

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B1)

(11) 特許番号

特許第3803366号
(P3803366)

(45) 発行日 平成18年8月2日(2006.8.2)

(24) 登録日 平成18年5月12日(2006.5.12)

(51) Int. Cl.	F I
HO4N 5/93 (2006.01)	HO4N 5/93 Z
G11B 20/12 (2006.01)	G11B 20/12
G11B 20/10 (2006.01)	G11B 20/10 321Z
HO4N 5/92 (2006.01)	G11B 20/12 103
HO4N 5/85 (2006.01)	G11B 20/10 A
請求項の数 20 (全 49 頁) 最終頁に続く	

(21) 出願番号	特願2005-518547 (P2005-518547)	(73) 特許権者	000005821
(86) (22) 出願日	平成16年4月27日(2004.4.27)		松下電器産業株式会社
(86) 国際出願番号	PCT/JP2004/006074		大阪府門真市大字門真1006番地
(87) 国際公開番号	W02004/098193	(74) 代理人	100090446
(87) 国際公開日	平成16年11月11日(2004.11.11)		弁理士 中島 司朗
審査請求日	平成17年8月23日(2005.8.23)	(72) 発明者	マクロッサン ジョセフ
(31) 優先権主張番号	60/465,972		アメリカ合衆国、カリフォルニア州 91
(32) 優先日	平成15年4月28日(2003.4.28)		501、#シー バーバンク、サン ホセ
(33) 優先権主張国	米国 (US)		アベニュー 472
		(72) 発明者	岡田 智之
			奈良県奈良市富雄元町1-8-19-30
			3
		(72) 発明者	小川 智輝
			兵庫県尼崎市西難波町3-28-9
		最終頁に続く	

(54) 【発明の名称】 記録媒体、再生装置、記録方法、再生方法、プログラム、集積回路

(57) 【特許請求の範囲】

【請求項1】

データの格納に用いられる記録媒体であって、

前記データは、ビデオストリームと、グラフィクスストリームとを多重してなるデジタルストリームを含み、

前記ビデオストリームは、動画像を表すデータであり、動画像を構成する複数ピクチャからなり、

前記グラフィクスストリームは、複数のディスプレイセットからなり、

個々のディスプレイセットは、1画面分のグラフィクスを構成するデータの集合であり、

前記複数のディスプレイセットのうち先頭のタイプはエポックススタートのディスプレイセットであり、

前記エポックススタートのディスプレイセットはグラフィクスデータと、前記グラフィクスデータの描画領域を定義するウィンドウ定義データとを含み、

前記グラフィクスデータは、前記ピクチャに合成されるグラフィクスを表し、

前記ウィンドウ定義データは、前記ピクチャとの合成のためのプレーンメモリにおけるウィンドウの位置、縦幅、横幅を示し、

前記エポックススタートのディスプレイセットに後続するディスプレイセットは、前記ウィンドウの範囲内においてグラフィクス表示を行う、

データを格納することを特徴とする記録媒体。

【請求項 2】

前記ウィンドウの縦幅、横幅は、

前記複数ディスプレイセットによるウィンドウのアップデートレートと、ビデオストリームにおけるピクチャのフレームレートとの比率に基づいて定められる

ことを特徴とする請求項 1 記載の記録媒体。

【請求項 3】

各ディスプレイセットによるウィンドウのアップデートレートと、ピクチャの表示レートとの比率を、"x"とした場合、前記ウィンドウの縦幅、横幅は、ピクチャに対応する平面の $1/x$ の大きさになるように定められる

ことを特徴とする請求項 2 記載の記録媒体。

10

【請求項 4】

各ディスプレイセットは、パケットに格納された制御データを含み、パケットは、デコードタイムスタンプと、プレゼンテーションタイムスタンプとを含み、

デコードタイムスタンプは、再生時間軸において、対応するディスプレイセットの始点は何時であるかを示し、

プレゼンテーションタイムスタンプは、再生時間軸において、そのディスプレイセットの終点は何時であり、そのディスプレイセットによるグラフィクス表示を何時実行するかを示す

ことを特徴とする請求項 1 記載の記録媒体。

【請求項 5】

前記グラフィクスストリームは、エポックスタートのディスプレイセットを2以上含み、ウィンドウ定義データにより定義されるウィンドウは、

再生時間軸において、一のエポックスタートのディスプレイセットから、その直後のエポックスタートのディスプレイセットの前までの間、有効になる

ことを特徴とする請求項 4 記載の記録媒体。

20

【請求項 6】

前記グラフィクスストリームにおいて、前記エポックスタートのディスプレイセットに後続するディスプレイセットが、ノーマルケースのディスプレイセットである場合、

ノーマルケースのディスプレイセットは、前記グラフィクスデータ、及び、ウィンドウ定義データは含まず、制御データのみを含むディスプレイセットであり、

ノーマルケースのディスプレイセットにおける制御データは、エポックスタートのディスプレイセット内のウィンドウ定義データで定義されるウィンドウにおいて、エポックスタートのディスプレイセット内のグラフィクスデータを用いたグラフィクス表示を行うように再生装置に命じる、

ことを特徴とする請求項 1 記載の記録媒体。

30

【請求項 7】

前記再生装置は、動画像との合成のためのプレーンメモリの他に、オブジェクトバッファを有しており、

前記オブジェクトバッファは、エポックスタートのディスプレイセット内のグラフィクスデータを復号することにより得られたグラフィクスオブジェクトを格納しておくバッファであり、

前記ノーマルケースのディスプレイセットに存在する制御データは、クロップ情報を含み、オブジェクトバッファに存在するグラフィクスオブジェクト全体のうち、どの部分を切り出して、プレーンメモリに転送するかを示す、ことを特徴とする請求項 6 記載の記録媒体。

40

【請求項 8】

前記ノーマルケースのディスプレイセットに存在する制御データは、クロップ情報によりグラフィクスオブジェクトから切り出されたグラフィクスの一部分をウィンドウ内のどこに表示させるかを示す位置情報を含み、

クロップ情報で示される切り出し枠は、ウィンドウにおいて位置情報により指定される

50

位置に存在する

ことを特徴とする請求項 8 記載の記録媒体。

【請求項 9】

ビデオストリーム、グラフィクスストリームが多重化されたデジタルストリームの再生に用いられる再生装置であって、

ビデオストリームをデコードして複数のピクチャからなる動画像を得るビデオデコーダと、

グラフィクスデコーダと、

グラフィクスと動画像との合成のためのプレーンメモリとを備え、

前記グラフィクスストリームは、複数のディスプレイセットからなり、

個々のディスプレイセットは、1画面分のグラフィクス表示を構成するデータの集合であり、

前記複数のディスプレイセットのうち先頭のタイプはエポックスタートのディスプレイセットであり、前記エポックスタートのディスプレイセットはグラフィクスデータと、前記グラフィクスデータの描画領域を定義するウィンドウ定義データとを含み、

前記グラフィクスデータは、前記ピクチャに合成されるグラフィクスを表し、

前記ウィンドウ定義データは、前記ピクチャとの合成のためのプレーンメモリにおけるウィンドウの位置、縦幅、横幅を示し、

前記グラフィクスデコーダは、前記エポックスタートのディスプレイセットに後続するディスプレイセットとにおけるグラフィクス表示を、前記ウィンドウ定義データで定義されたウィンドウの範囲内で実行する

ことを特徴とする再生装置。

【請求項 10】

前記ウィンドウの縦幅、横幅は、前記複数グラフィクスデコーダによるウィンドウのアップデートレートと、ビデオデコーダにおけるピクチャのフレームレートとの比率に基づいて定められる

ことを特徴とする請求項 9 記載の再生装置。

【請求項 11】

各ディスプレイセットによるウィンドウのアップデートレートと、ピクチャの表示レートとの比率を、"x"とした場合、前記ウィンドウの縦幅、横幅は、ピクチャに対応する平面の1/xの大きさになるように定められる

ことを特徴とする請求項 10 記載の再生装置。

【請求項 12】

前記エポックスタートタイプのディスプレイセット内のグラフィクスデータは、圧縮状態にあるグラフィクスであり、

前記グラフィクスデコーダは、

圧縮状態にあるグラフィクスをデコードするプロセッサと、

ウィンドウをクリアする処理、及び、ウィンドウに対するグラフィクス書き込みの処理を実行する制御部とを備える

ことを特徴とする請求項 9 記載の再生装置。

【請求項 13】

各ディスプレイセットは、パケットに格納された制御データを含み、

パケットは、デコードタイムスタンプと、プレゼンテーションタイムスタンプとを含み、

デコードタイムスタンプは、再生時間軸において、対応するディスプレイセットの始点は何時であるかを示し、

プレゼンテーションタイムスタンプは、再生時間軸において、そのディスプレイセットの終点は何時であるかを示し、そのディスプレイセットによるグラフィクス表示を何時実行するかを示し、

前記制御部は、ウィンドウをクリアする処理を、デコードタイムスタンプに示される時

10

20

30

40

50

点に開始し、

ウィンドウに対するグラフィクス書き込み処理を、プレゼンテーションタイムスタンプに示される時点までに完了する

ことを特徴とする請求項 1 2 記載の再生装置。

【請求項 1 4】

前記グラフィクスストリームは、エポックスタートのディスプレイセットを2以上含み、

前記制御部は、ウィンドウ定義データにより定義されるウィンドウを、再生時間軸において、一のエポックスタートのディスプレイセットから、その直後のエポックスタートのディスプレイセットの前までの間、有効にする

ことを特徴とする請求項 1 3 記載の再生装置。

10

【請求項 1 5】

前記グラフィクスデコーダは、動画像との合成のためのプレーンメモリの他に、オブジェクトバッファを備え、

前記オブジェクトバッファは、エポックスタートのディスプレイセット内のグラフィクスデータを復号することにより得られたグラフィクスオブジェクトを格納しておくバッファであり、

前記グラフィクスストリームにおいて、前記エポックスタートのディスプレイセットに後続するディスプレイセットが、ノーマルケースのディスプレイセットである場合、

ノーマルケースのディスプレイセットは、前記グラフィクスデータ、及び、ウィンドウ定義データは含まず、制御データのみを含むディスプレイセットであり、

20

前記グラフィクスデコーダにおける制御部は、オブジェクトバッファからグラフィクスオブジェクトを読み出して、エポックスタートのディスプレイセット内のウィンドウ定義データで定義されるウィンドウ内に書き込む処理を、ノーマルケースのディスプレイセットにおける制御データに基づいて実行する

ことを特徴とする請求項 1 2 記載の再生装置。

【請求項 1 6】

前記ノーマルケースのディスプレイセットに存在する制御データは、クロップ情報を含み、

前記制御部は、オブジェクトバッファに存在するグラフィクスオブジェクト全体のうち、クロップ情報示にされる部分を切り出して、プレーンメモリに転送する

30

ことを特徴とする請求項 1 5 記載の再生装置。

【請求項 1 7】

記録媒体の記録方法であって、

アプリケーションデータを作成するステップと、

作成したデータを記録媒体に記録するステップとを有し、

前記アプリケーションデータは、

ビデオストリームと、グラフィクスストリームとを多重してなるデジタルストリームを含み、

前記ビデオストリームは、動画像を表すデータであり、動画像を構成する複数ピクチャからなり、

40

前記グラフィクスストリームは、複数のディスプレイセットからなり、

個々のディスプレイセットは、1画面分のグラフィクス表示を構成するデータの集合であり、

前記複数のディスプレイセットのうち先頭のタイプはエポックスタートのディスプレイセットであり、前記エポックスタートのディスプレイセットはグラフィクスデータと、前記グラフィクスデータの描画領域を定義するウィンドウ定義データとを含み、

前記グラフィクスデータは、前記ピクチャに合成されるグラフィクスを表し、

前記ウィンドウ定義データは、前記ピクチャとの合成のためのプレーンメモリにおけるウィンドウの位置、縦幅、横幅を示し、

前記エポックスタートのディスプレイセットに後続するディスプレイセットは、前記ウ

50

インドウの範囲内においてグラフィクス表示を行う、
ことを特徴とする前記記録媒体の記録方法。

【請求項 18】

ビデオストリーム、グラフィクスストリームが多重化されたデジタルストリームの再生に
用いられるプログラムであって、

ビデオストリームをデコードして複数のピクチャからなる動画像を得るビデオデコード
と、

グラフィクスデコードとをコンピュータに行わせるものであり、

前記グラフィクスストリームは、複数のディスプレイセットからなり、

個々のディスプレイセットは、1画面分のグラフィクス表示を構成するデータの集合で
あり、

前記複数のディスプレイセットのうち先頭のタイプはエポックスタートのディスプレイ
セットであり、前記エポックスタートのディスプレイセットはグラフィクスデータと、描
画領域を定義するウィンドウ定義データとを含み、

前記グラフィックスデータは、前記ピクチャに合成されるグラフィックスを表し、

前記ウィンドウ定義データは、前記ピクチャとの合成のためのプレーンメモリにおける
ウィンドウの位置、縦幅、横幅を示し、

グラフィクスデコードにおいて、

前記エポックスタートのディスプレイセットと、これに後続するディスプレイセットと
におけるグラフィクス表示を、ウィンドウ定義データで定義されたウィンドウの範囲内で
実行するよう、コンピュータに命じる

ことを特徴とするプログラム。

【請求項 19】

ビデオストリーム、グラフィクスストリームが多重化されたデジタルストリームについて
の再生方法であって、

前記再生は、

ビデオストリームをデコードして複数のピクチャからなる動画像を得るビデオデコード
と、

グラフィクスデコードとを含み、

前記グラフィクスストリームは、複数のディスプレイセットからなり、

個々のディスプレイセットは、1画面分のグラフィクス表示を構成するデータの集合で
あり、

前記複数のディスプレイセットのうち先頭のタイプはエポックスタートのディスプレイ
セットであり、前記エポックスタートのディスプレイセットはグラフィクスデータと、前
記グラフィクスデータの描画領域を定義するウィンドウ定義データとを含み、

