性のエレメンタリストリームの本数の情報は、再生中でのエレメンタリストリームの切り替え（オーディオ切り替え、字幕切り替え等）に使用される。また、字幕ストリームは、クリップストリームファイル中に存在しない（コンテンツに字幕が含まれない）場合があり、字幕ストリームが存在するかどうかの判断に、「字幕ストリーム」の属性のエレメンタリストリームの本数の情報が使用される。

【０４１５】

プレイヤ制御モジュール２１２は、以上のようなStaticInfo()の調査の結果に基づいて、再生するエレメンタリストリームを選択、決定するが、いまの場合、クリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームの中に、「ビデオストリーム」と「オーディオストリーム」の属性のエレメンタリストリームは、それぞれ１本しかないので、「ビデオストリーム」と「オーディオストリーム」の属性のエレメンタリストリームについては、選択の余地がなく、その１本のビデオストリームとオーディオストリーム（ATRACオーディオストリーム）が、再生するエレメンタリストリームとして決定される。

【０４１６】

また、「字幕ストリーム」の属性のエレメンタリストリームについては、クリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームの中に２本存在するので、その２本の字幕ストリームのうちのいずれか１本の字幕ストリームが、再生するエレメンタリストリームとして選択、決定される。ここでは、例えば、２本の字幕ストリームのうちの、クリップ情報ファイル”00001.CLP”での出現順で最初の字幕ストリームが選択されることとする。

【０４１７】

ここで、上述のように、クリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームの属性と本数を認識するにあたっては、その４本のエレメンタリストリームそれぞれを特定する必要があるが、プレイヤ制御モジュール２１２は、クリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームの特定を、stream_idと必要なprivate_stream_idによって行う。

【０４１８】

即ち、プレイヤ制御モジュール２１２は、クリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームのうちの、「ビデオストリーム」の属性のエレメンタリストリームであるビデオストリームを、図３０でクリップ情報ファイル”00001.CLP”について説明したように、0xE0となっているstream_idで特定する。

【０４１９】

また、プレイヤ制御モジュール２１２は、クリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームのうちの、「オーディオストリーム」の属性のエレメンタリストリームであるATRACオーディオストリームを、図３０でクリップ情報ファイル”00001.CLP”について説明したように、0xBDとなっているstream_id、および0x00となっているprivate_stream_idで特定する。

【０４２０】

さらに、プレイヤ制御モジュール２１２は、クリップストリームファイル”00001.PS”

に多重化されている４本のエレメンタリストリームにおける「字幕ストリーム」の属性のエレメンタリストリームである２本の字幕ストリームそれぞれを、図３０でクリップ情報ファイル”00001.CLP”について説明したように、0xBDとなっているstream_id、および0x80となっているprivate_stream_idと、0xBDとなっているstream_id、および0x81となっているprivate_stream_idで、それぞれ特定する。

【０４２１】

以上のように、クリップストリームファイルに対応するクリップ情報ファイルのメタデータとして記述されるstream_idとprivate_stream_idの組み合わせによって、そのクリップストリームファイルに多重化されているエレメンタリストリームを特定することができる。

【０４２２】

ここで、stream_idとprivate_stream_idの組み合わせは、MPEG2-Systemの多重化を拡張するために設けたメカニズムである。このstream_idとprivate_stream_idの組み合わせを、メタデータ（データベース）において、エレメンタリストリームを特定するために使用することにより、エレメンタリストリームを確実に特定することが可能になる。また、将来、private_stream_idの意味を拡張し、対応するエレメンタリストリームの本数や種類（属性）を増やした場合にも現在のメカニズムをそのまま使用可能であるため、拡張性において勝っている。

【０４２３】

即ち、例えば、BD(Blue ray Disc)規格では、データの特定に、MPEG2規格のトランスポートストリーム(Transport Stream)のPID(Packet ID)が用いられるため、MPEG2規格に拘束される。また、例えば、DVD-Video規格では、private_stream_idに類似するsub_stream_idが定義されているが、sub_stream_idは、ストリームの特定のためにデータベース上に記述することができるようにはなっておらず、8本あるいは32本のストリーム情報を記述する固定的な領域に記述することができるにすぎないため（たとえばVI4-49、Table 4.2.1-2（VTS-AST_ATRT）やVI4-52、Table 4.2.1-3（VTS_SPST_ATRT）等参照）、拡張性に乏しい。

【０４２４】

これに対して、stream_idとprivate_stream_idの組み合わせは、メタデータが記述される、例えば、図１２のクリップ情報ファイルClip()において、number_of_streamsで表すことができる数だけ記述することができ、従って、クリップストリームファイルに多重化されているエレメンタリストリームを、その数によらず（但し、number_of_streamsで表すことができる数の範囲）、クリップ情報ファイルClip()に記述されたメタデータとしてのstream_idとprivate_stream_idの組み合わせから特定することが可能となる。

【０４２５】

なお、本実施の形態では、stream_idとprivate_stream_idの組み合わせは、図１２のクリップ情報ファイルにおいて、対応するクリップストリームファイルに多重化されているエレメンタリストリームを特定するのに使用される他、例えば、図９のPlayListMark()におけるentry_ES_stream_idとentry_ES_private_stream_idの組み合わせとして、Mark()を関連付けるエレメンタリストリームの特定にも使用される。さらに、stream_idとprivate_stream_idの組み合わせは、その他、例えば、図１６のEP_map()において、デコード可能開始点の情報を記述するエレメンタリストリームの特定にも使用される。

【０４２６】

「出力属性の制御処理」

その後、ステップＳ１２５からＳ１２６に進み、プレイヤ制御モジュール２１２は、再生対象のエレメンタリストリーム、即ち、ステップＳ１２５で再生すると決定したエレメンタリストリームの出力属性の制御処理を行う。

【０４２７】

具体的には、プレイヤ制御モジュール２１２は、まず、再生対象のエレメンタリストリーム、即ち、ステップＳ１２５で再生すると決定したビデオストリーム、ATRACオーディ

オストリーム、字幕ストリームそれぞれについて、出力属性が記述されるDynamicInfo()（図１５）の数を表すnumber_of_DynamicInfo（図１２）を調査する。

【０４２８】

ここで、いまの場合、再生対象のビデオストリーム、ATRACオーディオストリーム、字幕ストリームは、クリップストリームファイル”00001.PS”に多重化されているエレメンタリストリームであり、それらのnumber_of_DynamicInfoは、図３０の”00001.CLP”で説明したように、いずれも0になっている。このように、再生対象のエレメンタリストリームのすべてについて、number_of_DynamicInfoが0である場合、プレイヤ制御モジュール２１２は、再生対象のエレメンタリストリームの出力属性の制御処理としては、特に処理を行わない。

【０４２９】

なお、再生対象のエレメンタリストリームについてのnumber_of_DynamicInfoが0でない場合に、そのエレメンタリストリームの出力属性の制御として行われる処理については、後述する。

【０４３０】

「再生開始の準備処理」

ステップＳ１２６の処理後は、ステップＳ１２７に進み、プレイヤ制御モジュール２１２は、再生対象のエレメンタリストリームの再生開始の準備処理を行う。

【０４３１】

即ち、プレイヤ制御モジュール２１２は、コンテンツデータ供給モジュール２１３に対し、再生対象のエレメンタリストリームが多重化されているクリップストリームファイル”00001.PS”のファイル名と、ステップＳ１２２で決定した再生開始位置であるEP_map()に記述されたRPN_EP_start（=305）を与える。

【０４３２】

さらに、プレイヤ制御モジュール２１２は、再生対象のエレメンタリストリームが多重化されているクリップストリームファイル”00001.PS”に格納されたプログラムストリームの、バッファ制御モジュール２１５への供給が開始される前に、バッファ制御モジュール２１５を初期化する。

【０４３３】

具体的には、バッファ制御モジュール２１５（図５）では、データ先頭ポインタ記憶部２３１に記憶されるデータ先頭ポインタ、データ書き込みポインタ記憶部２３２に記憶されるデータ書き込みポインタ、ビデオ読み出しポインタ記憶部２４１に記憶されるビデオ読み出しポインタ、オーディオ読み出しポインタ記憶部２５１に記憶されるオーディオ読み出しポインタ、字幕読み出しポインタ記憶部２６２に記憶される字幕読み出しポインタに、同じ値が代入される。

【０４３４】

これにより、データ先頭ポインタ記憶部２３１に記憶されたデータ先頭ポインタと、データ書き込みポインタ２３２に記憶されたデータ書き込みポインタとは、バッファ制御モジュール２１５のバッファ２１５Ａの同一の位置を指す。これは、バッファ２１５Ａに、有効なデータが蓄積されていない状態を表す。

【０４３５】

さらに、プレイヤ制御モジュール２１２は、再生対象のエレメンタリストリームを識別（特定）するための識別情報としてのstream_id、さらには、必要に応じて、private_stream_idを、バッファ制御モジュール２１５に供給する。

【０４３６】

即ち、上述したように、再生対象のエレメンタリストリームのうちの、「ビデオストリーム」の属性のビデオストリームは、0xE0となっているstream_idによって特定され、「オーディオストリーム」の属性のATRACオーディオストリームは、0xBDとなっているstream_id、および0x00となっているprivate_stream_idによって特定され、「字幕ストリーム」の属性の字幕ストリームは、0xBDとなっているstream_id、および0x80となっているpri

vate_stream_idによって特定される。プレイヤ制御モジュール２１２は、これらのstream_idとprivate_stream_idを、バッファ制御モジュール２１５に供給する。

【０４３７】

バッファ制御モジュール２１５（図５）では、ビデオ読み出し機能部２３３が、プレイヤ制御モジュール２１２からの、ビデオストリームについての0xE0となっているstream_idを、stream_idレジスタ２４２に記憶させる。また、オーディオ読み出し機能部２３４が、プレイヤ制御モジュール２１２からの、0xBDとなっているstream_idと、0x00となっているprivate_stream_idを、stream_idレジスタ２５２とprivate_stream_idレジスタ２５３に、それぞれ記憶させる。さらに、字幕読み出し機能部２３５が、プレイヤ制御モジュール２１２からの、0xBDとなっているstream_idと、0x80となっているprivate_stream_idを、stream_idレジスタ２６３とprivate_stream_idレジスタ２６４に、それぞれ記憶させる。

【０４３８】

なお、プレイヤ制御モジュール２１２は、バッファ制御モジュール２１５に供給した再生対象のエレメンタリストリームのstream_idとprivate_stream_idを、今後の処理のために記憶する。プレイヤ制御モジュール２１２は、これらのstream_idやprivate_stream_idを、後述するストリーム切り替えを要求するメッセージの発生時や、後述するマーク処理において現在再生中のストリームを特定するために使用する。

【０４３９】

プレイヤ制御モジュール２１２は、バッファ制御モジュール２１５（図５）の初期化として、さらに、再生対象のエレメンタリストリームが多重化されているクリップストリームファイルに応じた値の字幕読み出し機能フラグを、字幕読み出し機能フラグ記憶部２６１にセットする。

【０４４０】

即ち、いまの場合、再生対象のエレメンタリストリームが多重化されているクリップストリームファイル”00001.PS”には、字幕ストリームが含まれるため、字幕読み出し機能部２３５を機能させるために、値が１の字幕読み出し機能フラグが、字幕読み出し機能フラグ記憶部２６１にセットされる。なお、再生対象のエレメンタリストリームが多重化されているクリップストリームファイルに字幕ストリームが含まれていない場合、字幕読み出し機能フラグ記憶部２６１には、値が０の字幕読み出し機能フラグがセットされる。この場合、字幕読み出し機能部２３５は機能しない（特に処理を行わない）。

【０４４１】

また、プレイヤ制御モジュール２１２は、スクリプト制御モジュール２１１から再生を指示された１番目のPlayList#0に含まれる１番目のPlayItem#0（図２９）のIN_timeである180,090と、OUT_timeである27,180,090とを、デコード制御モジュール２１４に対して与える。デコード制御モジュール２１４では、IN_timeは、PlayItem()によって再生されるクリップのデコード開始の制御に、OUT_timeは、そのクリップのデコード終了、さらには、後述するPlayItem乗り換えの制御に、それぞれ使用される。

【０４４２】

さらに、プレイヤ制御モジュール２１２は、グラフィクス処理モジュール２１９に対する字幕ストリームの表示方式の指示を初期化する。即ち、プレイヤ制御モジュール２１２は、字幕ストリームの表示方式を、例えば、デフォルトの表示方式とするように、グラフィクス処理モジュール２１９を制御する。

【０４４３】

「データ読み込み開始」

その後、ステップＳ１２７からＳ１２８に進み、プレイヤ制御モジュール２１２は、コンテンツデータ供給モジュール２１３を制御し、これにより、コンテンツデータ供給モジュール２１３は、オペレーティングシステム２０１の機能を使用して、再生対象のエレメンタリストリームが多重化されたプログラムストリームが格納されたクリップストリームファイルを読み出す。すなわち、コンテンツデータ供給モジュール２１３は、ディスク１

０１（図６）の”VIDEO”ディレクトリの下にある”STREAM”ディレクトリのクリップストリームファイル”00001.PS”を指定し、さらに、ステップＳ１２２で決定された再生開始位置である305セクタを指定して、オペレーティングシステム２０１に対してファイル読み出しを要求する。また、コンテンツデータ供給モジュール２１３は、ディスク１０１から読み出したデータを、バッファ制御モジュール２１５に供給するように指定する。

【０４４４】

これにより、ディスク１０１からの、クリップストリームファイル”00001.PS”に格納されたプログラムストリームの読み出しが開始され、そのプログラムストリームは、バッファ制御モジュール２１５に供給される。

【０４４５】

バッファ制御モジュール２１５（図５）は、ディスク１０１から読み出されて供給されたプログラムストリームを、バッファ２１５Ａのデータ書き込みポインタ記憶部２３２のデータ書き込みポインタが指す位置に書き込み、書き込んだデータのサイズだけデータ書き込みポインタをインクリメントする。

【０４４６】

ここで、以下、特に断らない限り、コンテンツデータ供給モジュール２１３は、バッファ制御モジュール２１５のバッファ２１５Ａに空きがあれば、ディスク１０１からデータを読み出し、バッファ制御モジュール２１５のバッファ２１５Ａに供給して記憶させることとする。従って、バッファ２１５Ａには、常時、十分なデータが蓄積されているとする。

【０４４７】

「デコーダ制御開始」

以上のようにして、ディスク１０１からのデータの読み出しが開始され、そのデータがバッファ制御モジュール２１５のバッファ２１５Ａに蓄積され始めると、ステップＳ１２８からＳ１２９に進み、デコード制御モジュール２１４は、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、字幕デコーダ制御モジュール２１８を制御し、デコード動作の前段階として、バッファ２１５Ａからのデータの読み出しを開始させる。

【０４４８】

即ち、これにより、ビデオデコーダ制御モジュール２１６は、バッファ制御モジュール２１５（図５）のビデオ読み出し機能部２３３にデータを要求し、その要求に応じてバッファ制御モジュール２１５から渡される、バッファ２１５Ａに記憶された１つのビデオアクセスユニット、そのビデオアクセスユニットに付加されているPTSとDTS（以下、適宜、タイムスタンプという）、およびデコード開始可能点の直前に配置されているprivate_stream_2のPES_packet()に記述された情報（以下、適宜、付加情報ともいう）であるpic_struct_copyや、au_ref_flag、AU_lengthなどを得る。なお、タイムスタンプは、ビデオデコーダ制御モジュール２１６がビデオアクセスユニットを得る毎に、ビデオデコーダ制御モジュール２１６からデコード制御モジュール２１４に渡される。

【０４４９】

ここで以下の時刻更新のために使用するpic_struct_copyは、ビデオ読み出し機能部２３３から渡されたものであるが、構文解析の結果得られたビットストリーム中に含まれるpic_structを使用することも可能である。

【０４５０】

一方、オーディオデコーダ制御モジュール２１７も、バッファ制御モジュール２１５（図５）のオーディオ読み出し機能部２３４にデータを要求し、その要求に応じてバッファ制御モジュール２１５から渡される、バッファ２１５Ａに記憶された１つの(ATRAC)オーディオアクセスユニットと、そのオーディオアクセスユニットに付加されているタイムスタンプ（PTS, DTS）を得る。なお、タイムスタンプは、オーディオデコーダ制御モジュール２１７がオーディオアクセスユニットを得る毎に、オーディオデコーダ制御モジュール２１７からデコード制御モジュール２１４に渡される。

【０４５１】
さらに、字幕デコーダ制御モジュール２１８は、バッファ制御モジュール２１５（図５）の字幕読み出し機能部２３５にデータを要求し、その要求に応じてバッファ制御モジュール２１５から渡される、バッファ２１５Ａに記憶された１つの字幕アクセスユニットと、その字幕アクセスユニットに付加されているタイムスタンプを得る。なお、タイムスタンプは、字幕デコーダ制御モジュール２１８が字幕アクセスユニットを得る毎に、字幕デコーダ制御モジュール２１８からデコード制御モジュール２１４に渡される。また、再生対象のエレメンタリストリームに、字幕ストリームが存在しない場合や、バッファ２１５Ａに、字幕アクセスユニットが記憶されていない場合は、バッファ制御モジュール２１５から字幕デコーダ制御モジュール２１８には、データは渡されない。

【０４５２】
ここで、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、および字幕デコーダ制御モジュール２１８は、バッファ制御モジュール２１５に対してデータを要求する毎に、そのデータの要求に対する結果を、デコード制御モジュール２１４に渡す。

【０４５３】
また、バッファ制御モジュール２１５から、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、および字幕デコーダ制御モジュール２１８に対してデータが渡されるときの、そのデータのバッファ２１５Ａからの読み出しの詳細については、後述する。

【０４５４】
「デコード開始」
以上のように、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、字幕デコーダ制御モジュール２１８が、バッファ制御モジュール２１５のバッファ２１５Ａからデータを読み出し始めると、ステップＳ１２９からＳ１３０に進み、そのデータのデコードが開始される。

【０４５５】
即ち、デコード制御モジュール２１４は、ステップＳ１２７でプレイヤ制御モジュール２１２から与えられた、PlayList#0に含まれる１番目のPlayItem#0のIN_timeである180,090、さらには、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、字幕デコーダ制御モジュール２１８からステップＳ１２９で説明したように渡されるタイムスタンプに基づき、同期を確保するために必要であればタイミングをずらして、デコード開始を、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、および字幕デコーダ制御モジュール２１８に指令する。

【０４５６】
ここで、同期を確保するためのタイミングをずらしたデコード開始の指令の方法は、例えば、特許第3496725号に記載されており、簡単には、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、字幕デコーダ制御モジュール２１８それぞれから渡されたタイムスタンプのうちの最小値を、計時部２１４Ａによって計時される時刻の初期値として設定して時刻の計時を開始し、計時部２１４Ａによって計時される時刻と、タイムスタンプとが一致した時点で、デコード開始を指令する方法がある。

【０４５７】
ビデオデコーダ制御モジュール２１６は、デコード制御モジュール２１４からのデコード開始の指示を受け、その指示に応じて、バッファ制御モジュール２１５（図５）のビデオ読み出し機能部２３３から得た１つのビデオアクセスユニットを、ビデオデコーダ１１６（図１）に渡してデコードさせる。さらに、ビデオデコーダ制御モジュール２１６は、ビデオデコーダ１１６によるデコードの結果得られるビデオデータを、グラフィクス処理モジュール２１９に供給する。

【０４５８】
以後、ビデオデコーダ制御モジュール２１６は、バッファ制御モジュール２１５のビデ

オ読み出し機能部２３３から得られる１ずつのビデオアクセスユニットを、ビデオデコーダ１１６で順次デコードし、そのデコードの結果得られるビデオデータを、グラフィクス処理モジュール２１９に供給していく。

【０４５９】

このとき、ビデオデコーダ１１６内部では、デコードと出力でビデオデータの順番入れ替え（リオーダリング）が発生している。例えば図３５に示すように、デコードの順番はI1，B0，P3，B2，P5，B4だが、表示の順番はB0，I1，B2，P3，B4，P5になることがある。このためにビデオデコーダ１１６内にはデコード後の画像をstoreしておくためのdecoded picture bufferが設けられている。図３５において、Inは、ｎ番目のIピクチャを示し、Bnは、ｎ番目のBピクチャを示し、Pnは、ｎ番目のPピクチャを示している。

【０４６０】

一方、オーディオデコーダ制御モジュール２１７も、デコード制御モジュール２１４からのデコード開始の指示を受け、その指示に応じて、バッファ制御モジュール２１５（図５）のオーディオ読み出し機能部２３４から得た１つのオーディオアクセスユニットを、オーディオデコーダ１１７（図１）に渡してデコードさせる。さらに、オーディオデコーダ制御モジュール２１７は、オーディオデコーダ１１７によるデコードの結果得られるオーディオデータを、オーディオ出力モジュール２２１に供給する。

【０４６１】

以後、オーディオデコーダ制御モジュール２１７は、バッファ制御モジュール２１５のオーディオ読み出し機能部２３４から得られる１ずつのオーディオアクセスユニットを、オーディオデコーダ１１７で順次デコードし、そのデコードの結果得られるオーディオデータを、オーディオ出力モジュール２２１に供給していく。

【０４６２】

また、字幕デコーダ制御モジュール２１８も、デコード制御モジュール２１４からのデコード開始の指示を受け、その指示に応じて、バッファ制御モジュール２１５（図５）の字幕読み出し機能部２３５から得た１つの字幕アクセスユニットを、内部に持つ字幕デコードソフトウェアでデコードし、そのデコードの結果得られる字幕データ（字幕の画像データ）を、グラフィクス処理モジュール２１９に供給する。

【０４６３】

以後、字幕デコーダ制御モジュール２１８は、バッファ制御モジュール２１５の字幕読み出し機能部２３５から得られる１ずつの字幕アクセスユニットを、内部に持つ字幕デコードソフトウェアで順次デコードし、そのデコードの結果得られる字幕データを、グラフィクス処理モジュール２１９に供給していく。

【０４６４】

「グラフィクス処理」

その後、ステップＳ１３０からＳ１３１に進み、グラフィクス処理モジュール２１９は、上述したようにして、ビデオデコーダ制御モジュール２１６から供給されるビデオデータ、さらには、必要に応じて、字幕デコーダ制御モジュール２１８から供給される字幕データを対象に、グラフィクス処理を行う。

【０４６５】

即ち、グラフィクス処理モジュール２１９は、まず字幕デコーダ制御モジュール２１８からの字幕データを、プレイヤ制御モジュール２１２からの表示方式の指示に従って、拡大や縮小等する字幕処理を行う。プレイヤ制御モジュール２１２から、表示方式の指示がない場合、またはデフォルトの表示方式の指示があった場合、グラフィクス処理モジュール２１９は、字幕デコーダ制御モジュール２１８からの字幕データを、そのまま保存する。

【０４６６】

さらに、グラフィクス処理モジュール２１９は、ビデオデコーダ制御モジュール２１６からのビデオデータと、字幕デコーダ制御モジュール２１８からの字幕データ、または字幕処理後の字幕データとを加算し、ビデオデコーダ制御モジュール２１６からのビデオデ

ータに字幕データがオーバーレイされた出力ビデオデータを得て、ビデオ出力モジュール２２０に供給する。

【０４６７】

なお、グラフィクス処理モジュール２１９は、スクリプト制御モジュール２１１やプレイヤ制御モジュール２１２から、例えば、メニューや、メッセージ、タイムコード、チャプタまたはインデクスの番号等の情報の表示の指示を受けた場合は、その情報を生成し、出力ビデオデータにオーバーレイして、ビデオ出力モジュール２２０に供給する。

【０４６８】

「出力処理」

ステップＳ１３１の処理後は、ステップＳ１３２に進み、ビデオ出力モジュール２２０は、ステップＳ１３１で説明したようにしてグラフィクス処理モジュール２１９から供給される出力ビデオデータを、FIF0２２０Ａに順次記憶させ、そのFIF0２２０Ａに記憶された出力ビデオデータを、あらかじめ決められた出力レートで順次出力する。

【０４６９】

ビデオ出力モジュール２２０は、FIF0２２０Ａの記憶容量（残量）に余裕がある限り、グラフィクス処理モジュール２１９からの出力ビデオデータを受け入れるが、余裕がない場合には、出力ビデオデータの受け入れの停止を、グラフィクス処理モジュール２１９に要求する。これにより、グラフィクス処理モジュール２１９は、処理を停止するとともに、処理の停止を、ビデオデコーダ制御モジュール２１６および字幕デコーダ制御モジュール２１８に要求する。これにより、ビデオデコーダ制御モジュール２１６および字幕デコーダ制御モジュール２１８が処理を停止する。

【０４７０】

ビデオ出力モジュール２２０は、出力ビデオデータの受け入れの停止を、グラフィクス処理モジュール２１９に要求した後に、FIF0２２０Ａからの出力ビデオデータの出力が進み、FIF0２２０Ａに余裕ができた時点で、出力ビデオデータの受け入れを、グラフィクス処理モジュール２１９に要求する。この要求は、出力ビデオデータの受け入れの停止の要求と同様に、グラフィクス処理モジュール２１９から、ビデオデコーダ制御モジュール２１６および字幕デコーダ制御モジュール２１８に伝えられる。これにより、グラフィクス処理モジュール２１９、さらには、ビデオデコーダ制御モジュール２１６および字幕デコーダ制御モジュール２１８は、停止していた処理を再開する。

【０４７１】

一方、オーディオ出力モジュール２２１も、ステップＳ１３０で説明したようにしてオーディオデコーダ制御モジュール２１７から供給されるオーディオデータを、FIF0２２１Ａに順次記憶させ、そのFIF0２２１Ａに記憶されたオーディオデータを、あらかじめ決められた出力レート（サンプリング周波数）で順次出力する。

【０４７２】

オーディオ出力モジュール２２１は、FIF0２２１Ａの記憶容量（残量）に余裕がある限り、オーディオデコーダ制御モジュール２１７からのオーディオデータを受け入れるが、余裕がない場合には、オーディオデータの受け入れの停止を、オーディオデコーダ制御モジュール２１７に要求する。これにより、オーディオデコーダ制御モジュール２１７は、処理を停止する。

【０４７３】

オーディオ出力モジュール２２１は、オーディオデータの受け入れの停止を、オーディオデコーダ制御モジュール２１７に要求した後に、FIF0２２１Ａからのオーディオデータの出力が進み、FIF0２２１Ａに余裕ができた時点で、オーディオデータの受け入れを、オーディオデコーダ制御モジュール２１７に要求する。これにより、オーディオデコーダ制御モジュール２１７は、停止していた処理を再開する。

【０４７４】

以上のようにして、ビデオ出力モジュール２２０およびオーディオ出力モジュール２２１からデータが出力されるにつれて、エレメンタリストリームのデコードが行われていく

。
【０４７５】
［ビデオデコーダ１１６の内部構造の説明］
図３６にビデオデコーダ１１６の内部構造を示す。この例ではビデオデコーダ１１６の内部はビデオデコードエンジン１１６ＡとDPB（Decoded Picture Buffer）１１６Ｂで構成されている。Decoded Picture Buffer１１６Ｂの内部はさらに細分化されており、DPB１１６Ｂ－１乃至DPB１１６Ｂ－ｎ（以降において、特に区別する必要が無い場合、単にDBP１１６Ｂと称するものとする）で構成されている。さらに、図３７で示されるように、DPB１１６Ｂの内部はビデオバッファ３０１と付加情報バッファ３０２に分けられている。
【０４７６】
ビデオデコードエンジン１１６Ａはビデオデータのデコード処理に際して、デコード中のビデオデータを一時保存する、あるいは将来の参照画像として使用するために保持する等の用途のために、DPB１１６Ｂのビデオバッファ３０１を使用する。このとき、付加情報バッファ３０２には、ビデオバッファ３０１に保存されたビデオデータに対応するべき、ビデオ読み出し機能部２３３から得た付加情報や、アクセスユニットを構文解析して得られたパラメータ（例えばpic_struct等）が記録される。
【０４７７】
次に、図１のディスク再生装置がディスク１０１を再生するときの全体の処理または動作の流れは、図３３および図３４で説明したとおりであるが、以下、ディスク再生装置においてディスク１０１の再生が行われているときの、その他の処理または動作について説明する。
【０４７８】
［時刻情報をデコード制御モジュール２１４に渡す］
以下、時計（計時部２１４Ａ）の更新について説明する。ビデオデコーダ制御モジュール２１６は、入力したビデオアクセスユニット（ビデオのストリームデータ）をビデオデコーダ１１６に指示してデコードさせる。ビデオデコーダ１１６によるデコード及びリオーダリング処理の後、１フレーム（２フィールド）分のビデオデータがグラフィクス処理モジュール２１９に出力されると同時に、該当ビデオデータのタイムスタンプ（PTS/DTS）とpic_struct情報がビデオデコーダ制御モジュール２１６からデコード制御モジュール２１４に渡される。
【０４７９】
アクセスユニットのpic_structが１あるいは２の場合にはこれらは1フィールド分のアクセスユニットであるため、１フレーム分とするために２つ分のアクセスユニットの出力が行われる段階で先行フィールドのpic_structと、先行フィールドのアクセスユニットにタイムスタンプが有った場合にはタイムスタンプが、ビデオデコーダ制御モジュール２１６からデコード制御モジュール２１４に渡される。ここで先行フィールドがタイムスタンプを持たない場合には、タイムスタンプがないという情報が渡される。上述したように孤立フィールドは許していないため、pic_structが１あるいは２のフィールドの直後にはpic_structが２あるいは１のフィールドが置かれている。この２のフィールドを１にまとめに扱うときには、先行しているフィールドのタイムスタンプを代表値として使用する。
【０４８０】
また、アクセスユニットのpic_structが０，３，４，５，６の場合には、１のアクセスユニットの出力が行われる段階で１のpic_structと、該当アクセスユニットにタイムスタンプが有った場合にはタイムスタンプが、ビデオデコーダ制御モジュール２１６からデコード制御モジュール２１４に渡される。タイムスタンプを持たないアクセスユニットでは、タイムスタンプがないという情報が渡される。
【０４８１】
デコード制御モジュール２１４は、受け取ったタイムスタンプとpic_structの情報を用いて計時部２１４Ａを更新する。

【０４８２】
以下、図３８のフローチャートを参照して更新の方法を説明する。
【０４８３】
デコード制御モジュール２１４は、受け取ったアクセスユニットがタイムスタンプが付加されているか否かを判定する（ステップＳ１４１）。例えば、アクセスユニットにタイムスタンプが付加されていた場合、デコード制御モジュール２１４は、タイムスタンプ(PTS)の値を計時部２１４Ａに設定する（ステップＳ１４２）。上述したようにデコード開始直後は必ずタイムスタンプが存在するため、初回でも問題は発生しない。アクセスユニットにタイムスタンプが付加されていなかった場合、現在時刻に前回のpic_structで決まる値を加算する（ステップＳ１４４）。その後、今回のpic_structを次回の処理のために保存し終了する（ステップＳ１４３）。
【０４８４】
pic_structで決まる値は、図３９で示されるように、保存してあったpic_structが０，３あるいは４だった場合、計時部２１４Ａは２フィールド時間分の時間を加算する。またpic_structが５あるいは６であった場合、計時部２１４Ａは３フィールド時間分の時間を加算する。さらに、保存してあったpic_structが１あるいは２だった場合、計時部２１４Ａは２フィールド時間分の時間を加算する。
【０４８５】
このような時刻変更処理により、計時部２１４Ａの示す時刻の値は、ビデオデコーダ制御モジュール２１６からグラフィクス処理モジュール２１９への出力が終了した（１フレーム分の）アクセスユニットの表示開始時刻を示す。すなわち、該当ビデオデータがタイムスタンプを持つ場合にはPTSが代入される。またタイムスタンプを持たない場合には、表示順で直前のビデオデータの表示間隔が加算される。
【０４８６】
この例ではビデオ符号化方式としてAVCを使用しているが、例えばMPEG2-Videoにおいてもrepeat_first_fieldを使用することによりアクセスユニットの表示durationを知ることができる。
【０４８７】
なお、上述したように、この場合も、FIFO２２０Ａの記憶容量に余裕がない場合にはビデオデコーダ制御モジュール２１６からのビデオデータの出力が停止する。この場合には計時部２１４Ａの更新も自動的に停止する。また、FIFO２２０Ａへのビデオデータの更新が再開すると計時部２１４Ａの更新も自動的に再開する。
【０４８８】
すなわち、ユーザからの指示により再生モードがポーズ状態に遷移した場合、ビデオ出力モジュール２２０の更新が止められることにより、連動してビデオデコーダ制御モジュール２１６が停止し、さらにそれとリンクしている時計（計時部２１４Ａ）も連動して止められることを意味している。また、ポーズ状態から通常再生状態へ復帰した場合、ビデオ出力モジュールの更新が許可されることにより、連動してビデオデコーダ制御モジュール２１６の動作およびビデオデータの出力が再開され、さらにそれとリンクしている時計（計時部２１４Ａ）も連動して更新が再開されることを意味している。
【０４８９】
このような動作はスロー再生に対しても期待できる。すなわち、スロー再生はポーズと通常再生を短い周期で交互に繰り返している状態であり、その際にも時計（計時部２１４Ａ）はビデオ出力に同期して更新される。
【０４９０】
なお、この例では計時部２１４Ａの更新をビデオデコーダ制御モジュール２１６からのビデオデータ出力と同期して行うと説明した。しかしながら、ビデオデコーダ制御モジュール２１６以降のディレイ、ここではグラフィクス処理モジュール２１９およびビデオ出力モジュール２２０で生じるディレイが大きい場合、実際にユーザに対して提供されているビデオデータと時計（計時部２１４Ａ）の関係がずれてしまう可能性がある。そのよう

な場合には、時計（計時部２１４Ａ）の更新をビデオ出力モジュール２２０からのビデオデータ出力と同期して行うという方式を採用することによって防ぐことが可能である。

【０４９１】

具体的には、グラフィクス処理モジュール２１９とビデオ出力モジュール２２０およびFIFO２２０Ａでのビデオデータの処理部分に、図３７を参照して説明した付加情報バッファ３０２を追加し、出力にいたるまで、ビデオデータと付加情報を常に一組として扱う。さらに、ビデオ出力モジュール２２０からのビデオデータの出力に際して、対応する付加情報がデコード制御モジュール２１４に渡される。デコード制御モジュールは上述したアルゴリズムにより時計（計時部２１４Ａ）を更新する。