前記グラフィックスデータは、前記ピクチャに合成されるグラフィックスを表し、

前記ウィンドウ定義データは、前記ピクチャとの合成のためのプレーンメモリにおける
ウィンドウの位置、縦幅、横幅を示し、

グラフィクスデコードにおいてエポックスタートのディスプレイセットと、これに後続
するディスプレイセットとにおけるグラフィクス表示を、ウィンドウ定義データで定義さ
れたウィンドウの範囲内で実行する

ことを特徴とする再生方法。

【請求項 20】

ビデオストリーム、グラフィクスストリームが多重化されたデジタルストリームについて
の集積回路であって、

ビデオストリームをデコードして複数のピクチャからなる動画像を得るビデオデコーダ
と、

グラフィクスデコーダと、

グラフィックスを動画像と合成させるための平面に対応するプレーンメモリとを備え、

前記グラフィクスストリームは、複数のディスプレイセットからなり、

10

20

30

40

50

個々のディスプレイセットは、1画面分のグラフィクス表示を構成するデータの集合であり、

前記複数のディスプレイセットのうち先頭のタイプはエポックススタートのディスプレイセットであり、前記エポックススタートのディスプレイセットはグラフィクスデータと、前記グラフィクスデータの描画領域を定義するウィンドウ定義データとを含み、

前記グラフィクスデータは、前記ピクチャに合成されるグラフィックスを表し、

前記ウィンドウ定義データは、前記ピクチャとの合成のためのプレーンメモリにおけるウィンドウの位置、縦幅、横幅を示し、

グラフィクスデコーダは、エポックススタートのディスプレイセットと、これに後続するディスプレイセットとにおけるグラフィクス表示を、ウィンドウ定義データで定義されたウィンドウの範囲内で実行する

ことを特徴とする集積回路。

【発明の詳細な説明】

【技術分野】

【0001】

BD-ROM等の記録媒体、再生装置に関し、特に、ビデオストリームと、グラフィクスストリームとを多重してなるデジタルストリームを再生して字幕表示を行う技術に関する。

【背景技術】

【0002】

グラフィクスストリームの描画により実現される字幕表示は、ある言語圏で制作された映画作品を、他の言語圏の人々に鑑賞してもらうための、必要不可欠な手段である。字幕表示を実現する先行技術には、ETSI EN 300 743標準規格(ETSI:European Telecommunication Standards Institute)におけるピクセルバッファのメモリ割り当て方式がある。ここでピクセルバッファとは、伸長されたグラフィックスを一旦格納しておくメモリであり、再生時において再生装置は、このメモリ内のグラフィックスを、“グラフィックスプレーン”と呼ばれる表示用メモリに、書き込む。これによりグラフィックスは描画される。上述したメモリ割り当て方式は、ピクセルバッファに“Region”を定義し、伸長されたグラフィックスのうち、この“Region”に包含される部分をグラフィックスプレーンに書き込むというものである。例えば、「Good bye・・・」という字幕がピクセルバッファに存在しており、このうち「Go」の部分を包含するよう“Region”の位置・大きさを定義すれば、この「Go」の部分がグラフィックスプレーンに書き込まれて画面に表示される。また「Good」を包含するよう“Region”を定義すれば、この「Good」の部分が画面に表示される。

【0003】

“Region”の定義と、グラフィックスプレーンへの書き込みとを繰り返し行えば、字幕「Good bye・・・」は、「Go」「Good」「Good bye」「Good bye・・・」というように、余韻をもって徐々に画面に現れる。このような描画により、ワイプインという表示効果を実現することができる。これらの技術を記載した文献には、以下の非特許文献に掲げたものがある。

【非特許文献1】“Digital Video Broadcasting(DVB) Subtitling systems;Final draft ETSI EN 300 743” ETSI STANDARDS,EUROPEAN TELECOMMUNICATION STANDARDS INSTITUTE, SOPHIA-ANTIPOLIS,FR

【発明の開示】

【発明が解決しようとする課題】

【0004】

ところで従来のETSI EN 300 743標準規格には、グラフィックスプレーンに対する書き込み負荷が大きい場合に、グラフィックスの表示と、ピクチャの表示との同期をどのように保証するかという考えが全くなかった。グラフィックスプレーンには、非圧縮状態のグラフィックスが書き込まれるので、グラフィックスの解像度が高ければ高い程、書き込み負荷が大きくなる。例えば、BD-ROMが予定しているグラフィックス(1920×1080という高解像度をもつグラフィックス)を描画しようとする、グラフィックスプレーンに書き込むべきグラ

10

20

30

40

50

フィクスのサイズは最大2Mバイトになる。このような、多大なサイズをもつグラフィックスを、ピクチャの表示と同期させるには、グラフィックスプレーンへの書き込みに多大なバンド幅が必要となる。しかしグラフィックスプレーンへの書き込みに、高いバンド幅を要求することは、再生装置の低コスト化の妨げになる。仮に再生装置が常に合理的な書き込みを行うなら、グラフィックスプレーン書き込み時のバンド幅を低下させることができる。ここで合理的な書き込みとは、前回の表示内容との差分のみ、グラフィックスプレーンに書き込んでゆくという書き込みをいう。しかし、再生装置に常に合理的書き込み手法を要求するのも、ソフトウェアの実装の自由度を奪う結果になる。

【0005】

このように、グラフィックスプレーンへの書き込み負荷が多大であると、高いバンド幅での動作や、合理的な手法での書き込みを再生装置に強いることになり、再生装置の製品開発を著しく阻害することになる。

本発明の目的は、グラフィックスプレーンに書き込むべきデータ量が多大であっても、ピクチャの表示と同期したグラフィックスのアップデートを実現することができる記録媒体を提供することである。

【課題を解決するための手段】

【0006】

上記目的を達成するため、本発明にかかる記録媒体は、ビデオストリームと、グラフィックスストリームとを多重してなるデジタルストリームを含み、前記ビデオストリームは、動画像を表すデータであり、動画像を構成する複数ピクチャからなり、前記グラフィックスストリームは、複数のディスプレイセットからなり、個々のディスプレイセットは、1画面分のグラフィックスを構成するデータの集合であり、前記複数のディスプレイセットのうち先頭のタイプはエポックスタートのディスプレイセットであり、前記エポックスタートのディスプレイセットはグラフィックスデータと、前記グラフィックスデータの描画領域を定義するウィンドウ定義データとを含み、前記グラフィックスデータは、前記ピクチャに合成されるグラフィックスを表し、前記ウィンドウ定義データは、前記ピクチャとの合成のためのプレーンメモリにおけるウィンドウの位置、縦幅、横幅を示し、前記エポックスタートのディスプレイセットに後続するディスプレイセットは、前記ウィンドウの範囲内においてグラフィックス表示を行う、ことを特徴としている。

【発明の効果】

【0007】

ピクチャに対応する平面(Plane)の一部をグラフィックスの表示のためのウィンドウとして指定するので、再生装置は、プレーン全体のグラフィックス描画を行う必要はない。ある限られた大きさのウィンドウに対してのみ、グラフィックス描画を行えばよい。表示用の平面のうち、ウィンドウ以外の部分の描画を省くことができるので、再生装置側のソフトウェアの負担は遥かに軽くなる。

【0008】

グラフィックスのアップデートがワーストケースで行われる場合であっても、ピクチャとの同期が保証されるような大きさにウィンドウを定めておけば、オーサリングを行う制作者は、あらゆる再生装置での同期保証を実現することができる。

またウィンドウ情報によりウィンドウの位置や大きさを規定できるので、ピクチャ上の余白にあたる位置に、ウィンドウが現れるようオーサリング時に調整しておくことができる。ピクチャ内の絵柄の邪魔にならないように、グラフィックスを表示させることができるので、ピクチャの絵柄に時間的な変動があっても、グラフィックスの見易さを維持することができる。これにより映画作品の品質を高めることができる。

【0009】

ここでワーストケースでのアップデートとは、全クリア+再描画という最も効率が悪い処理でグラフィックスがアップデートされることをいう。ワーストケースを想定して、ウィンドウの大きさを決める場合、前記ウィンドウの縦幅、横幅は、ピクチャに対応する平面の1/xの大きさになるように定め、xは、ウィンドウのアップデートレートと、ピクチャ

10

20

30

40

50

の表示レートとの比率に基づく実数とすることが望ましい。

【 0 0 1 0 】

このようにウィンドウの大きさを定めれば、再生装置において、グラフィックスプレーンへの書き込みに必要なバンド幅は、ある決まった値になる。ミニマムスタンダードとして、このバンド幅を満たすよう、再生装置を構成すれば、ソフトウェアの実装がどのようなものであっても、ピクチャとの同期を実現することができる。

このように再生装置構成のためのミニマムスタンダードを提供することができるので、かかる転送レートというミニマムスタンダードさえ満たせば、その他の実装は、開発者が自由に定めることができるので、再生装置の開発にあたっての自由度が高まる。

【 発明を実施するための最良の形態 】

10

【 0 0 1 1 】

(第 1 実施形態)

以降、本発明に係る記録媒体の実施形態について説明する。先ず始めに、本発明に係る記録媒体の実施行為のうち、使用行為についての形態を説明する。図 1 は、本発明に係る記録媒体の、使用行為についての形態を示す図である。図 1 において、本発明に係る記録媒体は、BD-ROM 1 0 0 である。このBD-ROM 1 0 0 は、再生装置 2 0 0、テレビ 3 0 0、リモコン 4 0 0 により形成されるホームシアターシステムに、映画作品を供給するという用途に供される。

【 0 0 1 2 】

以上が本発明に係る記録媒体の使用形態についての説明である。

20

続いて本発明に係る記録媒体の実施行為のうち、生産行為についての形態について説明する。本発明に係る記録媒体は、BD-ROMの応用層に対する改良により実施することができる。図 2 は、BD-ROMの構成を示す図である。

本図の第 4 段目にBD-ROMを示し、第 3 段目にBD-ROM上のトラックを示す。本図のトラックは、BD-ROMの内周から外周にかけて螺旋状に形成されているトラックを、横方向に引き伸ばして描画している。このトラックは、リードイン領域と、ボリウム領域と、リードアウト領域とからなる。本図のボリウム領域は、物理層、ファイルシステム層、応用層というレイヤモデルをもつ。ディレクトリ構造を用いてBD-ROMの応用層フォーマット(アプリケーションフォーマット)を表現すると、図中の第 1 段目のようになる。本図に示すようにBD-ROMには、ROOTディレクトリの下にBDMVディレクトリがあり、BDMVディレクトリの配下には、AVClipを格納したファイル(XXX.M2TS)、AVClipの管理情報を格納したファイル(XXX.CLPI)、AVClipにおける論理的な再生経路(PL)を定義したファイル(YYY.MPLS)が存在する。本図に示すようなアプリケーションフォーマットを作成することにより、本発明に係る記録媒体は生産される。尚、XXX.M2TS、XXX.CLPI、YYY.MPLSといったファイルが、それぞれ複数存在する場合は、BDMVディレクトリの配下に、STREAMディレクトリ、CLIPINFディレクトリ、PLAYLISTディレクトリという3つのディレクトリを設け、STREAMディレクトリにXXX.M2TSと同じ種別のファイルを、CLIPINFディレクトリにXXX.CLPIと同じ種別のファイルを、PLAYLISTディレクトリにYYY.MPLSと同じ種別のファイルを格納することが望ましい。

30

【 0 0 1 3 】

40

このアプリケーションフォーマットにおけるAVClip(XXX.M2TS)について説明する。

AVClip(XXX.M2TS)は、MPEG-TS(Transport Stream)形式のデジタルストリームであり、ビデオストリーム、1つ以上のオーディオストリーム、プレゼンテーショングラフィックスストリームを多重化することで得られる。ビデオストリームは映画の動画部分を、オーディオストリームは映画の音声部分を、プレゼンテーショングラフィックスストリームは、映画の字幕をそれぞれ示している。図 3 は、AVClipがどのように構成されているかを模式的に示す図である。

【 0 0 1 4 】

AVClipは(中段)、複数のビデオフレーム(ピクチャpj1,2,3)からなるビデオストリーム、複数のオーディオフレームからなるオーディオストリームを(上 1 段目)、PESパケット

50

列に変換し(上2段目)、更にTSパケットに変換し(上3段目)、同じくプレゼンテーショングラフィックスストリーム(下1段目)を、PESパケット列に変換し(下2段目)、更にTSパケットに変換して(下3段目)、これらを多重化することで構成される。

【0015】

本図において多重されるプレゼンテーショングラフィックスストリームは1つであるが、BD-ROMが多言語対応型である場合、その言語毎のプレゼンテーショングラフィックスストリームがAVClipに多重される。かかる過程を経て生成されたAVClipは、通常のコンピュータファイル同様、複数のエクステンツに分割され、BD-ROM上の領域に記録される。

続いてプレゼンテーショングラフィックスストリームについて説明する。図4(a)は、プレゼンテーショングラフィックスストリームの構成を示す図である。第1段目は、AVClipを構成するTSパケット列を示す。第2段目は、グラフィックスストリームを構成するPESパケット列を示す。第2段目におけるPESパケット列は、第1段目におけるTSパケットのうち、所定のPIDをもつTSパケットからペイロードを取り出して、連結することにより構成される。

【0016】

第3段目は、グラフィックスストリームの構成を示す。グラフィックスストリームは、PCS(Presentation Composition Segment)、PDS(Palette Definition Segment)、ODS(Object Definition Segment)、END(END of Display Set Segment)と呼ばれる機能セグメントからなる。これらの機能セグメントのうち、PCSは、画面構成セグメントと呼ばれ、WDS、PDS、ODS、ENDは定義セグメントと呼ばれる。PESパケットと機能セグメントとの対応関係は、1対1の関係、1対多の関係である。つまり機能セグメントは、1つのPESパケットに変換されてBD-ROMに記録されるか、又は、フラグメント化され、複数PESパケットに変換されてBD-ROMに記録される。

【0017】

図4(b)は、機能セグメントを変換することで得られるPESパケットを示す図である。図4(b)に示すようにPESパケットは、パケットヘッダと、ペイロードとからなり、このペイロードが機能セグメント実体にあたる。またパケットヘッダには、この機能セグメントに対応するDTS、PTSが存在する。尚以降の説明では、機能セグメントが格納されるPESパケットのヘッダ内に存在するDTS及びPTSを、機能セグメントのDTS及びPTSとして扱う。

【0018】

これら様々な種別の機能セグメントは、図5のような論理構造を構築する。図5は、様々な種別の機能セグメントにて構成される論理構造を示す図である。本図は第3段目に機能セグメントを、第2段目にDisplay Setを、第1段目にEpochをそれぞれ示す。

第2段目のDisplay Set(DSと略す)とは、グラフィックスストリームを構成する複数機能セグメントのうち、一画面分のグラフィックスを構成するものの集合をいう。図中の破線は、第3段目の機能セグメントが、どのDSに帰属しているかという帰属関係を示す。PCS-WDS-PDS-ODS-ENDという一連の機能セグメントが、1つのDSを構成していることがわかる。再生装置は、このDSを構成する複数機能セグメントをBD-ROMから読み出せば、一画面分のグラフィックスを構成することができる。

【0019】

第1段目のEpochとは、AVClipの再生時間軸上においてメモリ管理の連続性をもっている一つの期間、及び、この期間に割り当てられたデータ群をいう。ここで想定しているメモリとは、一画面分のグラフィックスを格納しておくためのグラフィックスプレーン、伸長された状態のグラフィックスデータを格納しておくためのオブジェクトバッファである。これらについてのメモリ管理に、連続性があるというのは、このEpochにあたる期間を通じてこれらグラフィックスプレーン及びオブジェクトバッファのフラッシュは発生せず、グラフィックスプレーン内のある決められた矩形領域内でのみ、グラフィックスの消去及び再描画が行われることをいう(ここでフラッシュとは、プレーン及びバッファの格納内容を全部クリアしてしまうことである。)。この矩形領域の縦横の大きさ及び位置は、Epochにあ

10

20

30

40

50

たる期間において、終始固定されている。グラフィックスプレーンにおいて、この固定化された領域内で、グラフィックスの消去及び再描画を行っている限り、映像とグラフィックスとの同期が保障される。つまりEpochは、映像 - グラフィックスの同期の保障が可能な再生時間軸上の一単位ということができる。グラフィックスプレーンにおいて、グラフィックスの消去・再描画を行うべき領域を変更したい場合は、再生時間軸上においてその変更時点を定義し、その変更時点以降を、新たなEpochにせねばならない。この場合、2つのEpochの境界では、映像 - グラフィックスの同期は保証されない。

【 0 0 2 0 】

Epochにおける字幕の位置関係にたとえば、再生時間軸上において、画面上のある決まった矩形領域内に字幕が出現している期間が、Epochということができる。図6は、字幕の表示位置と、Epochとの関係を示す図である。本図では、動画の各ピクチャの絵柄に応じて字幕の位置を変更するという配慮がなされている。つまり5つの字幕「本当は」「ウソだった」「あなたが」「ずっと」「好きだった」のうち、3つの字幕「本当は」「ウソだった」「あなたが」は画面の下側に、「ずっと」「好きだった」は画面の上側に配置されている。これは画面の見易さを考え、画面中の余白にあたる位置に字幕を配置することを意図している。かかる時間的な変動がある場合、AVClipの再生時間軸において、下側の余白に字幕が出現している期間が1つのEpoch1、上側の余白に字幕が出現している期間が別のEpoch2になる。これら2つのEpochは、それぞれ独自の字幕の描画領域をもつことになる。Epoch1では、画面の下側の余白が字幕の描画領域(window1)になる。一方Epoch2では、画面の上側の余白が字幕の描画領域(window2)になる。これらのEpoch1,2において、バッファ・プレーンにおけるメモリ管理の連続性は保証されているので、上述した余白での字幕表示はシームレスに行われる。以上がEpochについての説明である。続いてDisplay Setについて説明する。

【 0 0 2 1 】

図5における破線hk1,2は、第2段目の機能セグメントが、どのEpochに帰属しているかという帰属関係を示す。Epoch Start, Acquisition Point, Normal Caseという一連のDSは、第1段目のEpochを構成していることがわかる。『Epoch Start』、『Acquisition Point』、『Normal Case』は、DSの類型である。本図におけるAcquisition Point、Normal Caseの順序は、一例にすぎず、どちらが先であってもよい。

【 0 0 2 2 】

『Epoch Start』は、「新表示」という表示効果をもたらすDSであり、新たなEpochの開始を示す。そのためEpoch Startは、次の画面合成に必要な全ての機能セグメントを含んでいる。Epoch Startは、映画作品におけるチャプター等、頭出しがなされることが判明している位置に配置される。

『Acquisition Point』は、「表示リフレッシュ」という表示効果をもたらすDisplay Setであり、先行するEpoch Startと全く同じDisplay Setをいう。Acquisition PointたるDSは、Epochの開始時点ではないが、次の画面合成に必要な全ての機能セグメントを含んでいるので、Acquisition PointたるDSから頭出しを行えば、グラフィックス表示を確実に実現することができる。つまりAcquisition PointたるDSは、Epochの途中からの画面構成を可能するという役割をもつ。

【 0 0 2 3 】

Acquisition PointたるDisplay Setは、頭出し先になり得る位置に組み込まれる。そのような位置には、タイムサーチにより指定され得る位置がある。タイムサーチとは、何分何秒という時間入力をユーザから受け付けて、その時間入力に相当する再生時点から頭出しを行う操作である。かかる時間入力は、10分単位、10秒単位というように、大まかな単位でなされるので、10分間隔の再生位置、10秒間隔の再生位置がタイムサーチにより指定され得る位置になる。このようにタイムサーチにより指定され得る位置にAcquisition Pointを設けておくことにより、タイムサーチ時のグラフィックスストリーム再生を好適に行うことができる。

【 0 0 2 4 】

『Normal Case』は、“表示アップデート”という表示効果をもたらすDSであり、前の画面合成からの差分のみを含む。例えば、あるDSvの字幕は、先行するDSuと同じ内容であるが、画面構成が、この先行するDSuとは異なる場合、PCSのみのDSv、又は、PCSのみのDSvを設けてこのDSvをNormal CaseのDSにする。こうすれば、重複するODSを設ける必要はなくなるので、BD-ROMにおける容量削減に寄与することができる。一方、Normal CaseのDSは、差分にすぎないので、Normal Case単独では画面構成は行えない。

【0025】

続いてDefinition Segment(ODS,WDS,PDS)について説明する。

『Object_Definition_Segment』は、グラフィクスオブジェクトを定義する機能セグメントである。このグラフィクスオブジェクトについて以下説明する。BD-ROMに記録されているAVClipは、ハイビジョン並みの高画質をセールスポイントにしているため、グラフィクスオブジェクトの解像度も、1920×1080画素という高精細な大きさに設定されている。1920×1080という解像度があるので、BD-ROMでは、劇場上映用の字幕の字体、つまり、手書きの味わい深い字体の字幕表示を鮮やかに再現できる。画素の色にあたっては、一画素当たりのインデックス値(赤色差成分(Cr値),青色差成分(Cb値),輝度成分Y値,透明度(T値))のビット長が8ビットになっており、これによりフルカラーの16,777,216色から任意の256色を選んで画素の色として設定することができる。グラフィクスオブジェクトによる字幕は、透明色の背景に、文字列を配置することで描画される。

【0026】

ODSによるグラフィクスオブジェクトの定義は、図7(a)に示すようなデータ構造をもってなされる。ODSは、図7(a)に示すように自身がODSであることを示す『segment_type』と、ODSのデータ長を示す『segment_length』と、EpochにおいてこのODSに対応するグラフィクスオブジェクトを一意に識別する『object_id』と、EpochにおけるODSのバージョンを示す『object_version_number』と、『last_insequence_flag』と、グラフィクスオブジェクトの一部又は全部である連続バイト長データ『object_data_fragment』とからなる。

【0027】

『object_id』は、EpochにおいてこのODSに対応するグラフィクスオブジェクトを一意に識別する。グラフィクスストリームのEpochには、同じIDをもつODSが複数含まれる。同じIDをもつ複数のODSは縦横のサイズが同じであり、オブジェクトバッファの共通の領域が割り当てられる。同じIDをもつ複数ODSのうち1つが、この領域に読み出されれば、オブジェクトバッファにおけるODSは、同じIDをもつ後続のODSにより上書きされる。このように、ビデオストリームの再生進行に伴い、オブジェクトバッファ上のODSが、同じ識別子をもつ後続のODSにて上書きされることにより、ODSにおける絵柄は、時間的な変遷を遂げる。同じIDをもつグラフィクスオブジェクトの縦横サイズを同じにすると制限は、1つのEpoch内の制限に過ぎない。あるEpochに属するグラフィクスオブジェクトと、次のEpochに属するグラフィクスオブジェクトとでは縦横のサイズが変わっても良い。

【0028】

『last_insequence_flag』、『object_data_fragment』について説明する。PESパケットのペイロードの制限から、字幕を構成する非圧縮グラフィクスが1つのODSでは格納できない場合がある。そのような場合、グラフィクスを分割することにより得られた1部分(フラグメント)がobject_data_fragmentに設定される。1つのグラフィクスオブジェクトを複数ODSで格納する場合、最後のフラグメントを除く全てのフラグメントは同じサイズになる。つまり最後のフラグメントは、それ以前のフラグメントサイズ以下となる。これらフラグメントを格納したODSは、DSにおいて同じ順序で出現する。グラフィクスオブジェクトの最後は、last_insequence_flagをもつODSにより指示される。上述したODSのデータ構造は、前のPESパケットからフラグメントを詰めてゆく格納法を前提にしているが、各PESパケットに空きが生じるように、詰めてゆくという格納法であっても良い。以上がODSの説明である。

【0029】

10

20

30

40

50

『Palette Definition Segment(PDS)』は、色変換用のパレットを定義する情報である。PDSのデータ構造を図7(b)に示す。図7(b)に示すようにPDSは、自身がPDSであることを示す『segment_type』、PDSのデータ長を示す『segment_length』、このPDSに含まれるパレットを一意に識別する『pallet_id』、EpochにおけるEpochのPDSのバージョンを示す『pallet_version_number』、各エントリーについての情報『pallet_entry』からなる。『pallet_entry』は、各エントリーにおける赤色差成分(Cr値)、青色差成分(Cb値)、輝度成分Y値、透明度(T値)を示す。

【0030】

続いてWDSについて説明する。

『window_definition_segment』は、グラフィックスプレーンの矩形領域を定義するための機能セグメントである。Epochでは、クリア及び再描画が、グラフィックスプレーンにおけるある矩形領域内で行われている場合のみ、メモリ管理に連続性が生ずることは既に述べている。このグラフィックスプレーンにおける矩形領域は“window”と呼ばれ、このWDSで定義される。図8(a)は、WDSのデータ構造を示す図である。本図に示すようにWDSは、グラフィックスプレーンにおいてウィンドウを一意に識別する『window_id』と、グラフィックスプレーンにおける左上画素の水平位置を示す『window_horizontal_position』と、グラフィックスプレーンにおける左上画素の垂直位置を示す『window_vertical_position』と、グラフィックスプレーンにおけるウィンドウの横幅を示す『window_width』と、グラフィックスプレーンにおける縦幅を示す『window_height』とを用いて表現される。

【0031】

window_horizontal_position、window_vertical_position、window_width、window_heightがとりうる値について説明する。これらが想定している座標系は、グラフィックスプレーンの内部領域であり、このグラフィックスプレーンの内部領域においてウィンドウは、縦:video_height、横:video_widthという二次元状の大きさをもつ。

window_horizontal_positionは、グラフィックスプレーンにおける左上画素の水平アドレスであるので、0~video_width-1の値をとり、window_vertical_positionは、グラフィックスプレーンにおける左上画素の垂直アドレスであるので0~video_height-1の値をとる。

【0032】

window_widthは、グラフィックスプレーンにおけるウィンドウの横幅であるので、0~video_width-window_horizontal_position-1の値をとり、window_heightは、グラフィックスプレーンにおける縦幅であるので0~video_height-window_vertical_position-1の値をとる。

WDSのwindow_horizontal_position、window_vertical_position、window_width、window_heightにより、グラフィックスプレーンの何処にウィンドウを配置するか、ウィンドウの大きさをどれだけにするかをEpoch毎に規定することができる。そのため、あるEpochに属するピクチャが表示されている期間において、ピクチャ内の絵柄の邪魔にならないように、ピクチャ上の余白にあたる位置に、ウィンドウが現れるようオーサリング時に調整しておくことができる。これによりグラフィックスによる字幕表示を見易くすることができる。WDSはEpoch毎に定義可能なので、ピクチャの絵柄に時間的な変動があっても、その変動に応じて、グラフィックスを見易く表示することができる。そのため、結果として、字幕を映像本体に組み込むのと同じレベルにまで映画作品の品質を高めることができる。

【0033】

続いて『END of Display Set Segment』について説明する。END of Display Set Segmentは、Display Setの伝送の終わりを示す指標であり、Display Setにおける機能セグメントのうち、最後のODSの直後に配置される。このEND of Display Set Segmentの内部構成は、自身がEND of Display Set Segmentであることを示す『segment_type』と、当該機能セグメントのデータ長を示す『segment_length』とからなり、これといって説明が必要な構成要素はない。故に図示は省略する。