【０４９２】

このような方法をとることにより、ビデオデコーダ以降のディレイの多少にかかわらず、表示されるビデオデータと時計（計時部２１４Ａ）の同期を合わせることが可能になる。

【０４９３】

結果として、ストリームデータを再生する装置で、単独にカウントする時計を備えていない状態であっても、正確にストリームデータを再生することが可能となり、CPU１１２などの処理負荷を低減させることが可能となる。

【０４９４】

［PlayItem乗り換え］

図３３および図３４で説明したようにして、図２９における１番目のPlayList#0の１番目のPlayItem#0の再生が始まるが、PlayList#0によれば、その１番目のPlayItem#0の再生が終了すると、２番目のPlayItem#1の再生が開始される。即ち、PlayItem#0からPlayItem#1にPlayItemを乗り換えるPlayItem乗り換えが行われる。

【０４９５】

次に、図４０のフローチャートを参照して、このPlayItem乗り換えの処理について説明する。

【０４９６】

図３３および図３４で説明したようにして、図２９におけるPlayList#0の１番目のPlayItem#0（のクリップ）の再生が開始されると、デコード制御モジュール２１４（図２）は、その１番目のPlayItem#0の再生が行われている間、内蔵する計時部２１４Ａが計時している時刻を確認し続けている。

【０４９７】

「PlayItem#0の再生終了」

そして、デコード制御モジュール２１４は、計時部２１４Ａが計時している時刻が、図３４のステップ１２７でプレイヤ制御モジュール２１２から与えられた１番目のPlayItem#0のOUT_timeである27,180,090（図２９）に等しくなると、ステップＳ１５１において、デコード中断制御を行い、PlayItem#0の再生を終了する。

【０４９８】

ところで、計時部２１４Ａは、90kHzで変化するものではない場合、すなわち、ビデオデータの出力に応じて時刻を更新するという方式である場合、計時部２１４Ａが計時している時刻は、厳密には、PlayItem#0のOUT_timeと等しくならないことがある。そのようなとき、PlayItem#0のOUT_timeの時刻と計時部２１４Ａの計時する時刻が近傍の値となったタイミングで、デコード中断制御が行われ、PlayItem#0の再生が終了されることになる。尚、この処理については、図５１，図５２を参照して詳細を後述する。

【０４９９】

即ち、デコード制御モジュール２１４は、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、字幕デコーダ制御モジュール２１８を操作して、デコード動作を停止させる。さらに、デコード制御モジュール２１４は、ビデオ出力モジュール２２０を制御し、現在出力中の出力ビデオデータを引き続き出力させる。

【０５００】

また、デコード制御モジュール２１４は、１番目のPlayItem#0の再生が終了した旨のメッセージを、プレイヤ制御モジュール２１２に伝える。

【０５０１】

「PlayItem#1の再生開始」

プレイヤ制御モジュール２１２は、上述したように、図３３のステップＳ１０５で、１番目のPlayList #0に、１番目のPlayItem#0と２番目のPlayItem#1とが存在することを認識しており、デコード制御モジュール２１４から、１番目のPlayItem#0の再生が終了した旨のメッセージが伝えられると、ステップＳ１５１からＳ１５２に進み、２番目のPlayItem#1の再生を、上述した１番目のPlayItem#0における場合と同様にして開始する。

【０５０２】

即ち、２番目のPlayItem#1の再生手順を概説すれば、まず、プレイヤ制御モジュール２１２は、図３４のステップＳ１２２における場合と同様にして、２番目のPlayItem#1について、EP_map()に記述されたRPN_EP_startのうちのいずれかを、再生開始位置として決定する。

【０５０３】

さらに、プレイヤ制御モジュール２１２では、図３４のステップＳ１２４で説明したようにして、２番目のPlayItem#1に属するMark()の認識や、図３４のステップＳ１２５で説明したようにして、PlayItem#1によって再生されるクリップストリームファイル”00002.PS”に多重化されている各属性のエレメンタリストリームの本数の認識、さらには、再生する（再生対象の）エレメンタリストリームの決定等が行われる。

【０５０４】

そして、プレイヤ制御モジュール２１２は、図３４のステップＳ１２７における場合と同様の処理を行う。

【０５０５】

即ち、プレイヤ制御モジュール２１２は、再生開始位置として決定したEP_map()のRPN_EP_startと、再生対象のエレメンタリストリームが多重化されているクリップストリームファイルのファイル名、即ち、いまの場合、２番目のPlayItem#1（図２９）のClip_Information_file_nameに記述された”00002.CLP”に対応するクリップストリームファイル”00002.PS”のファイル名を、コンテンツデータ供給モジュール２１３に対して与える。

【０５０６】

さらに、プレイヤ制御モジュール２１２は、再生対象のエレメンタリストリームが多重化されているクリップストリームファイル”00002.PS”に格納されたプログラムストリームの、バッファ制御モジュール２１５への供給が開始される前に、バッファ制御モジュール２１５を初期化する。

【０５０７】

即ち、これにより、バッファ制御モジュール２１５（図５）において、データ先頭ポインタ記憶部２３１に記憶されるデータ先頭ポインタ、データ書き込みポインタ記憶部２３２に記憶されるデータ書き込みポインタ、ビデオ読み出しポインタ記憶部２４１に記憶されるビデオ読み出しポインタ、オーディオ読み出しポインタ記憶部２５１に記憶されるオーディオ読み出しポインタ、字幕読み出しポインタ記憶部２６２に記憶される字幕読み出しポインタに、同じ値が代入される。

【０５０８】

さらに、プレイヤ制御モジュール２１２は、再生対象のエレメンタリストリームを識別するための識別情報としてのstream_id、さらには、必要に応じて、private_stream_idを、バッファ制御モジュール２１５に供給する。

【０５０９】

バッファ制御モジュール２１５（図５）では、ビデオ読み出し機能部２３３が、プレイヤ制御モジュール２１２からの、再生対象のエレメンタリストリームのうちのビデオストリームについてのstream_idを、stream_idレジスタ２４２に記憶させる。また、オーディオ読み出し機能部２３４が、プレイヤ制御モジュール２１２からの、再生対象のエレメン

タリストリームのうちのオーディオストリームのstream_idとprivate_stream_idを、stream_idレジスタ２５２とprivate_stream_idレジスタ２５３に、それぞれ記憶させる。

【０５１０】

さらに、いま再生対象となっているエレメンタリストリームが多重化されているクリップストリームファイル”00002.PS”には、字幕ストリームが含まれるため、プレイヤ制御モジュール２１２から字幕読み出し機能部２３５には、再生対象のエレメンタリストリームのうちの字幕ストリームのstream_idとprivate_stream_idが供給され、字幕読み出し機能部２３５は、そのstream_idとprivate_stream_idを、stream_idレジスタ２６３とprivate_stream_idレジスタ２６４に、それぞれ記憶させる。

【０５１１】

そして、プレイヤ制御モジュール２１２は、バッファ制御モジュール２１５（図５）の初期化として、さらに、再生対象のエレメンタリストリームが多重化されているクリップストリームファイルに応じた値の字幕読み出し機能フラグを、字幕読み出し機能フラグ記憶部２６１にセットする。

【０５１２】

即ち、いまの場合、再生対象のエレメンタリストリームが多重化されているクリップストリームファイル”00002.PS”には、字幕ストリームが含まれるため、字幕読み出し機能部２３５を機能させるために、値が１の字幕読み出し機能フラグが、字幕読み出し機能フラグ記憶部２６１にセットされる。

【０５１３】

また、プレイヤ制御モジュール２１２は、再生しようとしている２番目のPlayItem#1（図２９）のIN_timeである90,000と、OUT_timeである27,090,000とを、デコード制御モジュール２１４に対して与える。

【０５１４】

さらに、プレイヤ制御モジュール２１２は、グラフィクス処理モジュール２１９に対する字幕ストリームの表示方式の指示を初期化する。即ち、プレイヤ制御モジュール２１２は、字幕ストリームの表示方式をデフォルトの表示方式とするように、グラフィクス処理モジュール２１９を制御する。

【０５１５】

なお、再生対象の字幕ストリームについて、configurable_flag（図１４）が、表示方式の変更を許可する1になっている場合には、プレイヤ制御モジュール２１２からグラフィクス処理モジュール２１９に対する字幕ストリームの表示方式の指示は、現在の表示方式のままとするようにしても良い。

【０５１６】

以下、２番目のPlayItem#1の再生は、１番目のPlayItem#0の再生と同様にして行われていく。そして、デコード制御モジュール２１４は、その２番目のPlayItem#1の再生が行われている間、内蔵する計時部２１４Ａが計時している時刻を確認し続けており、計時部２１４Ａが計時している時刻が、ステップＳ１５２（図４０）でプレイヤ制御モジュール２１２から与えられた２番目のPlayItem#1のOUT_timeである27,090,000（図２９）に等しくなると、ステップＳ１５１における場合と同様のデコード中断制御を行い、PlayItem#1の再生を終了する。尚、上述したように、計時部２１４Ａが計時している時刻は、厳密には、PlayItem#0のOUT_timeと等しくならないことがある。そのようなとき、PlayItem#0のOUT_timeの時刻と計時部２１４Ａの計時する時刻が近傍の値となったタイミングで、デコード中断制御が行われ、PlayItem#0の再生が終了されることになる。尚、この処理については、図５１，図５２を参照して詳細を後述する。

【０５１７】

［タイムコードの表示］

次に、上述したように、図３４のステップＳ１２３において、タイムコードの表示が開始されるが、このタイムコードの表示は、順次更新されていく。

【０５１８】

そこで、図４１のフローチャートを参照して、タイムコードの表示の処理について説明する。

【０５１９】

デコード制御モジュール２１４（図２）は、その内蔵する計時部２１４Ａによって1秒が計時されると、ステップＳ１７１において、１秒が経過した旨のメッセージとともに、その計時部２１４Ａによって計時されている現在時刻を、プレイヤ制御モジュール２１２に供給し、ステップＳ１７２に進む。ステップＳ１７２では、プレイヤ制御モジュール２１２は、デコード制御モジュール２１４からのメッセージと現在時刻を受信し、その現在時刻を、タイムコードに換算して、ステップＳ１７３に進む。

【０５２０】

ステップＳ１７３では、プレイヤ制御モジュール２１２は、ステップＳ１７２で得たタイムコードを表示するように、グラフィクス処理モジュール２１９を制御し、ステップＳ１７１に戻る。

【０５２１】

これにより、タイムコードは、１秒ごとに更新される。なお、タイムコードの更新の間隔は、１秒に限定されるものではない。

【０５２２】

［ストリーム切り替え］

次に、図２９で説明した１番目のPlayList#0を構成する１番目のPlayItem#0によって再生されるクリップストリームファイル”00001.PS”や、２番目のPlayItem#1によって再生されるクリップストリームファイル”00002.PS”には、図３０で説明したように、２本の字幕ストリームが多重化されている。

【０５２３】

このように、クリップストリームファイルに、複数の、同一の属性のエレメンタリストリームが多重化されている場合においては、再生対象のエレメンタリストリームを、その複数の、同一の属性のエレメンタリストリームのうちの１つから、他の１つに切り替えるストリーム切り替えを行うことができる。

【０５２４】

そこで、図４２のフローチャートを参照して、ストリーム切り替えの処理について説明する。

【０５２５】

ストリーム切り替えの要求は、例えば、”SCRIPT.DAT”ファイル（図６）に、ストリーム切り替えの指示がスクリプトプログラムとして記述されている場合に、スクリプト制御モジュール２１１が、そのスクリプトプログラムを実行することによって、あるいは、ユーザがリモコンを操作することによって、プレイヤ制御モジュール２１２に与えられる。

【０５２６】

即ち、スクリプト制御モジュール２１１は、ストリーム切り替えの指示が記述されているスクリプトプログラムを実行すると、ストリーム切り替えを要求するメッセージを、プレイヤ制御モジュール２１２に供給する。また、入力インターフェース１１５は、ユーザがリモコンを操作することによって、リモコンから、ストリーム切り替えを指示する信号を受信すると、ストリーム切り替えを要求するメッセージを、プレイヤ制御モジュール２１２に供給する。

【０５２７】

例えば、いま、プレイヤ制御モジュール２１２に対して、字幕ストリームの切り替えを要求するメッセージである字幕ストリーム切り替えのメッセージが供給されたとすると、プレイヤ制御モジュール２１２は、ステップＳ１９１において、図３４のステップＳ１２５で行われた再生対象のエレメンタリストリームの決定のときに認識した字幕ストリームの本数をチェックする。

【０５２８】

プレイヤ制御モジュール２１２は、字幕ストリームの本数をチェックした結果、その本

数が１本以下である場合、字幕ストリーム切り替えのメッセージを無視し、従って、以降のステップＳ１９２乃至Ｓ１９４の処理は行われない。

【０５２９】

一方、字幕ストリームの本数が２本以上である場合、ステップＳ１９２乃至Ｓ１９４に順次進み、再生する字幕ストリームが、現在再生されている字幕ストリームから、他の字幕ストリームに切り替えられる。

【０５３０】

即ち、ステップＳ１９２において、プレイヤ制御モジュール２１２は、現在再生中の字幕ストリームを、クリップ情報ファイル上で特定する。具体的には、例えば、いま、図２９で説明した１番目のPlayList#0を構成する２番目のPlayItem#1によって、クリップストリームファイル”00002.PS”に多重化された、stream_idが0xBDで、private_stream_idが0x80の字幕ストリームが再生されていることとすると、ステップＳ１９２では、現在再生中の字幕ストリームが、クリップストリームファイル”00002.PS”に多重化された２本の字幕ストリームのうちの、図３０のクリップ情報ファイル”00002.CLP”上で３本目の字幕ストリームであるstream#2であることが特定される。

【０５３１】

そして、ステップＳ１９３に進み、プレイヤ制御モジュール２１２は、ステップＳ１９２で特定した字幕ストリームの、クリップ情報ファイル上で次の字幕ストリームを、次に再生する字幕ストリームとして認識（特定）する。図３０では、クリップ情報ファイル”00002.CLP”上で、３本目の字幕ストリームstream#2の次の字幕ストリームは、４本目の字幕ストリームstream#3であるから、ステップＳ１９３では、この４本目の字幕ストリームstream#3が、次に再生する字幕ストリームとして認識される。

【０５３２】

なお、現在再生中の字幕ストリームが、クリップストリームファイル”00002.PS”に多重化された２本の字幕ストリームのうちの、図３０のクリップ情報ファイル”00002.CLP”上で４本目の字幕ストリームであるstream#3であることが特定された場合は、例えば、３本目の字幕ストリームstream#2が、次に再生する字幕ストリームとして認識される。

【０５３３】

その後、ステップＳ１９４に進み、プレイヤ制御モジュール２１２は、ステップＳ１９３で認識した次に再生する字幕ストリームのstream_idとprivate_stream_idを、バッファ制御モジュール２１５（図５）の字幕読み出し機能部２３５に対して与え、そのstream_idとprivate_stream_idを、次回からの、字幕アクセスユニットのバッファ２１５Ａからの読み出しから使用するように指示する。

【０５３４】

バッファ制御モジュール２１５（図５）の字幕読み出し機能部２３５では、ステップＳ１９４でプレイヤ制御モジュール２１２から与えられるstream_idとprivate_stream_idを、stream_idレジスタ２６３とprivate_stream_idレジスタ２６４に、それぞれ新たにセットし、次回以降のバッファ２１５Ａからの読み出しは、そのstream_idレジスタ２６３とprivate_stream_idレジスタ２６４にそれぞれ新たにセットされたstream_idとprivate_stream_idによって特定される字幕アクセスユニットを対象として行われる。

【０５３５】

以上のようにして、再生する字幕ストリームが、現在再生されている字幕ストリームから、他の字幕ストリームに切り替えられる。

【０５３６】

［バッファ制御モジュール２１５の処理］

次に、図４３乃至図４７を参照して、バッファ制御モジュール２１５（図５）の処理、即ち、バッファ２１５Ａへのデータの書き込みと、バッファ２１５Ａからのデータの読み出しについて説明する。

【０５３７】

バッファ制御モジュール２１５は、図５で説明したように、バッファ２１５Ａに対する

データの読み書きを行うための５つのポインタを有している。

【０５３８】

即ち、図４３および図４４に示すように、バッファ制御モジュール２１５は、データ先頭ポインタ記憶部２３１に記憶されるデータ先頭ポインタ、データ書き込みポインタ記憶部２３２に記憶されるデータ書き込みポインタ、ビデオ読み出しポインタ記憶部２４１に記憶されるビデオ読み出しポインタ、オーディオ読み出しポインタ記憶部２５１に記憶されるオーディオ読み出しポインタ、および字幕読み出しポインタ記憶部２６２に記憶される字幕読み出しポインタを有している。

【０５３９】

なお、図４３および図４４では、図５におけるビデオ読み出し機能部２３３のstream_idレジスタ２４２およびau_information()レジスタ２４３、オーディオ読み出し機能部２３４のstream_idレジスタ２５２およびprivate_stream_idレジスタ２５３、並びに字幕読み出し機能部２３５の字幕読み出し機能フラグ記憶部２６１、stream_idレジスタ２６３、およびprivate_stream_idレジスタ２６４の図示は、省略してある。

【０５４０】

データ先頭ポインタ記憶部２３１に記憶されたデータ先頭ポインタは、バッファ２１５Ａに残る最も古いデータ（読み出す必要があるデータであって、まだ読み出されていないデータのうちの最も古いデータ）の位置を表す。データ書き込みポインタ記憶部２３２に記憶されたデータ書き込みポインタは、バッファ２１５Ａへのデータの書き込みの位置を示し、この位置は、バッファ２１５Ａで最も新しいデータが書き込まれる位置である。

【０５４１】

ビデオ読み出しポインタ記憶部２４１に記憶されたビデオ読み出しポインタは、バッファ２１５Ａから読み出すビデオストリームの位置を表す。また、オーディオ読み出しポインタ記憶部２５１に記憶されたオーディオ読み出しポインタは、バッファ２１５Ａから読み出すオーディオストリームの位置を表し、字幕読み出しポインタ記憶部２６２に記憶された字幕読み出しポインタは、バッファ２１５Ａから読み出す字幕ストリームの位置を表す。

【０５４２】

なお、図５で説明したように、データ先頭ポインタ、データ書き込みポインタ、ビデオ読み出しポインタ、オーディオ読み出しポインタ、および字幕読み出しポインタは、いずれも、バッファ２１５Ａを右回りに移動する。

【０５４３】

さらに、本実施の形態では、データ先頭ポインタは、図４４に示すように、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタのうちの、最も古いデータの位置を指しているものと同一の位置を指すように、常時更新されるものとする。ここで、図４４では、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタのうちの、オーディオ読み出しポインタが、最も古いデータの位置を指しており、データ先頭ポインタは、そのオーディオ読み出しポインタと一致している。

【０５４４】

以上のようなデータ先頭ポインタ、データ書き込みポインタ、ビデオ読み出しポインタ、オーディオ読み出しポインタ、および字幕読み出しポインタを有するバッファ制御モジュール２１５では、データ書き込みポインタは、ディスク１０１から新たなデータが読み出され、バッファ２１５Ａに書き込まれると、その書き込まれた新たなデータの直後の位置を指すように、右回りに更新される。

【０５４５】

さらに、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタは、バッファ２１５Ａから、ビデオストリーム、オーディオストリーム、または字幕ストリームが読み出されると、その読み出し量に応じた分だけ、それぞれ、右回りに更新される。ここで読み出し量に応じた分とは、実際に読み出したビデオ、オーディオ、

字幕のデータに対応する部分と、読み出したデータの間に含まれており、読み出しの際には読み飛ばしを行った、他のストリームのデータの部分をあわせたものとなる。

【０５４６】

また、データ先頭ポインタは、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタが更新されると、そのビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタのうちの、最も古いデータの位置を指しているものと同一の位置を指すように更新される。

【０５４７】

ここで、バッファ制御モジュール２１５は、バッファ２１５Ａへのデータの書き込みについては、データ書き込みポインタがデータ先頭ポインタを追い越さないように、バッファ２１５Ａへのデータの書き込みを制御する。

【０５４８】

即ち、データ書き込みポインタによるデータ先頭ポインタの追い越しが発生しない限り、バッファ制御モジュール２１５では、ディスク１０１から読み出されたデータが、データ書き込みポインタが指すバッファ２１５Ａの位置に書き込まれ、データ書き込みポインタが更新されていく。一方、データ書き込みポインタによるデータ先頭ポインタの追い越しが発生しそうになると、バッファ制御モジュール２１５では、コンテンツデータ供給モジュール２１３に対して、ディスク１０１からのデータの読み出しの停止（中断）が要求され、さらに、バッファ２１５Ａへのデータの書き込みが停止される。これにより、バッファ２１５Ａのオーバーフローを防止することができる。

【０５４９】

以上のように、ディスク１０１から読み出されたデータの、バッファ２１５Ａへの書き込みは、データ先頭ポインタとデータ書き込みポインタとの２つのポインタの位置関係だけで制御される。

【０５５０】

一方、バッファ制御モジュール２１５は、バッファ２１５Ａからのデータの読み出しについては、ビデオ読み出しポインタ、オーディオ読み出しポインタ、および字幕読み出しポインタ、ひいては、データ先頭ポインタが、データ書き込みポインタを追い越さないように、バッファ２１５Ａからのデータの読み出しを制御する。

【０５５１】

即ち、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタによるデータ書き込みポインタの追い越しが発生しない限り、バッファ制御モジュール２１５では、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、または字幕デコーダ制御モジュール２１８からの要求に応じて、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタが指すバッファ２１５Ａの位置からデータが読み出され、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタが更新されるとともに、必要に応じて、データ先頭ポインタが更新される。一方、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタによるデータ書き込みポインタの追い越しが発生しそうになると、バッファ制御モジュール２１５では、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、または字幕デコーダ制御モジュール２１８からの要求が、例えば凍結され、十分なデータが用意されるまで待たされる。これにより、バッファ２１５Ａのアンダーフローを防止することができる。

【０５５２】

以上から、バッファ２１５Ａには、データ先頭ポインタが指す位置から、右回りに、データ書き込みポインタが指す位置までの範囲（図４３および図４４において影を付してある部分）に、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、および字幕デコーダ制御モジュール２１８に供給すべきデータが記憶されており、さらに、その範囲内に、ビデオ読み出しポインタ、オーディオ読み出しポインタ、および字幕読み出しポインタは存在する。

【０５５３】
なお、上述の場合には、データ先頭ポインタは、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタが指している位置のうちの、最も古いデータの位置を指すように更新することとしたが、その他、データ先頭ポインタの更新は、例えば、その最も古いデータの位置から、所定の時間（例えば、１秒）分だけ過去のデータの位置を指すように行うことが可能である。
【０５５４】
即ち、一般には、ビデオ読み出しポインタ、オーディオ読み出しポインタ、または字幕読み出しポインタのうちの、ビデオ読み出しポインタやオーディオ読み出しポインタが、最も古いデータの位置を指すことが多いと予想される。
【０５５５】
従って、データ先頭ポインタを、ビデオ読み出しポインタまたはオーディオ読み出しポインタが指す最も古いデータの位置から、例えば、１秒分だけ過去のデータの位置を指すように更新した場合、図４３に示すように、ビデオ読み出しポインタまたはオーディオ読み出しポインタが指す最も古いデータの位置から過去１秒分のデータを、バッファ２１５Ａに残しておくことができる。ここで、図４３では、オーディオ読み出しポインタが、最も古いデータの位置を指しており、データ先頭ポインタは、その位置から１秒分だけ過去のデータの位置を指している。
【０５５６】
以上のように、１秒分だけ過去のデータの位置を指すように、データ先頭ポインタを更新することにより、ディスク再生装置の応答性を向上させることができる。
【０５５７】
即ち、図４４に示したように、オーディオ読み出しポインタが指している最も古いデータの位置を指すように、データ先頭ポインタを更新する場合には、例えば、リバース方向への特殊再生が指示されたときに、バッファ２１５Ａからの読み出しが終了したデータを、ディスク１０１から再度読み出す必要があるため、特殊再生が指示されてから、その特殊再生が可能となるまでに、ある程度の時間がかかる。
【０５５８】
これに対して、図４３に示したように、オーディオ読み出しポインタが指している最も古いデータの位置から１秒分だけ過去のデータの位置を指すように、データ先頭ポインタを更新する場合には、リバース方向への特殊再生が指示されたときに、その特殊再生を開始するのに必要なデータが、バッファ２１５Ａに記憶されている１秒分だけ過去のデータであれば、上述したようなディスク１０１からのデータの再読み出しを行わずに、即座に、特殊再生を開始することが可能となる。
【０５５９】
なお、オーディオ読み出しポインタが指している最も古いデータの位置から１秒分だけ過去のデータの位置を指すように、データ先頭ポインタを更新する場合であっても、特殊再生を開始するのに必要なデータが、バッファ２１５Ａに記憶されていないことがあり得る。この場合には、特殊再生を開始するのに必要なデータが、ディスク１０１から再度読み出される。
【０５６０】
次に、バッファ２１５Ａからのビデオストリーム、オーディオストリーム、字幕ストリームそれぞれの読み出しの詳細について説明する。
【０５６１】
図３４のステップＳ１２７で説明したように、クリップストリームファイルの再生が開始されるときに、バッファ制御モジュール２１５においては、データ先頭ポインタ、データ書き込みポインタ、ビデオ読み出しポインタ、オーディオ読み出しポインタ、字幕読み出しポインタが、すべて、バッファ２１５Ａ上の同じ位置を指すように初期化される。
【０５６２】
そして、ディスク１０１からクリップストリームファイルに格納されたプログラムスト

リーム(MPEG2-System Program Stream)が読み出され、バッファ制御モジュール２１５に供給されると、バッファ制御モジュール２１５では、そのプログラムストリームが、バッファ２１５Ａのデータ書き込みポインタが指す位置に記憶されるとともに、データ書き込みポインタが、右回りに更新されていく。

【０５６３】

さらに、バッファ制御モジュール２１５（図５）では、ビデオ読み出し機能部２３３が、バッファ２１５Ａに記憶されたプログラムストリームの構文解析を行い、ビデオデコーダ制御モジュール２１６からの要求に応じて、バッファ２１５Ａに記憶されたプログラムストリームから、ビデオストリーム（ビデオアクセスユニット）を抽出（分離）して読み出し、ビデオデコーダ制御モジュール２１６に供給する。

【０５６４】

同様に、オーディオ読み出し機能部２３４も、バッファ２１５Ａに記憶されたプログラムストリームの構文解析を行い、オーディオデコーダ制御モジュール２１７からの要求に応じて、バッファ２１５Ａに記憶されたプログラムストリームから、オーディオストリーム（オーディオアクセスユニット）を抽出して読み出し、オーディオデコーダ制御モジュール２１７に供給する。字幕読み出し機能部２３５も、バッファ２１５Ａに記憶されたプログラムストリームの構文解析を行い、字幕デコーダ制御モジュール２１８からの要求に応じて、バッファ２１５Ａに記憶されたプログラムストリームから、字幕ストリーム（字幕アクセスユニット）を抽出して読み出し、字幕デコーダ制御モジュール２１８に供給する。

【０５６５】

「ビデオストリームの読み出し」

次に、図４５のフローチャートを参照して、ビデオ読み出し機能部２３３（図５）による、バッファ２１５Ａからのビデオストリームの読み出し処理の詳細について説明する。

【０５６６】

ビデオ読み出し機能部２３３は、まず最初に、ステップＳ２１１において、バッファ２１５Ａに記憶されたプログラムストリーム中のprivate_stream_2のPES_packet()を探索して見つけ出す。すなわち、private_stream_2のPES_packet()のstream_idは、図２３で説明したように、10111111B(=0xBF)であり、ビデオ読み出し機能部２３３は、stream_idが10111111Bとなっているを探索して見つけ出す。

【０５６７】

ここで、例えば、いま、上述したように、クリップストリームファイル”00001.PS”に格納されたプログラムストリームに多重化されたエレメンタリストリームが、再生対象のエレメンタリストリームであるとすると、そのプログラムストリームをディスク１０１から読み出して、バッファ２１５Ａに記憶させるときに、図３４のステップＳ１２２において、クリップストリームファイル”00001.PS”のEP_map()（図３１）に記述されたデコード開始可能点の情報から、305セクタが再生開始位置として決定され、さらに、図３４のステップＳ１２８において、再生開始位置である305セクタが指定され、オペレーティングシステム２０１に対して、クリップストリームファイル”00001.PS”に格納されたプログラムストリームの読み出しが要求される。

【０５６８】

また、ビデオストリームについては、EP_map()に記述されたデコード開始可能点の情報は、実際のデコード開始可能点の直前に配置されたprivate_stream_2のPES_packet()の位置を表す。

【０５６９】

従って、クリップストリームファイル”00001.PS”に格納されたプログラムストリームがディスク１０１から読み出され、バッファ２１５Ａに記憶された直後においては、データ先頭ポインタやビデオ読み出しポインタが指すバッファ２１５Ａの位置には、private_stream_2のPES_packet()が記憶されている。

【０５７０】

ビデオ読み出し機能部２３３は、ステップＳ２１１において、private_stream_2のPES_packet()が見つかると、ステップＳ２１２に進み、そのprivate_stream_2のPES_packet()のPES_packet_data_byteであるprivate_stream2_PES_payload()（図２６）に記述されているvideo_stream_idを抜き出し、そのvideo_stream_idが、図３４のステップＳ１２７でstream_idレジスタ２４２（図５）に記憶された、再生対象のビデオストリームのstream_idと一致するかどうかを判定する。

【０５７１】

ステップＳ２１２において、private_stream2_PES_payload()に記述されているvideo_stream_idが、stream_idレジスタ２４２に記憶されたstream_idと一致しないと判定された場合、即ち、直前のステップＳ２１１で見つけ出されたprivate_stream_2のPES_packet()が、再生対象のビデオストリームのデコード開始点に配置されたものでない場合、ステップＳ２１１に戻り、バッファ２１５Ａに記憶されたプログラムストリーム中の他のprivate_stream_2のPES_packet()の探索が行われ、以下、同様の処理が繰り返される。

【０５７２】

一方、ステップＳ２１２において、private_stream2_PES_payload()に記述されているvideo_stream_idが、stream_idレジスタ２４２に記憶されたstream_idと一致すると判定された場合、即ち、直前のステップＳ２１１で見つけ出されたprivate_stream_2のPES_packet()が、再生対象のビデオストリームのデコード開始点に配置されたものである場合、ステップＳ２１３に進み、ビデオ読み出し機能部２３３は、そのprivate_stream_2のPES_packet()のprivate_stream2_PES_payload()に記述されているau_information()を、バッファ２１５Ａから読み出し、au_information()レジスタ２４３（図５）に記憶させ、ステップＳ２１４に進む。

【０５７３】

ステップＳ２１４では、ビデオ読み出し機能部２３３は、直前のステップＳ２１１で見つけ出したprivate_stream_2のPES_packet()（video_stream_id（図２６）が、stream_idレジスタ２４２（図５）に記憶されたstream_idと一致するprivate_stream_2のPES_packet()）のサイズだけ、ビデオ読み出しポインタ記憶部２３１に記憶されたビデオ読み出しポインタを更新する。

【０５７４】

即ち、クリップストリームファイルでは、private_stream_2のPES_packet()の直後に、そのvideo_stream_idと一致するstream_idのビデオストリーム（PES_packet()）が配置されており、従って、ステップＳ２１４では、ビデオ読み出しポインタは、ビデオストリームの実際のデコード開始可能点の位置を指すように更新される。

【０５７５】

その後、ステップＳ２１４からＳ２１５に進み、ビデオ読み出し機能部２３３は、ビデデコーダ制御モジュール２１６から、データの要求があったかどうかを判定し、ないと判定した場合、ステップＳ２１５に戻り、同様の処理を繰り返す。

【０５７６】

また、ステップＳ２１５において、ビデオデコーダ制御モジュール２１６から、データの要求があったと判定された場合、ステップＳ２１６に進み、ビデオ読み出し機能部２３３は、ビデオ読み出しポインタが指しているバッファ２１５Ａの位置からのプログラムストリームの構文解析を行いつつ、au_information()レジスタ２４３に記憶されたau_information()のAU_lengthに記述されたバイト数のデータ、つまり１つのビデオアクセスユニットを、バッファ２１５Ａから読み出し、ビデオデコーダ制御モジュール２１６に供給するとともに、ビデオ読み出しポインタを、バッファ２１５Ａから読み出した１つのビデオアクセスユニットのサイズ分だけ更新する。

【０５７７】

即ち、au_information()には、図２７で説明したように、それを含むprivate_stream_2のPES_packet()から、次のprivate_stream_2のPES_packet()までの間に含まれるビデオアクセスユニット（ピクチャ）の数を表すnumber_of_access_unitが記述されている。

【０５７８】
さらに、au_information()には、図２７で説明したように、そのnumber_of_access_unitの数だけのビデオアクセスユニットそれぞれに関する情報としてのpic_struct_copy, au_ref_flag、およびAU_lengthが記述されている。
【０５７９】
au_information()にnumber_of_access_unitの数だけ記述されているAU_lengthそれぞれは、図２７で説明したように、それを含むprivate_stream_2のPES_packet()から、次のprivate_stream_2のPES_packet()までの間に含まれる、number_of_access_unitの数のビデオアクセスユニットそれぞれのサイズを表すから、ビデオ読み出し機能部２３３は、そのAU_lengthを用いることで、ビデオストリームの構文解析を行うことなく、アクセスユニットの切り出しを行うことが出来る。
【０５８０】
即ち、従来、MPEG2-VideoやMPEG4-AVCのアクセスユニットを切り出す場合には、ビデオストリームの構文を知った上で、ビデオストリームの構文解析を行う必要があったが、ディスク１０１に記録されたクリップストリームファイルに格納されたプログラムストリームは、ビデオアクセスユニット単位のビデオストリームにおける１以上の実際のデコード開始可能点それぞれの直前に、ビデオアクセスユニットのサイズを表すAU_lengthが記述されたprivate_stream_2のPES_packet()を含んでいるので、ビデオ読み出し機能２３３は、そのprivate_stream_2のPES_packet()に記述されたAU_lengthに基づき、ビデオストリームの構文解析を行うことなく、バッファ２１５Ａから、ビデオアクセスユニット（単位のビデオストリーム）を読み出し、ビデオデコーダ制御モジュール２１６に供給することができる。
【０５８１】
なお、ビデオ読み出し機能部２３３は、ステップＳ２１６において、ビデオアクセスユニットを、ビデオデコーダ制御モジュール２１６に供給するときに、そのビデオアクセスユニットに関する情報としてau_information()に記述されているpic_struct_copy, au_ref_flag、およびAU_lengthと、ビデオアクセスユニット単位に付加されているタイムスタンプ（PTS, DTS）も、ビデオデコーダ制御モジュール２１６に供給する。
【０５８２】
ステップＳ２１６において、バッファ２１５Ａから１つのビデオアクセスユニットが読み出され、ビデオデコーダ制御モジュール２１６に供給された後は、ステップＳ２１７に進み、ビデオ読み出し機能部２３３は、au_information()レジスタ２４３に記憶されたau_information()（図２７）のnumber_of_access_unitが表す数だけのアクセスユニットを処理したかどうかを判定する。
【０５８３】
ステップＳ２１７において、number_of_access_unitが表す数だけのアクセスユニットを、まだ処理していないと判定された場合、即ち、number_of_access_unitが表す数だけのアクセスユニットを、まだ、バッファ２１５Ａから読み出してビデオデコーダ制御モジュール２１６に供給していない場合、ステップＳ２１５に戻り、以下、同様の処理が繰り返される。
【０５８４】
また、ステップＳ２１７において、number_of_access_unitが表す数だけのアクセスユニットを処理したと判定された場合、即ち、number_of_access_unitが表す数だけのアクセスユニットを、バッファ２１５Ａから読み出してビデオデコーダ制御モジュール２１６に供給した場合、ステップＳ２１１に戻り、次のprivate_stream_2のPES_packet()の探索が行われ、以下、同様の処理が繰り返される。
【０５８５】
「オーディオストリームの読み出し」
次に、図４６のフローチャートを参照して、オーディオ読み出し機能部２３４（図５）による、バッファ２１５Ａからのオーディオストリームの読み出し処理の詳細について説

明する。

【０５８６】

オーディオ読み出し機能部２３４は、まず最初に、ステップＳ２３０において、図３４のステップＳ１２７でstream_idレジスタ２５２（図５）に記憶された、再生対象のオーディオストリームのstream_idが、private_stream_1のPES_packet()を表しているかどうかを判定する。

【０５８７】

ステップＳ２３０において、stream_idレジスタ２５２に記憶されたstream_idが、private_stream_1のPES_packet()を表していない判定された場合、即ち、stream_idレジスタ２５２に記憶されたstream_idが、図２３で説明したように、MPEG規格にしたがって符号化されたオーディオストリームに割り当てられる110xxxxxBである場合、ステップＳ２３１に進み、オーディオ読み出し機能部２３４は、バッファ２１５Ａに記憶されたプログラムストリームから、MPEG Audioで定められたオーディオフレーム先頭を表す同期コードを探す。同期コードの位置がオーディオフレーム先頭なので、オーディオ読み出し機能部２３４は、オーディオ読み出しポインタを、オーディオフレーム先頭の位置を示すように更新し、ステップＳ２３１からＳ２３２に進む。ステップＳ２３２では、オーディオ読み出し機能部２３４は、バッファ２１５Ａに記憶されたプログラムストリーム中の、stream_idレジスタ２５２に記憶されたstream_idに一致するPES_packet()を、オーディオ読み出しポインタが示す位置から探索して見つけ出し、ステップＳ２３３に進む。

【０５８８】

ステップＳ２３３では、オーディオ読み出し機能部２３４は、オーディオ読み出しポインタ記憶部２５１に記憶されたオーディオ読み出しポインタを、直前のステップＳ２３２で見つけ出したPES_packet()のPES_packet_data_byte（図１９乃至図２１）の先頭を指すように更新し、ステップＳ２３７に進む。

【０５８９】

ステップＳ２３７では、オーディオ読み出し機能部２３４は、オーディオデコーダ制御モジュール２１７から、データの要求があったかどうかを判定し、ないと判定した場合、ステップＳ２３７に戻り、同様の処理を繰り返す。

【０５９０】

また、ステップＳ２３７において、オーディオデコーダ制御モジュール２１７から、データの要求があったと判定された場合、ステップＳ２３８に進み、オーディオ読み出し機能部２３４は、オーディオ読み出しポインタが指しているバッファ２１５Ａの位置からのプログラムストリームの構文解析を行いつつ、既知の固定長の１つのオーディオアクセスユニットを、バッファ２１５Ａから読み出し、そのオーディオアクセスユニットに付加されているタイムスタンプ（PTS，DTS）とともに、オーディオデコーダ制御モジュール２１７に供給する。

【０５９１】

そして、オーディオ読み出し機能部２３４は、バッファ２１５Ａから読み出した１つのオーディオアクセスユニットのサイズ分だけ、オーディオ読み出しポインタを更新して、ステップＳ２３７に戻り、以下、同様の処理が繰り返される。

【０５９２】

一方、ステップＳ２３０において、stream_idレジスタ２５２に記憶されたstream_idが、private_stream_1のPES_packet()を表していると判定された場合、即ち、stream_idレジスタ２５２に記憶されたstream_idが、10111101B(=0xBD)であり、図２３で説明したように、private_stream_1のPES_packet()を表している場合、ステップＳ２３４に進み、オーディオ読み出し機能部２３４は、バッファ２１５Ａに記憶されたプログラムストリーム中のprivate_stream_1のPES_packet()を探索して見つけ出す。すなわち、オーディオ読み出し機能部２３４は、stream_idが10111101Bとなっているを探索して見つけ出す。

【０５９３】

オーディオ読み出し機能部２３４は、ステップＳ２３４において、private_stream_1のPES_packet()が見つかると、ステップＳ２３５に進み、そのprivate_stream_1のPES_packet()のPES_packet_data_byteであるprivate_stream1_PES_payload()（図２４）に記述されているprivate_stream_idを抜き出し、そのprivate_stream_idが、図３４のステップＳ１２７でprivate_stream_idレジスタ２５３（図５）に記憶された、再生対象のオーディオストリームのprivate_stream_idと一致するかどうかを判定する。

【０５９４】

ステップＳ２３５において、private_stream1_PES_payload()に記述されているprivate_stream_idが、private_stream_idレジスタ２５３に記憶されたprivate_stream_idと一致しないと判定された場合、即ち、直前のステップＳ２３４で見つけ出されたprivate_stream_1のPES_packet()が、再生対象のオーディオストリームではない場合、ステップＳ２３４に戻り、バッファ２１５Ａに記憶されたプログラムストリーム中の他のprivate_stream_1のPES_packet()の探索が行われ、以下、同様の処理が繰り返される。

【０５９５】

一方、ステップＳ２３５において、private_stream1_PES_payload()に記述されているprivate_stream_idが、private_stream_idレジスタ２５３に記憶されたprivate_stream_idと一致すると判定された場合、即ち、直前のステップＳ２３４で見つけ出されたprivate_stream_1のPES_packet()が、再生対象のオーディオストリームである場合、ステップＳ２３６に進み、オーディオ読み出し機能部２３４は、そのprivate_stream_1のPES_packet()のprivate_stream1_PES_payload()（図２４）に記述されているAU_locatorを、バッファ２１５Ａから読み出し、そのAU_locatorの直後の位置と、そのAU_locatorが表す値とを加算することで、オーディオアクセスユニットの先頭位置を求める。

【０５９６】

即ち、AU_locatorは、図２４で説明したように、そのAU_locatorの直後の位置を基準として、private_stream1_PES_payload()のprivate_payload()に格納されるオーディオアクセスユニット（あるいは字幕アクセスユニット）の先頭位置を表すから、AU_locatorの直後の位置に、そのAU_locatorが表す値を加算することにより、オーディオアクセスユニットの（絶対的な）先頭位置を求めることができる。

【０５９７】

オーディオ読み出し機能部２３４は、さらに、ステップＳ２３６において、以上のようにして求めたオーディオアクセスユニットの先頭位置を指すように、オーディオ読み出しポインタ記憶部２５１に記憶されたオーディオ読み出しポインタを更新し、ステップＳ２３７に進む。

【０５９８】

ステップＳ２３７では、オーディオ読み出し機能部２３４は、オーディオデコーダ制御モジュール２１７から、データの要求があったかどうかを判定し、ないと判定した場合、ステップＳ２３７に戻り、同様の処理を繰り返す。

【０５９９】

また、ステップＳ２３７において、オーディオデコーダ制御モジュール２１７から、データの要求があったかと判定された場合、ステップＳ２３８に進み、オーディオ読み出し機能部２３４は、オーディオ読み出しポインタが指しているバッファ２１５Ａの位置からのプログラムストリームの構文解析を行いつつ、既知の固定長の１つのオーディオアクセスユニットを、バッファ２１５Ａから読み出し、そのオーディオアクセスユニットに付加されているタイムスタンプとともに、オーディオデコーダ制御モジュール２１７に供給する。

【０６００】

そして、オーディオ読み出し機能部２３４は、バッファ２１５Ａから読み出した１つのオーディオアクセスユニットのサイズ分だけ、オーディオ読み出しポインタを更新して、ステップＳ２３７に戻り、以下、同様の処理が繰り返される。

【０６０１】

「字幕ストリームの読み出し」

次に、図４７のフローチャートを参照して、字幕読み出し機能部２３５（図５）による、バッファ２１５Ａからの字幕ストリームの読み出し処理の詳細について説明する。

【０６０２】

字幕読み出し機能部２３５は、まず最初に、ステップＳ２５１において、図３４のステップＳ１２７で字幕読み出し機能フラグ記憶部２６１に記憶された字幕読み出し機能フラグを判定する。ステップＳ２５１において、字幕読み出し機能フラグが0であると判定された場合、即ち、例えば、再生対象のエレメンタリストリームが多重化されているクリップストリームファイルに字幕ストリームが含まれておらず、図３４のステップＳ１２７で字幕読み出し機能フラグ記憶部２６１に、0がセットされた場合、字幕読み出し機能部２３５は特に処理を行わない。

【０６０３】

一方、ステップＳ２５１において、字幕読み出し機能フラグが1であると判定された場合、即ち、例えば、再生対象のエレメンタリストリームが多重化されているクリップストリームファイルに字幕ストリームが含まれており、図３４のステップＳ１２７で字幕読み出し機能フラグ記憶部２６１に、1がセットされた場合、ステップＳ２５２に進み、字幕読み出し機能部２３５は、stream_idレジスタ２６３（図５）に記憶された、再生対象の字幕ストリームのstream_idに一致するPES_packet()を、バッファ２１５Ａに記憶されたプログラムストリームから探索する。

【０６０４】

ここで、図３４のステップＳ１２７で説明したように、stream_idレジスタ２６３（図５）には、再生対象の字幕ストリームのstream_idが記憶されるが、字幕ストリームのstream_idは、図２３で説明したように、private_stream_1のPES_packet()を表す10111101B(=0xBD)である。

【０６０５】

従って、ステップＳ２５２では、バッファ２１５Ａに記憶されたプログラムストリーム中のprivate_stream_1のPES_packet()が探索されることになる。

【０６０６】

ステップＳ２５２において、private_stream_1のPES_packet()の探索が行われ、private_stream_1のPES_packet()が見つかると、ステップＳ２５３に進み、字幕読み出し機能部２３５は、そのprivate_stream_1のPES_packet()のPES_packet_data_byteであるprivate_stream1_PES_payload()（図２４）に記述されているprivate_stream_idを抜き出し、そのprivate_stream_idが、図３４のステップＳ１２７でprivate_stream_idレジスタ２６４（図５）に記憶された、再生対象の字幕ストリームのprivate_stream_idと一致するかどうかを判定する。

【０６０７】

ステップＳ２５３において、private_stream1_PES_payload()に記述されているprivate_stream_idが、private_stream_idレジスタ２６４に記憶されたprivate_stream_idと一致しないと判定された場合、即ち、直前のステップＳ２５２で見つかったprivate_stream_1のPES_packet()が、再生対象の字幕ストリームではない場合、ステップＳ２５２に戻り、バッファ２１５Ａに記憶されたプログラムストリーム中の他のprivate_stream_1のPES_packet()の探索が行われ、以下、同様の処理が繰り返される。

【０６０８】

一方、ステップＳ２５３において、private_stream1_PES_payload()に記述されているprivate_stream_idが、private_stream_idレジスタ２６４に記憶されたprivate_stream_idと一致すると判定された場合、即ち、直前のステップＳ２５２で見つかったprivate_stream_1のPES_packet()が、再生対象の字幕ストリームである場合、ステップＳ２５４に進み、字幕読み出し機能部２３５は、そのprivate_stream_1のPES_packet()のprivate_stream1_PES_payload()（図２４）に記述されているAU_locatorを、バッファ２１５Ａから読み出し、そのAU_locatorの直後の位置と、そのAU_locatorが表す値とを加算することで、字

幕アクセスユニットの先頭位置を求める。

【０６０９】

即ち、AU_locatorは、図２４で説明したように、そのAU_locatorの直後の位置を基準として、private_stream1_PES_payload()のprivate_payload()に格納される字幕アクセスユニット（あるいはオーディオアクセスユニット）の先頭位置を表すから、AU_locatorの直後の位置に、そのAU_locatorが表す値を加算することにより、字幕アクセスユニットの（絶対的な）先頭位置を求めることができる。

【０６１０】

字幕読み出し機能部２３５は、さらに、ステップＳ２５４において、以上のようにして求めた字幕アクセスユニットの先頭位置を指すように、字幕読み出しポインタ記憶部２６２に記憶された字幕読み出しポインタを更新し、ステップＳ２５５に進む。

【０６１１】

ステップＳ２５５では、字幕読み出し機能部２３５は、字幕デコーダ制御モジュール２１８から、データの要求があったかどうかを判定し、ないと判定した場合、ステップＳ２５５に戻り、同様の処理を繰り返す。

【０６１２】

また、ステップＳ２５５において、字幕デコーダ制御モジュール２１８から、データの要求があったかと判定された場合、ステップＳ２５６に進み、字幕読み出し機能部２３５は、字幕読み出しポインタが指しているバッファ２１５Ａの位置からのプログラムストリームの構文解析を行いつつ、字幕アクセスユニットの先頭に記述されているサイズ分の１つの字幕アクセスユニットを、バッファ２１５Ａから読み出し、その字幕アクセスユニットに付加されているタイムスタンプとともに、字幕デコーダ制御モジュール２１８に供給する。即ち、字幕アクセスユニットの先頭には、図２で説明したように、その字幕アクセスユニットのサイズが記述されており、字幕読み出し機能部２３５は、そのサイズ分のデータを、字幕読み出しポインタが指しているバッファ２１５Ａの位置から読み出し、その読み出したデータである字幕アクセスユニットを、その字幕アクセスユニットに付加されているタイムスタンプとともに、字幕デコーダ制御モジュール２１８に供給する。

【０６１３】

そして、字幕読み出し機能部２３５は、バッファ２１５Ａから読み出した１つの字幕アクセスユニットのサイズ分だけ、字幕読み出しポインタを更新して、ステップＳ２５５に戻り、以下、同様の処理が繰り返される。

【０６１４】

［再同期処理］

次に、図２のデコード制御モジュール２１４による、ビデオデータとオーディオデコーダとの同期制御について説明する。

【０６１５】

図３４のＳ１３０で説明したように、デコード制御モジュール２１４は、同期を確保するために必要であればタイミングをずらして、デコード開始を、ビデオデコーダ制御モジュール２１６、オーディオデコーダ制御モジュール２１７、および字幕デコーダ制御モジュール２１８に指令するが、例えば、その後のビデオデコーダ１１６とオーディオデコーダ１１７のデコード処理の進行程度によって、ビデオデータの出力と、そのビデオデータと同期して出力されるべき出力データとしてのオーディオデータの出力とがずれることがある。

【０６１６】

そこで、デコード制御モジュール２１４では、ビデオデータの出力と、そのビデオデータと同期して出力されるべきオーディオデータの出力とに生じたずれを補正し、ビデオデータとオーディオデータとが同期して出力されるようにするための再同期処理が行われる。

【０６１７】

図４８のフローチャートを参照して、再同期処理について説明する。

【０６１８】
再同期処理では、まず最初に、ステップＳ２７１において、デコード制御モジュール２１４は、ビデオデコーダ制御モジュール２１６からのビデオアクセスユニットのタイムスタンプと、オーディオ制御モジュール２１７からのオーディオアクセスユニットのタイムスタンプとのずれが大であるかどうかを判定する。
【０６１９】
即ち、図３４のステップＳ１２９で説明したように、ビデオデコーダ制御モジュール２１６は、バッファ制御モジュール２１５からビデオアクセスユニットを得るたびに、そのビデオアクセスユニットのタイムスタンプを、デコード制御モジュール２１４に供給する。同様に、オーディオ制御モジュール２１７も、バッファ制御モジュール２１５からオーディオアクセスユニットを得るたびに、そのオーディオアクセスユニットのタイムスタンプを、デコード制御モジュール２１４に供給する。
【０６２０】
ステップＳ２７１では、デコード制御モジュール２１４は、ビデオデコーダ制御モジュール２１６とオーディオ制御モジュール２１７とのそれぞれから、同一タイミングで（同一タイミングとみなすことができる、ある時間内に）供給されるタイムスタンプどうしを比較し、それらのタイムスタンプのずれが大であるかどうかを判定する。
【０６２１】
ステップＳ２７１において、ビデオデコーダ制御モジュール２１６からのビデオアクセスユニットのタイムスタンプと、オーディオ制御モジュール２１７からのオーディオアクセスユニットのタイムスタンプとのずれが大でないと判定された場合、即ち、ビデオアクセスユニットのタイムスタンプと、オーディオアクセスユニットのタイムスタンプとのずれが、あらかじめ定められた同期がとれているとみなすことができる範囲内である、例えば、２ビデオフレーム（約６６ミリ秒）である場合、ステップＳ２７１に戻り、タイムスタンプどうしのずれの判定（監視）が続行される。
【０６２２】
一方、ステップＳ２７１において、ビデオデコーダ制御モジュール２１６からのビデオアクセスユニットのタイムスタンプと、オーディオ制御モジュール２１７からのオーディオアクセスユニットのタイムスタンプとのずれが大であると判定された場合、即ち、ビデオアクセスユニットのタイムスタンプと、オーディオアクセスユニットのタイムスタンプとのずれが、あらかじめ定められた同期がとれているとみなすことができる範囲外である場合、ステップＳ２７２に進み、デコード制御モジュール２１４は、ビデオデコーダ制御モジュール２１６からのビデオアクセスユニットのタイムスタンプと、オーディオ制御モジュール２１７からのオーディオアクセスユニットのタイムスタンプとを比較することにより、ビデオデータの出力（デコード）と、オーディオデータの出力とのうちのいずれが遅れているかを判定する。
【０６２３】
ステップＳ２７２において、ビデオデータの出力が、オーディオデータの出力よりも遅れていると判定された場合、ステップＳ２７３に進み、デコード制御モジュール２１４は、１ビデオアクセスユニットだけ、ビデオアクセスユニットの処理を進めるために、ビデオデコーダ制御モジュール２１６に対して、ビデオアクセスユニットのデコードと出力（表示）を行わない旨の指示、即ち、ビデオアクセスユニットの処理のスキップの指示を出力して、ステップＳ２７４に進む。
【０６２４】
ステップＳ２７４では、ビデオデコーダ制御モジュール２１６は、デコード制御モジュール２１４からのスキップの指示を受信し、そのスキップの指示に応じて、バッファ制御モジュール２１５からのビデオアクセスユニットとともに供給されるau_ref_flag（図２７）を検査する。
【０６２５】
即ち、private_stream_2のPES_packet()のprivate_stream2_PES_payload()（図２６）

に配置されたau_information()（図２７）には、アクセスユニットに関する情報としてのau_ref_flagが含まれており、バッファ制御モジュール２１５は、図３４のステップＳ１２９や、図４５のステップＳ２１６で説明したように、ビデオアクセスユニットとともに、そのビデオアクセスユニットのau_ref_flagを、ビデオデコーダ制御モジュール２１６に供給する。

【０６２６】

ステップＳ２７４では、このように、アクセスユニットとともに供給される、そのアクセスユニットのau_ref_flagが検査される。

【０６２７】

そして、ステップＳ２７４からＳ２７５に進み、ビデオデコーダ制御モジュール２１６は、バッファ制御モジュール２１５から供給されたビデオアクセスユニットのau_ref_flagの検査の結果に基づき、そのビデオアクセスユニットが、他のピクチャのデコードにあたって参照されない非参照画像であるかどうかを判定する。

【０６２８】

ここで、図２７で説明したように、ビデオアクセスユニットのau_ref_flagは、そのアクセスユニットが参照画像であるか否かを表し、参照画像である場合には１とされ、参照画像でない場合、即ち、非参照画像である場合には０とされる。

【０６２９】

ステップＳ２７５において、バッファ制御モジュール２１５から供給されたビデオアクセスユニットが非参照画像（のビデオアクセスユニット）でないと判定された場合、即ち、バッファ制御モジュール２１５から供給されたビデオアクセスユニットが参照画像である場合、ステップＳ２７６に進み、ビデオデコーダ制御モジュール２１６は、通常通り、そのビデオアクセスユニットを、ビデオデコーダ１１６に処理させ、次のビデオアクセスユニットが、バッファ制御モジュール２１５から供給されるのを待って、ステップＳ２７４に戻る。

【０６３０】

また、ステップＳ２７５において、バッファ制御モジュール２１５から供給されたビデオアクセスユニットが非参照画像であると判定された場合、ステップＳ２７７に進み、ビデオデコーダ制御モジュール２１６は、そのビデオアクセスユニットの、ビデオデコーダ１１６による処理をスキップさせ、次のビデオアクセスユニットが、バッファ制御モジュール２１５から供給されるのを待って、ステップＳ２７１に戻る。

【０６３１】

このように、ビデオアクセスユニットの処理がスキップされることにより、ビデオアクセスユニットの処理が、ほぼ１ビデオアクセスユニット分だけ進められる（処理時間が短縮される）。その結果、オーディオデータの出力よりも遅れていたビデオデータの出力が早まることになる。

【０６３２】

一方、ステップＳ２７２において、ビデオデータの出力が、オーディオデータの出力よりも遅れていないと判定された場合、即ち、オーディオデータの出力が、ビデオデータの出力よりも遅れている場合、ステップＳ２７８に進み、デコード制御モジュール２１４は、ビデオアクセスユニットの処理を待たせるために、ビデオデコーダ制御モジュール２１６に対して、いまデコードされているビデオアクセスユニットに対応するビデオデータを繰り返して出力する繰り返し出力の指示を出力して、ステップＳ２７９に進む。

【０６３３】

ステップＳ２７９では、ビデオデコーダ制御モジュール２１６は、デコード制御モジュール２１４からの繰り返し出力の指示を受信し、その繰り返し出力の指示に応じて、いまビデオデコーダ１１６でデコードされているビデオアクセスユニットに対応するビデオデータを繰り返して、グラフィクス処理モジュール２１９に出力し、次のビデオアクセスユニットが、バッファ制御モジュール２１５から供給されるのを待って、ステップＳ２７１に戻る。

【０６３４】
以上のように、デコード制御モジュール２１４では、ビデオデータの出力が、オーディオデータの出力よりも遅れているか否かを判定し、ビデオデータの出力が、オーディオデータの出力よりも遅れている場合には、１つのアクセスユニットの処理のスキップを、ビデオデコーダ制御モジュール２１６に指示する。そして、ビデオデコーダ制御モジュール２１６では、スキップが指示されたアクセスユニットのau_ref_flagに基づき、そのアクセスユニットが参照画像であるか、または非参照画像であるかを判定し、非参照画像である場合に、ビデオデコーダ１１６に、スキップが指示されたアクセスユニットの処理をスキップさせる。従って、ビデオデータの出力と、オーディオデータの出力との同期を、容易にとることができる。

【０６３５】
即ち、処理をスキップするアクセスユニットが参照画像である場合、そのアクセスユニットに対応するビデオデータは、その後にデコードされる他のアクセスユニットのデコード時に参照するためにデコードする必要がある。従って、ビデオデータの出力と、オーディオデータの出力との同期をとるための同期制御において、参照画像のアクセスユニットの処理をスキップしてしまうと、その参照画像を参照する他のアクセスユニットをデコードすることができず、その結果、ビデオデータの表示において、同期制御がノイズとして現れてしまう。

【０６３６】
このため、処理をスキップするのは、参照画像でないアクセスユニット、即ち、非参照画像のアクセスユニットとするのが望ましい。

【０６３７】
一方、従来のエレメンタリストリームについて、非参照画像のアクセスユニットを探すためには、エレメンタリストリームの構文解析を行う必要があるが、例えば、MPEG4-AVCなどにしたがった符号化により得られるエレメンタリストリームは、構文が非常に複雑であるため、構文解析に、多大なコストがかかる。

【０６３８】
これに対して、ディスク１０１に記録されたクリップストリームファイルに格納されたプログラムストリームには、ビデオアクセスユニットがPES_packet_data_byteに配置されるPES_packet()（図１９乃至図２１）とは別に、PES_packet_data_byteを拡張したprivate_stream2_PES_payload()（図２６）が配置されたprivate_stream_2のPES_packet()が多重化されており、そのprivate_stream2_PES_payload()のau_information()（図２７）には、ビデオアクセスユニットごとに、そのビデオアクセスユニットが参照画像であるか、または非参照画像であるかを表すau_ref_flagが記述されている。そして、そのau_ref_flagは、対応するビデオアクセスユニットとともに、バッファ制御モジュール２１５からビデオデコーダ制御モジュール２１６に供給される。従って、ビデオデコーダ制御モジュール２１６は、ビデオアクセスユニットとともに供給される、そのビデオアクセスユニットのau_ref_flagを検査することにより、コストをほとんどかけずに、ビデオアクセスユニットが参照画像であるか、または非参照画像であるかを認識することができる。

【０６３９】
［マーク処理］
次に、図４９のフローチャートを参照して、PlayListMark()（図９）に記述されたMark()に基づいて行われるマーク処理について説明する。

【０６４０】
デコード制御モジュール２１４は、内蔵する計時部２１４Ａによって計時されている現在時刻を、常時確認しており、ステップＳ３０１において、現在時刻が、PlayListMark()（図９）に記述されたいずれかのMark()のmark_time_stampに一致したか否かを判定する。

【０６４１】
即ち、図３４のステップＳ１２４で説明したように、プレイヤ制御モジュール２１２は

、図２９に示した１番目のPlayList#0の１番目のPlayItem#0を再生しようとするときに、図３２上側に示したPlayListMark()に含まれる７つのMark()のうちの１番目から４番目までの４つのMark()が、PlayList#0の１番目のPlayItem#0に属していることを認識し、その４つのMark()のmark_time_stampである{180,090}、{5,580,090}、{10,980,090}、{16,380,090}を、そのmark_time_stampが表す時刻の属性が「マーク処理」である旨とともに、デコード制御モジュール２１４に渡している。

【０６４２】

ステップＳ３０１では、デコード制御モジュール２１４において、現在時刻が、上述のようにしてプレイヤ制御モジュール２１２から供給された「マーク処理」の属性の時刻(mark_time_stamp)のうちのいずれかと一致するかどうかが判定される。

【０６４３】

ステップＳ３０１において、現在時刻が、「マーク処理」の属性の時刻のうちのいずれとも一致しないと判定された場合、ステップＳ３０１に戻り、同様の処理が繰り返される。

【０６４４】

［マーク処理における一致判定］

ここで、マーク処理に関して、ステップＳ３０１においてデコード制御モジュール２１４は、現在時刻とmark_time_stampのいずれかが一致しているか否かの判定を行っている。しかしながら、本実施例においては、図４に示すように、計時部２１４Ａの示す時刻が離散的な値を示すため、単純な一致判定では問題が発生することがある。

【０６４５】

図５０を用い、少し単純化した例で説明する。図５０の上部でI0，P1，P2，P3はビデオアクセスユニットである。これらのpic_structはすべて3であり、すなわち表示durationは1フレーム（90kHz換算で3003）とする。また、この例ではデコード順と表示順が同じであるとしており、リオーダリングは発生しない。また、I0は「再生準備処理」で説明したようにEP_map()に登録されているアクセスユニットであり、タイムスタンプを持っておりPTS＝180,090であるとする。P1，P2，P3のアクセスユニットはタイムスタンプを持っていない。

【０６４６】

このようなビデオデータが処理される場合、計時部２１４Ａの示す時計は図５０の下部で示されるように更新される。すなわち、I0の出力時は、I0のPTSとI0のpic_structが渡され、PTSが存在するので計時部２１４ＡにPTSが代入されて180,090となる。P1の出力時には、PTSは存在せず、P1のpic_structが渡され、I0のpic_structが３であるため、1フレーム分の時間（90kHzで3003）が計時部２１４Ａに加算され、183,093となる。同様にP2の出力時にはP1のpic_structが3であることから3003が加算されて186,096、P3の出力時にも3003が加算されて189,099となる。

【０６４７】

ここで、PlayListMark()（図９及び図３２）に登録されたうちのひとつのマークのmark_time_stampが186,000であったときの処理を考える。上述したように時計（計時部２１４Ａ）の出力する値は、180,090、183,093、186,096、189,099であるため、マークの時刻と一致する186,000という時刻は出力しない。このため、mark_time_stampと時刻の単純な比較、すなわち差分が0であるか否かの確認では問題が発生する。

【０６４８】

このため、ここでは時刻の一致判定に対して一定のルールが適用される。すなわち、ここでは、所定のイベントのmark_time_stampが、所定の画像の表示duration中に含まれる場合、その所定のイベントは該当する画像の表示開始時刻に発生すると定義する。例えば上記の例では、mark_time_stamp=186,000は、画像P1の表示durationに含まれている。このため、このイベントはP1の表示開始時刻すなわち183,093に発生するとする。

【０６４９】

このような定義の下で一致判定を行うためのデコード制御モジュール２１６の判定アル

ゴリズムについて説明する。

【０６５０】

なお、この例の場合、時刻（計時部２１４Ａ）の更新はビデオデータの更新の際にのみ発生する。つまり、図４９のステップＳ３０１は、時刻の更新の際にのみ行われることになる。ソフトウェアで構成する再生機器では、処理を大幅に減らすことができ有利である。

【０６５１】

さて、時刻の更新に応じて図４９の処理が呼び出される。デコード制御モジュール２１４は、ステップＳ３０１において、現在時刻に一致と判定されるイベントが存在するか否かの確認を行う。すなわち、現在時刻と、表示中の画像の表示durationから現在表示中の画像の表示durationに含まれているイベントが存在するかどうかの確認を行い、一致と判定されるイベントが存在しない場合、処理はステップＳ３０１に戻る。また、一致と判定されるイベントが存在する場合、処理はステップＳ３０２に進む。尚、イベントが存在しない場合には、処理が終了されるようにしても良い。

【０６５２】

具体的には、例えばI0を表示中の場合に、時刻は180,090であり、I0のpic_structが3であることからI0の表示durationは3003であることが分かっている。このため、180,090≦mark_time_stamp＜180,090+3003を満たすmark_time_stampを探す。このとき例示したイベント時刻186,000はこの式を満たさないため、一致と判定されない。

【０６５３】

次に、例えばI1を表示中の場合に、時刻は183,093であり、I1のpic_structが3であることからI0の表示durationは3003であることが分かっている。このため、183,093≦mark_time_stamp＜183,093+3003を満たすmark_time_stampを探す。このとき例示したイベント時刻186,000はこの式を満すので、時刻が一致すると判定され、ステップＳ３０２以降の処理が行われる。

【０６５４】

この例では、時刻の一致判定の１の例を示したが、他の定義を適用することも可能である。例えば、あるイベントのmark_time_stampが、”ある画像の表示開始－α”の時刻以上、”次の表示画像の表示開始時刻－α”の時刻未満に含まれる場合、そのイベントは該当する画像の表示開始時刻に発生すると定義することも可能である。また判定基準はそのままで、イベントの発生時刻を例えば”該当画像の表示開始時刻－α”に発生すると定義することも可能である。

【０６５５】

なお、このような定義を導入することにより、マークの時刻すなわちmark_time_stampの設定時にビデオストリームの時刻を知る必要が無くなる。このため、オーサリングにおけるビデオエンコードとデータベース作成の独立性が高まり、作業を分離することが可能となる。

【０６５６】

また、ステップＳ３０１において、現在時刻が、「マーク処理」の属性の時刻のうちのいずれかと一致すると判定された場合、デコード制御モジュール２１４は、現在時刻が、「マーク処理」の属性の時刻となった旨のメッセージと、現在時刻と一致した「マーク処理」の属性の時刻とを、プレイヤ制御モジュール２１２に供給して、ステップＳ３０２に進む。

【０６５７】

ステップＳ３０２では、プレイヤ制御モジュール２１２が、現在時刻が、「マーク処理」の属性の時刻となった旨のメッセージと、現在時刻と一致した「マーク処理」の属性の時刻(mark_time_stamp)とを、デコード制御モジュール２１４から受信し、mark_time_stampが現在時刻に一致したMark()を、マーク処理の処理対象とするMark()（以下、適宜、処理対象markという）として認識する。

【０６５８】

即ち、プレイヤ制御モジュール２１２は、現在再生されているPlayList()のPlayItem()を認識しており、そのPlayList()およびPlayItem()と、デコード制御モジュール２１４からの、現在時刻と一致した「マーク処理」の属性の時刻(mark_time_stamp)（以下、適宜、マーク時刻という）とから、”PLAYLIST.DAT”ファイル（図７）のPlayListMark()（図９）を参照することにより、処理対象markを認識する。