10

20

30

40

50

【 0 0 3 4 】

以上がODS、PDS、WDS、ENDについての説明である。続いてPCSについて説明する。

PCSは、対話的な画面を構成する機能セグメントである。PCSは、図 8 (b) に示すデータ構造で構成される。本図に示すようにPCSは、『segment_type』と、『segment_length』と、『composition_number』と、『composition_state』と、『pallet_update_flag』と、『pallet_id』と、『window情報(1)～(m)』とから構成される。

【 0 0 3 5 】

『composition_number』は、0から15までの数値を用いてDisplay Setにおけるグラフィクスアップデートを識別する。どのように識別するかというと、Epochの先頭から本PCSまでにグラフィクスアップデートが存在すれば、それらグラフィクスアップデートを経由する度に、インクリメントされるというルールでcomposition_numberは設定される。

『composition_state』は、本PCSから始まるDisplay Setが、Normal Caseであるか、Acquisition Pointであるか、Epoch Startであるかを示す。

【 0 0 3 6 】

『pallet_update_flag』は、本PCSにおいてPalletOnly Display Updateがなされているかどうかを示す。PalletOnly Display Updateとは、直前のパレットのみを新たなものに切り換えることでなされるアップデートをいう。本PCSでかかるアップデートがなされれば、本フィールドは " 1 " に設定される。

『pallet_id』は、PalletOnly Display Updateに用いられるべきパレットを示す。

【 0 0 3 7 】

『window情報(1)・・・(n)』は、このPCSが属するDisplay Set内の、個々のウィンドウをどのように制御するかを示す情報である。図 8 (b) の破線wd1は、任意のwindow情報(i)の内部構成をクローズアップしている。この破線wd1に示すように、window情報(i)は、『object_id』、『window_id』、『object_cropped_flag』、『object_horizontal_position』、『object_vertical_position』、『cropping_rectangle情報(1)(2)・・・(n)』からなる。

【 0 0 3 8 】

『object_id』は、window情報(i)に対応するウィンドウ内に存在するODSの識別子を示す。

『window_id』は、本PCSにおいて、グラフィクスオブジェクトに割り当てられるべきウィンドウを示す。1つのウィンドウには最大2つのグラフィクスオブジェクトが割り当てられる。

【 0 0 3 9 】

『object_cropped_flag』は、オブジェクトバッファにおいてクロップされたグラフィクスオブジェクトを表示するか、グラフィクスオブジェクトを非表示とするかを切り換えるフラグである。 " 1 " と設定された場合、オブジェクトバッファにおいてクロップされたグラフィクスオブジェクトが表示され、 " 0 " と設定された場合、グラフィクスオブジェクトは非表示となる。

【 0 0 4 0 】

『object_horizontal_position』は、グラフィックスプレーンにおけるグラフィクスオブジェクトの左上画素の水平位置を示す。

『object_vertical_position』は、グラフィックスプレーンにおける左上画素の垂直位置を示す。

『cropping_rectangle情報(1)(2)・・・(n)』は、『object_cropped_flag』が1に設定されている場合に有効となる情報要素である。破線wd2は、任意のcropping_rectangle情報(i)の内部構成をクローズアップしている。この破線に示すようにcropping_rectangle情報(i)は、『object_cropping_horizontal_position』、『object_cropping_vertical_address』、『object_cropping_width』、『object_cropping_height』からなる。

【 0 0 4 1 】

『object_cropping_horizontal_position』は、グラフィックスプレーンにおけるクロ

10

20

30

40

50

ップ矩形の左上画素の水平位置を示す。クロップ矩形は、グラフィックスオブジェクトの一部を切り出すための枠であり、ETSI EN 300 743標準規格における "Region" に相当する。

『object_cropping_vertical_address』は、グラフィックスプレーンにおけるクロップ矩形の左上画素の垂直位置を示す。

【 0 0 4 2 】

『object_cropping_width』は、グラフィックスプレーンにおけるクロップ矩形の横幅を示す。

『object_cropping_height』は、グラフィックスプレーンにおけるクロップ矩形の縦幅を示す。

10

以上がPCSのデータ構造である。続いてPCSの具体的な記述について説明する。この具体例は、図 6 に示した字幕の表示、つまり動画の再生進行に伴い、三回のグラフィックスプレーンへの書き込みで『ほんと』 『ウソだった』 『あなたが』というように徐々に表示させるというものである。図 9 は、かかる字幕表示を実現するための記述例である。本図におけるEpochは、DS1(EPOCH Start)、DS2(Normal Case)、DS3(Normal Case)を有する。DS1は、字幕の表示枠となるwindowを定義するWDS、台詞『ほんと ウソだった あなたが』を表すODS、1つ目のPCSを備える。DS2(Normal Case)は、2つ目のPCSを有する。DS3(Normal Case)は3つ目のPCSを有する。

【 0 0 4 3 】

次に個々のPCSをどのように記述するかについて説明する。Display Setに属するWDS、PCSの記述例を図 1 0 ~ 図 1 2 に示す。図 1 0 は、DS1におけるPCSの記述例を示す図である。

20

図 1 0 において、WDSのwindow_horizontal_position、window_vertical_positionは、グラフィックスプレーンにおけるウィンドウの左上座標LP1を、window_width、window_heightは、ウィンドウの表示枠の横幅、縦幅を示す。

【 0 0 4 4 】

図 1 0 におけるクロップ情報のobject_cropping_horizontal_position、object_cropping_vertical_positionは、オブジェクトバッファにおけるグラフィックスオブジェクトの左上座標を原点とした座標系においてクロップ範囲の基準ST1を示している。そして基準点からobject_cropping_width、object_cropping_heightに示される範囲(図中の太枠部分)がクロップ範囲になる。クロップされたグラフィックスオブジェクトは、グラフィックスプレーンの座標系においてobject_cropping_horizontal_position、object_cropping_vertical_positionを基準点(左上)とした破線の範囲cp1に配置される。こうすることにより、『本当は』がグラフィックスプレーンにおけるウィンドウ内に書き込まれる。これにより字幕『本当は』は動画像と合成され表示される。

30

【 0 0 4 5 】

図 1 1 は、DS2におけるPCSの記述例を示す図である。本図におけるWDSの記述は、図 1 0 と同じなので説明を省略する。クロップ情報の記述は、図 1 0 と異なる。図 1 1 におけるクロップ情報のobject_cropping_horizontal_position、object_cropping_vertical_positionは、オブジェクトバッファ上の字幕『本当はウソだった。あなたが』のうち、『ウソだった』の左上座標を示し、object_cropping_height、object_cropping_widthは、『ウソだった』の横幅、縦幅を示す。こうすることにより、『ウソだった』がグラフィックスプレーンにおけるウィンドウ内に書き込まれる。これにより字幕『ウソだった』は動画像と合成され表示される。

40

【 0 0 4 6 】

図 1 2 は、DS3におけるPCSの記述例を示す図である。本図におけるWDSの記述は、図 1 0 と同じなので説明を省略する。クロップ情報の記述は、図 1 0 と異なる。図 1 2 におけるクロップ情報のobject_cropping_horizontal_position、object_cropping_vertical_positionは、オブジェクトバッファ上の字幕『本当はウソだった。あなたが』のうち、『あなたが』の左上座標を示し、object_cropping_height、object_cropping_widthは、『あな

50

たが』の横幅、縦幅を示す。こうすることにより、『あなたが』がグラフィックスプレーンにおけるウィンドウ内に書き込まれる。これにより字幕『あなたが』は動画像と合成され表示される。

【 0 0 4 7 】

DS1,2,3のPCSを以上のように記述することで、字幕の表示効果を実現することができる。その他の表示効果を実現するにあたっての、PCSの記述作法について説明する。

先ず初めにCut-In/Outの記述作法について説明する。図 1 3 は、時間軸に沿った連続写真的な表記で、Cut-In/Outを実行する際のDisplay Setの記述例を示す図である。

本図におけるwindow(x,y,u,v)におけるx,yは、window_vertical_position、window_horizontal_positionの値であり、u,vは、window_width、window_heightの値を示す。また本図のCropping Rectangle(a,b,c,d)におけるa,bは、object_cropping_vertical_position、object_cropping_horizontal_positionの値であり、c,dはobject_cropping_width、object_cropping_heightの値である。本図の再生時間軸の時点t11,t12,t13には、DS11,DS12,DS13が存在している。

【 0 0 4 8 】

時点t11に存在するDS11は、Composition_StateがEpoch Startに設定され、Object_Cropped_Flagが0(No Cropping Rectangle Visible)に設定されたPCS#0、グラフィックスプレーンの(100,100)の位置に、横700×縦500のwindowを宣言するWDS#0、PDS#0、字幕「Credit:」を表すODS#0、ENDを有する。

時点t12に存在しているDS12は、Composition_StateがNormal Caseであり、オブジェクトバッファの(0,0)から横600×縦400の範囲でクロップを行い(Cropping Rectangle#0(0,0,600,400))、クロップされたグラフィクスオブジェクトをグラフィックスプレーンの座標(0,0)に配置する(on window(0,0))という手順を示すPCS#1を含む。

【 0 0 4 9 】

時点t13に存在しているDS13は、Composition_StateがNormal Caseであり、クロップされたグラフィクスオブジェクトを消すよう、object_cropped_flagが0に設定されたPCS#2を含む(No Cropping Rectangle Visible)。

以上のDisplay Setにより、t11では字幕「Credit:」は非表示になっているが、t12で表示され、t13で再び非表示になる。これによりCut-In/Cut-Outという表示効果が実現される。

【 0 0 5 0 】

続いてFade-In/Outという表示効果を実現する際の、Display Setの記述作法について説明する。図 1 4 は、時間軸に沿った連続写真的な表記で、Fade-In/Outを実行する際のDisplay Setの記述例を示す図である。本図の時点t21,t22,t23,t24には、DS21,DS22,DS23,DS24が存在している。

時点t21に存在するDS21は、Composition_StateがEpoch Startに設定され、オブジェクトバッファの(0,0)から横600×縦400の範囲でクロップを行い(Cropping Rectangle#0(0,0,600,400))、クロップされたグラフィクスオブジェクトをグラフィックスプレーンの座標(0,0)に配置する(on window(0,0))という手順を示すPCS#0、グラフィックスプレーンの(100,100)の位置に、横700×縦500のwindowを宣言するWDS#0、PDS#0、字幕「Fin」を表すODS#0、ENDを有する。

【 0 0 5 1 】

時点t22に存在しているDS22は、Composition_StateがNormal CaseであるPCS#1と、PDS#1とを含む。このPDS#1は、PDS#0と同じ色彩の赤色差、青色差を示しておりながら、高い輝度値をもっているものとする。

時点t23に存在しているDS23は、Composition_StateがNormal CaseであるPCS#3と、PDS#2と、ENDとを含む。このPDS#2は、PDS#0と同じ色彩の赤色差、青色差を示しておりながら、低い輝度値をもっているものとする。

【 0 0 5 2 】

時点t24に存在するDS24は、Composition_StateがNormal Caseに設定され、Object_Crop

10

20

30

40

50

ped_Flagが0(No Cropping Rectangle Visible)に設定されたPCS、ENDを有する。

これら一連のDisplay Setは、1つ前のDisplay Setと異なるPDSを指定しているため、Epoch内の複数PCSにおいて表示されるグラフィクスオブジェクトは、徐々に輝度値が高められたり、弱められることになる。これによりFade-In/Outという表示効果の実現が可能になる。

【 0 0 5 3 】

続いてScrollingの記述作法について説明する。図 1 5 は、時間軸に沿った連続写真的な表記で、Scrollingがどのように行われるかを記述した図である。本図における表記は、図 1 3 の一例に準ずる。

AVClip再生時間軸の時点t31に存在するDS31は、Composition_StateがEpoch Startに設定され、Object_Cropped_Flagが0(No Cropping Rectangle Visible)に設定されたPCS#0、グラフィックスプレーンの(100,100)の位置に、横700×縦500のwindowを宣言するWDS#0、PDS#0、字幕「Credit: Company」を表すODS#0、ENDを有する。

10

【 0 0 5 4 】

時点t32に存在するDS32は、Composition_StateがNormal Caseであり、オブジェクトバッファにおける(0,0)から横600×縦400の範囲でクロップを行い(Cropping Rectangle#0(0,0,600,400))、クロップされたグラフィクスオブジェクトをグラフィックスプレーンの座標(0,0)に配置する(on window#0(0,0))という手順を示すPCS#1を含む。オブジェクトバッファにおいて(0,0)から横600×縦400の範囲には、「Credit:」「Company」という2段表記の字幕のうち「Credit:」が存在するので、この「Credit:」がグラフィックスプレーン

20

【 0 0 5 5 】

時点t33に存在するDS33は、Composition_StateがNormal Caseであり、オブジェクトバッファ上の座標(0,100)から横600×縦400の範囲を切り出し(Cropping Rectangle#0(0,100,600,400))、ウィンドウにおける座標(0,0)に配置する(on window#0(0,0))という手順を示すPCS#2を含む。オブジェクトバッファ上の座標(0,100)から横600×縦400の範囲には、「Credit:」「Company」という2段表記の字幕のうち、「Credit:」の下半分と、「Company」の上半分とが存在するので、これらの部分がグラフィックスプレーンに出現することになる。

【 0 0 5 6 】

時点t34に存在するDS34は、Composition_StateがNormal Caseであり、オブジェクトバッファの座標(0,200)から横600×縦400の範囲を切り出し(Cropping Rectangle#0(0,200,600,400))、ウィンドウにおける座標(0,0)に配置する(on window#0(0,0))という手順を示すPCS#3を含む。オブジェクトバッファ上の座標(0,200)から横600×縦400の範囲には、「Credit:」「Company」という2段表記の字幕のうち、「Company」が存在するので、2段表記の字幕のうち「Company」の部分が画面上に現れることになる。以上のPCSの記述により、「Credit:」「Company」という2段表記の字幕の下スクロールがなされることになる。