【０６５９】

具体的には、例えば、いま、図２９に示した１番目のPlayList#0の１番目のPlayItem#0が再生されているとすると、そのことにより、プレイヤ制御モジュール２１２は、マーク時刻が、図３２上側に示したPlayListMark()に含まれる７つのMark()のうちの１番目から４番目までの４つのMark()のうちのいずれかのmark_time_stampであることを認識する。

【０６６０】

そして、デコード制御モジュール２１４からプレイヤ制御モジュール２１２に供給されたマーク時刻が、例えば、16,380,090であったとすると、プレイヤ制御モジュール２１２は、図３２上側に示したPlayListMark()に含まれる１番目から４番目までの４つのMark()のうちの、mark_time_stampが、マーク時刻である16,380,090に一致する４番目のMark()を、処理対象markとして認識する。

【０６６１】

プレイヤ制御モジュール２１２は、以上のようにして、処理対象markを認識すると、ステップＳ３０２からＳ３０３に進み、処理対象markにおいて、エレメンタリストリームを特定するentry_ES_stream_idとentry_ES_private_stream_id（図９）が記述されているかどうかを判定する。

【０６６２】

ステップＳ３０３において、処理対象markに、エレメンタリストリームを特定するentry_ES_stream_idとentry_ES_private_stream_id（図９）が記述されていないと判定された場合、即ち、entry_ES_stream_idとentry_ES_private_stream_idが、いずれも0x00である場合、ステップＳ３０４をスキップして、ステップＳ３０５に進み、以下、処理対象markに応じた処理が行われる。

【０６６３】

また、ステップＳ３０３において、処理対象markに、エレメンタリストリームを特定するentry_ES_stream_idとentry_ES_private_stream_id（図９）が記述されていると判定された場合、ステップＳ３０４に進み、プレイヤ制御モジュール２１２は、再生中のエレメンタリストリームに、そのentry_ES_stream_id、さらには、必要に応じてentry_ES_private_stream_idによって特定されるエレメンタリストリームが含まれるかどうかを判定する。

【０６６４】

ステップＳ３０４において、再生中のエレメンタリストリームに、処理対象markのentry_ES_stream_idとentry_ES_private_stream_idによって特定されるエレメンタリストリームが含まれないと判定された場合、ステップＳ３０１に戻る。即ち、処理対象markのentry_ES_stream_idとentry_ES_private_stream_idによって特定されるエレメンタリストリームが再生されていない場合、処理対象markは、無視される。

【０６６５】

一方、ステップＳ３０４において、再生中のエレメンタリストリームに、処理対象markのentry_ES_stream_idとentry_ES_private_stream_idによって特定されるエレメンタリストリームが含まれると判定された場合、即ち、処理対象markのentry_ES_stream_idとentry_ES_private_stream_idによって特定されるエレメンタリストリームが再生されている場合、処理対象markは有効であるとして、ステップＳ３０５に進み、以下、その処理対象markに応じた処理が行われる。

【０６６６】

即ち、ステップＳ３０５では、プレイヤ制御モジュール２１２は、処理対象markのmark_type（図９）を参照することにより、その処理対象markを判定する。

【０６６７】
ステップＳ３０５において、処理対象markが、チャプタマークまたはインデクスマークであると判定された場合、即ち、処理対象markのmark_typeが、'Chapter'または'Index'である場合、ステップＳ３０６に進み、プレイヤ制御モジュール２１２は、グラフィクス処理モジュール２１９に命じて、チャプタまたはインデクスの番号の表示を、処理対象markであるチャプタマークまたはインデクスマークが表すチャプタまたはインデクスの番号に更新させて、ステップＳ３０１に戻る。
【０６６８】
また、ステップＳ３０５において、処理対象markが、イベントマークであると判定された場合、即ち、処理対象markのmark_typeが、'Event'である場合、ステップＳ３０７に進み、プレイヤ制御モジュール２１２は、イベントの発生を表すイベントメッセージと、処理対象markのmark_dataを、スクリプト制御モジュール２１１に通知（供給）して、ステップＳ３０８に進む。
【０６６９】
ステップＳ３０８では、スクリプト制御モジュール２１１が、プレイヤ制御モジュール２１２からのイベントメッセージとmark_dataとを受信し、イベントメッセージを割り込み要求として、あらかじめ”SCRIPT.DAT”ファイルに記述された一連の処理を、mark_dataを引数として行って、ステップＳ３０１に戻る。
【０６７０】
即ち、スクリプト制御モジュール２１１では、mark_dataに対応した処理が行われる。
【０６７１】
具体的には、例えば、図３２下側に示したPlayList #1のPlayListMark()では、２番目のMark()（Mark#1）と３番目のMark()（Mark#2）とは、いずれも、mark_typeが'Event'であるが、mark_dataは、それぞれ1（Mark#1）と2（Mark#2）で異なっている。
【０６７２】
スクリプト制御モジュール２１１は、２番目のMark()に対応するイベントメッセージを受信した場合と、３番目のMark()に対応するイベントメッセージを受信した場合の、いずれも場合も、そのイベントメッセージに応じて、同一のイベントハンドラ（割り込み処理ルーチン）で処理を行うが、イベントハンドラ内において、イベントメッセージとともに供給されるmark_dataを検査することにより、イベントメッセージに対して、mark_dataごとに異なる処理を行う。
【０６７３】
具体的には、例えば、mark_dataが1である場合には、スクリプト制御モジュール２１１は、グラフィクス処理モジュール２１９を制御して、第１の種類のアイコンの表示を行わせる。また、例えば、mark_dataが2である場合、スクリプト処理モジュール２１１は、グラフィクス処理モジュール２１９を制御して、第２の種類のアイコンの表示を行わせる。
【０６７４】
なお、mark_dataは、１や２に限定されるものではなく、また、mark_dataに対応して行われる処理も、上述したような、単なるアイコンの表示限定されるものではない。
【０６７５】
即ち、例えば、mark_dataが3乃至18の範囲の値である場合には、スクリプト制御モジュール２１１は、グラフィクス処理モジュール２１９を制御し、第１の種類のアイコンの表示を、mark_dataから2を減じた値（1乃至16の数値）に対応する明るさで行わせる。また、例えば、mark_dataが19乃至34の範囲の値である場合には、スクリプト制御モジュール２１１は、グラフィクス処理モジュール２１９を制御し、第２の種類のアイコンの表示を、mark_dataから18を減じた値（1乃至16の数値）に対応する明るさで行わせる。
【０６７６】
その他、例えば、入力インターフェース１１５（図１）に、ユーザが操作するコントローラが接続されており、そのコントローラが、DC(Direct Current)モータの軸に偏芯させたおもりを取り付けた、DCモータを動作させると振動が発生する振動モータを内蔵する場

合には、mark_dataが35乃至42の範囲の値であるときに、その振動モータを、mark_dataから34を減じた値（1乃至8の数値）に応じた動作時間だけ動作させることができる。

【０６７７】

mark_dataは数値であり、その使用法やアルゴリズムは、スクリプト制御モジュール２１１が実行するスクリプトプログラムにより記述することができる。従って、mark_dataは、事前に取り決められたルールで使用する他、ディスク１０１の製造者、あるいはディスク１０１に記録されるデータを提供するコンテンツプロバイダなどが独自に設定したルールで使用することが可能である。

【０６７８】

以上のように、マーク処理では、現在時刻が、「マーク処理」の属性の時刻と一致すると、その「マーク処理」の属性の時刻であるマーク時刻から、処理対象markが認識される。さらに、処理対象markにおいて、エレメンタリストリームを特定するentry_ES_stream_idとentry_ES_private_stream_idが記述されていない場合には、処理対象markのmark_typeに応じた処理が行われる。また、処理対象markにおいて、エレメンタリストリームを特定するentry_ES_stream_idとentry_ES_private_stream_idが記述されている場合であっても、そのentry_ES_stream_idとentry_ES_private_stream_idによって特定されるエレメンタリストリームが再生中であれば、処理対象markのmark_typeに応じた処理が行われる。

【０６７９】

従って、例えば、いま、図２９に示した２番目のPlayList#1の再生が行われているとすると、以下のようなマーク処理が行われる。

【０６８０】

即ち、２番目のPlayList#1のPlayListMark()においては、図３２下側に示したように、mark_time_stampがそれぞれ90,000, 27,090,000, 27,540,000に指定されている１番目のMark()(Mark#0)、２番目のMark()（Mark#1）、３番目のMark()（Mark#2）が記述されている。

【０６８１】

さらに、図３２下側のPlayListMark()においては、２番目のMark()と３番目のMark()のentry_ES_stream_idには、それぞれ、0xE0と0xE1が記述されているから、２番目のMark()と３番目のMark()は、それぞれ、stream_idが0xE0と0xE1で特定されるエレメンタリストリームが関連付けられている。

【０６８２】

ここで、図２９で説明したように、２番目のPlayList#1には、１つのPlayItem()(PlayItem#0)だけが記述され、そのPlayItem#0によれば、クリップストリームファイル”00003.PS”が再生される。そして、クリップストリームファイル”00003.PS”には、そのクリップストリームファイル”00003.PS”に対応する図３０のクリップ情報ファイル”00003.CLP”で説明したように、0xE0となっているstream_idで特定されるビデオストリームstream#0、0xE1となっているstream_idで特定されるビデオストリームstream#1、0xBDとなっているstream_idおよび0x00となっているprivate_stream_idで特定されるオーディオストリームstream#2の３つのエレメンタリストリームが多重化されている。

【０６８３】

従って、図３２下側のPlayListMark()の２番目のMark()には、クリップストリームファイル”00003.PS”に多重化されている、stream_idが0xE0となっているビデオストリームstream#0が関連付けられており、３番目のMark()には、クリップストリームファイル”00003.PS”に多重化されている、stream_idが0xE1となっているビデオストリームstream#1が関連付けられている。

【０６８４】

図２９の２番目のPlayList#1のPlayItem#0の再生が開始される場合、図３４のステップＳ１２４で説明したようにして、プレイヤ制御モジュール２１２は、図３２下側に示したPlayListMark()に含まれる３つのMark()が、PlayList#1のPlayItem#0に属していることを認識し、その３つのMark()のmark_time_stampである{90,000}、{27,090,000}、{27,540,0

00}を、そのmark_time_stampが表す時刻の属性が「マーク処理」である旨とともに、デコード制御モジュール２１４に渡している。

【０６８５】

マーク処理では、デコード制御モジュール２１４が、PlayList#1のPlayItem#0の再生中に、計時部２１４Ａによって計時される現在時刻が、属性が「マーク処理」の時刻{90,000}、{27,090,000}、{27,540,000}のうちのいずれかに一致するかを、常時確認しており（ステップＳ３０１）、現在時刻が、属性が「マーク処理」の時刻に一致すると、現在時刻と一致した「マーク処理」の属性の時刻であるマーク時刻と、現在時刻が、「マーク処理」の属性の時刻となった旨のメッセージとを、プレイヤ制御モジュール２１２に供給する。

【０６８６】

即ち、例えば、いま、現在時刻が、「マーク処理」の属性の時刻{90,000}、{27,090,000}、{27,540,000}のうちの、27,090,000に一致したとすると、デコード制御モジュール２１４は、現在時刻と一致した「マーク処理」の属性の時刻であるマーク時刻27,090,000と、現在時刻が、「マーク処理」の属性の時刻となった旨のメッセージとを、プレイヤ制御モジュール２１２に供給する。

【０６８７】

プレイヤ制御モジュール２１２は、PlayList#1のPlayItem#0が現在再生されていることを認識しており、そのPlayList#1の図３２下側に示したPlayListMark()に記述されたMark()のうちの、PlayItem#0に属する３つのMark()のmark_time_stampである90,000, 27,090,000, 27,540,000それぞれと、デコード制御モジュール２１４からのマーク時刻である27,090,000とを比較することにより、そのマーク時刻27,090,000に、mark_time_stampが一致するMark()、即ち、図３２下側のPlayListMark()に記述された２番目のMark()(Mark#1)を、処理対象markとして認識する（ステップＳ３０２）。

【０６８８】

処理対象markである、図３２下側のPlayListMark()に記述された２番目のMark()においては、entry_ES_stream_idとして、0xE0が指定されている。この0xE0となっているentry_ES_stream_idは、上述したことから、クリップストリームファイル”00003.PS”に多重化されている、stream_idが0xE0となっているビデオストリームstream#0（図３０）を表すものであり、プレイヤ制御モジュール２１２は、再生中のエレメンタリストリームの中に、そのビデオストリームstream#0が含まれるかどうかを判定する（ステップＳ３０３，Ｓ３０４）。

【０６８９】

そして、再生中のエレメンタリストリームの中に、ビデオストリームstream#0が含まれない場合には、処理対象markは無視される（ステップＳ３０４）。

【０６９０】

一方、再生中のエレメンタリストリームの中に、ビデオストリームstream#0が含まれる場合には、処理対象markは有効であるとして、その処理対象markに応じた処理が行われる（ステップＳ３０５乃至Ｓ３０８）。

【０６９１】

即ち、いまの場合、処理対象markである、図３２下側のPlayListMark()に記述された２番目のMark()は、そのmark_typeが'Event'になっているからイベントマークであり、従って、プレイヤ制御モジュール２１２は、イベントの発生を表すイベントメッセージと、処理対象markのmark_dataを、スクリプト制御モジュール２１１に供給する（ステップＳ３０５，Ｓ３０７）。そして、スクリプト制御モジュール２１１では、プレイヤ制御モジュール２１２からのイベントメッセージを割り込み要求として、あらかじめ”SCRIPT.DAT”ファイルに記述された一連の処理を、そのイベントメッセージとともに供給されたmark_dataを引数として行う（ステップＳ３０８）。

【０６９２】

以上のように、マーク処理によれば、PlayList()の時間軸上の１つの再生時刻を表すma

rk_time_stampと、Mark()のタイプを表すmark_typeと、イベントマークの引数となるmark_dataとを含む０以上のMark()を有するPlayListMark()（図９）を含むPlayList()（図７）にしたがって再生されているクリップストリームファイルの再生時刻である現在時刻が、mark_time_stampに一致するか否かが判定され、現在時刻が、mark_time_stampに一致する場合に、その一致した現在時刻であるマーク時刻に等しいmark_time_stampを有するMark()が、処理対象markとして認識される。さらに、その処理対象markが有するmark_typeが、イベントを発生させるタイプを表している場合、即ち、処理対象markが、イベントマークである場合、処理対象markが有するmark_dataとイベントメッセージとが通知され、そのmark_dataに応じた処理が実行される。従って、クリップストリームファイルの再生時刻に応じ、mark_dataに応じた処理を実行することが可能となる。

【０６９３】

［out_time処理における一致判定］

ところで、上述したように、デコード制御モジュール２１４は計時部２１４Ａが計時している時刻が、プレイヤ制御モジュール２１２から与えられたPlayItemのOUT_timeに等しくなると、デコード中断制御が行われ、PlayItemの再生が終了されるという制御が行われる。本実施例では、PlayItem#0の終了については、図４０のフローチャートにおけるステップＳ１５１で説明されている。

【０６９４】

この場合にも、時刻とOUT_timeの単純に比較での一致判定では、問題が生じる可能性がある。このため、時刻とOUT_timeの比較においても上述した一致判定の定義を使用する。

【０６９５】

すなわち、図５１で示されるようにplayListEndに対応するPlayItemのOUT_timeが、playListで最後に表示されるFoCFP（再生中のビデオストリーム内のframe or complementary field pair）のPETより小さい場合、PlayListEndの時刻に相当するOUT_timeを表示durationに含むFoCFPの表示開始時刻(PTS)にPlayListEndイベントが発生するとき、すなわち、$PTS_{FOCFP[3]} \leqq OUT_time < PET_{FOCFP[3]}$であるとき、FoCFP[3]の表示開始時刻$PTS_{FOCFP[3]}$で、PlayListEndイベントが発生する。ここで$PET_{FOCFP[k]}$は「$PTS_{FOCFP[k]}$にpic_structで決まる表示durationを加えた時刻」を示す。

【０６９６】

これにより、時刻の一致判定がビデオ出力の際に限定されるため、処理が軽くなる。また、上述したようにビデオストリームの準備とデータベースの準備の独立性が高まる。

【０６９７】

さらに、PlayItemの再生終了がデコード制御モジュール２１４からプレイヤ制御モジュール２１２に伝えられ、プレイヤ制御モジュール２１２がそれを再生中の最後のPlayListのPlayItemであると判断すると、スクリプト制御モジュール２１１に対してplayListEndのイベントを発生する。

【０６９８】

スクリプト制御モジュール２１１は、playListEndのイベントを受けるとることにより、再生を指示したPlayListの再生が終了したことを知り、その後プログラミングされてある動作を続行する。すなわち、別のplayListを再生する、メニューを表示する、動作を終了する等が実行される。

【０６９９】

なお、図５２で示されるような場合、すなわち、OUT_timeがPlayItem中で最後に表示される画像の表示終了時刻に等しい場合には、上記の一致判定では処理できない場合がある。図５２において、例えば、FoCFP[2]を表示してpauseしているとき、playStop()がcallされると、FoCFP[3]を表示してpauseする。その後再度playStop()がcallされると表示される画像には変更なくplayListEndが発生する。

【０７００】

すなわち、最後の画像の表示開始時刻＋pic_structで決まるduration＝OUT_timeとなってしまうため、最後の画像の表示開始時刻＋pic_structで決まるduration＜OUT_timeを満

たさない。
【０７０１】
このような場合には、ビデオデコーダ制御モジュール２１６が最後の画像を出力した後、該当画像の表示duration経過後に表示終了を示す情報をデコード制御モジュール２１４に渡し、これにより時計を”最後の画像の表示開始時刻＋pic_structで決まるduration”に進めることで、一致条件を満たすようにすることが出来る。
【０７０２】
［字幕デコード］
字幕デコーダ制御モジュール２１８は、バッファ制御モジュール２１５（図５）の字幕読み出し機能部２３５から、バッファ２１５Ａに記憶された１つの字幕アクセスユニットと、その字幕アクセスユニットに付加されているタイムスタンプを得るたびに、内部に持つ字幕デコードソフトウェアでデコードを開始し、同時にタイムスタンプとdurationをデコード制御モジュール２１４に渡す。
【０７０３】
デコード制御モジュール２１４は、ビデオデコーダ制御モジュール２１６からの情報により時計（計時部２１４Ａ）の時刻を変更した際に、字幕デコーダ制御モジュール２１８から渡された字幕アクセスユニットのＰＴＳを吟味する。すなわち、字幕アクセスユニットのPTSと時刻が上述の一致判定基準に照らした上で一致と判断されると、デコード制御モジュール２１４はグラフィクス処理モジュール２１９に対して字幕入力の指示を行うと同時に、字幕デコーダ制御モジュール２１８に対して出力の指示を行う。
【０７０４】
デコード制御モジュール２１４から出力の指示を受けた字幕デコーダ制御モジュール２１８は、デコードされた字幕画像データをグラフィクス処理モジュール２１９に供給する。グラフィクス処理モジュール２１９は入力された字幕データを蓄積し、以降入力するビデオデータに対して合成を行う。
【０７０５】
デコード制御モジュール２１４はさらに字幕の表示durationの検査を行う。すなわち、”該当字幕の表示開始時刻＋表示duration”の値と現在時刻が、上述の一致判定基準に照らした上で一致と判断されると、デコード制御モジュール２１４はグラフィクス処理モジュール２１９に対して字幕の消去を命じる。グラフィクス処理モジュール２１９は蓄積している入力された字幕データを消去し、以降入力するビデオデータに対しての合成は行わない。
【０７０６】
［マーク間隔の必然性］
以上に説明した一致判定基準では、一定の範囲の時刻を単一時刻に量子化している。すなわち、あるビデオデータの表示開始時刻≦ｔ＜表示終了時刻を見たす全てのｔの時刻は該当ビデオデータの表示開始時刻に量子化（rouding）が行われている。
【０７０７】
このため、隣り合う二つのイベント（あるいはplaylistEnd）の位置関係によっては、同一の時刻に丸められてしまうことがある。例えば図５０の例では、186,000に置かれたイベントの直前のイベントのmark_time_stampが184,000であった場合、この二つのイベントはいずれもP1の表示開始時刻に発生すると定義される。
【０７０８】
このような状況を避けるためには、単一のビデオに対しては単一のイベントのみを設定できることを保証する必要がある。このため、隣あったイベントの間隔を3フィールド以上（pic_structで設定される最大表示時間以上）とすることで、上記条件を保証する。
【０７０９】
図５３は、上記条件の一例を示している。すなわち、図５３においては、Case１は、フレームレート（frame rate）が5005/240000（プログレッシブの23.976Hz）であって、その場合、90kHzで表現した最小イベント間隔は、7507となり、Case２は、フレームレート

（frame rate）が4004/240000（インタレースの59.94Hz）であって、その場合、90kHzで表現した最小イベント間隔は、6006となることが示されている。

【０７１０】

AVCやMPEG2 Video等のビデオ符号化方式では、2-3プルダウン画像を効率よく符号化するために、1フレーム信号を3フィールドの時間だけ表示するという機能を持っている。このため1フレームの信号の最大のdurationは3フィールドである。すなわち、隣り合ったイベントの間隔を3フィールド以上の時間だけ離すことによって、この二つの隣り合ったイベントが単一のビデオデータの表示開始時刻に発生すると判断されることを防ぐというものである。

【０７１１】

また、3フィールドを越える間隔で定義することも可能である。例えば隣り合ったイベントの間隔を2フレーム以上と定義することも可能である。

【０７１２】

なお、全てのイベントが対応するビデオデータを調査し、それらが重複していないということを確認することによって上記を保証する方法も考えられる。

【０７１３】

［出力属性の制御処理］

次に、図５４のフローチャートを参照して、図３４のステップＳ１２６などで行われる出力属性の制御処理の詳細について説明する。

【０７１４】

図３４のステップＳ１２６で説明したように、プレイヤ制御モジュール２１２は、まず、再生対象の１以上のエレメンタリストリーム、即ち、図３４のステップＳ１２５で再生すると決定した１以上のエレメンタリストリームそれぞれについて、出力属性が記述されるDynamicInfo()（図１５）の数を表すnumber_of_DynamicInfo（図１２）を調査する。

【０７１５】

そして、再生対象の１以上のエレメンタリストリームのすべてについて、number_of_DynamicInfoが0になっている場合、プレイヤ制御モジュール２１２は、特に処理を行わない。

【０７１６】

一方、再生対象のエレメンタリストリームについてのnumber_of_DynamicInfoが0でない場合、プレイヤ制御モジュール２１２は、図５４のフローチャートにしたがった出力属性の制御処理を行う。

【０７１７】

従って、ディスク１０１に記録された３つのクリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”が、例えば、図３０に示したようになっている場合に、クリップ情報ファイル”00001.CLP”に対応するクリップストリームファイル”00001.PS”（を再生する１番目のPlayList#0の１番目のPlayItem#0）が再生されるときには、クリップ情報ファイル”00001.CLP”（図３０）では、クリップストリームファイル”00001.PS”に多重化されている４つのエレメンタリストリームstream#0乃至stream#3のすべてについて、number_of_DynamicInfoが0になっているから、出力属性の制御処理は行われない。

【０７１８】

同様に、クリップ情報ファイル”00002.CLP”に対応するクリップストリームファイル”00002.PS”（を再生する１番目のPlayList#0の２番目のPlayItem#1）が再生されるときも、クリップ情報ファイル”00002.CLP”（図３０）では、クリップストリームファイル”00002.PS”に多重化されている４つのエレメンタリストリームstream#0乃至stream#3のすべてについて、number_of_DynamicInfoが0になっているから、出力属性の制御処理は行われない。

【０７１９】

一方、クリップ情報ファイル”00003.CLP”に対応するクリップストリームファイル”00003.PS”（を再生する２番目のPlayList#1のPlayItem#0）が再生されるときは、クリッ

プ情報ファイル”00003.CLP”（図３０）において、クリップストリームファイル”00003.PS”に多重化されている３つのエレメンタリストリームstream#0乃至stream#2のうちの、１番目のエレメンタリストリームであるビデオストリームstream#0と、３番目のエレメンタリストリームであるオーディストリームstream#2について、number_of_DynamicInfoが0でない2と3に、それぞれなっているから、出力属性の制御処理が行われる。

【０７２０】

即ち、出力属性の制御処理では、まず最初に、ステップＳ３２０において、プレイヤ制御モジュール２１２は、再生対象のクリップストリームファイルに対応するクリップ情報ファイルClip()（図１２）に記述されたpts_change_pointを、「DynamicInfo()処理」の属性の時刻である旨とともに、デコード制御モジュール２１４に渡し、デコード制御モジュール２１４は、プレイヤ制御モジュール２１２からの「DynamicInfo()処理」の属性の時刻であるpts_change_pointを受信して、ステップＳ３２１に進む。

【０７２１】

ステップＳ３２１では、デコード制御モジュール２１４が、計時部２１４Ａによって計時されている現在時刻が、「DynamicInfo()処理」の属性の時刻であるpts_change_point（のいずれか）に一致したかどうかを判定し、一致していないと判定した場合、ステップＳ３２１に戻る。

【０７２２】

また、ステップＳ３２１において、現在時刻が、「DynamicInfo()処理」の属性の時刻（のいずれか）に一致したと判定された場合、デコード制御モジュール２１４は、現在時刻が、「DynamicInfo()処理」の属性の時刻となった旨のメッセージと、現在時刻と一致した「DynamicInfo()処理」の属性の時刻（以下、適宜、DynamicInfo時刻という）とを、プレイヤ制御モジュール２１２に供給して、ステップＳ３２２に進む。

【０７２３】

ステップＳ３３２では、プレイヤ制御モジュール２１２が、現在時刻が、「DynamicInfo()処理」の属性の時刻となった旨のメッセージと、DynamicInfo時刻とを、デコード制御モジュール２１４から受信し、そのDynamicInfo時刻に一致するpts_change_point（図１２）とセットになっているDynamicInfo()を、処理対象のDynamicInfo()である処理対象DynamicInfo()として認識して、ステップＳ３２３に進む。

【０７２４】

ステップＳ３２３では、プレイヤ制御モジュール２１２は、処理対象DynamicInfo()となっているDynamicInfo()（図１５）に記述された出力属性を、グラフィクス処理モジュール２１９またはオーディオ出力モジュール２２１に供給して、ステップＳ３２４に進む。

【０７２５】

ステップＳ３２４では、グラフィクス処理モジュール２１９またはオーディオ出力モジュール２２１が、直前のステップＳ３２３でプレイヤ制御モジュール２１２から供給された出力属性にしたがって、ビデオデータまたはオーディオデータの出力の制御を、それぞれ開始し、ステップＳ３２１に戻る。

【０７２６】

これにより、ビデオデータが、出力属性（表示方式）として記述された、例えばアスペクト比に応じて出力され、あるいは、オーディオデータが、出力属性（出力方式）として記述された、例えば、ステレオまたはデュアル（二ヶ国語）に応じて出力される。

【０７２７】

次に、図５５を参照して、出力属性の制御処理の詳細について、さらに説明する。

【０７２８】

即ち、図５５は、図３０のクリップ情報ファイル”00003.CLP”に記述されているpts_change_pointとDynamicInfo()とのセット（図１２）を示している。

【０７２９】

ここで、上述したように、クリップストリームファイル”00003.PS”に多重化されてい

る３つのエレメンタリストリームstream#0乃至stream#2のうちの、１番目のエレメンタリストリームであるビデオストリームstream#0と、３番目のエレメンタリストリームであるオーディストリームstream#2については、図３０のクリップ情報ファイル”00003.CLP”において、number_of_DynamicInfoが、それぞれ2と3になっている。従って、クリップ情報ファイル”00003.CLP”において、クリップストリームファイル”00003.PS”の１番目のビデオストリームstream#0については、２セットのpts_change_pointおよびDynamicInfo()が記述されており、３番目のオーディオストリームstream#2については、３セットのpts_change_pointおよびDynamicInfo()が記述されている。

【０７３０】

図５５上側は、クリップストリームファイル”00003.PS”の１番目のビデオストリームstream#0について記述されている２セットのpts_change_pointおよびDynamicInfo()を示しており、図５５下側は、クリップストリームファイル”00003.PS”の３番目のオーディオストリームstream#2について記述されている３セットのpts_change_pointおよびDynamicInfo()を示している。

【０７３１】

なお、図５５上側では、１番目のビデオストリームstream#0について記述されている２セットのpts_change_pointおよびDynamicInfo()の他に、そのビデオストリームstream#0について、図３０のクリップ情報ファイル”00003.CLP”に記述されているstream_id(=0xE0), private_stream_id(=0x00), number_of_DynamicInfo(=2)も、図示してある。同様に、図５５下側でも、３番目のオーディオストリームstream#2について記述されている３セットのpts_change_pointおよびDynamicInfo()の他に、そのオーディオストリームstream#2について、図３０のクリップ情報ファイル”00003.CLP”に記述されているstream_id(=0xBD), private_stream_id(=0x00), number_of_DynamicInfo(=3)も、図示してある。

【０７３２】

図５５上側において、ビデオストリームstream#0について記述されている２セットのpts_change_pointおよびDynamicInfo()のうちの１セット目では、pts_change_pointが90,000になっており、DynamicInfo()のdisplay_aspect_ratio（図１５）が'4:3'になっている。さらに、その２セット目では、pts_change_pointが54,090,000になっており、DynamicInfo()のdisplay_aspect_ratioが'16:9'になっている。

【０７３３】

一方、図５５下側において、オーディオストリームstream#2について記述されている３セットのpts_change_pointおよびDynamicInfo()のうちの１セット目では、pts_change_pointが90,000になっており、DynamicInfo()のchannel_assignment（図１５）が'Dual'になっている。さらに、その２セット目では、pts_change_pointが27,090,000になっており、DynamicInfo()のchannel_assignmentが'Stereo'になっている。また、その３セット目では、pts_change_pointが32,490,000になっており、DynamicInfo()のchannel_assignmentが'Dual'になっている。

【０７３４】

例えば、いま、図３４のステップＳ１２５において、クリップストリームファイル”00003.PS”の、0xE0となっているstream_idで特定される１番目のビデオストリームstream#0と、0xBDとなっているstream_idおよび0x00となっているprivate_stream_idで特定される３番目のオーディオストリームstream#2とが、再生対象のストリームとして決定されたとする。

【０７３５】

この場合、プレイヤ制御モジュール２１２は、0xE0となっているstream_idで特定されるビデオストリームstream#0について記述されている図５５上側の２セットのpts_change_pointおよびDynamicInfo()と、0xBDとなっているstream_idおよび0x00となっているprivate_stream_idで特定されるオーディオストリームstream#2について記述されている図５５下側の３セットのpts_change_pointおよびDynamicInfo()との中のpts_change_pointを調査し、初期値を認識する。

【０７３６】
即ち、0xE0となっているstream_idで特定されるビデオストリームstream#0について記述されている図５５上側の２セットのpts_change_pointおよびDynamicInfo()のうちの１セット目では、pts_change_pointが90,000になっている。そして、この90,000という時刻は、ビデオストリームstream#0が多重化されているクリップストリームファイル”00003.PS”に対応する図３０のクリップ情報ファイル”00003.CLP”において、クリップストリームファイル”00003.PS”の先頭の時刻を表すpresentation_start_timeに記述されている時刻90,000に一致する。
【０７３７】
同様に、0xBDとなっているstream_idおよび0x00となっているprivate_stream_idで特定されるオーディオストリームstream#2について記述されている図５５下側の３セットのpts_change_pointおよびDynamicInfo()のうちの１セット目では、pts_change_pointが90,000になっている。そして、この90,000という時刻は、オーディオストリームstream#2が多重化されているクリップストリームファイル”00003.PS”に対応する図３０のクリップ情報ファイル”00003.CLP”において、クリップストリームファイル”00003.PS”の先頭の時刻を表すpresentation_start_timeに記述されている時刻90,000に一致する。
【０７３８】
プレイヤ制御モジュール２１２は、クリップストリームファイル”00003.PS”の先頭の時刻を表すpresentation_start_timeに記述されている時刻90,000に一致するpts_change_pointを、初期値として認識する。従って、図５５上側の２セットのpts_change_pointおよびDynamicInfo()のうちの１セット目のpts_change_pointと、図５５下側の３セットのpts_change_pointおよびDynamicInfo()のうちの１セット目のpts_change_pointが、初期値として認識される。
【０７３９】
そして、プレイヤ制御モジュール２１２は、クリップストリームファイル”00003.PS”の再生が開始される前に（図３４のステップＳ１２６で）、初期値として認識したpts_change_pointとセットになっているDynamicInfo()にしたがって、対応するエレメンタリストリームの出力属性を指示する。
【０７４０】
即ち、0xE0となっているstream_idで特定されるビデオストリームstream#0については、図５５上側で、初期値である90,000になっているpts_change_pointとセットになっているDynamicInfo()において、display_aspect_ratioが'4:3'になっている。この場合、プレイヤ制御モジュール２１２は、display_aspect_ratioが'4:3'になっている旨、即ち、ビデオストリームstream#0が、４：３のアスペクト比のビデオデータである旨の出力属性の情報を、グラフィクス処理モジュール２１９を制御する。
【０７４１】
また、0xBDとなっているstream_idおよび0x00となっているprivate_stream_idで特定されるオーディオストリームstream#2については、図５５下側で、初期値である90,000になっているpts_change_pointとセットになっているDynamicInfo()において、channel_assignmentが'Dual'になっている。この場合、プレイヤ制御モジュール２１２は、channel_assignmentが'Dual'になっている旨、即ち、オーディオストリームstream#2が、デュアルのオーディオデータである旨の出力属性の情報を、オーディオ出力モジュール２２１に供給する。
【０７４２】
ここで、図３４のステップＳ１２６では、以上のような初期値としてのpts_change_pointを対象とした出力属性の制御処理が行われる。
【０７４３】
その後、プレイヤ制御モジュール２１２は、ビデオストリームstream#0についての図５５上側の２つのpts_change_pointである90,000および54,090,000と、オーディオストリームstream#2についての図５５下側の３つのpts_change_pointである90,000, 27,090,000、

および32,490,000のうちの、初期値90,000以外の時刻である{27,090,000}，{32,490,000}，{54,090,000}を、「DynamicInfo()処理」の属性の時刻である旨とともに、デコード制御モジュール２１４に渡す（ステップＳ３２０）。

【０７４４】

デコード制御モジュール２１４は、プレイヤ制御モジュール２１２からの、「DynamicInfo()処理」の属性の時刻{27,090,000}，{32,490,000}，{54,090,000}を受信し、さらに、ビデオストリームstream#0およびオーディオストリームstream#2の再生（クリップストリームファイル”00003.PS”を再生する２番目のPlayList#1のPlayItem#0の再生）の開始後、計時部２１４Ａによって計時されている現在時刻の監視を開始する。

【０７４５】

そして、デコード制御モジュール２１４は、現在時刻が、「DynamicInfo()処理」の属性の時刻{27,090,000}，{32,490,000}，{54,090,000}のうちのいずれかに一致した場合、その現在時刻と一致した「DynamicInfo()処理」の属性の時刻であるDynamicInfo時刻を、プレイヤ制御モジュール２１２に供給する（ステップＳ３２１）。

【０７４６】

即ち、例えば、現在時刻が、27,090,000になったとすると、デコード制御モジュール２１４は、「DynamicInfo()処理」の属性の時刻のうちの、現在時刻と一致する27,090,000を、DynamicInfo時刻として、プレイヤ制御モジュール２１２に供給する

【０７４７】

プレイヤ制御モジュール２１２は、デコード制御モジュール２１４からのDynamicInfo時刻である27,090,000を受信し、ビデオストリームstream#0についての図５５上側の２つのpts_change_pointと、オーディオストリームstream#2についての図５５下側の３つのpts_change_pointとの中から、DynamicInfo時刻である27,090,000に一致するpts_change_pointを調査し、その27,090,000に一致するpts_change_pointとセットになっているDynamicInfo()、即ち、オーディオストリームstream#2についての図５５下側の２番目のDynamicInfo()を、処理対象DynamicInfo()として認識する（ステップＳ３２２）。

【０７４８】

処理対象DynamicInfo()が、ビデオストリームについてのDynamicInfo()である場合、プレイヤ制御モジュール２１２は、処理対象DynamicInfo()に記述されている出力属性を、グラフィクス処理モジュール２１９に供給する（ステップＳ３２３）。また、処理対象DynamicInfo()が、オーディオストリームについてのDynamicInfo()である場合、プレイヤ制御モジュール２１２は、処理対象DynamicInfo()に記述されている出力属性を、オーディオ出力モジュール２２１に供給する（ステップＳ３２３）。

【０７４９】

グラフィクス処理モジュール２１９は、プレイヤ制御モジュール２１２から出力属性が供給されると、その出力属性にしたがって、ビデオデータの出力の制御を開始する（ステップＳ３２４）。

【０７５０】

即ち、グラフィクス処理モジュール２１９は、例えば、プレイヤ制御モジュール２１２からの出力属性が表す、ビデオデータのアスペクト比の指示(display_aspect_ratio（図１５）)と、図１のビデオ出力端子１２０に接続されたビデオ出力装置のアスペクト比とに基づいて、ビデオ出力モジュール２２０に出力するビデオデータのアスペクト比の変換を行う。

【０７５１】

具体的には、例えば、ビデオ出力装置のアスペクト比が16:9である場合において、出力属性としてのビデオデータのアスペクト比の指示が4:3のアスペクト比を表しているときには、グラフィクス処理モジュール２１９は、ビデオ出力モジュール２２０に出力するビデオデータを、横方向にスクイーズ処理し、左右に黒味を入れて出力する。また、例えば、ビデオ出力装置のアスペクト比が4:3である場合において、出力属性としてのビデオデータのアスペクト比の指示が16:9のアスペクト比を表しているときには、グラフィクス処

理モジュール２１９は、ビデオ出力モジュール２２０に出力するビデオデータを、縦方向にスクイーズ処理し、上下に黒味を入れて出力する。さらに、例えば、ビデオ出力装置のアスペクト比と、出力属性としてのビデオデータのアスペクト比の指示が表すアスペクト比とが、いずれも、4:3や16:9で、同一である場合、グラフィクス処理モジュール２１９は、ビデオ出力モジュール２２０に出力するビデオデータを、スクイーズ処理することなく、そのまま出力する。

【０７５２】

ここで、図５５上側において、0xE0となっているstream_idで特定されるビデオストリームstream#0について記述されている２セットのpts_change_pointおよびDynamicInfo()によれば、ビデオストリームstream#0の再生開始時である時刻90,000から、時刻54,090,000の直前までは、ビデオストリームstream#0から、４：３のアスペクト比のビデオデータが得られる。そして、時刻54,090,000以後は、ビデオストリームstream#0から、１６：９のアスペクト比のビデオデータが得られる。

【０７５３】

従って、例えば、図１のビデオ出力端子１２０に接続されたビデオ出力装置のアスペクト比が4:3であるとすると、グラフィクス処理モジュール２１９では、時刻90,000から時刻54,090,000の直前までは、ビデオストリームstream#0から得られる４：３のアスペクト比のビデオデータが、そのまま４：３のアスペクト比のビデオ出力装置に供給されて表示される。

【０７５４】

そして、時刻54,090,000以後は、ビデオストリームstream#0から得られる１６：９のアスペクト比のビデオデータが、縦方向にスクイーズ処理され、さらに、上下に黒味が入った４：３のアスペクト比のビデオ信号に変換され、４：３のアスペクト比のビデオ出力装置に供給されて表示される。

【０７５５】

一方、オーディオ出力モジュール２２１は、プレイヤ制御モジュール２１２から出力属性が供給されると、その出力属性にしたがって、オーディオデータの出力の制御を開始する（ステップＳ３２４）。

【０７５６】

即ち、オーディオ出力モジュール２２１は、例えば、プレイヤ制御モジュール２１２からの出力属性が表す、オーディオデータのチャネル割り当ての指示(channel_assignment（図１５）)と、ユーザがリモコンを操作することによって入力インターフェース１１５（図１）を介してプレイヤ制御モジュール２１２から供給される音声出力モードとに基づいて、オーディオデコーダ制御モジュール２１７からのオーディオデータを処理し、オーディオ出力端子１２１（図１）に出力する。

【０７５７】

具体的には、例えば、出力属性が表すオーディオデータのチャネル割り当ての指示が、左チャネルが「主音声」のオーディオデータで、右チャネルの「副音声」のオーディオデータであるデュアル(Dual)（二ヶ国語）モードを表している場合、オーディオ出力モジュール２２１は、プレイヤ制御モジュール２１２から供給される音声出力モードにしたがって、オーディオデコーダ制御モジュール２１７からのオーディオデータを処理して、オーディオ出力端子１２１に出力する。

【０７５８】

即ち、音声出力モードとして、例えば、「主音声」が指定されているときには、オーディオ出力モジュール２２１は、オーディオデコーダ制御モジュール２１７からのオーディオデータのうちの左チャネルのオーディオデータを、右チャネルのオーディオデータとしてコピーし、その左チャネルと右チャネルのオーディオデータ（「主音声」のオーディオデータ）を、オーディオ出力端子１２１に出力する。また、音声出力モードとして、「副音声」が指定されているときには、オーディオ出力モジュール２２１は、オーディオデコーダ制御モジュール２１７からのオーディオデータのうちの右チャネルのオーディオデー

タを、左チャネルのオーディオデータとしてコピーし、その左チャネルと右チャネルのオーディオデータ（「副音声」のオーディオデータ）を、オーディオ出力端子１２１に出力する。さらに、音声出力モードとして、「主・副」が指定されているときには、オーディオ出力モジュール２２１は、オーディオデコーダ制御モジュール２１７からのオーディオデータを、そのまま、オーディオ出力端子１２１に出力する。

【０７５９】

また、例えば、出力属性が表すオーディオデータのチャネル割り当ての指示が、ステレオ(Stereo)モードを表している場合、オーディオ出力モジュール２２１は、プレイヤ制御モジュール２１２から供給される音声出力モードにかかわらず、オーディオデコーダ制御モジュール２１７からのオーディオデータを、そのまま、オーディオ出力端子１２１に出力する。

【０７６０】

ここで、図５５下側において、0xBDとなっているstream_idおよび0x00となっているprivate_stream_idで特定されるオーディオストリームstream#2について記述されている３セットのpts_change_pointおよびDynamicInfo()によれば、オーディオストリームstream#2の再生開始時である時刻90,000から、時刻27,090,000の直前までは、オーディオストリームstream#2から、デュアルのオーディオデータが得られる。また、時刻27,090,000から、時刻32,490,000の直前までは、オーディオストリームstream#2から、ステレオのオーディオデータが得られ、時刻32,490,000以後は、オーディオストリームstream#2から、デュアルのオーディオデータが得られる。

【０７６１】

従って、例えば、音声出力モードとして、「主音声」が指定されているとすると、オーディオ出力出力モジュール２２１では、時刻90,000から、時刻27,090,000の直前までは、オーディオストリームstream#2から得られるデュアルのオーディオデータのうちの左チャネルのオーディオデータが、右チャネルのオーディオデータとしてコピーされ、その左チャネルと右チャネルのオーディオデータが、オーディオ出力端子１２１に出力される。

【０７６２】

また、時刻27,090,000から、時刻32,490,000の直前までは、オーディオストリームstream#2から得られるステレオのオーディオデータが、そのまま、オーディオ出力端子１２１に出力される。

【０７６３】

そして、時刻32,490,000以後は、オーディオストリームstream#2から得られるデュアルのオーディオデータのうちの左チャネルのオーディオデータが、右チャネルのオーディオデータとしてコピーされ、その左チャネルと右チャネルのオーディオデータが、オーディオ出力端子１２１に出力される。

【０７６４】

以上のように、出力属性の制御処理では、クリップストリームファイルに多重化されているエレメンタリストリームごとに、そのエレメンタリストリームの再生時刻を表すpts_change_pointと、そのエレメンタリストリームの出力属性を含むDynamicInfo()とのセットを０セット以上含むクリップ情報ファイルClip()（図１２）の記述に基づき、再生中のエレメンタリストリームの再生時刻が、pts_change_pointに一致するか否かが判定される。そして、再生中のエレメンタリストリームの再生時刻が、pts_change_pointに一致する場合、そのpts_change_pointとセットになっているDynamicInfo()が認識され、その認識されたDynamicInfo()に含まれる出力属性にしたがって、再生中のエレメンタリストリームの出力が制御される。従って、エレメンタリストリームの再生時刻と出力属性に応じて、そのエレメンタリストリームの出力を制御することが可能となる。

【０７６５】

［字幕表示制御処理］

次に、図５６のフローチャートを参照して、字幕ストリームに対応する字幕データの表示を制御する字幕表示制御処理について説明する。

【０７６６】
PlayList()（図７）（のPlayItem()）の再生が開始されると、プレイヤ制御モジュール２１２は、ステップＳ３４１において、グラフィクス処理モジュール２１９に対する字幕データの表示方式の指示を初期化する。即ち、プレイヤ制御モジュール２１２は、字幕データの表示方式をデフォルトの表示方式とするように、グラフィクス処理モジュール２１９を制御する。なお、ステップＳ３４１で行われる表示方式の指示の初期化は、図３４の１２７で説明した表示方式の指示の初期化に対応する。

【０７６７】
ステップＳ３４１の処理後は、ステップＳ３４２に進み、プレイヤ制御モジュール２１２は、ユーザがリモコンを操作することにより入力インターフェース１１５から、字幕データの表示について、新たな表示方式の指示があったかどうかを判定する。

【０７６８】
ステップＳ３４２において、新たな表示方式の指示があったと判定された場合、ステップＳ３４３に進み、プレイヤ制御モジュール２１２は、字幕ストリーム（に対応する字幕データ）を、現在再生しているかどうかを判定する。

【０７６９】
ステップＳ３４３において、字幕ストリームが再生されていないと判定された場合、ステップＳ３４２に戻る。

【０７７０】
また、ステップＳ３４３において、字幕ストリームが再生されていると判定された場合、ステップＳ３４５に進み、プレイヤ制御モジュール２１２は、新たな表示方式の指示が、デフォルトの表示方式の指示であるかどうかを判定する。ステップＳ３４３において、新たな表示方式の指示が、デフォルトの表示方式の指示であると判定された場合、ステップＳ３４１に戻り、上述したように、プレイヤ制御モジュール２１２は、字幕データの表示方式をデフォルトの表示方式とするように、グラフィクス処理モジュール２１９を制御する。

【０７７１】
一方、ステップＳ３４５において、新たな表示方式の指示が、デフォルトの表示方式の指示でないと判定された場合、即ち、新たな表示方式の指示が、例えば、字幕データを拡大や縮小して表示する、あるいは明るさを変えて見やすくする等、デフォルトでない表示方式の指示である場合、ステップＳ３４６に進み、プレイヤ制御モジュール２１２は、現在再生している字幕ストリームが多重化されたクリップストリームファイルに対応するクリップ情報ファイルClip()（図１２）のStaticInfo()（図１４）のうちの、現在再生している字幕ストリームについてのStaticInfo()を取得し、ステップＳ３４７に進む。

【０７７２】
ステップＳ３４７では、プレイヤ制御モジュール２１２は、ステップＳ３４６で取得したStaticInfo()のconfigurable_flagを判定する。

【０７７３】
ステップＳ３４７において、configurable_flagが、字幕データの表示方式の変更を許可しない旨の0になっていると判定された場合、ステップＳ３４８に進み、プレイヤ制御モジュール２１２は、グラフィクス処理モジュール２１９を制御することにより、出力ビデオデータに、字幕データの表示方式を変更することができない旨のエラーメッセージをオーバーレイさせ、ステップＳ３４２に戻る。これにより、エラーメッセージが表示される。

【０７７４】
一方、ステップＳ３４７において、configurable_flagが、字幕データの表示方式の変更を許可する旨の1になっていると判定された場合、ステップＳ３４９に進み、プレイヤ制御モジュール２１２は、ユーザがリモコンを操作することにより入力インターフェース１１５から供給された新たな表示方式の指示を、グラフィクス処理モジュール２１９に供給して、ステップＳ３５０に進む。

【０７７５】
ステップＳ３５０では、グラフィクス処理モジュール２１９は、字幕デコーダ制御モジュール２１８から供給される字幕データを、直前のステップＳ３４９でプレイヤ制御モジュール２１２から供給された表示方式の指示にしたがって拡大または縮小等あるいは明るさを変える等の処理を開始し、ステップＳ３４２に戻る。これにより、字幕データは、ユーザがリモコンを操作することによって指示した表示方式にしたがった表示サイズや、表示位置、表示色等で表示される。
【０７７６】
一方、ステップＳ３４２において、新たな表示方式の指示がなかったと判定された場合、ステップＳ３５１に進み、プレイヤ制御モジュール２１２は、図４０で説明したPlayItem()の乗り換えが行われたかどうかを判定し、行われていないと判定した場合、ステップＳ３４２に戻る。
【０７７７】
また、ステップＳ３５１において、PlayItem()の乗り換えが行われたと判定された場合、ステップＳ３４１に戻り、上述したように、プレイヤ制御モジュール２１２は、字幕データの表示方式をデフォルトの表示方式とするように、グラフィクス処理モジュール２１９を制御する。即ち、この場合、PlayItem()の乗り換えが行われたときには、字幕データの表示方式は、デフォルトの表示方式に戻される。
【０７７８】
以上のように、字幕表示制御処理においては、字幕ストリームのconfigurable_flagが、表示方式の変更を許可する旨の1になっている場合にのみ、その字幕ストリームに対応する字幕データの表示方式が、例えば、ユーザがリモコンを操作することにより入力される表示方式の指示に応じて変更される。
【０７７９】
従って、例えば、図３０に示したクリップ情報ファイル”00001.CLP”によれば、対応するクリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームのうちの３本目のエレメンタリストリームである字幕ストリームstream#2についてのconfigurable_flagは、表示方式の変更を許可しない旨の0になっているので、その字幕ストリームstream#2が表示されているときに、ユーザが字幕の表示を変更するようにリモコンを操作しても、その表示は変更されない。
【０７８０】
一方、例えば、クリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームのうちの４本目のエレメンタリストリームである字幕ストリームstream#3についてのconfigurable_flagは、表示方式の変更を許可する旨の1になっているので、その字幕ストリームstream#3が表示されているときに、ユーザが字幕の表示を変更するようにリモコンを操作すると、その操作に応じて、字幕の表示サイズ等が変更される。
【０７８１】
即ち、例えば、いま、図２９の１番目のPlayList#0の１番目のPlayItem#0にしたがい、クリップストリームファイル”00001.PS”が再生されているとする。また、図３０でクリップ情報ファイル”00001.CLP”について説明したように、クリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームのうちの、３本目と４本目が字幕ストリームであるが、その３本目の字幕ストリームstream#2と、４本目の字幕ストリームstream#3のうちの、例えば、３本目の字幕ストリームstream#2が、現在再生されているとする。
【０７８２】
ユーザが、リモコンを操作することにより、字幕の表示方式の指示を入力すると（ステップＳ３４２）、その表示方式の指示は、入力インターフェース１１５（図１）からプレイヤ制御モジュール２１２に供給される。プレイヤ制御モジュール２１２は、表示方式の指示が供給されると、再生中の字幕ストリームに対応するStaticInfo()（図１２）を、クリップ情報ファイルから探し出す（ステップＳ３４６）。

【０７８３】
即ち、いまの場合、再生中の字幕ストリームは、クリップストリームファイル”00001.PS”に多重化されている３本目の字幕ストリームstream#2であり、プレイヤ制御モジュール２１２は、対応するクリップ情報ファイル”00001.CLP”から、３本目の字幕ストリームstream#2についてのStaticInfo()を探し出す。
【０７８４】
さらに、プレイヤ制御モジュール２１２は、図３０において３本目の字幕ストリームstream#2についてのStaticInfo()に記述されている、0になっているconfigurable_flagを判定し（ステップＳ３４７）、これにより、３本目の字幕ストリームstream#2については、表示方式の変更が許可されていないことを認識する。
【０７８５】
この場合、プレイヤ制御モジュール２１２は、再生中の字幕ストリーム（に対応する字幕データ）が拡大縮小等に対応していないと判断し、グラフィクス処理モジュール２１９を制御することにより、その旨のエラーメッセージを生成させ（ステップＳ３４８）、ビデオデータにオーバーレイして出力させる。
【０７８６】
一方、クリップストリームファイル”00001.PS”に多重化されている４本のエレメンタリストリームの３本目の字幕ストリームstream#2と、４本目の字幕ストリームstream#3のうちの、３本目の字幕ストリームstream#2ではなく、４本目の字幕ストリームstream#3が、現在再生されている場合には、ユーザがリモコンを操作することによって表示方式の指示の供給を受けたプレイヤ制御モジュール２１２は、対応するクリップ情報ファイル”00001.CLP”から、４本目の字幕ストリームstream#3についてのStaticInfo()を探し出す。
【０７８７】
さらに、プレイヤ制御モジュール２１２は、図３０において４本目の字幕ストリームstream#3についてのStaticInfo()に記述されている、1になっているconfigurable_flagを判定し（ステップＳ３４７）、これにより、４本目の字幕ストリームstream#3については、表示方式の変更が許可されていることを認識する。
【０７８８】
この場合、プレイヤ制御モジュール２１２は、再生中の字幕ストリーム（に対応する字幕データ）が拡大縮小等に対応していると判断し、ユーザがリモコンを操作することによって供給された表示方式の指示を、グラフィクス処理モジュール２１９に供給する（ステップＳ３４９）。
【０７８９】
これにより、その後、グラフィックス処理制御モジュール２１９は、プレイヤ制御モジュール２１２からの表示方式の指示にしたがい、字幕デコーダ制御モジュール２１８からの字幕データを拡大または縮小等し、ビデオデコーダ制御モジュール２１６からのビデオデータにオーバーレイして出力する。
【０７９０】
なお、プレイヤ制御モジュール２１２は、PlayList()の最初のPlayItem()の再生開始時に、グラフィクス処理モジュール２１９に対する字幕データの表示方式の指示を初期化する（ステップＳ３４１）。即ち、プレイヤ制御モジュール２１２は、字幕データの表示方式をデフォルトの表示方式とするように、グラフィクス処理モジュール２１９を制御する。
【０７９１】
さらに、プレイヤ制御モジュール２１２は、PlayItem()の乗り換え時にも、グラフィクス処理モジュール２１９に対する字幕データの表示方式の指示を初期化する（ステップＳ３４１，Ｓ３５１）。
【０７９２】
但し、PlayItem()の乗り換え時においては、その後に新たに再生されるPlayItem()にしたがって再生される新たな字幕ストリームについてのconfigurable_flagを調査し、confi

gurable_flagが0である場合には、グラフィクス処理モジュール２１９に対する字幕データの表示方式の指示を初期化し、configurable_flagが1である場合には、グラフィクス処理モジュール２１９に対する表示方式の指示を、PlayItem()の乗り換え前のまま維持するようにすることが可能である。

【０７９３】

また、図５６の字幕表示制御処理では、ユーザがリモコンを操作することにより、新たな表示方式の指示が入力された場合に、その新たな表示方式の指示を、グラフィクス処理モジュール２１９に供給するようにしたが（ステップＳ３４９）、表示方式の指示は、例えば、メモリ１１３（図１）を構成する不揮発性メモリに記憶し、その不揮発性メモリに記憶された表示方式の指示を、グラフィクス処理モジュール２１９に供給するようにすることが可能である。

【０７９４】

即ち、例えば、図１のディスク再生装置の初期設定として、不揮発性メモリに、ユーザ設定の表示方式の指示を記憶させておき、ユーザがリモコンを操作することにより、新たな表示方式の指示が入力された場合には、不揮発性メモリに記憶された表示方式の指示を、新たな表示方式の指示に更新する一方、その不揮発性メモリに記憶された表示方式の指示を、グラフィクス処理モジュール２１９に供給するようにすることが可能である。この場合、不揮発性メモリには、前回の再生終了時における表示方式の指示が保持されるので、次回のPlayList()の再生時に、ユーザがリモコンを操作することにより、前回の再生終了時における表示方式の指示を入力しなくても、その表示方式で、字幕データの表示が開始される。

【０７９５】

なお、この場合、不揮発性メモリに記憶させる表示方式の指示には、例えば、字幕データを拡大または縮小するときの拡大率または縮小率等が含まれるものとする。

【０７９６】

以上のように、字幕表示制御処理によれば、クリップ情報ファイルClip()（図１２）に含まれる、エレメンタリストリームごとの、そのエレメンタリストリームの再生中に変化しないStaticInfo()のうちの、字幕データのStaticInfo()が取得され、そのStaticInfo()に含まれる、字幕データの表示をデフォルトの表示方式から変更することを許可するか否かを表すconfigurable_flagに基づき、再生中の字幕データの表示をデフォルトの表示方式から変更することが許可されているか否かが判定される。そして、再生中の字幕データの表示をデフォルトの表示方式から変更することが許可されている場合には、字幕データの表示方式の変更の指示にしたがって、その字幕データの表示処理、即ち、例えば、字幕データを拡大または縮小、あるいは表示色を変更する等して表示する処理が行われる。従って、字幕データの表示方式の変更を制御することができる。

【０７９７】

［キャプチャ制御処理］

次に、図５７のフローチャートを参照して、ビデオストリームに対応するビデオデータのキャプチャを制御するキャプチャ制御処理について説明する。なお、図５７には、キャプチャ制御処理を説明するフローチャートとともに、そのキャプチャ制御処理によってキャプチャされたビデオデータを２次利用する処理の例であるバックグラウンド／スクリーンセーバ処理を説明するフローチャートも、図示してある。

【０７９８】

キャプチャ制御処理は、例えば、ユーザがリモコンを操作することにより、ビデオデータのキャプチャを指示するキャプチャ指示が、入力インターフェース１１５（図１）を介して、プレイヤ制御モジュール２１２に供給されると開始される。

【０７９９】

即ち、キャプチャ制御処理では、まず最初に、ステップＳ３７１において、プレイヤ制御モジュール２１２が、ビデオストリームを再生中であるかどうかを判定し、再生中でないと判定した場合、キャプチャ制御処理は終了する。

【０８００】
一方、ステップＳ３７１において、ビデオストリームを再生中であると判定された場合、ステップＳ３７２に進み、プレイヤ制御モジュール２１２は、再生中のビデオストリームに対応するPlayList()（図７）から、capture_enable_flag_PlayListを取得するとともに、再生中のビデオストリームに対応するクリップ情報ファイルClip()（図１２）から、capture_enable_flag_Clipを取得する。
【０８０１】
ここで、PlayList()におけるcapture_enable_flag_PlayListは、図７で説明したように、そのPlayList()によって再生されるビデオストリームに対応するビデオデータ（PlayList()に属するビデオデータ）の２次利用を許可するか否かを表す。また、クリップ情報ファイルClip()におけるcapture_enable_flag_Clipは、図１２で説明したように、そのクリップ情報ファイルClip()に対応するクリップストリームファイルに格納されているビデオストリームに対応するビデオデータの２次利用を許可するか否かを表す。
【０８０２】
ステップＳ３７２の処理後は、ステップＳ３７３に進み、プレイヤ制御モジュール２１２は、直前のステップＳ３７３で取得されたcapture_enable_flag_PlayListとcapture_enable_flag_Clipとに基づき、キャプチャ指示が入力インターフェース１１５（図１）から入力されたときに再生されていたビデオデータのピクチャのキャプチャの可否を判定する。
【０８０３】
ステップＳ３７３において、キャプチャ指示が入力インターフェース１１５から入力されたときに再生されていたビデオデータのピクチャのキャプチャが不可であると判定された場合、即ち、直前のステップＳ３７３で取得されたcapture_enable_flag_PlayListまたはcapture_enable_flag_Clipのうちの少なくとも一方が、ビデオデータの２次利用を許可しない旨の0になっている場合、ステップＳ３７４に進み、プレイヤ制御モジュール２１２は、グラフィクス処理モジュール２１９を制御することにより、ビデオデータのキャプチャが不可である旨のエラーメッセージをオーバーレイさせ、キャプチャ制御処理を終了する。これにより、エラーメッセージが表示される。
【０８０４】
一方、ステップＳ３７３において、キャプチャ指示が入力インターフェース１１５から入力されたときに再生されていたビデオデータのピクチャのキャプチャが可能であると判定された場合、即ち、直前のステップＳ３７３で取得されたcapture_enable_flag_PlayListおよびcapture_enable_flag_Clipの両方が、ビデオデータの２次利用を許可する旨の1になっている場合、ステップＳ３７５に進み、プレイヤ制御モジュール２１２は、キャプチャ指示が入力インターフェース１１５から入力されたときに再生されていたビデオデータのピクチャのキャプチャの指示を、グラフィクス処理モジュール２１９に供給し、ステップＳ３７６に進む。
【０８０５】
ステップＳ３７６では、グラフィクス処理モジュール２１９は、プレイヤ制御モジュール２１２からのキャプチャの指示にしたがい、ビデオデコーダ制御モジュール２１６からのビデオデータのピクチャをキャプチャし、メモリ１１３（図１）に記憶させて、キャプチャ制御処理を終了する。なお、capture_enable_flagが複数ビット構成になっており、使用条件の制約が行われている場合にはこの時点で対応が行われる。すなわちキャプチャした画像の大きさに制限がある場合には、この時点で縮小した画像がキャプチャされる。また使用するアプリケーションに制約がある場合にはその旨を知らせるフラグが同時に記録される。
【０８０６】
以上のように、キャプチャ制御処理では、ユーザからのキャプチャ指示があったときに再生されているビデオストリームに対応するPlayList()（図７）とクリップ情報ファイルClip()（図１２）それぞれのcapture_enable_flag_PlayListとcapture_enable_flag_Clip

との論理積をとって、その論理積が1である場合、即ち、capture_enable_flag_PlayListとcapture_enable_flag_Clipが、いずれも、２次利用を許可する1になっている場合にのみ、ビデオデータの２次利用が可能であると判断され、キャプチャが行われる。

【０８０７】

従って、例えば、図２９における１番目のPlayList#0の１番目のPlayItem#0にしたがって、ビデオストリームの再生、即ち、クリップストリームファイル”00001.PS”に多重化されたビデオストリームの再生が行われている場合に、ユーザからのキャプチャ指示があったときには、１番目のPlayList#0におけるcapture_enable_flag_PlayListは1であり、その１番目のPlayItem#0によって再生されるクリップストリームファイル”00001.PS”に対応する図３０のクリップ情報ファイル”00001.CLP”におけるcapture_enable_flag_Clipは1であるから、再生中のビデオデータ（クリップストリームファイル”00001.PS”に多重化されたビデオストリームに対応するビデオデータ）の２次利用は可能であると判断され、キャプチャが行われる。

【０８０８】

また、例えば、図２９における１番目のPlayList#0の２番目のPlayItem#1にしたがって、ビデオストリームの再生、即ち、クリップストリームファイル”00002.PS”に多重化されたビデオストリームの再生が行われている場合に、ユーザからのキャプチャ指示があったときには、１番目のPlayList#0におけるcapture_enable_flag_PlayListは1であり、その２番目のPlayItem#1によって再生されるクリップストリームファイル”00002.PS”に対応する図３０のクリップ情報ファイル”00002.CLP”におけるcapture_enable_flag_Clipは0であるから、再生中のビデオデータ（クリップストリームファイル”00002.PS”に多重化されたビデオストリームに対応するビデオデータ）の２次利用は不可であると判断され、キャプチャが行われない。

【０８０９】

さらに、例えば、図２９における２番目のPlayList#1のPlayItem#0にしたがって、ビデオストリームの再生、即ち、クリップストリームファイル”00003.PS”に多重化されたビデオストリームの再生が行われている場合に、ユーザからのキャプチャ指示があったときには、２番目のPlayList#1におけるcapture_enable_flag_PlayListは0であり、２番目のPlayList#1のPlayItem#0によって再生されるクリップストリームファイル”00003.PS”に対応する図３０のクリップ情報ファイル”00003.CLP”におけるcapture_enable_flag_Clipは1であるから、再生中のビデオデータ（クリップストリームファイル”00003.PS”に多重化されたビデオストリームに対応するビデオデータ）の２次利用は不可であると判断され、キャプチャは行われない。

【０８１０】

なお、この場合、２番目のPlayList#1におけるcapture_enable_flag_PlayListが0であることが確認された時点で、ビデオデータの２次利用は不可であると判断することができるので、２番目のPlayList#1のPlayItem#0によって再生されるクリップストリームファイル”00003.PS”に対応する図３０のクリップ情報ファイル”00003.CLP”におけるcapture_enable_flag_Clipの確認は省略することができる。

【０８１１】

キャプチャ制御処理によってキャプチャされ、メモリ１１３に記憶されたピクチャは、バックグラウンド／スクリーンセーバ処理において２次利用することができる。

【０８１２】

バックグラウンド／スクリーンセーバ処理は、例えば、プレイヤ制御モジュール２１２が動作しているが、エレメンタリストリームの再生が行われていない状態、即ち、ディスクドライブ１０２（図１）にディスク１０１が挿入されていない状態、あるいはエレメンタリストリームの再生が終了した状態となったときなどに行われる。

【０８１３】

即ち、バックグラウンド／スクリーンセーバ処理では、ステップＳ３８０において、プレイヤ制御モジュール２１２は、キャプチャ制御処理によってメモリ１１３に記憶された

ピクチャを表示するように、グラフィクス処理モジュール２１９を制御する。グラフィクス処理モジュール２１９は、プレイヤ制御モジュール２１２からの制御にしたがい、キャプチャ制御処理によってメモリ１１３に記憶されたピクチャを表示させる。

【０８１４】

ここで、グラフィクス処理モジュール２１９において、メモリ１１３に記憶されたピクチャを静止画で表示させれば、いわゆる壁紙(バックグラウンド)が実現され、一定周期で拡大や縮小、移動等しながら表示させれば、スクリーンセーバーが実現される。また、キャプチャ制御処理によってメモリ１１３に記憶されたピクチャの表示を行うバックグラウンド／スクリーンセーバ処理は、プレイヤ制御モジュール２１２ではなく、他の独立したアプリケーションによって行うことが可能である。

【０８１５】

また、このときメモリ１１３に記憶されたピクチャに使用制限を表すフラグが付加されている場合にはその制限に従う。

【０８１６】

以上のように、ビデオアクセスユニット単位より大きな単位の、例えば、PlayList()やPlayItem()に対応するビデオデータの２次利用を許可するか否かを表す、再生中のビデオデータに対するcapture_enable_flag_PlayListやcapture_enable_flag_Clipが取得され、そのcapture_enable_flag_PlayListやcapture_enable_flag_Clipに基づき、再生中のビデオデータの２次利用が許可されているか否かが判定される。そして、再生中のビデオデータの２次利用が許可されていると判定された場合、再生中のビデオデータがキャプチャされ、そのキャプチャされたビデオデータを利用したバックグラウンド／スクリーンセーバ処理が実行される。従って、ビデオデータの２次利用の制御が可能となる。

【０８１７】

なお、図５７のキャプチャ制御処理では、PlayList()（図７）において、capture_enable_flag_PlayListを設けるとともに、PlayItem()によって再生されるクリップストリームファイルに対応するクリップ情報ファイルClip()（図１２）において、capture_enable_flag_Clipを設け、そのcapture_enable_flag_PlayListとcapture_enable_flag_Clipとの両方を用いて、２次利用の許可（可否）を判定するようにしたが、PlayList()（図７）において、capture_enable_flag_PlayListを設けるだけか、または、PlayItem()によって再生されるクリップストリームファイルに対応するクリップ情報ファイルClip()（図１２）において、capture_enable_flag_Clipを設けるだけにして、capture_enable_flag_PlayListまたはcapture_enable_flag_Clipの一方だけを用いて、２次利用の可否を判定するようにすることも可能である。