30

【 0 0 5 7 】

続いてWipe-In/Outの記述作法について説明する。図 1 6 は、時間軸に沿った連続写真的な表記で、Wipe-In/Outがどのように行われるかを記述した図である。本図における表記は、図 1 3 の一例に準ずる。

40

本図の時点t51に存在するDS51は、Composition_StateがEpoch Startに設定され、Object_Cropped_Flagが0(No Cropping Rectangle Visible)に設定されたPCS#0、グラフィックスプレーンの(100,100)の位置に、横700×縦500のwindowを宣言するWDS#0、PDS#0、字幕「Fin」を表すODS#0、ENDを有する。

【 0 0 5 8 】

時点t52に存在するDS52は、Composition_StateがNormal Caseであり、オブジェクトバッファの(0,0)から横600×縦400の範囲でクロップを行い(Cropping Rectangle#0(0,0,600,400))、クロップされたグラフィクスオブジェクトをグラフィックスプレーンの座標(0,0

50

)に配置する(on window#0(0,0))という手順を示すPCS#1を含む。(0,0)から横600,縦400の範囲には、「Fin」という字幕が存在するので、この「Fin」がグラフィックスプレーンに出現することになる。

【 0 0 5 9 】

時点t53に存在するDS53は、Composition_StateがNormal Caseであり、オブジェクトバッファにおける(200,0)から縦400×横400の範囲でクロップを行い(Cropping Rectangle#0(200,0,400,400))、座標(200,0)を基準にして、表示を行わせるPCS#2を含む(on window#0(200,0))。そうすると、(200,0)～(400,400)が表示範囲内になり、(0,0)～(199,400)は、非表示範囲になる。

【 0 0 6 0 】

時点t54に存在するDS54は、Composition_StateがNormal Caseであり、オブジェクトバッファにおけるグラフィックスオブジェクトの(400,0)から横200×縦400の範囲でクロップを行い(Cropping Rectangle#0(400,0,200,400))、座標(400,0)を基準にして、ウィンドウの表示を行わせる(on window#0(400,0))PCS#3を含む。そうすると、(0,0)～(399,400)は、非表示範囲になる。

【 0 0 6 1 】

グラフィックスプレーンの非表示範囲が広がるに伴い、表示範囲が狭まるのでWipe-In/Outが実現されることになる。

このようにPCSの記述次第でCut-In/Out,Fade-In/Out,Wipe-In/Out,Scrollというように多様な表示効果を実現することができるので、字幕描画に様々な工夫を凝らすことができる。

【 0 0 6 2 】

かかる表示効果実現にあたっての制約について説明する。図13～図16の表示効果のうち、Scrollingの実現には、ウィンドウクリアと、ウィンドウ再描画とが必要になる。図15のt32 - t33間で考えれば、Scrollingの実現のためには、t32における「Credit:」なるグラフィックスオブジェクトをグラフィックスプレーンから消し去るというウィンドウクリアと、「Credit:」の下半分、「Company」の上半分をグラフィックスプレーンに書き込むというウィンドウ再描画とをt32 - t33の時間間隔で実現せねばならない。この時間間隔がビデオフレームの時間間隔である場合、オブジェクトバッファと、グラフィックスプレーンとの間の転送レートはどれだけ必要であろうか。

【 0 0 6 3 】

ここでウィンドウをどれだけの大きさとするかの制限について検討する。オブジェクトバッファ - グラフィックスプレーン間の転送レートをRcとすると、ワーストケースでは、この転送レートRcでウィンドウクリアと、ウィンドウ再描画とを行わねばならない。そうするとウィンドウクリア、ウィンドウ再描画のそれぞれをRcの半分の転送レート(Rc/2)で実現せねばならない。

【 0 0 6 4 】

これらウィンドウクリア - ウィンドウ再描画をビデオフレームと同期させるには、
ウィンドウサイズ×フレームレート Rc/2

を満たす必要がある。このフレームレートが29.97であるなら、

Rcは、ウィンドウサイズ×2×29.97になる。

字幕の表示にあたっては、グラフィックスプレーン全体に対し、最低でも25%～33%程度の大きさが必要となる。ここでグラフィックスプレーンの総画素数は1920×1080であり、一画素当たりのインデックスのビット長を8ビットとすると、グラフィックスプレーンの総容量は2Mバイト(1920×1080×8)になる。

【 0 0 6 5 】

この総容量の1/4をウィンドウサイズとすると、500kバイト(=2Mバイト/4)になる。こ

10

20

30

40

50

れを上記した式に代入してRcを導けば、Rcは256Mbps(500Kバイト×2×29.97)と算出することができる。

この25%～33%という大きさであれば、256Mbpsという転送レートで字幕の表示を行っている限り、如何なる表示効果を実現する場合であっても、動画との同期を維持することができる。

【0066】

仮に、ウィンドウクリア及び再描画のレートがビデオフレームのフレームレートの1/2, 1/4でよいなら、Rcがたとえ同じであってもウィンドウサイズを2倍,4倍にすることができる。以上がwindowの大きさについての説明である。続いて、windowの位置、範囲について説明する。上述したように、Epochにおいてウィンドウの位置、範囲は一貫している。

10

Epochにおいてウィンドウの位置、範囲を一貫させておくのは以下の理由による。ウィンドウの位置・範囲を変えれば、グラフィックスプレーンに対する書込先アドレスを変えねばならず、オーバーヘッドが発生するので、かかるオーバーヘッドによりオブジェクトバッファからグラフィックスプレーンへの転送レートが低下するからである。

【0067】

ウィンドウには、グラフィックスオブジェクトの個数が制限されている。この個数制限は、デコードされたグラフィックスオブジェクトの転送にあたってのオーバーヘッドを低減する目的で設けられている。ここでのオーバーヘッドは、グラフィックスオブジェクトのエッジ部分のアドレスを設定する際に発生する。そうすれば、エッジの部分が多く存在する程、オーバーヘッドの発生回数が増える。

20

【0068】

図17は、ウィンドウ内にグラフィックスオブジェクトが4つ存在する場合と、2つの存在する場合とを対比して示す図である。グラフィックスオブジェクトが4つ存在する場合、エッジの数は8つになるのでグラフィックスオブジェクトが2つ存在する場合と比べると、エッジの数は2倍になっている。

ウィンドウにおけるグラフィックスオブジェクトの数に制限がないと、グラフィックス転送にあたって発生するオーバーヘッド数が未知数になり、転送負荷の増減が激しくなる。一方、ウィンドウにおけるグラフィックスの個数が2つまでであると、最悪4つのオーバーヘッドが発生すると見込んで転送レートを設定すればよいので、ミニマムスタンダードたる転送レートを数値化し易くなる。

30

【0069】

以上がウィンドウについての説明である。続いてこれらPCS、ODSを有したDisplay Setが、AVClipの再生時間軸上にどのように割り当てられるかについて説明する。Epochは、再生時間軸上においてメモリ管理が連続する期間であり、Epochは1つ以上のDisplay Setから構成されるので、Display SetをどうやってAVClipの再生時間軸に割り当てるかが問題になる。ここでAVClipの再生時間軸とは、AVClipに多重されたビデオストリームを構成する個々のピクチャデータのデコードタイミング、再生タイミングを規定するための想定される時間軸をいう。この再生時間軸においてデコードタイミング、再生タイミングは、90KHzの時間精度で表現される。Display Set内のPCS、ODSに付加されたDTS、PTSは、この再生時間軸において同期制御を実現すべきタイミングを示す。このPCS、ODSに付加されたDTS、PTSを用いて同期制御を行うことが、再生時間軸へのDisplay Setの割り当てである。

40

【0070】

先ず、ODSに付加されたDTS、PTSにより、どのような同期制御がなされるかについて説明する。

DTSは、ODSのデコードを開始すべき時間を90KHzの時間精度で示しており、PTSはデコード終了時刻を示す。

ODSのデコードは、瞬時には完了せず、時間的な長さをもっている。このデコード期間の開始点・終了点を明らかにしたいとの要望から、ODSについてのDTS、PTSはデコード開始時刻、デコード終了時刻を示している。

50

【 0 0 7 1 】

PTSの値はデッドラインであるので、PTSに示される時刻までにODSのデコードがなされて、非圧縮状態のグラフィックオブジェクトが、再生装置上のオブジェクトバッファに得られなければならない。

DSnに属する任意のODSjのデコード開始時刻は、90KHzの時間精度でDTS(DSn[ODSj])に示されるので、これにデコードを要する最長時間を加えた時刻が、Display SetのODSjのデコード終了時刻になる。

【 0 0 7 2 】

ODSjのサイズを " SIZE(DSn[ODSj]) "、ODSのデコードレートを " Rd " とすると、デコードに要する最長時間(秒)は、 " SIZE(DSn[ODSj])//Rd " になる。ここで " // " は、小数点以下切り上げの割り算を示す演算子である。

10

この最長時間を90KHzの時間精度に変換し、ODSjのDTSに加算することにより、PTSで示されるべきデコード終了時刻(90KHz)は算出される。

DSnに属するODSjのPTSを、数式で表すと、以下の式のようにになる。

$$PTS(DS[ODSj]) = DTS(DSn[ODSj]) + 90,000 \times (SIZE(DSn[ODSj]) // Rd)$$

そして互いに隣接する2つのODS(ODSj, ODSj+1)との間では、以下の関係を満たす必要がある。

20

$$PTS(DSn[ODSj]) \leq DTS(DSn[ODSj+1])$$

続いてPCSのDTS、PTSの設定について説明する。

【 0 0 7 3 】

PCSは、DSnにおける最初のODS(ODS1)のデコード開始時点(DTS(DSn[ODS1]))以前、及び、DSnにおける最初のPDS(PDS1)が有効になる時点(PTS(DSn[PDS1]))以前に、再生装置上のバッファにロードされねばならない。よって以下の式の関係を満たす値に、設定されねばならない。

$$DTS(DSn[PCS]) \leq DTS(DSn[ODS1])$$

30

$$DTS(DSn[PCS]) \leq PTS(DSn[PDS1])$$

そしてDSnにおけるPCSのPTSは、以下の数式から算出される。

$$PTS(DSn[PCS]) = DTS(DSn[PCS]) + decodeduration(DSn)$$

ここでdecodeduration(DSn)は、PCSのアップデートに用いられる全グラフィクスオブジェクトのデコード時間である。このデコード時間は、固定値ではない。しかし各再生装置の状態や再生装置の実装により変動するものでもない。本DSn.PCSnの画面合成に用いられるオブジェクトをDSn.PCSn.OBJ[j]とした場合、decodeduration(DSn)は、ウィンドウクリアに要する時間(i)、DSn.PCSn.OBJのデコード期間(ii)、DSn.PCSn.OBJの書き込みに要する時間(iii)により変動を受ける値になる。Rd,Rcさえ予め定められていれば、どのような実装の再生装置であっても、同じ値になる。そのためオーサリング時にあたっては、これらの期間の長さを算出して、この値からPTSを計算している。

40

【 0 0 7 4 】

decode_durationの計算は、図18のプログラムに基づき行われる。また図19、図20(a)(b)は、このプログラムのアルゴリズムを図式化したフローチャートである。以降、これらの図を参照しながらdecode_durationの算出について説明する。図19のフローチャートにおいて、先ず初めに、PLANEINITIALIZATINTIME関数を呼び出し、戻り値をdecode_durationに加算する(図19のステップS1)。PLANEINITIALIZATINTIME関数(図20(a))は、Display Setを表示するにあたってグラフィックスプレーンの初期化に要す

50

る時間を求める関数を呼出するための関数であり、図 19 のステップ S 1 では、DSn,DSnPCS.OBJ[0],decode_durtationを引数に設定して、この関数を呼び出す。

【 0 0 7 5 】

続いて図 20 (a) を参照しながらPLANEINITIALIZATINTIME関数について説明する。図 20 (a) においてinitialize_durationとは、PLANEINITIALIZATINTIME関数の戻り値を示す変数である。

図 20 のステップ S 2 は、DSnのPCSにおけるcomposition_stateがEpoch Startかどうかにより、処理を切り換えるif文である。もしcomposition_stateがEpoch Startであるなら(図 18 のDSn.PCS.composition_state==EPOCH_START、ステップ S 2 = Yes)、グラフィックスプレーンのクリアに要する時間をinitialize_durationに設定する(ステップ S 3)。 10

【 0 0 7 6 】

オブジェクトバッファ - グラフィックスプレーン間の転送レートRcを上記したように256,000,000とし、グラフィックスプレーンの総サイズをvideo_width*video_heightとすると、クリアに要する時間(秒)は、「video_width*video_height//256,000,000」になる。これに、90.000Hzを乗じPTSの時間精度で表現すれば、グラフィックスプレーンのクリアに要する時間は「90000×video_width*video_height//256,000,000」になる。この時間を、initialize_durationに加えて、initialize_durationを戻り値としてリターンする。

【 0 0 7 7 】

もしcomposition_stateがEpoch Startでないなら(ステップ S 2 = No)、WDSにて定義されるwindow[i]のクリアに要する時間をinitialize_durationに加えるという処理を全てのwindowについて繰り返す(ステップ S 4)。オブジェクトバッファ - グラフィックスプレーン間の転送レートRcを上記したように256,000,000とし、WDSに属するウィンドウ[i]の総サイズをSIZE(WDS.WIN[i])とすると、クリアに要する時間(秒)は、「SIZE(WDS.WIN[i])//256,000,000」になる。これに、90.000Hzを乗じPTSの時間精度で表現すれば、WDSに属する全ウィンドウのクリアに要する時間は「90000×SIZE(WDS.WIN[i])//256,000,000」になる。この時間を、initialize_durationに加えて、initialize_durationを戻り値としてリターンする。以上がPLANEINITIALIZATINTIME関数である。 20

【 0 0 7 8 】

図 19 のステップ S 5 は、DSnに属するグラフィクスオブジェクトの数が2であるか、1であるかで処理を切り換えるものであり(図 18 のif(DSn.PCS.num_of_object==2),if(DSn.PCS.num_of_object==1))、もし " 1 " であるなら(ステップ S 5 = 1)、グラフィクスオブジェクトのデコードを行うための待ち時間をdecode_durationに加える(ステップ S 6)。この待ち時間の算出は、WAIT関数の呼出(図 18 のdecode_duration +=WAIT(DSn,DSn.PCS.OBJ[0],decode_duration))で実現される。この関数呼出にあたっての引数はDSn,DSn.PCS.OBJ[0],decode_durationであり、待ち時間を示すwait_durationが戻り値として返される。 30

【 0 0 7 9 】

図 20 (b) は、WAIT関数の処理手順を示すフローチャートである。

このWAIT関数においてcurrent_durationとは、呼出元のdecode_durationが設定される。object_define_ready_timeは、Display Setのグラフィクスオブジェクト[i]のPTSが設定される変数である。 40

current_timeとは、DSnのPCSのDTSに、current_durationを足した値が設定される変数である。このcurrent_timeよりobject_define_ready_timeが大きい場合(ステップ S 7 が Yes、if(current_time < object_define_ready_time))、戻り値たるwait_durationは、object_define_ready_timeとcurrent_timeとの差分が設定されることになる(ステップ S 8、wait_duration += object_define_ready_time - current_time)。以上がWait関数である。ステップ S 6 においてdecode_durationには、このwait関数の戻り値と、window再描画に必要な時間を足し合わせた時間(90,000*(SIZE(DSn.WDS.WIN[0]))//256,000,000)が設定されることになる。

【 0 0 8 0 】

以上はDSnのグラフィクスオブジェクトが1つである場合の処理である。図19のステップS5は、グラフィクスオブジェクトの数が2であるかどうかを判定している。DSnのグラフィクスオブジェクトが2以上であれば(図18のif(DSn.PCS.num_of_object==2))、PCSにおけるOBJ[0]を引数として、wait関数を呼び出し、その戻り値をdecode_durationに加算する(ステップS10)。

【0081】

続くステップS11は、DSnのOBJ[0]が属するwindowが、グラフィクスオブジェクト[1]が属するwindowと同一かどうかの判定であり(if(DSn.PCS.OBJ[0].window_id == DSn.PCS.OBJ[1].window_id)、もし同一であるなら、OBJ[1]を引数にしてwait関数を呼び出し、その戻り値wait_durationを、decode_durationに加算する(ステップS12)。OBJ[0]が属するwindowの再描画に必要な時間($90,000 \times (\text{SIZE}(\text{DSn.WDS.OBJ}[0].\text{window_id}) // 256,000,000)$)をdecode_durationに加える(ステップS13)。

10

【0082】

もし属するwindowが異なるなら(ステップS11で"異なる")、OBJ[0]が属するwindowの再描画に必要な時間($90,000 \times (\text{SIZE}(\text{DSn.WDS.OBJ}[0].\text{window_id}) // 256,000,000)$)をdecode_durationに加える(ステップS15)。OBJ[1]を引数にしてwait関数を呼び出し、その戻り値wait_durationを、decode_durationに加算する(ステップS16)。その後、OBJ[1]が属するwindowの再描画に必要な時間($90,000 \times (\text{SIZE}(\text{DSn.WDS.OBJ}[1].\text{window_id}) // 256,000,000)$)をdecode_durationに加える(ステップS17)。

【0083】

20

以上のアルゴリズムにより、Decode_Durationは算出される。このPCSのPTSが、具体的にどのように設定されるかについて説明する。

図21(a)は、1つのwindowに1つのODSが存在するケースを想定した図である。図21(b)(c)は、図18で引用した各数値の時間的な前後関係を示すタイミングチャートである。本チャートには3つの段があり、これらの段のうち、"グラフィックスプレーンアクセス"の段、"ODSデコード"の段は、再生時にパラレルになされる2つの処理を示す。上述したアルゴリズムは、これらの2つの処理のパラレル実行を想定している。

【0084】

グラフィックスプレーンアクセスは、クリア期間(1)、書き込み期間(3)からなる。このクリア期間(1)は、グラフィックスプレーン全体のクリアに要する期間($90,000 \times (\text{グラフィックスプレーンのサイズ} // 256,000,000)$)、グラフィックスプレーンにおける全windowのクリアに要する時間($(90,000 \times (\text{window}[i]\text{のサイズ} // 256,000,000))$)のどちらかを意味する。

30

【0085】

書き込み期間(3)は、window全体描画に要する期間($90,000 \times (\text{windowサイズ} // 256,000,000)$)を意味する。

一方、デコード期間(2)は、ODSのDTSからPTSまでに示される期間を意味する。これらクリア期間(1)～書き込み期間(3)は、クリアすべき範囲、デコードすべきODSのサイズ、グラフィックスプレーンに書き込むべきグラフィクスオブジェクトのサイズにより変化する。本図では、説明の簡略化のため、ODSのデコード期間(2)の始点は、クリア期間(1)の始点と同一であると仮定している。

40

【0086】

図21(b)はデコード期間(2)が長くなるケースであり、Decode_Durationはデコード期間(2) + 書き込み期間(3)になる。

図21(c)は、クリア期間(1)が長くなるケースであり、Decode_Durationはクリア期間(1) + 書き込み期間(3)の期間がDecode_Durationになる。

図22(a)～(c)は、1つのwindowに2つのODSが存在するケースを想定した図である。本図(b)(c)におけるデコード期間(2)は、2つのグラフィックスのデコードに要する期間の総和を意味する。グラフィックス書込期間(3)も、2つのグラフィックスをグラフィックスプレーンに書き込む期間の総和を意味する。

50

ODSが2つになっているものの、図2-1と同様に考えればDecode_Durationを算出することができる。2つのODSをデコードするためのデコード期間(2)が長い場合は、図2-2(b)に示すようにDecode_Durationはデコード期間(2) + 書き込み期間(3)に算出されることになる。

【0087】

クリア期間(1)が長い場合は、図2-2(c)に示すように、Decode_Durationはクリア期間(1) + 書き込み期間(3)となる。

図2-3(a)は、2つのwindowのそれぞれに、ODSが1つずつ存在するケースを想定している。この場合でもクリア期間(1)が、2つのODSをデコードするための総デコード期間(2)より長い場合、Decode_Durationはクリア期間(1) + デコード期間(2)になる。問題は、クリア期間(1)がデコード期間(2)より短くなるケースである。この場合デコード期間(2)の経過を待たずに、1つ目のwindowへの書き込みは可能になる。そのためクリア期間(1) + 書き込み期間(3)、デコード期間(2) + 書き込み期間(3)の長さにはならない。ここで1つ目のODSのデコードに要する期間を書込期間(31)、2つ目のODSのデコードに要する期間を書込期間(32)とする。図2-3(b)は、デコード期間(2)がクリア期間(1) + 書込期間(31)より長くなるケースを示す。この場合Decode_Durationは、デコード期間(2) + 書込期間(32)になる。

【0088】

図2-3(c)は、クリア期間(1) + 書込期間(31)がデコード期間(2)より長くなるケースを示す。この場合Decode_Durationはクリア期間(1) + 書込期間(31) + 書込期間(32)になる。

グラフィックスプレーンのサイズは、プレーヤモデルから予め判明しており、またwindowのサイズ、ODSのサイズ、個数もオーサリングの段階で判明しているので、これらからDecode_Durationがクリア期間(1) + 書き込み期間(3)、デコード期間(2) + 書き込み期間(3)、デコード期間(2) + 書込期間(32)、クリア期間(1) + 書込期間(31) + 書込期間(32)のどれかになる。こうしたDecode_Duration算出を基にPCSのPTSを設定すれば、ピクチャデータとの同期表示を高い時間精度で実現することができる。このような高精度な同期制御は、windowを定義し、クリア・再描画する範囲を、このwindowに限定することで成り立っているので、オーサリングの環境に、このwindowの概念を導入したことの意義は大きい。

【0089】

続いてDSnにおけるWDSのDTS、PTSの設定について説明する。WDSのDTSは、以下の数式を満たすように設定すればよい。

$$DTS(DSn[WDS]) = DTS(DSn[PCS])$$

一方、DSnにおけるWDSのPTSは、グラフィックスプレーンに対する書き込みを開始すべきデッドラインを示す。グラフィックスプレーンへの書き込みは、ウィンドウだけで足りるので、PCSのPTSに示される時刻から、WDSの書き込みに要する期間を差し引けば、グラフィックスプレーンへの書き込みを開始すべき時刻は定まる。WDSの総サイズを $SIZE(WDS.WIN[i])$ とすれば、これのクリア及び再描画に要する時間は、『 $SIZE(WDS.WIN[i]) / 256,000,000$ 』になる。そして、これを90.000KHzの時間精度で表現すると『 $90000 \times SIZE(WDS.WIN[i]) / 256,000,000$ 』になる。

【0090】

故に以下の数式から、WDSのPTSを算出すればよい。

$$PTS(DSn[WDS]) = PTS(DSn[PCS]) - 90000 \times SIZE(WDS.WIN[i]) / 256,000,000$$

このWDSに示されるPTSはデッドラインなので、これより早い時点からグラフィックスプレーンの書き込みを開始してもよい。つまり図2-3に示すように、2つのウィンドウのうち、1つのウィンドウに表示させるべきODSのデコードが完了したなら、その時点からデコ

10

20

30

40

50

ードにより得られたグラフィクスオブジェクトの書き込みを開始してもよい。

【 0 0 9 1 】

このようにWDSに付加されたDTS、PTSを用いてAVClipの再生時間軸の任意の時点に、ウィンドウを割り当てることができる。

以上が再生時間軸に対するDisplay Setの割り当てについての説明である。

Display Setに対するDTS、PTSの設定について、図 2 4 ~ 図 2 5 の具体例を交えながら説明する。図 2 4 は、4回のグラフィックスプレーンへの書き込みにより字幕を表示することを想定した具体例である。この具体例は、2つの字幕『what is blu-ray』、『blu-ray is everywhere』を1つずつ表示させるというアップデートを想定している。図 2 4 は、本具体例が想定しているアップデートの、時間的変遷がどのようなものであるかを示す図である。時点t1では『what』までを表示し、続く時点t2では『what is』まで、時点t3では『what is blu-ray』全体を表示させる。字幕の全貌を明らかにした上で時点t4において別の字幕『blu-ray is everywhere』を表示させるのである。

10

【 0 0 9 2 】

図 2 5 (a) は、かかるアップデートを実現するため、記述された4つのDisplay Setを示す図である。DS1は、時点t1におけるアップデートを制御するPCS1.2と、彩色のためのPDS1と、字幕『what is blu-ray』に相当するODS1と、DS1の終了コードであるENDとからなる。

DS2は、時点t2におけるアップデートを制御するPCS1.2と、ENDとからなる。DS3は、時点t3におけるアップデートを制御するPCS1.3と、ENDとからなる。DS4は、時点t2のアップデートを制御するためのPCS2と、色変換のためのPDS2と、字幕『blu-ray is everywhere』に相当するODS2と、ENDとからなる。

20

【 0 0 9 3 】

これら4つのDisplay Setに属する各機能セグメントのDTS、PTSをどのように設定するかを、図 2 5 (b) のタイミングチャートを参照して説明する。

このタイミングチャートの再生時間軸は、図 2 4 の再生時間軸と同じものである。このタイミングチャートにおいて、PTS(PCS1.1)、PTS(PCS1.2)、PTS(PCS1.3)、PTS(PCS2)は、それぞれ『what』の表示時点t1、『what is』の表示時点t2、『what is blu-ray』の表示時点t3、『blu-ray is everywhere』の表示時点t4を示すよう設定される。これは、それぞれの字幕の表示時点において各PCSに記述された制御(Cropなど)を実行する必要があるからである。

30

【 0 0 9 4 】

PTS(ODS1)、PTS(ODS2)は、PTS(PCS1.1)及びPTS(PCS2)の時点からDecode_Durationを差し引いた時点を示すよう設定されている。PTS(PCS)は、数式 $PTS(DSn[PCS]) - DTS(DSn[PCS]) + decodeduration(DSn)$ を満たすよう設定する必要があるからである。

図 2 5 (b) では、時点t4より手前の時点t5を示すよう、PTS(ODS2)は設定されており、時点t1より手前の時点t0を示すよう、PTS(ODS1)は設定されている。DTS(ODS1)、DTS(ODS2)は、PTS(ODS1)及びPTS(ODS2)の時点からデコード期間を差し引いた時点を示すよう設定されている。DTS(ODS)は、数式 $DTS(DSn[ODSj]) = PTS(DSn[ODSj]) + 90,000 \times (SIZE(DSn[ODSj]) / Rd)$ を満たすよう設定する必要があるからである。図 2 5 (b) では、時点t5より手前の時点t0を示すよう、PTS(ODS2)は設定されており、時点t0より手前の時点を示すよう、PTS(ODS1)は設定されている。

40

【 0 0 9 5 】

ここで $DTS(ODS2) = PTS(ODS1)$ の関係が満たされることがわかる。これにより再生装置の実装によっては、ODS1がオブジェクトバッファに読み出された後、このオブジェクトバッファ上のODS1を上書きする形で、ODS2を読み出すことができる。あるODSのPTSを、先に表示すべきODSのPTSに設定しておけば、古いODSを上書きする形で、新しいODSをメモリに読み出すという動作を再生装置が行うことにより、小メモリ規模での再生処理を実現することができる。かかる再生処理の実現により、再生装置におけるメモリ規模の選択の幅は広がる。