【０８１８】

また、図５７のキャプチャ制御処理では、ステップＳ３７６において、グラフィクス処理モジュール２１９が、プレイヤ制御モジュール２１２からのキャプチャの指示にしたがい、ビデオデコーダ制御モジュール２１６からのビデオデータのピクチャ、即ち、１つのピクチャだけをキャプチャするようにしたが、その他、複数のピクチャをキャプチャすることも可能である。つまり、ビデオデコーダ制御モジュール２１６が出力する時系列の複数のピクチャ（動画としての複数のピクチャのシーケンス）をキャプチャすることが可能である。この場合、一度にキャプチャされるピクチャの枚数は、例えば、あらかじめ決めておくことができる。あるいは、capture_enable_flag_PlayListやcapture_enable_flag_Clipのビットを拡張して、そのcapture_enable_flag_PlayListやcapture_enable_flag_Clipに、一度にキャプチャ可能なピクチャの枚数の情報を含めるようにしても良い。

【０８１９】

さらに、上述の場合には、ビデオデータの２次利用を許可するか否かの利用許可情報(capture_enable_flag_PlayList, capture_enable_flag_Clip)を、PlayList()や、クリップ情報ファイルClip()に記述し、その利用許可情報によって、PlayList()によって再生されるビデオデータ全体や、クリップ情報ファイルClip()に対応するクリップストリームファイルに多重化されたビデオストリームに対応するビデオデータ全体についての２次利用の

可否を判定するようにしたが、利用許可情報は、その他の任意の単位のビデオデータについて記述し、その利用許可情報によって、任意の単位のビデデータについての２次利用の可否を判定することが可能である。

【０８２０】

即ち、図５８は、利用許可情報が配置されたprivate_stream2_PES_payload()のシンタクスを示しており、図５９は、利用許可情報が配置されたau_information()のシンタクスを示している。

【０８２１】

なお、図５８のprivate_stream2_PES_payload()は、video_stream_idの直前に、利用許可情報としてのcapture_enable_flag_ps2が配置されている他は、図２６における場合と同様に構成されている。図５９のau_information()も、pic_struct_copyの直前に、利用許可情報としてのcapture_enable_flag_AUが配置されている他は、図２７における場合と同様に構成されている。

【０８２２】

図５８のprivate_stream2_PES_payload()に配置されたcapture_enable_flag_ps2は、そのprivate_stream2_PES_payload()を含むprivate_stream_2のPES_packet()から、次のprivate_stream_2のPES_packet()の直前までに配置されるビデオストリームに対応するビデオデータの２次利用を許可するか否かを表す。従って、図５８のprivate_stream2_PES_payload()に配置されたcapture_enable_flag_ps2によれば、あるデコード開始可能点から次のデコード開始可能点までの間のビデオデータについて、その２次利用を許可するか否かを判定することができる。

【０８２３】

また、図５９のau_information()に配置されたcapture_enable_flag_AUは、そのcapture_enable_flag_AUに対応するビデオアクセスユニットのビデオデータの２次利用を許可するか否かを表す。従って、図５９のau_information()に配置されたcapture_enable_flag_AUによれば、ビデオアクセスユニット単位のビデオデータについて、即ち、ピクチャ単位で、その２次利用を許可するか否かを判定することができる。

【０８２４】

ここで、PlayList()（図７）における利用許可情報としてのcapture_enable_flag_PlayList、クリップ情報ファイルClip()（図１２）における利用許可情報としてのcapture_enable_flag_Clip、private_stream2_PES_payload()（図５８）における利用許可情報としてのcapture_enable_flag_ps2, au_information()（図５９）における利用許可情報としてのcapture_enable_flag_AUは、そのうちの２以上を重複して採用することが可能であり、この場合、あるビデオデータのピクチャの２次利用の可否は、その重複して採用される２以上の利用許可情報の論理積等に基づいて判定することができる。

【０８２５】

また、図５９のau_information()が配置される図２６または図５８のprivate_stream2_PES_payload()を含むprivate_stream_2のPES_packet()は、図４５のステップＳ２１１で説明したように、バッファ制御モジュール２１５（図５）のビデオ読み出し機能部２３３が、バッファ２１５Ａに記憶されたプログラムストリーム中から探索する。従って、capture_enable_flag_ps2が配置された図５８のprivate_stream2_PES_payload()や、capture_enable_flag_AUが配置された図５９のau_information()を採用する場合には、プレイヤ制御モジュール２１２は、ビデオデータの２次利用の可否を判定するにあたって、capture_enable_flag_ps2やcapture_enable_flag_AUを、ビデオ読み出し機能部２３３に問い合わせる必要がある。

【０８２６】

次に、図６０を参照して、ディスク記録装置のハードウェアの構成について説明する。

【０８２７】

図６０のディスク記録装置は、例えば、ディスクプレーヤや、ゲーム装置、カーナビゲーションシステムその他に適用することができる。

【０８２８】
図６０のディスク記録装置において、ディスク４１０は、例えば、DVDなどの光ディスク、あるいは光磁気ディスク、磁気ディスクなどであり、ビデオデータや、オーディオデータ、字幕データなどのコンテンツデータ、さらには、コンテンツデータを記録することができる。尚、ディスク４１０は、各種のデータが記録されることにより、図１のディスク１０１として使用することが可能となる。
【０８２９】
ビデオ入力端子４００Ａは、図示せぬ撮像装置などのビデオ入力装置に接続されており、ビデオ入力装置より供給されるビデオデータをビデオ入力インタフェース４０１に供給する。オーディオ入力端子４００Ｂは、図示せぬマイクロフォンやアンプ（アンプリファイヤ）などのオーディオ入力装置に接続されており、入力されたオーディオデータを、オーディオ入力インタフェース４０２に供給する。
【０８３０】
ビデオ入力インターフェース４０１は、入力されるビデオデータに必要な処理を施し、バス４１１を介してビデオエンコーダ４０３に供給する。オーディオ入力インターフェース４０２は、入力されるオーディオデータに所定の処理を施し、バス４１１を介してオーディオエンコーダ４０４に供給する。
【０８３１】
ビデオエンコーダ４０３は、CPU４０５やビデオ入力インタフェース４０１より供給されるビデオデータをエンコードし、その結果得られる、圧縮符号化データ（符号化ビデオデータ：例えば、MPEG２ビデオストリーム）を、バス４１１を介してディスクドライブ４０９によりディスク４１０に記録させる。
【０８３２】
オーディオエンコーダ４０４は、CPU４０５やオーディオ入力インタフェース４０２より供給されるオーディオデータをエンコードし、その結果得られる、圧縮符号化データ（符号化オーディオデータ：例えば、MPEG２オーディオストリーム）を、バス４１１を介してディスクドライブ４０９によりディスク４１０に記録させる。
【０８３３】
CPU４０５およびメモリ４０６は、コンピュータシステムを形成している。即ち、CPU４０５は、メモリ４０６に記憶されたプログラムを実行し、ディスク記録装置全体を制御するとともに、後述する各種の処理を行う。メモリ４０６は、CPU４０５が実行するプログラムを記憶している。また、メモリ４０６は、CPU４０５の動作上必要なデータを一時記憶する。なお、メモリ４０６は、不揮発性メモリのみ、または揮発性メモリと不揮発性メモリとの組み合わせで構成することが可能である。また、図６０のディスク記録装置に、ハードディスクを設け、そのハードディスクに、CPU４０５が実行するプログラムを記録（インストール）しておく場合には、メモリ４０６は、揮発性メモリのみで構成すること

【０８３４】
ここで、CPU４０５が実行するプログラムは、ディスク記録装置に内蔵されている記録媒体としてのメモリ４０６に予め記録しておく（記憶させておく）ことができる。
【０８３５】
あるいはまた、プログラムは、ディスク４０９、さらには、ディスク４０９以外のフレキシブルディスク、CD-ROM(Compact Disc Read Only Memory), MO(Magneto Optical)ディスク、磁気ディスク、メモリカードなどのリムーバブル記録媒体に、一時的あるいは永続的に格納（記録）しておくことができる。このようなリムーバブル記録媒体は、いわゆるパッケージソフトウエアとして提供することができる。
【０８３６】
なお、プログラムは、メモリ４０６にあらかじめ記憶させておくこと、あるいは、上述したようなリムーバブル記録媒体からディスク記録装置にインストールすることができる。また、プログラムは、ダウンロードサイトから、ディジタル衛星放送用の人工衛星を介

して、ディスク再生装置に無線で転送したり、LAN(Local Area Network)、インターネットといったネットワークを介して、ディスク再生装置に有線で転送し、ディスク再生装置では、そのようにして転送されてくるプログラムを、入力インターフェース４０８で受信し、内蔵するメモリ４０６にインストールすることができる。

【０８３７】

さらに、プログラムは、１のCPUにより処理されるものであっても良いし、複数のCPUによって分散処理されるものであっても良い。

【０８３８】

ドライブインターフェース４０７は、CPU４０５の制御の下、ディスクドライブ４０９を制御し、これにより、バス４１１を介して、CPU４０５や、メモリ４０６、ビデオエンコーダ４０３、オーディオエンコーダ４０４より供給されるデータをディスクドライブ４０９に対して供給し、ディスク４１０に記録させる、または、ディスク４１０に記録されたデータを読み出し、バス４１１を介して、CPU４０５や、メモリ４０６に供給する。

【０８３９】

入力インターフェース４０８は、図示せぬキー（ボタン）や、リモコン（リモートコントローラ）がユーザに操作されることによって供給される信号を受信し、バス４１１を介して、CPU４０５に供給する。なお、入力インターフェース４０８は、その他、例えば、モデム（ADSL(Asymmetric Digital Subscriber Line)モデムを含む）や、NIC(Network Interface Card)などの通信インターフェースとしても機能する。

【０８４０】

なお、ビデオ入力装置とオーディオ入力装置からのディスク記録装置へのビデオデータとオーディオデータの供給は、有線または無線のいずれによって行うことも可能である。

【０８４１】

ディスクドライブ４０９には、ディスク４１０が着脱可能になっている。ディスクドライブ４０９は、図示せぬインターフェースを内蔵し、そのインターフェースを通じて、ドライブインターフェース４０７に接続されている。ディスクドライブ４０９は、そこに装着されたディスク４１０を駆動し、ドライブインターフェース４０７からの記録等の命令にしたがって、ディスク４１０にデータを記録する等の処理を行う。

【０８４２】

なお、ディスク４１０に記録されるデータ（記録データ）には、必要に応じて、コンピュータが実行可能なプログラムも含まれる。また、本実施の形態では、記録媒体として、ディスク状の記録媒体であるディスク４１０を採用するが、その他、記録媒体としては、例えば、半導体メモリや、テープ状の記録媒体であってもよい。

【０８４３】

バス４１１には、CPU(Central Processing Unit)４０５、メモリ４０６、ドライブインターフェース４０７、入力インターフェース４０８、ビデオエンコーダ４０３、オーディオエンコーダ４０４、ビデオ入力インターフェース４０１、オーディオ入力インターフェース４０２が接続されている。

【０８４４】

次に、図６１を参照して、本発明のデータ符号化方法を具現化したディスク記録装置により実現される機能について説明する。この図に示すディスク記録装置により実現される機能において、オーディオエンコーダ４０４はオーディオ入力端子４００Ｂおよびオーディオ入力インタフェース４０２を介して入力されたオーディオ信号を圧縮符号化して多重化器４２１に出力する。

【０８４５】

また、ビデオエンコーダ４０３はビデオ入力端子４００Ａおよびビデオ入力インタフェース４０１を介して入力されたビデオ信号を圧縮符号化して多重化器４２１に出力する。

【０８４６】

多重化器４２１は、入力されたMPEG２ビデオストリームとMPEG２オーディオストリームをパケット化して、上述した図１８乃至図２７を用いて説明したように時分割多重化（パ

ケタイズ）している。多重化器４２１は、ストリーム中のイントラピクチャを選び、およそ1秒に2回程度の頻度で図２６に示すprivate_stream_2のPES_packet()を挿入する。

【０８４７】

多重化器４２１は、多重化ストリームを、FIFO４２２を経由してRAPI書換器４２４に出力するとともに、RAPI情報抽出器４２３にも同時に出力する。RAPI情報抽出器４２３では、多重化ストリームに多重化されたビデオストリームのなかのprivate_stream_2のPES_packet()の開始位置と、private_stream_2のPES_packet()直後に置かれているイントラピクチャの持つタイムスタンプ（PTS）値、及び該当イントラピクチャと、それにつづく2枚目、3枚目、4枚目の参照画像の終了位置を検出し、記憶する。

【０８４８】

なお、ここでいうRAPIとは、private_stream_2のPES_packet()のことである。

【０８４９】

RAPI情報抽出器４２３は検出した該当イントラピクチャと、それにつづく2枚目、3枚目、4枚目の参照画像の終了位置をRAPI書換器４２４に出力する。RAPI書換器４２４では、RAPIの情報として図２６の1st_Ref_picture、2ndRef_picture、3rdRef_picture、4thRef_pictureのフィールドが上書きされ、それぞれ先頭、2枚目、3枚目、4枚目の参照画像の終了位置が、セクタ単位の数値で記録され、出力サーバ４２６に蓄積される。

【０８５０】

コントローラ４２５は、多重化ストリーム全体の処理が終了すると、RAPI情報抽出器４２３で抽出、蓄積された、多重化ストリームに多重化されたすべてのRAPIの開始位置と、RAPI直後に置かれているイントラピクチャ及びそれにつづく2枚目、3枚目、4枚目の参照画像の終了位置を取得する。

【０８５１】

コントローラ４２５は、入力された情報を使用して、図１６を用いて説明したEP_map()を作成する。

【０８５２】

コントローラ４２５は、RAPI情報抽出器４２３から入力されたデータから、RAPIに対して、そのアドレス、直後のイントラピクチャの持つPTS、そしてイントラピクチャ及びそれにつづく2枚目、3枚目、4枚目の参照画像の終了位置のうちの１を選び出して、クリップ情報ファイル内のEP_map()を作成し、出力サーバ４２６に蓄積する。

【０８５３】

ここで、図６２のフローチャートを参照して、EP_map（）生成処理について説明する。

【０８５４】

ステップＳ３８１において、ビデオエンコーダ４０３はビデオ入力端子４００Ａおよびビデオ入力インタフェース４０１を介して入力されたビデオ信号を圧縮符号化して多重化器４２１に出力する。また、オーディオエンコーダ４０４はオーディオ入力端子４００Ｂおよびオーディオ入力インタフェース４０２を介して入力されたオーディオ信号を圧縮符号化して多重化器４２１に出力する。この場合、オーディオエンコーダ４０４から出力されるストリームは、MPEG２オーディオストリーム（オーディオレイヤ）とされ、ビデオエンコーダ４０３から出力されるストリームは、MPEG２ビデオストリーム（ビデオレイヤ）とされる。

【０８５５】

ステップＳ３８２において、多重化器４２１は、入力されたMPEG２ビデオストリームとMPEG２オーディオストリームをパケット化して、図１８乃至図２７を用いて説明したように時分割多重化（パケタイズ）し、ストリーム中のイントラピクチャを選び、およそ1秒に2回程度の頻度で図２６に示すprivate_stream_2のPES_packet()を挿入する。ここでprivate_stream_2のPES_packet()は、ビデオのイントラピクチャ（デコード開始可能ピクチャ）が直後に置かれていることを示している。このとき、該当イントラピクチャにはタイムスタンプ（PTS/DTS）が必ず付与される。

【０８５６】

なお、この段階では、図２６で説明した1st_Ref_picture、2ndRef_picture、3rdRef_picture、4thRef_pictureのフィールドにはデータが記録されていない。さらに、図示されていないが、サブタイトルストリームを多重化器４２１に入力して、ビデオストリーム、オーディオストリームと共に多重化してもよい。

【０８５７】

ステップＳ３８３において、多重化器４２１は、多重化ストリームを、FIFO４２２を経由してRAPI書換器４２４に出力するとともに、RAPI情報抽出器４２３にも同時に出力する。RAPI情報抽出器４２３では、多重化ストリームに多重化されたビデオストリームのなかのprivate_stream_2のPES_packet()の開始位置と、private_stream_2のPES_packet()直後に置かれているイントラピクチャの持つタイムスタンプ（PTS）値、及び該当イントラピクチャと、それにつづく2枚目、3枚目、4枚目の参照画像の終了位置を検出し、記憶する。

【０８５８】

さらに、RAPI情報抽出器４２３は検出した該当イントラピクチャと、それにつづく2枚目、3枚目、4枚目の参照画像の終了位置をRAPI書換器４２４に出力する。RAPI書換器４２４は、RAPIの情報、うち図２６の1st_Ref_picture、2ndRef_picture、3rdRef_picture、4thRef_pictureのフィールドを上書きし、それぞれ先頭、2枚目、3枚目、4枚目の参照画像の終了位置を、セクタ単位の数値で記録し、さらに出力サーバ４２６に蓄積させる。

【０８５９】

ステップＳ３８５において、RAPI情報抽出器４２３で抽出、蓄積された、多重化ストリームに多重化されたすべてのRAPIの開始位置と、RAPI直後に置かれているイントラピクチャ及びそれにつづく2枚目、3枚目、4枚目の参照画像の終了位置が、コントローラ４２５に入力される。

【０８６０】

コントローラ４２５は、入力された情報を使用して、図１６を用いて説明したEP_map()を作成する。ここで作成するEP_map()はビデオストリームのみの情報が含まれているとする。ビデオに関するEP_map()は、ストリーム中に置かれた全てのRAPI、すなわちprivate_stream_2のPES_packet()の位置であり、この情報はRAPI情報抽出器４２３からコントローラ４２５に入力された情報で作成される。

【０８６１】

より詳細には、コントローラ４２５は、RAPI情報抽出器４２３から入力されたデータから、RAPIに対して、そのアドレス、直後のイントラピクチャの持つPTS、そしてイントラピクチャ及びそれにつづく2枚目、3枚目、4枚目の参照画像の終了位置のうちの１を選び出して、クリップ情報ファイル内のEP_map()を作成し、出力サーバ４２６に蓄積する。すなわち、コントローラ４２５は、４点の参照画像終了位置（1stRef_Picture、2ndRef_Picture、3rdRef_Picture、4thRef_Picture）のうち、所定のセクタ数（エンコード処理において、まとめて読出可能なセクタ数）に近い値を、N-th_Ref_picture_copyにコピーする。

【０８６２】

ステップＳ３８６において、コントローラ４２５は、N-th_Ref_picture_copyに基づいて、index_minus1を決定し記録し、ディスク４１０に記録させる。ここでは、出力サーバ４２６に蓄積されたストリームデータとデータベース系のファイルは、ドライブインタフェース４０７よりディスクドライブ４０９に供給されて、ディスク４１０に記録される。

【０８６３】

以上の処理により、図３１で示されるようにEP_mapが生成される。

【０８６４】

［1stRef_Picture、2ndRef_Picture、3rdRef_Picture、4thRef_Pictureの使い方］

次に、図６３のフローチャートを参照して、図３１のEP_map()を用いた早送り再生処理について説明する。

【０８６５】

図示せぬユーザがビデオコンテンツ再生プログラム２１０に対して早送りを指令したとする。プレイヤ制御モジュール２１２は再生中のストリームのクリップ情報ファイルから、EP_map()に登録されている再生可能開始位置の１を選択し（ステップＳ３９１）、そこに記述されているRPN_EP_startから、N-th_Ref_Picture_copyだけの大きさからの読み出しデータを決定する（ステップＳ３９２）。プレイヤ制御モジュール２１２は、この情報をコンテンツデータ制御モジュール２１３に伝え、さらにデコード制御モジュール２１４に対して早送り再生を指示する。

【０８６６】

コンテンツデータ制御モジュール２１３は、オペレーティングシステム２０１の機能を使用して、再生対象のエレメンタリストリームが多重化されたプログラムストリームが格納されたクリップストリームファイルを読み出してバッファ制御モジュール２１５に供給させる（ステップＳ３９３）。既にファイル名等は指示してあるので、ここでは繰り返さない。再生開始時とは異なり、読み込み開始のアドレスと転送するデータの大きさを伝えて読み込みを指示する。

【０８６７】

ビデオ読み出し機能部２３３は、バッファ制御モジュール２１５に入力された多重化データを多重化分離し（ステップＳ３９４）、ビデオストリームのみをビデオデコーダ制御モジュール２１６に供給する。この場合、早送りの処理であるため、オーディオデコーダ制御モジュール２１７、字幕デコーダ制御モジュール２１５、オーディオ読み出し機能部２３４、字幕読み出し機能部２３３は動作しない。

【０８６８】

入力されたデータには、１枚以上４枚以下の参照画像が含まれている。ここで、早送り処理では、図６４のフローチャートを参照して説明するエントリポイント選定処理（ステップＳ３９５）により選定されたエントリポイントに基づいて、参照画像のみがデコードされて表示されるが、デコーダ制御モジュール２１４にはあらかじめindex_N_minus1が送られており、デコーダ制御モジュール２１４は指定された数だけの参照画像をデコードし、順次後段に送って表示させる（ステップＳ３９６）。

【０８６９】

表示が終了すると、プレイヤ制御モジュール２１２は、次に表示するEP_map()エントリポイントを選択し、上記に説明したことを繰り返すことによって早送り画像を出力する（ステップＳ３９７）。

【０８７０】

このとき、次に表示するEP_map()内のジャンプ先のエントリポイントを選択する際に、index_N_minus1を使用する方法を説明する。既に説明したようにindex_N_minus1はN-th_Ref_Picture_copyまで読み込んだときに、そのデータに含まれる参照画像の枚数を示している。図３１のデータ例では、１番目、および３番目のエントリポイントではindex_N_minux1は0であるから、イントラピクチャ１枚であり、２番目、および４番目のエントリポイントでは、index_N_minus1は3であるから、４枚の参照画像が含まれている。

【０８７１】

早送り画像は、２枚以上の参照画像が出力されることにより、その主観品質が高まるという性質がある。しかしながら、多くの参照画像を出力するためにはデータ読み出し量を多くする必要があり、更新の頻度が遅くなるというトレードオフの関係にある。このため、プレイヤ制御モジュール２１２は次に表示するEP_map()内のエントリポイントを選択するにあたり、index_N_minus1の値を評価する。

【０８７２】

すなわち、早送りの速度が早いときには、EP_map()内のエントリポイントの間引き間隔が大きくなるが、その際にindex_N_minus1が大きいもの（すなわち、主観画質の高くなるもの）を優先的に選択する。また、早送りの速度が遅いときには、index_N_minus1の小さいものも選択する。

【０８７３】

上記に説明したindex_N_minus1決定のアルゴリズムでは、N-th_Ref_picture_copyの値が”30”に近いものを選択している。すなわち、このようなアルゴリズムの下で作成されたEP_map()では、N-th_Ref_picture_copyの通りに読み出すときにデータ読み出し量がほぼ”30”セクタになる。データ読み出し速度が支配的な場合には、一定の読み出し時間であることが重要であるため、こういうエントリポイントを選択する方法も有効である。

【０８７４】

次に、図６４のフローチャートを参照して、早送り時のエントリポイントの選択処理について説明する。

【０８７５】

プレイヤ制御モジュール２１２は、ステップＳ４０１で、早送りのモードが高速モードかあるいは低速モードか否かを判定する。早送りが高速モードであると判定された場合、処理は、ステップＳ４０２に進み、低速モードであると判定された場合、処理は、ステップＳ４１１に進む。

【０８７６】

＜低速モード早送りの場合の選択の説明＞

ステップＳ４０２では、プレイヤ制御モジュール２１２は、早送りが低速モードであるため、選択するエントリポイントの番号（該当エントリ（Ｎ））を前回の該当エントリから２だけインクリメントする（該当エントリ番号＋＝２）。そして、ステップＳ４０３において、プレイヤ制御モジュール２１２は、該当エントリＮ、１前のエントリ（Ｎ－１）、１先のエントリ（Ｎ＋１）からindex_N_minus1を読み取る。

【０８７７】

ステップＳ４０４では、プレイヤ制御モジュール２１２は、inde_N_minus(N)、すなわちＮ番目のエントリポイントの持つindex_N_minus1の値が0あるいは1であるか否かを判定する。ステップＳ４０４において、例えば、index_N_minus1の値が0あるいは1であった場合、処理は、Ｓ４０５に進み、プレイヤ制御モジュール２１２は、Ｎ番目のエントリポイントを選定して処理を終了する。一方、ステップＳ４０４において、index_N_minus1の値が0でも1でも無い場合、処理は、Ｓ４０６に進む。

【０８７８】

ステップＳ４０６では、プレイヤ制御モジュール２１２は、inde_N_minus(N+1)、すなわち(N+1)番目のエントリポイントの持つindex_N_minus1の値が、0あるいは1であるか否かを判定する。ステップＳ４０６において、inde_N_minus(N+1)が、0あるいは1であった場合、処理は、Ｓ４０７に進み、プレイヤ制御モジュール２１２は、(N+1)番目のエントリポイントを選定して処理を終了する。一方、ステップＳ４０６において、inde_N_minus(N+1)が、0でも1でも無い場合、処理は、Ｓ４０８に進む。

【０８７９】

ステップＳ４０８では、プレイヤ制御モジュール２１２は、inde_N_minus(N-1)、すなわち(N-1)番目のエントリポイントの持つindex_N_minus1の値が0あるいは1であるか否かを判定する。ステップＳ４０８において、index_N_minus1の値が0あるいは1であった場合、処理は、Ｓ４０９に進み、プレイヤ制御モジュール２１２は、(N-1)番目のエントリポイントを選定して処理を終了する。0でも1でも無い場合にはＳ４１０に進む。

【０８８０】

Ｓ４１０ではエントリN、(N+1)、(N-1)の全てのエントリポイントにおいてindex_N_minus1が0でも1でもないことが判明していることになるので、プレイヤ制御モジュール２１２は、Ｎ番目のエントリポイントを選定して処理を終了する。

【０８８１】

＜高速モード早送りの場合の選択の説明＞

ステップＳ４１１では、早送りが高速モードであるため、プレイヤ制御モジュール２１２は、選択するエントリポイントの番号（該当エントリ）（Ｎ）を前回の該当エントリポイントから５だけインクリメントする。（該当エントリ番号＋＝５）。そしてステップＳ４１２において、プレイヤ制御モジュール２１２は、該当エントリＮ、１前のエントリ（

Ｎ－１）、１先のエントリ（Ｎ＋１）からindex_N_minus1を読み取る。

【０８８２】

ステップＳ４１３では、プレイヤ制御モジュール２１２は、inde_N_minus(N)、すなわちＮ番目のエントリポイントの持つindex_N_minus1の値が3あるいは2であるか否かを判定する。ステップＳ４１３において、Ｎ番目のエントリポイントの持つindex_N_minus1の値が3あるいは2であった場合、処理は、Ｓ４１４に進み、プレイヤ制御モジュール２１２は、Ｎ番目のエントリポイントを選定して処理を終了する。ステップＳ４１３において、Ｎ番目のエントリの持つindex_N_minus1の値が3でも2でも無い場合、処理は、ステップＳ４１５に進む。

【０８８３】

ステップＳ４１５では、プレイヤ制御モジュール２１２は、inde_N_minus(N+1)、すなわち(N+1)番目のエントリポイントの持つindex_N_minus1の値が3あるいは2であるか否かを判定する。ステップＳ４１５において、(N+1)番目のエントリポイントの持つindex_N_minus1の値が3あるいは2であった場合、処理は、Ｓ４１６に進み、プレイヤ制御モジュール２１２は、(N+1)番目のエントリポイントを選定して処理を終了する。一方、ステップＳ４１５において、(N+1)番目のエントリポイントの持つindex_N_minus1の値が、3でも2でも無い場合、処理は、ステップＳ４１７に進む。

【０８８４】

ステップＳ４１７では、プレイヤ制御モジュール２１２は、inde_N_minus(N-1)、すなわち(N-1)番目のエントリポイントの持つindex_N_minus1の値が3あるいは2であるか否かを判定する。ステップＳ４１７において、(N-1)番目のエントリポイントの持つindex_N_minus1の値が3あるいは2であった場合、処理は、ステップＳ４１８に進み、プレイヤ制御モジュール２１２は、(N-1)番目のエントリポイントを選定して処理を終了する。3でも2でも無い場合にはＳ４１９に進む。

【０８８５】

ステップＳ４１９ではN、(N+1)、(N-1)の全てのエントリにおいてindex_N_minus1が3で2でもないことが判明していることになるので、プレイヤ制御モジュール２１２は、Ｎ番目のエントリポイントを選定して処理を終了する。

【０８８６】

すなわち、早送りの速度が早いときには、EP_map()内のエントリポイントの間引き間隔が大きくなるが、その際にindex_N_minus1が大きいもの（すなわち、主観画質の高くなるもの）を優先的に選択する。また、早送りの速度が遅いときには、index_N_minus1の小さいものも選択する。

【０８８７】

以上の処理により、主観画質を低下させること無く、高速で早送り再生を実現させるようにすることが可能となる。また、低速の早送り再生では、より多くの参照画像を用いて早送り再生させることが可能となるので、再生画像の画質の低減を抑制することが可能となる。

【０８８８】

以上の例においては、再生処理において、読み出されるエントリポイントの情報は、常に一定であることが前提となるが、このような場合、処理能力の高いディスク再生装置では、多くのエントリポイントが読み出されることで再生画像を向上させることが可能となるが、一方で、処理能力の低いディスク再生装置で読み込むエントリポイントの情報を大量にすると、処理速度が遅れてしまう恐れがある。そこで、読み込むエントリポイントの情報にプライオリティを設定し、高い処理能力を持つディスク再生装置では、全てのエントリポイントの情報を用いるようにし、逆に、処理能力の低いディスク再生装置では、プライオリティの高いエントリポイントのみを読み出して処理するようにしても良い。

【０８８９】

図６５は、エントリポイントにプライオリティを設定するようにした、ディスク記録装置の機能を説明する機能ブロック図である。尚、図６５のディスク記録装置において、図

６１のディスク記録装置の機能と同一の機能については、同一の符号を付しており、その説明は、適宜省略するものとする。

【０８９０】

Subtitleエンコーダ４４３は、字幕素材サーバ４４２より字幕の素材を読み取り、圧縮符号化して字幕データサーバ４４４に書き込む。字幕データはビデオデータやオーディオデータとは異なり、時間軸上に間欠的に存在している。このため、字幕素材サーバ４４２に記録された字幕素材および字幕データサーバ４４４に記録された字幕データのそれぞれの表示開始時刻及び表示durationは字幕タイミング情報４４５に用意される。

【０８９１】

多重化器４４１は、図６１の多重化器４２１と基本的には同様の機能を有するが、さらに、字幕データと字幕タイミング情報を多重化する。すなわち、多重化器４４１は、入力されたMPEG２ビデオストリームとMPEG２オーディオストリーム、および字幕データサーバ４４４から供給された字幕データと字幕タイミング情報４４５を読み込み、上述した図１８乃至図２７を用いて説明したように時分割多重化（パケタイズ）している。

【０８９２】

多重化器４４１は、ストリーム中のイントラピクチャを選び、およそ1秒に2回程度の頻度で図２３に示すprivate_stream_2のPES_packet()を挿入する。ここでprivate_stream_2のPES_packet()は、ビデオのイントラピクチャ（デコード開始可能ピクチャ）が直後に置かれていることを示している。このとき、該当イントラピクチャにはタイムスタンプ（PTS/DTS）が必ず付与される。

【０８９３】

また、字幕データの全てのアクセスユニットにはタイムスタンプが必ず付与されている。

【０８９４】

なお、この段階では、図２６で説明した1st_Ref_picture、2ndRef_picture、3rdRef_picture、4thRef_pictureのフィールドにはデータが記録されていない。さらに、図示されていないが、サブタイトルストリームを多重化器４４１に入力して、ビデオストリーム、オーディオストリームと共に多重化してもよい。

【０８９５】

多重化器４４１は、多重化ストリームをFIFO４２２を経由してRAPI書換器４２４に供給するとともに、RAPI情報抽出器４２３にも同時に供給する。RAPI情報抽出器４２４は、多重化ストリームに多重化されたビデオストリームの情報と字幕ストリームの情報を抽出し記憶する。すなわち、RAPI情報抽出器４２４は、ビデオストリームに関してはprivate_stream_2のPES_packet()の開始位置と、private_stream_2のPES_packet()直後に置かれているイントラピクチャの持つタイムスタンプ（ＰＴＳ）値、及び該当イントラピクチャと、それにつづく2枚目、3枚目、4枚目の参照画像の終了位置を検出し、記憶する。また、字幕ストリームに関しては、全ての字幕アクセスユニットの開始位置とタイムスタンプを検出し、記憶する。

【０８９６】

コントローラ４４６は入力された情報を使用して、例えば、図６６で示されるようなEP_map()を作成する。ここで作成するEP_map()はビデオストリームと字幕ストリームに関する情報が含まれているとする。ビデオに関するEP_map()の主たる情報は、ストリーム中に置かれた全てのRAPI、すなわちprivate_stream_2のPES_packet()の位置と直後のイントラピクチャのタイムスタンプであり、これらの情報はRAPI情報抽出器４２３からコントローラ４４６に入力した情報で作成することができる。また、字幕に関するEP_map()の主たる情報は、字幕アクセスユニットの位置とタイムスタンプであり、これらの情報もRAPI情報抽出器４２３からコントローラ４４６に入力した情報で作成することができる。

【０８９７】

コントローラ４４６はRAPI情報抽出器４２３から入力されたデータから、EP_map()の情報のうち、まだ確定していない（後述する）prioirty_flagを作成する。すなわち、全て

のビデオのエントリポイント（RAPIのエントリポイント）および字幕アクセスユニットに対してそのタイムスタンプを評価し（後述する）priority_flagを設定する。コントローラ４４６にはこのためにチャプタ・シーンチェンジ情報４４７が入力されている。

【０８９８】

［EP_mapの説明］

ここで、図６６を参照して、ファイルにプライオリティを設定するときのEP_mapについて説明する。図６６で示されるように、number_of_EP_entriesの後には、その直前のstream_idとprivate_stream_idで特定されるエレメンタリストリームのデコード開始可能点の情報としてのpriority_flag（2ビット）とreserved_for_future_use（14ビット）、さらにPTS_EP_start（32ビット）とRPN_EP_start（32ビット）とのセットが、number_of_EP_entriesが表す数だけ繰り返し配置される。