50

【 0 0 9 6 】

PCS1.1のDTSは、 $DTS(PCS1.1) = DTS(ODS1)$ になるように設定されている。これは、PCS1.1のDTS値は、 $DTS(ODS1)$ により示される時点以前であるなら、どの時点でもよいからである。

ODS1のPTS値、ODS2のDTS値、PCS1.2、PCS1.3、PCS2のPTS値は、 $PTS(ODS1) = DTS(ODS2) = PTS(PCS1.2) = PTS(PCS1.3) = DTS(PCS2)$ の関係を満たすよう、何れも時点 t_0 に設定されている。

【 0 0 9 7 】

これは、PCS1.2、PCS1.3のDTS値は、 $PTS(PCS1.2)$ 、 $PTS(PCS1.3)$ 以前であれば、どの時点でもよく、またPCS2のDTS値も、 $DTS(PCS2)$ 以前であれば、どの時点でもよいという自由度があるからである。これにより、複数のPCSをまとめて読み出しおき、前のPCSによるアップデートの実行が終われば続くPCSによるアップデートを即座に実行することができる。

10

【 0 0 9 8 】

PCSのDTS・PTS、ODSのDTS・PCSは、上述の数式で示したような不等号の関係さえ最低限満たせばよいので、 $DTS(ODS2) = PTS(ODS1)$ になるように設定したり、 $PTS(ODS1) = DTS(ODS2) = PTS(PCS1.2) = PTS(PCS1.3) = DTS(PCS2)$ になるように設定することも可能になる。こうしたタイムスタンプの設定により、デコード負荷が発生する期間やバッファの占有量が増える期間を、調整することができる。かかる調整が可能になるので、再生時になしえる再生制御の幅が広がり、オーサリングを行う技術者及び再生装置を製造する技術者にとってのメリットは大きい。これで、DTS,PTS設定の具体例の説明を終える。

20

【 0 0 9 9 】

以上説明したDisplay Set(PCS,WDS,PDS,ODS)のデータ構造は、プログラミング言語で記述されたクラス構造体のインスタンスであり、オーサリングを行う制作者は、Blu-ray Disc Prerecording Formatに規定された構文に従い、クラス構造体を記述することで、BD-ROM上のこれらのデータ構造を得ることができる。

以上が本発明に係る記録媒体の実施形態である。続いて本発明に係る再生装置の実施形態について説明する。図26は、本発明に係る再生装置の内部構成を示す図である。本発明に係る再生装置は、本図に示す内部に基づき、工業的に生産される。本発明に係る再生装置は、主としてシステムLSIと、ドライブ装置、マイコンシステムという3つのパーツからなり、これらのパーツを装置のキャビネット及び基板に実装することで工業的に生産することができる。システムLSIは、再生装置の機能を果たす様々な処理部を集積した集積回路である。こうして生産される再生装置は、BDドライブ1、Read Buffer2、PIDフィルタ3、Transport Buffer4 a,b,c、周辺回路4 d、ビデオデコーダ5、ビデオプレーン6、オーディオデコーダ7、グラフィクスプレーン8、CLUT部9、加算器10、グラフィクスデコーダ12、Coded Data Buffer13、周辺回路13 a、Stream Graphics Processor14、Object Buffer15、Composition Buffer16、Graphical Controller17から構成される。

30

【 0 1 0 0 】

BD-ROMドライブ1は、BD-ROMのローディング/リード/イジェクトを行い、BD-ROMに対するアクセスを実行する。

40

Read Buffer2は、FIFOメモリであり、BD-ROMから読み出されたTSパケットが先入れ先出し式に格納される。

PIDフィルタ3は、Read Buffer2から出力される複数TSパケットに対してフィルタリングを施す。PIDフィルタ3によるフィルタリングは、TSパケットのうち、所望のPIDをもつもののみをTransport Buffer4 a,b,cに書き込むことでなされる。PIDフィルタ3によるフィルタリングでは、バッファリングは必要ではない。従って、PIDフィルタ3に入力されたTSパケットは、時間遅延なく、Transport Buffer4 a,b,cに書き込まれる。

【 0 1 0 1 】

Transport Buffer4 a,b,cは、PIDフィルタ3から出力されたTSパケットを先入れ先出し

50

式に格納しておくメモリである。このTransport Buffer 4 a,b,cからTSパケットが取り出される速度を速度Rxとする

周辺回路 4 dは、Transport Buffer 4 a,b,cから読み出されたTSパケットを、機能セグメントに変換する処理を行うワイロジックである。変換により得られた機能セグメントはCoded Data Buffer 1 3に格納される。

【 0 1 0 2 】

ビデオデコーダ 5 は、PIDフィルタ 3 から出力された複数TSパケットを復号して非圧縮形式のピクチャを得てビデオプレーン 6 に書き込む。

ビデオプレーン 6 は、動画用のプレーンメモリである。

オーディオデコーダ 7 は、PIDフィルタ 3 から出力されたTSパケットを復号して、非圧縮形式のオーディオデータを出力する。

10

【 0 1 0 3 】

グラフィクスプレーン 8 は、一画面分の領域をもったプレーンメモリであり、一画面分の非圧縮グラフィクスを格納することができる。

CLUT部 9 は、グラフィクスプレーン 8 に格納された非圧縮グラフィクスにおけるインデックスカラーを、PDSに示されるY,Cr,Cb値に基づき変換する。

加算器 1 0 は、CLUT部 9 により色変換された非圧縮グラフィクスに、PDSに示されるT値(透過率)を乗じて、ビデオプレーン 6 に格納された非圧縮状態のピクチャデータと画素毎に加算し、合成画像を得て出力する。

【 0 1 0 4 】

20

グラフィクスデコーダ 1 2 は、グラフィクスストリームをデコードして、非圧縮グラフィクスを得て、これをグラフィクスオブジェクトとしてグラフィクスプレーン 8 に書き込む。グラフィクスストリームのデコードにより、字幕やメニューが画面上に現れることになる。このグラフィクスデコーダ 1 2 は、Coded Data Buffer 1 3、周辺回路 1 3 a、Stream Graphics Processor 1 4、Object Buffer 1 5、Composition Buffer 1 6、Graphical Controller 1 7 から構成される。

【 0 1 0 5 】

Coded Data Buffer 1 3 は、機能セグメントがDTS、PTSと共に格納されるバッファである。かかる機能セグメントは、Transport Buffer 4 a,b,cに格納されたトランスポートストリームの各TSパケットから、TSパケットヘッダ、PESパケットヘッダを取り除き、ペイロードをシーケンシャルに配列することにより得られたものである。取り除かれたTSパケットヘッダ、PESパケットヘッダのうち、PTS/DTSは、PESパケットと対応付けて格納される。

30

【 0 1 0 6 】

周辺回路 1 3 aは、Coded Data Buffer 1 3 - Stream Graphics Processor 1 4 間の転送、Coded Data Buffer 1 3 - Composition Buffer 1 6 間の転送を実現するワイロジックである。この転送処理において現在時点がODSのDTSに示される時刻になれば、ODSを、Coded Data Buffer 1 3 からStream Graphics Processor 1 4 に転送する。また現在時刻がPCS、PDSのDTSに示される時刻になれば、PCS、PDSをComposition Buffer 1 6 に転送するという処理を行う。

40

【 0 1 0 7 】

Stream Graphics Processor 1 4 は、ODSをデコードして、デコードにより得られたインデックスカラーからなる非圧縮状態の非圧縮グラフィクスをグラフィクスオブジェクトとしてObject Buffer 1 5 に書き込む。このStream Graphics Processor 1 4 によるデコードは、ODSに関連付けられたDTSの時刻に開始し、ODSに関連付けられたPTSに示されるデコード終了時刻までに終了する。上述したグラフィックオブジェクトのデコードレートRdは、このStream Graphics Processor 1 4 の出力レートである。

【 0 1 0 8 】

Object Buffer 1 5 は、ETSI EN 300 743標準規格におけるピクセルバッファに相当するバッファであり、Stream Graphics Processor 1 4 のデコードにより得られたグラフィク

50

スオブジェクトが配置される。Object Buffer 1 5 は、グラフィクスプレーン 8 の2倍 / 4 倍の大きさに設定せねばならない。何故ならScrollingを実現する場合を考えると、グラフィクスプレーン 8 の2倍、4倍のグラフィクスオブジェクトを格納しておかねばならないからである。

【 0 1 0 9 】

Composition Buffer 1 6 は、PCS、PDSが配置されるメモリである。

Graphical Controller 1 7 は、Composition Buffer 1 6 に配置されたPCSを解読して、PCSに基づく制御をする。この制御の実行タイミングは、PCSに付加されたPTSの値に基づく。以上が再生装置の構成要素である。

続いて、PIDフィルタ 3、Transport Buffer 4 a,b,c、グラフィクスプレーン 8、CLUT部 10 9、Coded Data Buffer 1 3 ~ Graphical Controller 1 7 を構成するための、転送レート、バッファサイズの推奨値について説明する。図 2 7 は、書込レートRx,Rc,Rd、グラフィクスプレーン 8、Coded Data Buffer 1 3、Object Buffer 1 5、Composition Buffer 1 6 のサイズを示す図である。

【 0 1 1 0 】

Object Buffer 1 5 - グラフィクスプレーン 8 間の転送レートRcは、本装置において最も高い転送レートであり、ウィンドウサイズ、フレームレートから256Mbps(=500Kバイト × 29.97 × 2)と算出される。

Stream Graphics Processor 1 4 - Object Buffer 1 5 間の転送レートRd(Pixel Decoding Rate)は、Rcとは異なり、ビデオフレームの周期によるアップデートは要求されずRcの1 20 /2,1/4でよい。故に128Mbps,64Mbpsになる。

【 0 1 1 1 】

Transport Buffer 4 a,b,c - Coded Data Buffer 1 3 間のTransport Buffer LeakレートRxは、圧縮状態たるODSの転送レートである。従ってTransport Buffer Leakレートは、Rdに圧縮率を乗じた転送レートでよい。ODSの圧縮率を25%と仮定すれば、16Mbps(=64Mbps × 25%)で足りる。

この図に示す転送レート、バッファ規模はあくまでもミニマムスタンダードであり、これより大きい値での実装を否定している訳ではない。

【 0 1 1 2 】

以上のように構成された再生装置において、各構成要素はパイプライン式にデコード処理を行うことができる。 30

図 2 8 は、再生装置によるパイプライン処理を示すタイミングチャートである。第 5 段目は、BD-ROMにおけるDisplay Setを示し、第 4 段目は、Coded Data Buffer 1 3 へのPCS、WDS、PDS、ODSの読出期間を示す。第 3 段目は、Stream Graphics Processor 1 4 による各ODSのデコード期間を、第 2 段目はComposition Buffer 1 6 の格納内容を、第 1 段目はGraphical Controller 1 7 の処理内容を示す。

【 0 1 1 3 】

ODS1,2に付与されたDTS(デコード開始時刻)は、図中のt31,t32の時点を示している。デコード開始時刻がDTSに規定されているので、各ODSは、自身のDTSに示される時刻までにCoded Data Buffer 1 3 に読み出されなければならない。そのためODS1の読み出しは、Coded Data Buffer 1 3 へのODS1のデコード期間dp1の直前までに完了している。Coded Data Buffer 1 3 へのODS2の読み出しは、ODS2のデコード期間dp2の直前までに完了している。 40

【 0 1 1 4 】

一方、ODS1,2に付与されたPTS(デコード終了時刻)は、図中のt32,t33の時点を示している。Stream Graphics Processor 1 4 によるODS1のデコードはt32までに完了し、ODS2のデコードは、t33に示される時刻までに完了する。以上のように、Stream Graphics Processor 1 4 は、各ODSのDTSに示される時刻までに、ODSをCoded Data Buffer 1 3 に読み出し、Coded Data Buffer 1 3 に読み出されたODSを、各ODSのPTSに示される時刻までに、デコードしてObject Buffer 1 5 に書き込む。

【 0 1 1 5 】

本図の第1段目における期間cd1は、Graphics Controller 17がグラフィクスプレーン8をクリアするのに要する期間である。また期間td1は、Object Buffer 15上にえられたグラフィクスオブジェクトを、グラフィクスプレーン8に書き込むのに要する期間である。WDSのPTSは、この書き込みの開始にあたってのデッドラインを示し、PCSのPTSはこの書き込みの終了時点及び表示タイミングを示す。このPCSのPTSに示される時点になれば、対話画面を構成する非圧縮グラフィクスがグラフィクスプレーン8上に得られることになる。この非圧縮グラフィクスの色変換をCLUT部9に行わせ、ビデオプレーン6に格納されている非圧縮ピクチャとの合成を加算器10に行わせれば、合成画像が得られることになる。

【0116】

10

グラフィクスデコーダ12において、Graphics Controller 17がグラフィクスプレーン8のクリアを実行している間においても、Stream Graphics Processor 14のデコードは継続して行われる。以上のようなパイプライン処理により、グラフィクスの表示を迅速に実施することができる。

図28では、グラフィックスプレーンのクリアが、ODSのデコードより早く終わる場合を想定したが、図29は、ODSのデコードが、グラフィックスプレーンのクリアより早く終わる場合を想定したパイプライン処理を示すタイミングチャートである。この場合、ODSのデコードが完了した段階では、グラフィックスプレーンへの書き込みを実行することができず、グラフィックスプレーンのクリアが完了した時点で、デコードにより得られたグラフィクスをグラフィックスプレーンに書き込むことができる。

20

【0117】

以上が再生装置の内部構成である。続いて制御部20及びグラフィクスデコーダ12を、どのようにして実装するかについて説明する。制御部20は、図30の処理手順を行うプログラムを作成し、汎用CPUに実行させることにより実装可能である。以降、図30を参照しながら、制御部20の処理手順について説明する。