【０８９９】

priority_flagは、図６７で示されるような意味を持つ。すなわち、ビデオストリームのエントリでは、priority_flagの値が3の場合、そのエントリがチャプタ先頭に対応していることを示す。また、priority_flagの値が2の場合、上記以外で1分おきの重要なシーンチェンジに該当するのエントリであることを示す。またpriority_flagの値が1の場合は、上記以外でシーンチェンジに該当する3秒おきのエントリであることを示す。また、それ以外のエントリは全てpriority_flagの値が0であるとする。

【０９００】

Subtitleストリームに対応するエントリも同様に、priority_flagの値が3の場合、そのエントリがチャプタ先頭に対応していることを示す。また、priority_flagの値が2の場合、上記以外で重要なシーンチェンジに該当するエントリであることを示す。またpriority_flagの値が1の場合、上記以外でシーンチェンジに該当するエントリであることを示す。また、それ以外のエントリは全てpriority_flagの値が0であるとする。

【０９０１】

例えば、このクリップが2時間の映画であり、1秒に2回のランダムアクセスポイントが存在するならば、エントリ数は全体で14400(＝２時間×３６００秒×２回)となる。また、チャプタ数が、およそ数十であるとした場合、priority_flag=3のエントリは数十個の同数となる。重要なシーンチェンジ（priority_flag=2）とその他のシーンチェンジ（priority_flag=1）の数はコンテンツに依存するために一概には言えないが、priority_flag=3あるいは2のエントリは約200、priority_flag=3あるいは2あるいは1のエントリは約2400、その他全てを含めると全部で14400となるここでは例えばprioiry_flag=2のものと=1のものの総計が1000であったとする。この場合、priority_flag=3，2，1のエントリだけを読み込むことにすると、全部を読み込んだ場合に比べて約1000/14,400＝14分の１のメモリ量となる。またこの例では（ひとつのエントリが10バイトなので）一本のビデオストリームに対して、10バイト×（14400-1000）＝120134キロバイトのメモリを節約することが可能となる。

【０９０２】

また、字幕に関しては2時間の映画に対して1000から2000の字幕が存在するといわれており、これに対してチャプタは数十なので、priority_flag=3のエントリだけを読み込むとすると（数十を1000あるいは2000で除して）数十分の一のメモリに節約することが可能となる。字幕ストリームはビデオストリームに比べて本数が多くなるため、削減効果も十分である。

【０９０３】

なお、この例ではフラグを3，2，1，0の値で示しているが、それぞれをビットで表現し、対応するビットを1とセットするという方法も考えられる。すなわち、このフィールドを3ビットとし、最上位ビットが1の場合にはチャプタ先頭であることを示し、次のビットが1の場合には1分おきのエントリであることを示し、最下位ビットが1の場合には5秒おきのエントリであることを示し、全てのビットが0であれば、それら三つのカテゴリに含まれないと定義することも考えられる。

【０９０４】
Subtitleストリームに対応するエントリでは、priority_flagの値が1の場合にはそのエントリがチャプタ先頭に対応していることを示す。それ以外のエントリでは全てpriority_flagの値が0であるとする。
【０９０５】
次に、図６８のフローチャートを参照して、prioirty_flag設定処理について説明する。
【０９０６】
ステップＳ４４１において、コントローラ４４６は、ビデオのエントリに対して、ビデオのチャプタの先頭位置であるか否か、すなわち、評価中のエントリがチャプタ・シーンチェンジ情報４４７にあるチャプタの時刻に該当しているか否かを判定する。ここで該当しているとは差分が0と定義する。例えば、チャプタの時刻に該当している場合、ステップＳ４４２において、コントローラ４４６は、priority_flag=3として設定し出力サーバ４２６に記憶させる。
【０９０７】
一方、ステップＳ４４１において、ビデオのチャプタの先頭位置ではないと判定された場合、ステップＳ４４３において、コントローラ４４６は、ビデオの重要なシーンチェンジの位置であるか否か、すなわち、次に評価中のエントリがチャプタ・シーンチェンジ情報４４７にある「重要なシーンチェンジ」先頭から1分おきの位置に該当するか否かを判定する。ステップＳ４４３において、例えば、「重要なシーンチェンジ」先頭から1分おきの位置に該当している場合、ステップＳ４４４において、コントローラ４４６は、priority_flag=2に設定する。
【０９０８】
ステップＳ４４３において、重要なシーンチェンジではないと判定された場合、ステップＳ４４５において、コントローラ４４６は、ビデオの通常のシーンチェンジであるか否か、すなわち、次に評価中のエントリがチャプタ・シーンチェンジ情報４４７にある「シーンチェンジ」の先頭から3秒おきの位置に該当するか否かを判定する。ステップＳ４４５において、例えば、「シーンチェンジ」の先頭から3秒おきの位置に該当している場合、ステップＳ４４６において、コントローラ４４６は、priority_flag=1に設定する。
【０９０９】
ステップＳ４４５において、通常のシーンチェンジではないと判定された場合、すなわち、いずれのシーンチェンジにも該当しないと判定された場合、ステップＳ４４７において、コントローラ４４６は、priority_flag=0に設定する。
【０９１０】
ステップＳ４４８において、コントローラ４４６は、全てのビデオの全てのエントリについて処理が終了したか否かを判定し、終了していないと判定された場合、処理は、ステップＳ４４１に戻り、それ以降の処理が繰り返される。すなわち、ビデオのエントリについて全てのpriority_flagが設定されるまで、ステップＳ４４１乃至Ｓ４４８の処理が繰り返される。
【０９１１】
そして、ステップＳ４４８において、全ての処理が終了したと判定された場合、すなわち、ビデオの全てのエントリについての処理が終了したと判定された場合、その処理は、ステップＳ４４９に進む。
【０９１２】
ステップＳ４４９において、コントローラ４４６は、字幕に関して、エントリがチャプタ先頭位置に該当するか否か、すなわち、評価中のエントリがチャプタシーンチェンジ情報４４７にあるチャプタの時刻に該当しているか否かを判定する。ステップＳ４４９において、エントリがチャプタの時刻に該当している場合、ステップＳ４５０において、コントローラ４４６は、priority_flag=3に設定する。
【０９１３】

一方、ステップＳ４４９において、字幕のチャプタの先頭位置ではないと判定された場合、ステップＳ４５１において、コントローラ４４６は、字幕の重要なシーンチェンジの位置であるか否か、すなわち、次に評価中のエントリがチャプタ・シーンチェンジ情報４４７にある「重要なシーンチェンジ」先頭から1分おきの位置に該当するか否かを判定する。ステップＳ４５１において、例えば、「重要なシーンチェンジ」先頭から1分おきの位置に該当している場合、ステップＳ４５２において、コントローラ４４６は、priority_flag=2に設定する。

【０９１４】

ステップＳ４５１において、重要なシーンチェンジではないと判定された場合、ステップＳ４５３において、コントローラ４４６は、字幕の通常のシーンチェンジであるか否か、すなわち、次に評価中のエントリがチャプタ・シーンチェンジ情報４４７にある「シーンチェンジ」の先頭から3秒おきの位置に該当するか否かを判定する。ステップＳ４５３において、例えば、「シーンチェンジ」の先頭から3秒おきの位置に該当している場合、ステップＳ４５４において、コントローラ４４６は、priority_flag=1に設定する。

【０９１５】

ステップＳ４５３において、通常のシーンチェンジではないと判定された場合、すなわち、いずれのシーンチェンジにも該当しないと判定された場合、ステップＳ４５５において、コントローラ４４６は、priority_flag=0に設定する。

【０９１６】

ステップＳ４５６において、コントローラ４４６は、字幕の全てのエントリについて処理が終了したか否かを判定し、終了していないと判定された場合、処理は、ステップＳ４４９に戻り、それ以降の処理が繰り返される。すなわち、字幕のエントリについて全てのpriority_flagが設定されるまで、ステップＳ４４９乃至Ｓ４５６の処理が繰り返され、ステップＳ４５６において、字幕の全てのエントリについて処理が終了したと判定されると、コントローラ４４６は、図６６の構文に従ったEP_map()のデータを出力サーバ４２６に対して出力する。

【０９１７】

［再生側の動作：EP_map()の間引き］

以上のように設定されたpriority_flagに基づいて、ディスク再生装置は、メモリ（例えば、図１のメモリ１１３）の大きさにより、EP_map()を間引く。すなわち、コストを削減するために機能を減らしているディスク再生装置では、高い値のpriority_flagを持つエントリのみをメモリに残す。もちろん、メモリの容量が、EP_map()全体を保持することができるでけ確保している機器ではこのような作業を必ずしも行う必要はない。

【０９１８】

例えば、図３３のフローチャートにおけるステップＳ１０６の処理において、ビデオに関してはpriority_flag=1以上のエントリをメモリに保持し、字幕に関してはpriority_flag=1以上のエントリをメモリに保持するとする。この場合、EP_map()の読み出しの際に、stream_idとprivate_stream_idの値により、プレイヤ制御モジュール２１２はストリームの種類がビデオであったとき、priority_flagの値が3，2、あるいは1のエントリは読み込み（メモリに読み込む）、priority_flagの値が0のエントリは読み捨てる（メモリに読み込まない）。またストリームの種類が字幕であった場合、prioirty_flag=3および21のエントリは読み込み、priority_flagの値が1，０のエントリは読み捨てる。

【０９１９】

以上の処理を行うことにより、一本のビデオストリームに対するEP_mapのために必要なメモリの容量は、1/6乃至1/10程度とすることが可能となる。また、一本の字幕ストリームに対するEP_mapのために必要なメモリ量はおよそ数十分の一程度とすることが可能となる。結果として、低コストのディスク再生装置でも、再生能力（メモリの容量）に応じてエントリを記憶させることができ、効率的な再生処理を実現させることが可能となる。

【０９２０】

なお、ここではpriority_flag=3をチャプタ先頭と意味づけしているが、チャプタ先頭

及び重要なシーンチェンジポイントなど任意の意味に定義することが可能である。

【０９２１】

なお、本実施の形態では、上述した一連の処理を、ソフトウェアによって行うこととしたが、上述した一連の処理は、専用のハードウェアにより行うこともできる。

【０９２２】

また、本実施の形態では、ビデオデコーダ１１６（図１）およびビデオエンコーダ４０３（図６０）として、ハードウェアデコーダを採用することとしたが、ビデオデコーダ１１６およびビデオエンコーダ４０３（図６０）としては、ソフトウェアデコーダを採用することも可能である。オーディオデコーダ１１７(図１）およびオーディオエンコーダ４０４（図６０）についても、同様である。

【０９２３】

さらに、本実施の形態では、字幕デコーダとして、ソフトウェアデコーダを採用することとしたが、字幕デコーダとしては、ハードウェアデコーダを採用することも可能である。

【図面の簡単な説明】

【０９２４】

【図１】本発明を適用したディスク再生装置の一実施の形態のハードウェア構成例を示すブロック図である。

【図２】CPU１１２が実行するソフトウェアモジュール群の構成例を示すブロック図である。

【図３】実際の時間経過と90kHzの時計の計時との関係を示した図である。

【図４】実際の時間経過とビデオデコーダのビデオデータの出力に応じて時刻を更新する時計の計時との関係を示した図である。

【図５】バッファ制御モジュール２１５の構成例を示すブロック図である。

【図６】ディスク１０１におけるディレクトリ構成例を示す図である。

【図７】”PLAYLIST.DAT”ファイルのシンタクスを示す図である。

【図８】PlayItem()のシンタクスを示す図である。

【図９】PlayListMark()のシンタクスを示す図である。

【図１０】mark_typeの値と、Mark()のタイプとの関係を示す図である。

【図１１】PlayList(), PlayItem()、クリップ、およびクリップストリームファイルに格納されたプログラムストリームの関係を示す図である。

【図１２】クリップ情報ファイルClip()のシンタクスを示す図である。

【図１３】エレメンタリストリームを識別するstream_idおよびprivate_stream_idと、エレメンタリストリームとの関係を示す図である。

【図１４】StaticInfo()のシンタクスを示す図である。

【図１５】DynamicInfo()のシンタクスを示す図である。

【図１６】EP_map()のシンタクスを示す図である。

【図１７】図１６のindex_N_minus1の値とlstRef_Picture乃至4thRef_Pictureとの関係を示す図である。

【図１８】MPEG-2 Systemのプログラムストリーム、プログラムストリームパック、およびプログラムストリームパックヘッダのシンタクスを示す図である。

【図１９】MPEG-2 SystemのPESパケットのシンタクスを示す図である。

【図２０】MPEG-2 SystemのPESパケットのシンタクスを示す図である。

【図２１】MPEG-2 SystemのPESパケットのシンタクスを示す図である。

【図２２】MPEG-2 SystemにおけるPES_packet()のstream_idに記述される値と、エレメンタリストリームの属性（種類）との関係を示す図である。

【図２３】ディスク再生装置が採用するstream_idを示す図である。

【図２４】private_stream1_PES_payload()のシンタクスを示す図である。

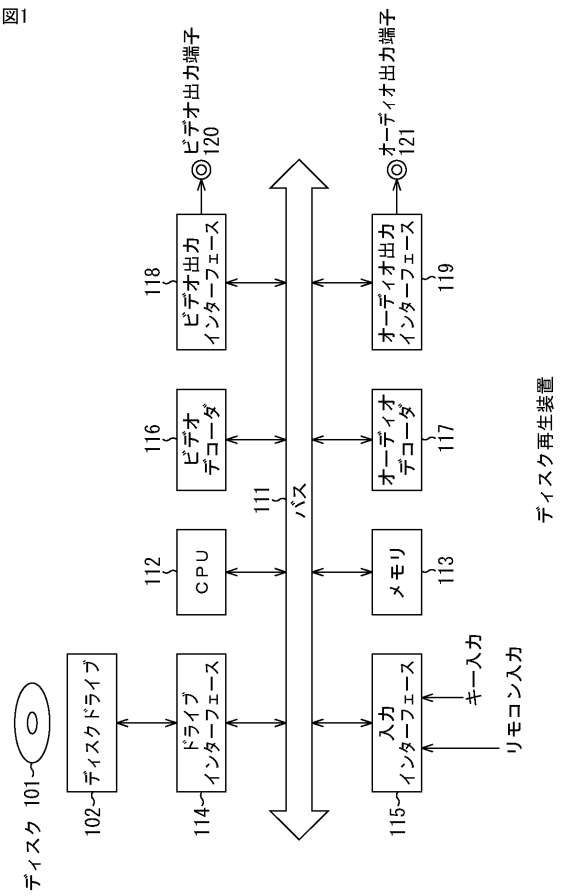
【図２５】private_stream_idの値と、private_payload()に格納されるエレメンタリストリームの属性との関係を示す図である。

【図２６】private_stream2_PES_payload()のシンタクスを示す図である。
【図２７】au_information()のシンタクスを示す図である。
【図２８】pic_structを説明する図である。
【図２９】”PLAYLIST.DAT”ファイルの具体例を示す図である。
【図３０】クリップ情報ファイル”00001.CLP”，”00002.CLP”，”00003.CLP”の具体例を示す図である。
【図３１】クリップ情報ファイル”00001.CLP”の中のEP_map()の具体例を示す図である。
【図３２】PlayList #0とPlayList #1の中のPlayListMark()の具体例を示す図である。
【図３３】再生前処理を説明するフローチャートである。
【図３４】再生処理を説明するフローチャートである。
【図３５】デコード順序と出力順序の関係を説明するフローチャートである。
【図３６】ビデオデコーダの構成を説明する図である。
【図３７】図３６のDPBの構成を説明する図である。
【図３８】時刻更新処理を説明するフローチャートである。
【図３９】pic_structの値に応じた時刻更新処理を説明する図である。
【図４０】PlayItem乗り換え処理を説明するフローチャートである。
【図４１】タイムコード表示処理を説明するフローチャートである。
【図４２】ストリーム切り替え処理を説明するフローチャートである。
【図４３】バッファ制御モジュール２１５の処理を説明するフローチャートである。
【図４４】バッファ制御モジュール２１５の処理を説明するフローチャートである。
【図４５】ビデオストリームの読み出しの処理を説明するフローチャートである。
【図４６】オーディオストリームの読み出しの処理を説明するフローチャートである。
【図４７】字幕ストリームの読み出しの処理を説明するフローチャートである。
【図４８】再同期処理を説明するフローチャートである。
【図４９】マーク処理を説明するフローチャートである。
【図５０】マーク処理における一致判定を説明する図である。
【図５１】playListEndタイミングを説明する図である。
【図５２】playListEndタイミングを説明する図である。
【図５３】イベントの間隔を説明する図である。
【図５４】出力属性の制御処理を説明するフローチャートである。
【図５５】クリップ情報ファイル”00003.CLP”に記述されているpts_change_pointとDynamicInfo()とのセットの具体例を示す図である。
【図５６】字幕表示制御処理を説明するフローチャートである。
【図５７】キャプチャ制御処理とバックグラウンド／スクリーンセーバ処理を説明するフローチャートである。
【図５８】private_stream2_PES_payload()の他のシンタクスを示す図である。
【図５９】au_information()の他のシンタクスを示す図である。
【図６０】ディスク記録装置のハードウェア構成例を示すブロック図である。
【図６１】図６０のディスク記録装置により実現される機能を説明するブロック図である。
【図６２】Ep_map生成処理を説明するフローチャートである。
【図６３】早送り再生処理を説明するフローチャートである。
【図６４】エントリポイント選定処理を説明するフローチャートである。
【図６５】図６０のディスク記録装置により実現されるその他の機能を説明するブロック図である。
【図６６】EP_map()のその他のシンタクスを示す図である。
【図６７】図６５のpriority_flagを説明する図である。
【図６８】プライオリティ設定処理を説明するフローチャートである。
【符号の説明】

【０９２５】
１０１　ディスク，　１０２　ディスクドライブ，　１１１　バス，　１１２　CPU，　１１３　メモリ，　１１４　ドライブインターフェース，　１１５　入力インターフェース，　１１６　ビデオデコーダ，　１１６A　ビデオでコードエンジン，　１１６Ｂ　DPB（Decoded Picture Baffer），　１１６Ｂ－０乃至１１６Ｂ－ｎ　DPB－０乃至DPB－ｎ，　１１７　オーディオデコーダ，　１１８　ビデオ出力インターフェース，　１１９　オーディオ出力インターフェース，　１２０　ビデオ出力端子，　１２１　オーディオ出力端子，　２０１　オペレーティングシステム，　２１０　ビデオコンテンツ再生プログラム，　２１１　スクリプト制御モジュール，　２１２　プレイヤ制御モジュール，　２１３　コンテンツデータ供給モジュール，　２１４　デコード制御モジュール，　２１４Ａ　計時部，　２１５　バッファ制御モジュール，　２１５Ａ　バッファ，　２１６　ビデオデコーダ制御モジュール，　２１７　オーディオデコーダ制御モジュール，　２１８　字幕デコーダ制御モジュール，　２１９　グラフィクス処理モジュール，　２２０　ビデオ出力モジュール，　２２０Ａ　FIFO，　２２１　オーディオ出力モジュール，　２２１Ａ　FIFO，　２３１　データ先頭ポインタ記憶部，　２３２　データ書き込みポインタ記憶部，　２３３　ビデオ読み出し機能部，　２３４　オーディオ読み出し機能部，　２３５　字幕読み出し機能部，　２４１　ビデオ読み出しポインタ記憶部，　２４２　stream_idレジスタ，　２４３　au_information()レジスタ２４３，　２５１　オーディオ読み出しポインタ記憶部，　２５２　stream_idレジスタ，　２５３　private_stream_idレジスタ，　２６１　字幕読み出し機能フラグ記憶部２６１，　２６２　字幕読み出しポインタ記憶部，　２６３　stream_idレジスタ，　２６４　private_stream_idレジスタ，　３０１　ビデオバッファ，　３０２　付加情報バッファ，　４００Ａ　ビデオ入力端子，　４００Ｂ　オーディオ入力端子，　４０１　ビデオ入力インタフェース，　４０２　オーディオ入力インタフェース，　４０３　ビデオエンコーダ，　４０４　オーディオエンコーダ，　４０５　CPU，　４０６　メモリ，　４０７　ドライブインタフェース，　４０８　入力インタフェース，　４０９　ディスクドライブ，　４１０　ディスク，　４１１　バス，　４２１　多重化器，　４２２　FIFO，　４２３　RAPI情報抽出器，　４２４　RAPI書換器，　コントローラ，　４２６　出力サーバ，　４４１　多重化器，　４４２　字幕素材サーバ，　４４３　Subtitleエンコーダ，　４４４　字幕データサーバ，　４４５　字幕タイミング情報，　４４６　コントローラ

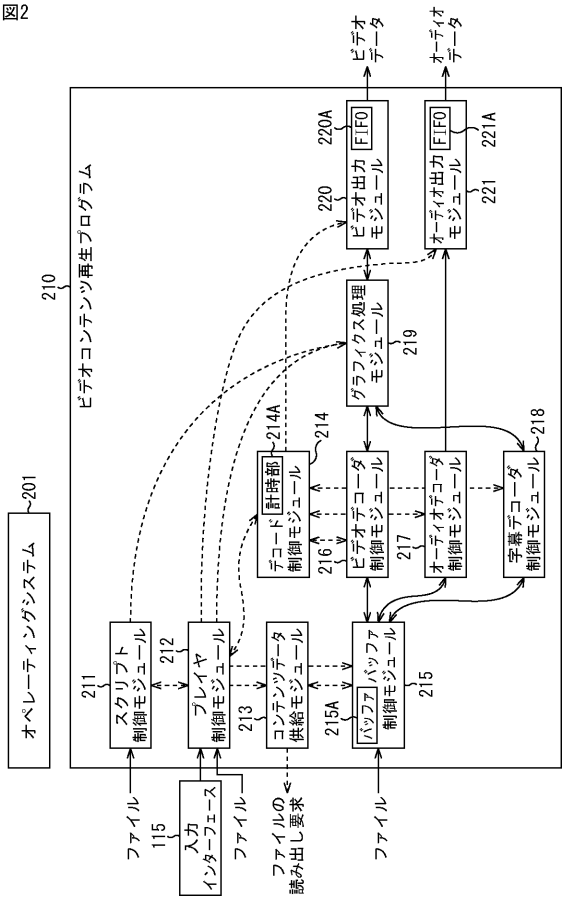
【図１】

図1



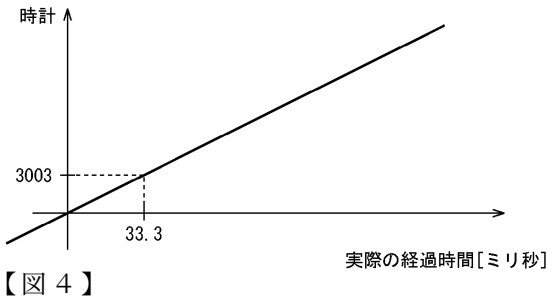
【図２】

図2



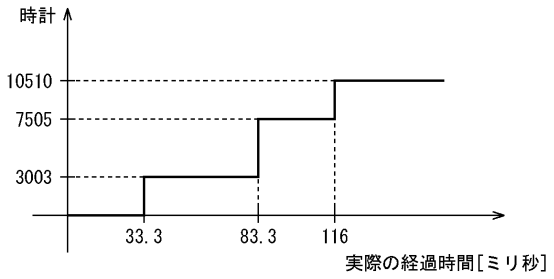
【図３】

図3



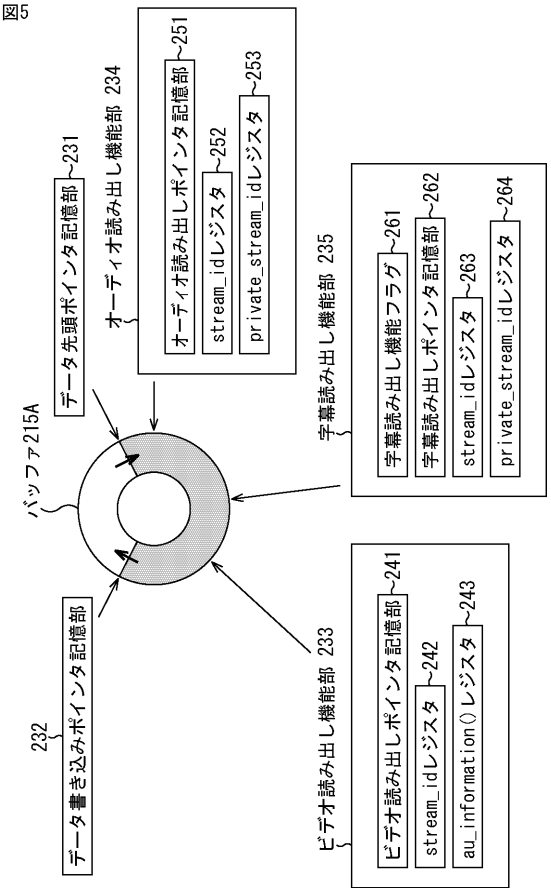
【図４】

図4

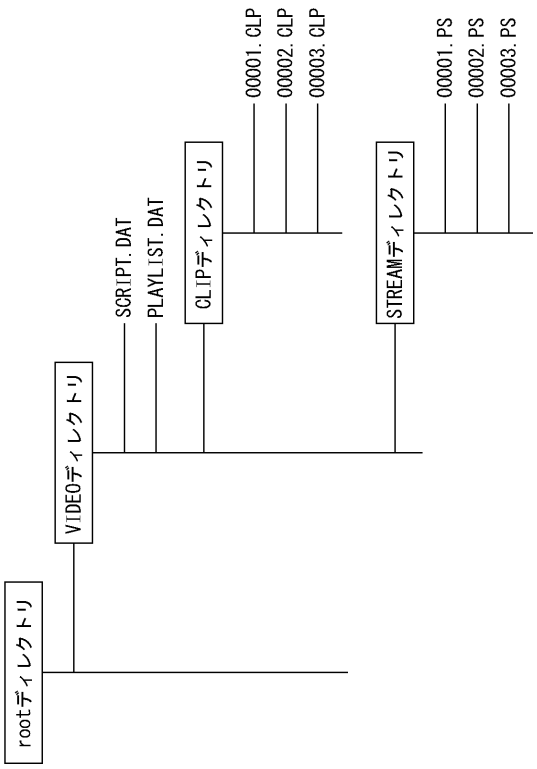


【図５】

図5



【図６】
図6



【図７】
図7

PlayList.PATファイル

Syntax	No.of bits	Mnemonic
“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
// 属性情報		
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()		
}		
}		
}		

【図８】
図8

PlayItem()

Syntax	No.of bits	Mnemonic
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	
}		

【図９】
図9

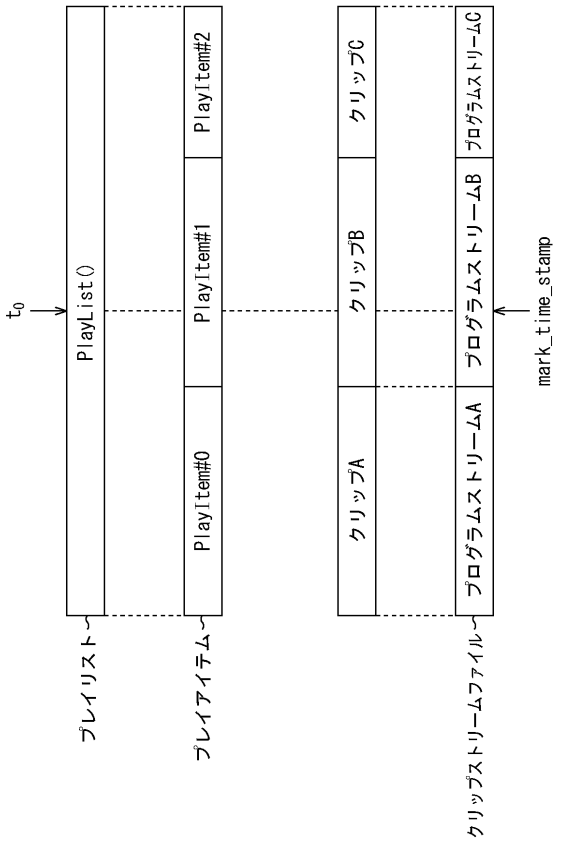
PlayListMark()

Syntax	No.of bits	Mnemonic
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
}		
}		
}		

【図１０】
図10

mark_type	stream coding
0	reserved
1	"Chapter"マーク
2	"Index"マーク
3	"Event"マーク
4～255	reserved

【図１１】
図11



【図１２】
図12

clip()「Clip情報ファイル」(.CLP)

Syntax	No.of bits	Mnemonic
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()		
}		

【図１３】
図13

エレメンタリ ストリーム種類	stream_id	private_stream_id
ビデオ	0xE0～0xEF	(なし)
ATRACオーディオ	0xBD	0x00～0x0F
LPCMオーディオ	0xBD	0x10～0x1F
字幕	0xBD	0x80～0x9F

【図１４】
図14

StaticInfo()

Syntax	No.of bits	Mnemonic
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
}		
}		

【図１５】
図15

DynamicInfo

Syntax	No.of bits	Mnemonic
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
}		
}		

【図１６】
図16

EP_map()

Syntax	No.of bits	Mnemonic
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++){		
if(isVideoStream){		
index_N_minus1	2	uimsbf
N-th_Ref_picture_copy	14	uimsbf
}else		
reserved_for_future_use	16	bslbf
}		
PTS_EP_start	32	uimsbf
RPN_EP_start	32	uimsbf
}		
}		
}		

【図１７】
図17

	index_N_minus1
1stRef_Picture	0
2ndRef_Picture	1
3rdRef_Picture	2
4thRef_Picture	3

【図１８】

図18

Table 2-31—Program Stream

Syntax	No. of bits	Mnemonic
MPEG2_program_stream() {		
do {		
pack()		
}while(nextbits()== pack_start_code)		
MPEG_program_end_code	32	bslbf
}		

Table 2-32—Program Stream pack

Syntax	No. of bits	Mnemonic
pack() {		
pack_header()		
while(nextbits()==packet_start_code_prefix){		
PES_packet()		
}		
}		

Table 2-33—Program Stream pack header

Syntax	No. of bits	Mnemonic
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()		
}		
}		

【図１９】

図19

Table 2-17—PES packet

Syntax	No. of bits	Mnemonic
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		
&& 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
}		

【図２０】

図20

Table 2-17—PES packet (continued)

Syntax	No. of bits	Mnemonic
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
}		
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
}		
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
}		

【図２１】

図21

Table 2-17—PES packet(concluded)

Syntax	No. of bits	Mnemonic
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
}		
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
}		
}		
}		

【図２２】
図22

Table 2-18—Stream_id assignments

stream_id	Note	stream coding
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

The notation x means that the values'0'or'1'are both permitted and results in the same stream type.The stream number is given by the values taken by the x's.
NOTE 1—PES packets of type program_stream_map have unique syntax specified in 2.5.4.1.
NOTE 2—PES packets of type private_stream_1 and ISO/IEC_13552_stream follow the same PES packet syntax as those for ITU-T Rec. H.262 | ISO/IEC 13818-2 video and ISO/IEC 13818-3 audio streams
NOTE 3—PES packets of type private_stream_2, ECM_stream and EMM_stream are similar to private_stream_1 except no syntax is specified after PES_packet_length field.
NOTE 4—PES packets of type program_stream_directory have a unique syntax specified in 2.5.5.
NOTE 5—PES packets of type DSM-CC_stream have a unique syntax specified in ISO/IEC 13818- 6.
NOTE 6—This stream_id is associated with stream_type 0x09 in Table 2-29.
NOTE 7—This stream_id is only used in PES packets, which carry data from a Program Stream or an ISO/IEC 11172-1 System Stream, in a Transport Stream (refer to 2.4.3.7).

【図２３】
図23

stream_id	stream coding(ストリームの種類)
1011 1101	private_stream_1
1011 1110	padding_stream
1011 1111	private_stream_2
110x xxxx	audio stream number x xxxx
1110 xxxx	video stream number xxxx

【図２４】
図24

Syntax	No.of bits	Mnemonic
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()		
}		

【図２５】
図25

private_stream_id	stream coding(ストリームの種類)
0000 xxxx	ATRAC Audio stream number xxxx
0001 xxxx	LPCM stream number xxxx
100x xxxx	Subtitle stream number x xxxx

【図２６】
図26

Syntax	No. of bits	Mnemonic
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
}		
}		

【図２７】
図27

Syntax	No. of bits	Mnemonic
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
}		

【図２８】
図28

Value	Indicated display of picture	Restrictions	NumClockTS
0	frame	field_pic_flag shall be 0	1
1	top field	field_pic_flag shall be 1, bottom_field_flag shall be 0	1
2	bottom field	field_pic_flag shall be 1, bottom_field_flag shall be 1	1
3	top field, bottom field, in that order	field_pic_flag shall be 0	2
4	bottom field, top field, in that order	field_pic_flag shall be 0	2
5	top field, bottom field, top field repeated, in that order	field_pic_flag shall be 0	3
6	bottom field, top field, bottom field repeated, in that order	field_pic_flag shall be 0	3
7	frame doubling	field_pic_flag shall be 0 fixed_frame_rate_flag shall be 1	2
8	frame tripling	field_pic_flag shall be 0 fixed_frame_rate_flag shall be 1	3
9..15	reserved		

【図２９】
図29

// "PLAYLIST.DAT"		
number_of_PlayLists	2	
// PlayList()	PlayList #0	PlayList #1
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	

【図３０】

図30

// 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_flage='Yes'	cc_flage='Yes'	cc_flage='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='日本語'	audio_language_code='日本語'	picture_size='720x480'
	channel_configuration='STEREO'	channel_configuration='STEREO'	frame_rate='29.97Hz'
	lfe_existence='NO'	lfe_existence='NO'	cc_flage='No'
	sampling_frequency='48KHz'	sampling_frequency='48KHz'	cc_flage='No'
number_of_DynamicInfo	0	0	0
// 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='日本語'	subtitle_language_code='日本語'	audio_language_code='日本語'
	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='日本語'	subtitle_language_code='日本語'	
	configurable_flag=1	configurable_flag=1	
number_of_DynamicInfo	0	0	

【図３１】

図31

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	1stRef_Picture	2ndRef_Picture	3rdRef_Picture	4thRef_Picture	index_N_minus1	N-th_Ref_picture_copy
	90000	0	28	37	48	58	0	28
	135045	244	10	18	25	31	3	31
	180090	305	25	44	50	54	0	25
	225135	427	8	15	22	29	3	29
	270180	701	26	32	41	48	1	32

【図３２】

図32

PlayList #0のPlayListMark()
number_of_PlayList_marks==7

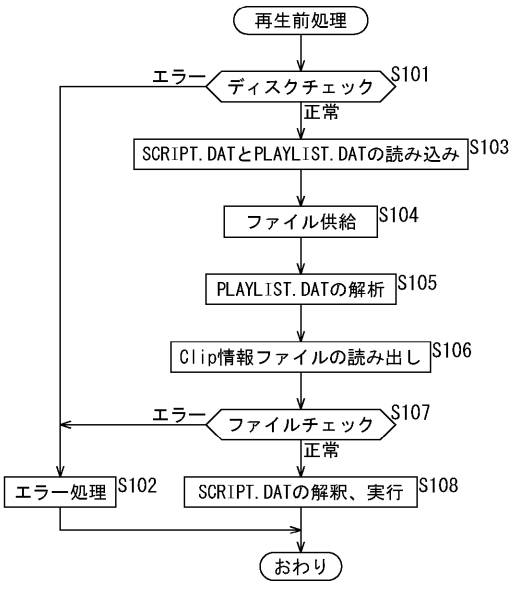
Mark()	mark_type	ref_to_PlayItem_id	mark_time_stamp	entry_ES_stream_id	entry_ES_private_stream_id	mark_data	
#0	Chapter	0	180,090	0	0	1	先頭
#1	Index	0	5,580,090	0	0	1	1分
#2	Index	0	10,980,090	0	0	2	2分
#3	Event	0	16,380,090	0	0	0	3分
#4	Chpater	1	90,000	0	0	2	先頭
#5	Index	1	5,490,000	0	0	1	1分
#6	Index	1	10,890,000	0	0	2	2分

PlayList #1のPlayListMark()
number_of_PlayList_marks==3

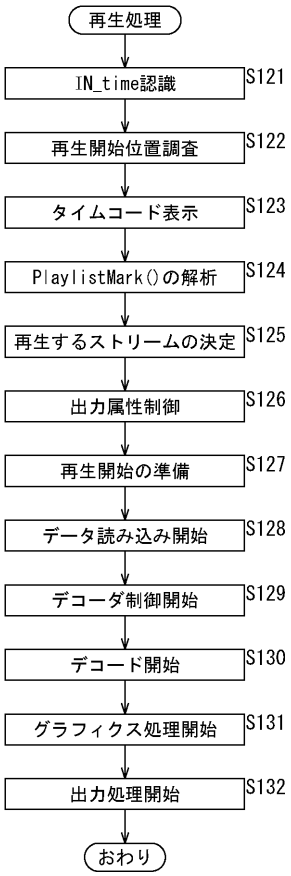
Mark()	mark_type	ref_to_PlayItem_id	mark_time_stamp	entry_ES_stream_id	entry_ES_private_stream_id	mark_data	
#0	Chapter	0	90,000	0	0	0	先頭
#1	Event	0	27,090,000	0xE0	0	1	5分
#2	Event	0	27,540,000	0xE1	0	2	5分5秒(305秒)

【図３３】

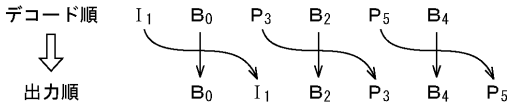
図33



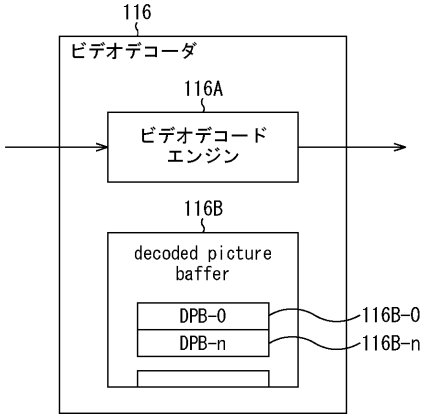
【図３４】
図34



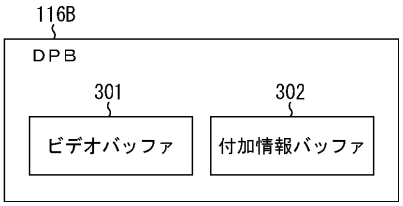
【図３５】
図35



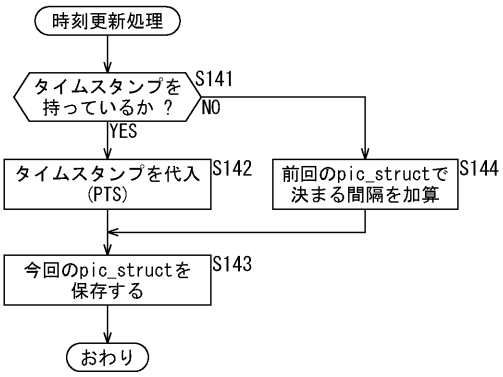
【図３６】
図36



【図３７】
図37



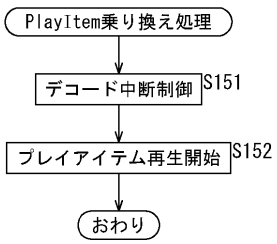
【図３８】
図38



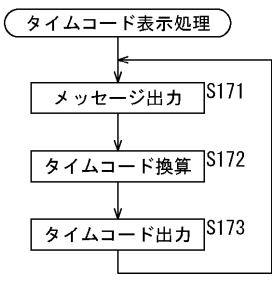
【図３９】
図39

PTS	保存してあった pic_structの値	動作
あり	—	PTSを代入
なし	0, 1, 2, 3, 4	時刻＋＝2フィールド(3003)
	5, 6	時刻＋＝3フィールド(4505)

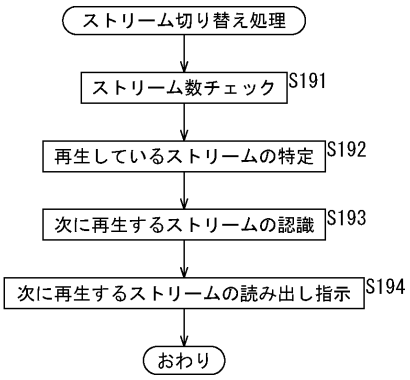
【図４０】
図40



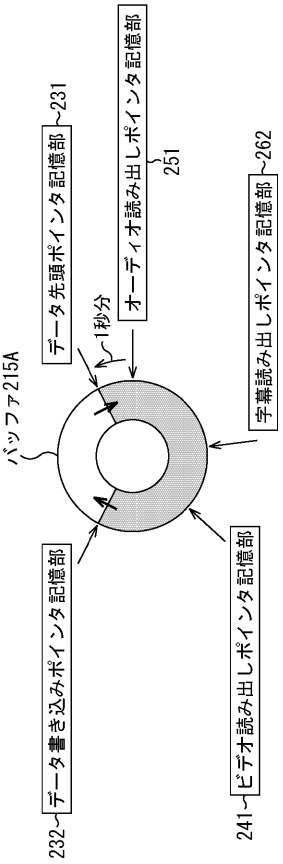
【図４１】
図41



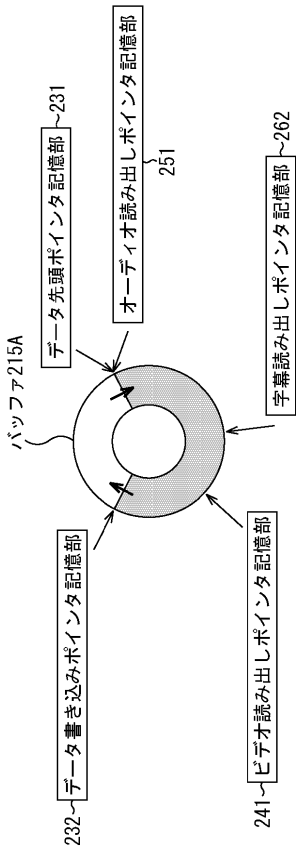
【図４２】
図42



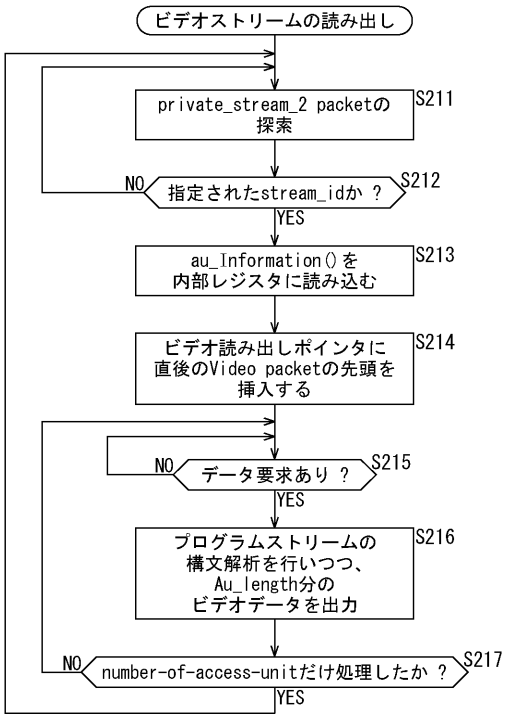
【図４３】
図43



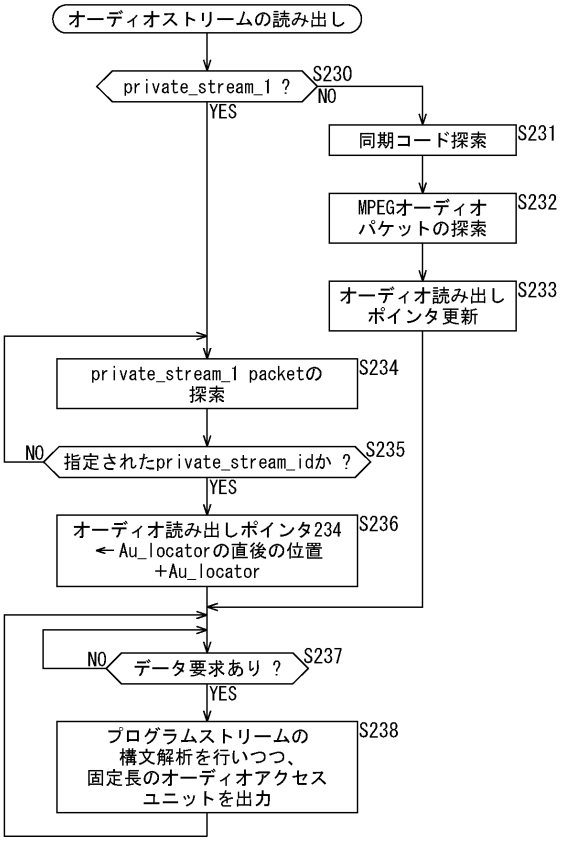
【図４４】
図44



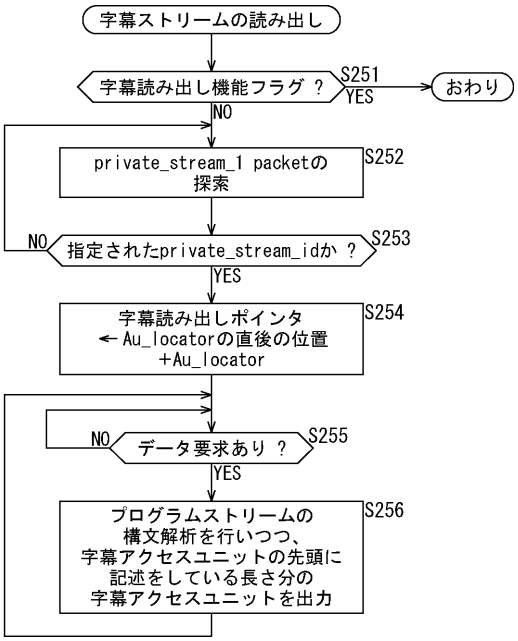
【図４５】
図45



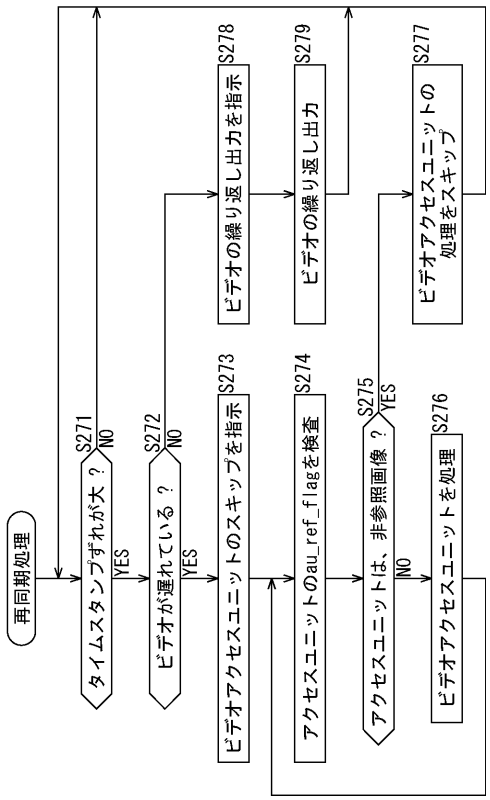
【図４６】
図46



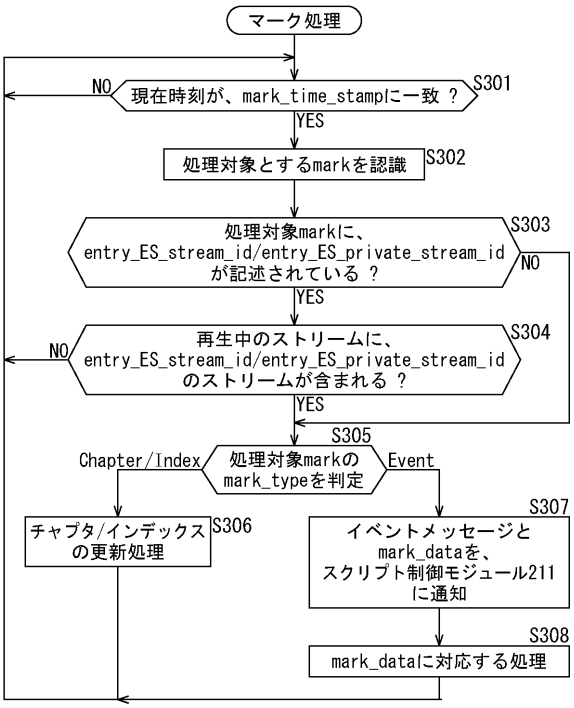
【図４７】
図47



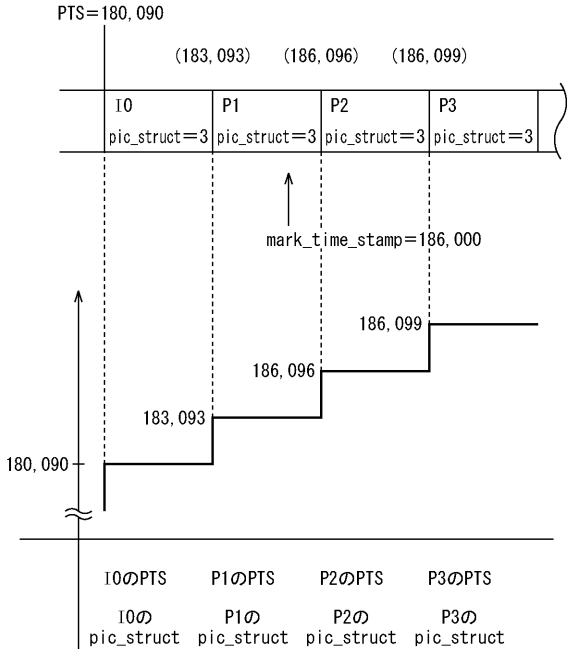
【図４８】
図48



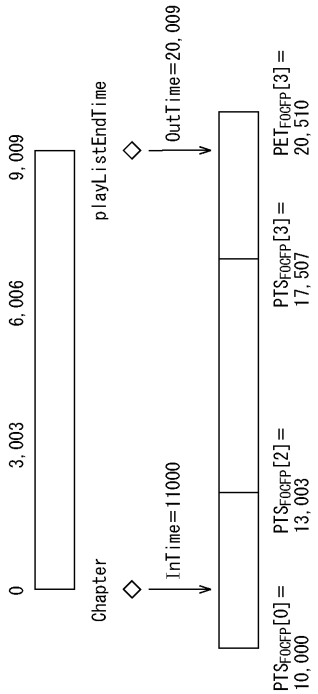
【図４９】
図49



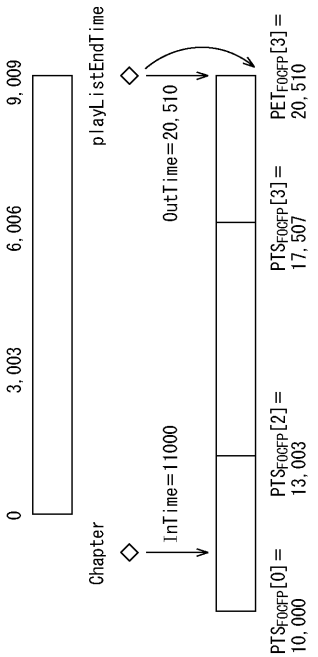
【図５０】
図50



【図５１】
図51



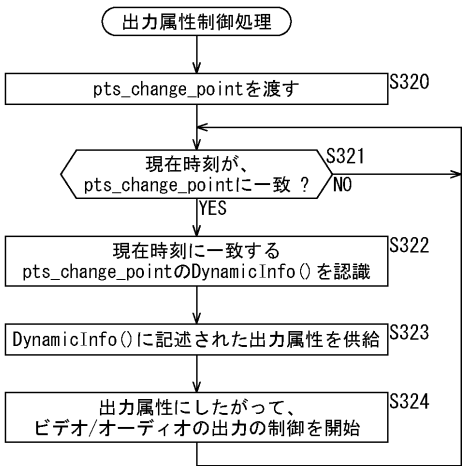
【図５２】
図52



【図５３】
図53

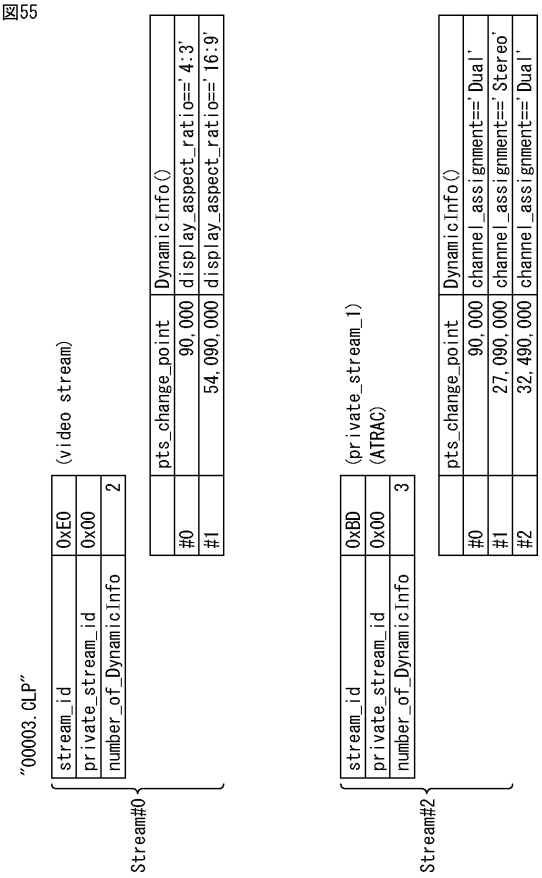
	Case A (23.976p)	Case B (59.94i)
frame rate	5005/240000	4004/240000
90kHzで表現した最小イベント間隔	7507	6006

【図５４】
図54



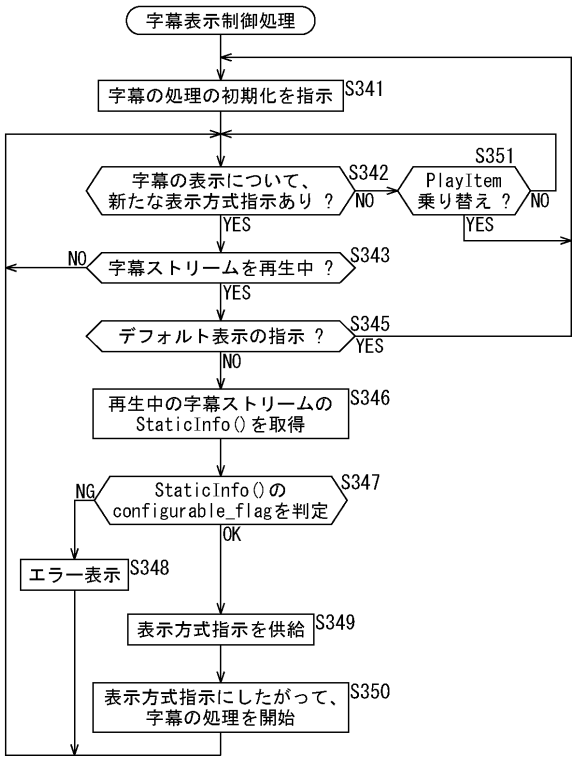
【図５５】

図55



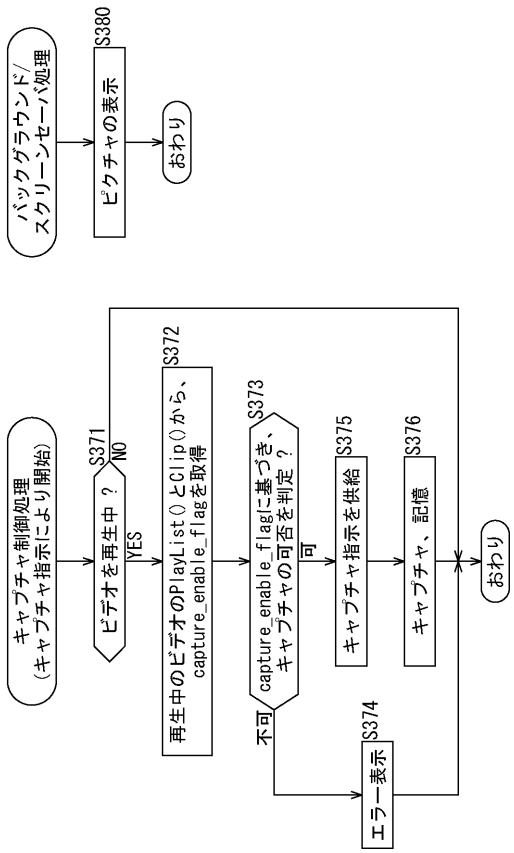
【図５６】

図56



【図５７】

図57



【図５８】

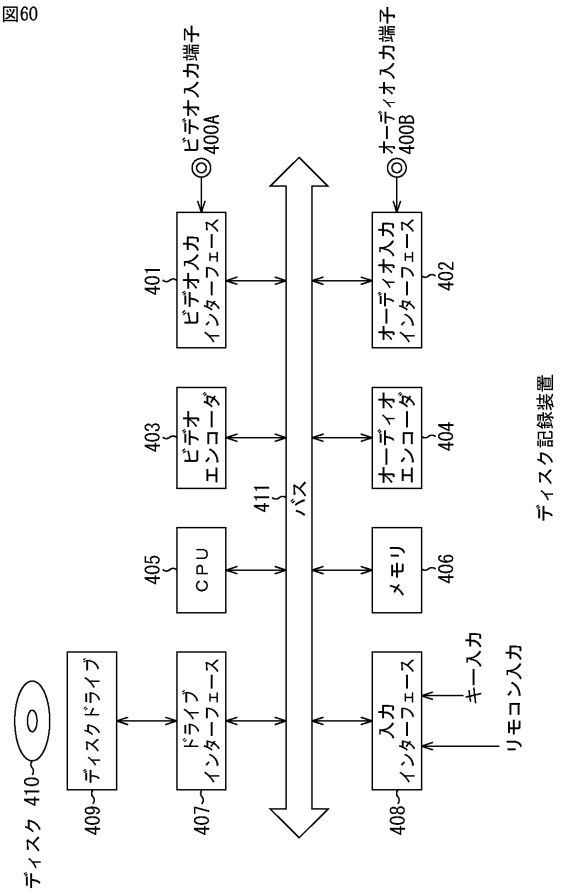
図58

Syntax	No.of bits	Mnemonic
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
}		
}		

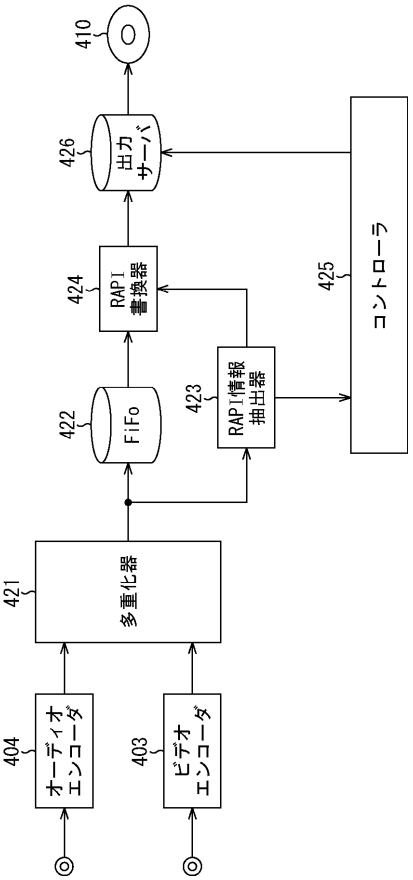
【図５９】
図59

Syntax	No.of bits	Mnemonic
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
}		

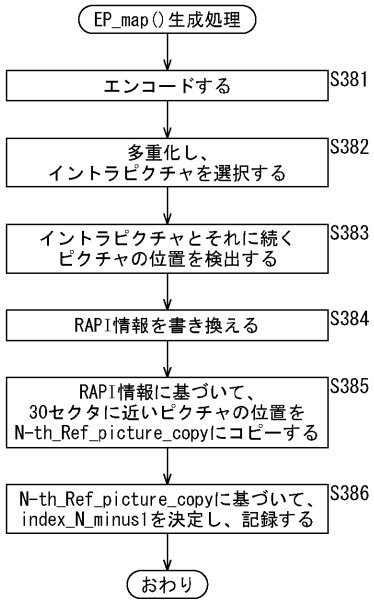
【図６０】
図60



【図６１】
図61

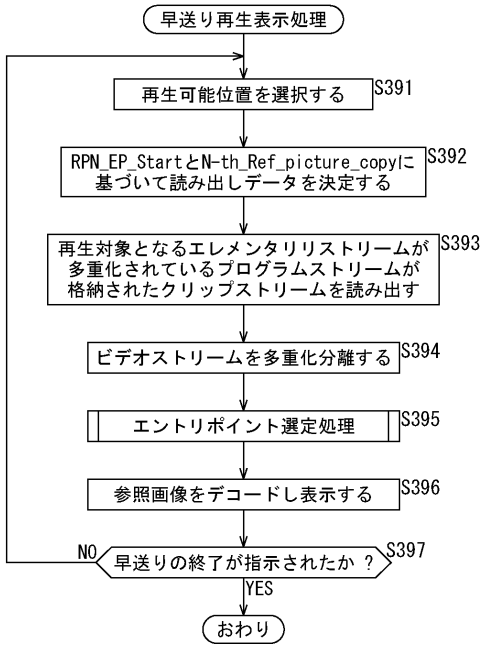


【図６２】
図62



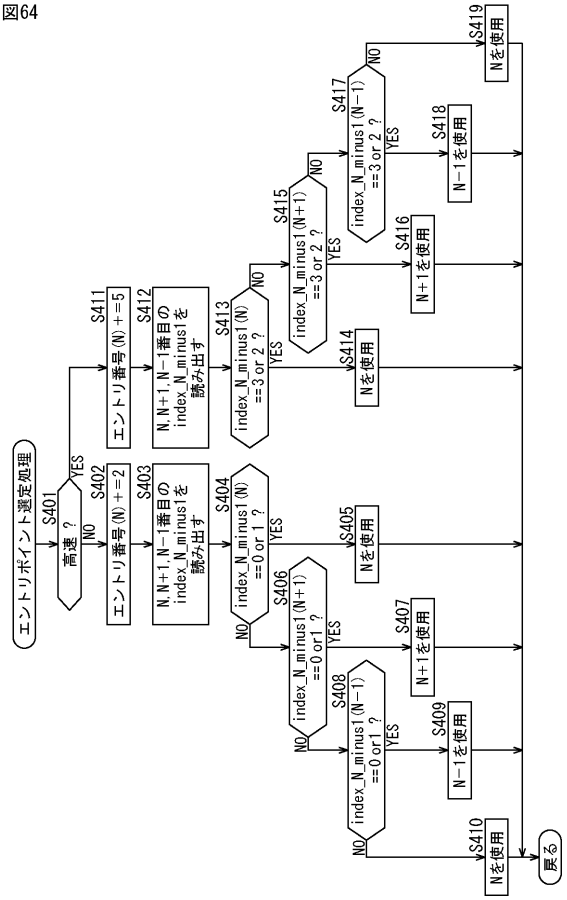
【図６３】

図63



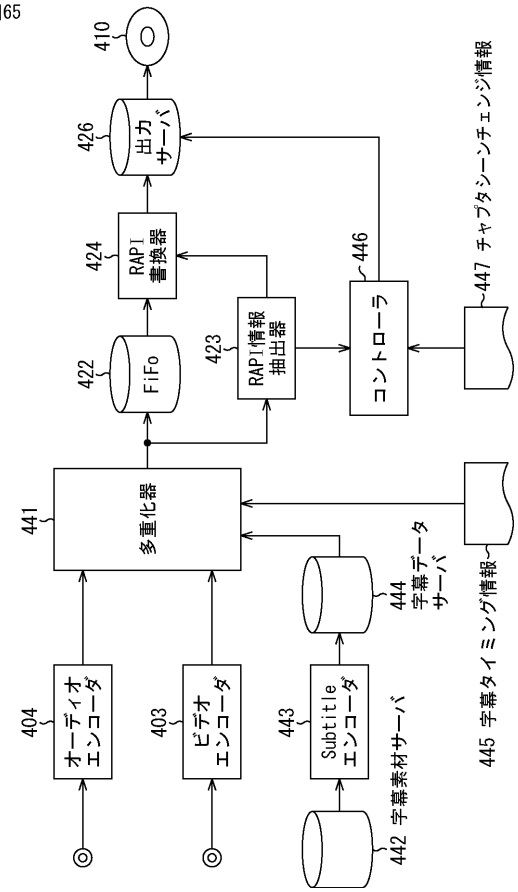
【図６４】

図64



【図６５】

図65



【図６６】

図66

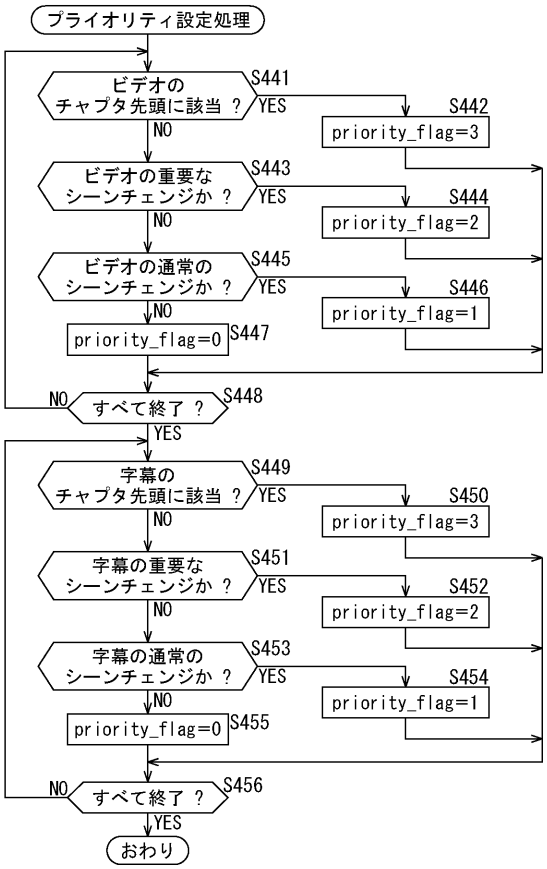
EP_map()

Syntax	No.of bits	Mnemonic
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++){		
priority_flag	2	uimsbf
reserved_for_future_use	14	bslbf
PTS_EP_start	32	uimsbf
RPN_EP_start	32	uimsbf
}		
}		
}		

【図６７】
図67

priority_flag	Videoストリームの場合	Subtitleストリームの場合
3	チャプタ先頭	チャプタ先頭
2	重要なシーンチェンジ	重要なシーンチェンジ
1	通常のシーンチェンジ	通常のシーンチェンジ
0	その他全てのエントリ	その他全てのエントリ

【図６８】
図68



フロントページの続き

(72)発明者　各務　辰哉
東京都港区南青山二丁目６番２１号　株式会社ソニー・コンピュータエンタテインメント内
(72)発明者　上田　暁彦
東京都港区南青山二丁目６番２１号　株式会社ソニー・コンピュータエンタテインメント内
(72)発明者　井原　宏ニ
東京都港区南青山二丁目６番２１号　株式会社ソニー・コンピュータエンタテインメント内
(72)発明者　内海　秀介
東京都港区南青山二丁目６番２１号　株式会社ソニー・コンピュータエンタテインメント内

審査官　早川　卓哉

(56)参考文献　特開２００２－１７１４９０（ＪＰ，Ａ）
特開平１０－３２０９１４（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ１１Ｂ　２７／００　－　２７／３４
Ｇ１１Ｂ　２０／１０
Ｇ１１Ｂ　２０／１２
Ｈ０４Ｎ　５／９１
Ｈ０４Ｎ　５／７６