図30は、機能セグメントのロード処理の処理手順を示すフローチャートである。本フローチャートにおいてSegmentKとは、AVClipの再生時において、読み出されたSegment(PCS, WDS, PDS, ODS)のそれぞれを意味する変数であり、無視フラグは、このSegmentKを無視するかロードするかを切り換えるフラグである。本フローチャートは、無視フラグを0に初期化した上で、ステップS21～S24、ステップS27～S31の処理を全てのSegmentKについて繰り返すループ構造を有している(ステップS25、ステップS26)。

30

【0118】

ステップS21は、SegmentKがPCSであるか否かの判定であり、もしSegmentKがPCSであれば、ステップS27、ステップS28の判定を行う。

ステップS22は、無視フラグが1かどうかの判定である。無視フラグが0であるならステップS23に移行し、1であるならステップS24に移行する。無視フラグが1であれば(ステップS22でYes)、ステップS23においてSegmentKをCoded Data Buffer 13にロードする。

【0119】

無視フラグが0に設定されていれば(ステップS22がNo)、ステップS24においてSegmentKが無視される。これによりDSに属する機能セグメントは全て、ステップS22がNoになって、無視されることになる(ステップS24)。

40

このように、SegmentKが無視されるか、ロードされるかは、無視フラグの設定により決まる。ステップS27～S31、S34、S35は、この無視フラグを設定する処理である。

【0120】

ステップS27は、PCSにおけるsegment_typeがAcquisition Pointであるか否かの判定である。SegmentKがAcquisition Pointであるなら、ステップS28に移行し、SegmentKがもしEpoch StartかNormal Caseであるなら、ステップS31に移行する。

ステップS28は、先行するDSがグラフィクスデコーダ12内のどれかのバッファ(Cod

50

ed Data Buffer 1 3、Stream Graphicsプロセッサ 1 4、Object Buffer 1 5、Composition Buffer 1 6)に存在するかどうかの判定であり、ステップ S 2 7 がYesである場合に実行される。グラフィクスデコーダ 1 2 内にDSが存在しないケースとは、頭出しがなされたケースをいう。この場合、Acquisition PointたるDSから、表示を開始せねばならないので、ステップ S 3 0 に移行する(ステップ S 2 8 でNo)。

【 0 1 2 1 】

ステップ S 3 0 は、無視フラグを0に設定し、ステップ S 2 2 に移行する。

グラフィクスデコーダ 1 2 内にDSが存在するケースとは、通常再生がなされたケースをいう。この場合、ステップ S 2 9 に移行する(ステップ S 2 8 でYes)。ステップ S 2 9 は、無視フラグを1に設定し、ステップ S 2 2 に移行する。

10

ステップ S 3 1 は、PCSにおけるsegment_typeがNormal Caseであるか否かの判定である。もしNormal Caseであるなら、ステップ S 3 4 に移行する。SegmentKがEpoch Startであるなら、ステップ S 3 0 において無視フラグを0に設定する。ステップ S 3 4 は、ステップ S 2 8 と同じであり、先行するDSがグラフィクスデコーダ 1 2 内に存在するかどうかの判定を行う。もし存在するなら、無視フラグを0に設定する(ステップ S 3 0)。存在しないなら、元々、対話画面を構成する十分な機能セグメントが得られないため、無視フラグを1に設定する(ステップ S 3 5)。かかるフラグ設定により、先行するDSがグラフィクスデコーダ 1 2 に存在しない場合、Normal Caseを構成する機能セグメントは無視されることになる。

【 0 1 2 2 】

20

DSが、図 3 1 のように多重化されている場合を想定して、DSの読み出しがどのように行われるかを説明する。図 3 1 の一例では、3つのDSが動画と多重化されている。この3つのDSのうち、初めのDS1は、segment_typeがEpoch_Start、DS10はAcquisition Point、DS20は、Normal Caseである。

かかる3つのDSが、動画と多重化されているAVClipにおいて、ピクチャデータpt10からの頭出しが矢印am1に示すように行われたものとする。この場合、頭出し位置に最も近いDS10が、図 3 0 のフローチャートの対象となる。ステップ S 2 7 においてsegment_typeはAcquisition Pointと判定されるが、先行するDSはCoded Data Buffer 1 3 上に存在しないため、無視フラグは0に設定され、このDS10が図 3 2 の矢印md1に示すように再生装置のCoded Data Buffer 1 3 にロードされる。一方、頭出し位置がDS10の存在位置より後である場合(図 3 1 の矢印am2)、DS20は、Normal CaseのDisplay Setであり、先行するDS20はCoded Data Buffer 1 3 に存在しないので、このDisplay Setは、無視されることになる(図 3 2 の矢印md2)。

30

【 0 1 2 3 】

図 3 3 のように通常再生が行われた場合のDS1,10,20のロードは、図 3 3 に示すものとなる。3つのDSのうち、PCSのsegment_typeがEpoch StartであるDS1は、そのままCoded Data Buffer 1 3 にロードされるが(ステップ S 2 3)、PCSのsegment_typeがAcquisition PointであるDS10については、無視フラグが1に設定されるため(ステップ S 2 9)、これを構成する機能セグメントはCoded Data Buffer 1 3 にロードされず無視される(図 3 4 の矢印rd2,ステップ S 2 4)。またDS20については、PCSのsegment_typeはNormal Caseなので、Coded Data Buffer 1 3 にロードされる(図 3 4 の矢印rd3)。

40

【 0 1 2 4 】

続いてGraphical Controller 1 7 の処理手順について説明する。図 3 5 ~ 図 3 7 は、Graphical Controller 1 7 の処理手順を示すフローチャートである。

ステップ S 4 1 ~ ステップ S 4 4 は、本フローチャートのメインルーチンであり、ステップ S 4 1 ~ ステップ S 4 4 に規定した何れかの事象の成立を待つ。

ステップ S 4 1 は、現在の再生時点がPCSのDTS時刻になっているか否かの判定であり、もしなっていれば、ステップ S 4 5 ~ ステップ S 5 3 の処理を行う。

【 0 1 2 5 】

ステップ S 4 5 は、PCSのcomposition_stateが、Epoch_Startであるか否かの判定であ

50

り、もしEpoch Startであるなら、ステップS 4 6においてグラフィクスプレーン 8 を全クリアする。それ以外であるなら、ステップS 4 7においてWDSのwindow_horizontal_position、window_vertival_position、window_width、window_heightに示されるwindowをクリアする。

【 0 1 2 6 】

ステップS 4 8 は、ステップS 4 6 又はステップS 4 7 でのクリア後の実行されるステップであり、任意のODSxのPTS時刻が既に経過しているか否かの判定である。つまり、グラフィクスプレーン 8 全体のクリアにあたっては、そのクリア時間に長時間を費するので、あるODS(ODSx)のデコードが既に完了していることもある。ステップS 4 8 はその可能性を検証している。もし経過していないなら、メインルーチンにリターンする。どれかのODSのデコード時刻を経過しているなら、ステップS 4 9 ~ステップS 5 1 を実行する。ステップS 4 9 は、object_crop_flagが0を示しているか否かの判定であり、もし0を示しているなら、グラフィクスオブジェクトを非表示とする(ステップS 5 0)。

【 0 1 2 7 】

もし0を示していないなら、object_cropping_horizontal_position、object_cropping_vertival_position、cropping_width、cropping_heightに基づきクロップされたグラフィクスオブジェクトを、グラフィクスプレーン 8 のwindowにおいて object_cropping_horizontal_position、object_cropping_vertival_positionに示される位置に書き込む(ステップS 5 1)。以上の処理により、ウィンドウに1つ以上のグラフィクスオブジェクトが描かれることになる。

【 0 1 2 8 】

ステップS 5 2 は、別のODSyのPTS時刻が経過しているか否かの判定である。ODSxをグラフィクスプレーン 8 に書き込んでいる際、別のODSのデコードが既に完了していれば、このODSyをODSxにして(ステップS 5 3)、ステップS 4 9 に移行する。これにより、別のODSに対しても、ステップS 4 9 ~ S 5 1 の処理が繰り返し行われる。

次に図 3 6 を参照して、ステップS 4 2、ステップS 5 4 ~ステップS 5 9 について説明する。

【 0 1 2 9 】

ステップS 4 2 は、現在の再生時点がWDSのPTSであるか否かの判定であり、もしWDSのPTSであるなら、ステップS 5 4 においてウィンドウが1つであるか否かを判定し、もし2つであれば、メインルーチンにリターンする。ウィンドウが1つであるなら、ステップS 5 5 ~ステップS 5 9 のループ処理を行う。このループ処理は、ウィンドウに表示される2つのグラフィクスオブジェクトのそれぞれについて、ステップS 5 7 ~ステップS 5 9 を実行するというものである。ステップS 5 7 は、object_crop_flagが0を示しているか否かの判定であり、もし0を示しているなら、グラフィクスオブジェクトを非表示とする(ステップS 5 8)。

【 0 1 3 0 】

もし0を示していないなら、object_cropping_horizontal_position、object_cropping_vertival_position、cropping_width、cropping_heightに基づきクロップされたグラフィクスオブジェクトを、グラフィクスプレーン 8 のwindowにおいて object_cropping_horizontal_position、object_cropping_vertival_positionに示される位置に書き込む(ステップS 5 9)。以上の処理を繰り返せば、ウィンドウに1つ以上のグラフィクスオブジェクトが描かれることになる。

【 0 1 3 1 】

ステップS 4 4 は、現在の再生時点がPDSのPTSに示される時点であるかの判定であり、もしそうであるなら、ステップS 6 0 においてPallet_update_flagが1を示しているか否かを判定する。もし1を示しているなら、pallet_idに示されるPDSをCLUT部に設定する(ステップS 6 1)。0を示しているなら、ステップS 6 1 をスキップする。

その後、グラフィクスプレーン 8 におけるグラフィクスオブジェクトの色変換をCLUT部に行わせて、動画像と合成する(ステップS 6 2)。

10

20

30

40

50

【 0 1 3 2 】

次に図 3 7 を参照して、ステップ S 4 3、ステップ S 6 4 ~ ステップ S 6 6 について説明する。

ステップ S 4 3 は、現在の再生時点が ODS の PTS であるか否かの判定であり、もし ODS の PTS であるなら、ステップ S 6 3 においてウィンドウが 2 つであるか否かを判定し、もし 1 つであれば、メインルーチンにリターンする。ウィンドウが 2 つであるなら、ステップ S 6 4 ~ ステップ S 6 6 を行う。ステップ S 6 4 は、object_crop_flag が 0 を示しているか否かの判定であり、もし示しているなら、グラフィクスオブジェクトを非表示とする (ステップ S 6 5)。

【 0 1 3 3 】

10

もし 0 を示していないなら、object_cropping_horizontal_position、object_cropping_vertical_position、cropping_width、cropping_height に基づきクロップされたグラフィクスオブジェクトを、グラフィクスプレーン 8 の window において object_cropping_horizontal_position、object_cropping_vertical_position に示される位置に書き込む (ステップ S 6 6)。以上の処理を繰り返せば、各ウィンドウにグラフィクスオブジェクトが描かれることになる。

【 0 1 3 4 】

以上、DSn に属する PCS の DTS、PTS、ODS の DTS、PTS の設定について説明したが、PDS の DTS、PTS、END の DTS、PTS については説明していない。先ず、DSn に属する各 PDS の DTS、PTS の設定について説明する。

20

DSn に属する各 PDS は、PCS が Composition Buffer 1 6 にロードされる時点 (DTS(DSn[PCS])) から、最初の ODS のデコード開始時点 (DTS(DSn[ODS1])) までに、CLUT 部 9 において、有効になればよい。このことから DSn に属する各 PDS (PDS1 ~ PDSlast) の PTS 値は、以下の関係を満たす値に、設定されねばならない。

$$DTS(DSn[PCS]) \quad PTS(DSn[PDS1])$$

$$PTS(DSn[PDSj]) \quad PTS(DSn[PDSj+1]) \quad PTS(DSn[PDSlast])$$

$$PTS(DSn[PDSlast]) \quad DTS(DSn[ODS1])$$

30

尚、PDS において DTS は再生時に参照されないが、MPEG2 規格を満たすため、PDS の DTS は、その PTS と同じ値に設定される。

以上の関係を満たすよう DTS、PDS が設定された場合、再生装置のパイプラインにおいてこれら DTS、PTS がどのような役割をもつかについて説明する。図 3 8 は PDS における PTS に基づく、再生装置におけるパイプラインを示す図である。この図 3 8 は、図 2 8 をベースにして作図されている。図 3 8 において第 1 段目は、CLUT 部 9 への PDS 設定を示している。以降の段は、図 2 8 の第 1 ~ 第 5 段目と同じである。PDS1 ~ last の CLUT 部 9 への設定は、PCS、WDS の転送後、ODS1 のデコードより前になされるので、矢印 up2、up3 に示すように ODS1 の DTS に示される時点より前に設定されている。

40

【 0 1 3 5 】

以上のように PDS の設定は、ODS のデコードの先立ちなされることがわかる。

続いて DSn に属する END of Display Set Segment の PTS の設定について説明する。DSn に属する END は、DSn の終わりを示すものだから、ODS2 のデコード終了時刻を示せばよい。このデコード終了時刻は、ODS2 (ODSlast) の PTS (PTS(DSn[ODSlast])) に示されているので、END の PTS は、以下の式に示される値に設定されねばならない。

$$PTS(DSn[END]) = PTS(DSn[ODSlast])$$

50

DSn,DSn+1に属するPCSとの関係で考えれば、DSnにおけるPCSは、最初のODS(ODS1)のロード時刻以前に、Composition Buffer 16にロードされるから、ENDのPTSは、DSnに属するPCSのロード時刻(DTS(DSn[PCS]))以降、DSn+1に属するPCSのロード時刻(DTS(DSn+1[PCS]))以前でなければならない。そのためENDのPTSは、以下の式の間を満たす必要がある。

$$DTS(DSn[PCS]) \leq PTS(DSn[END]) \leq DTS(DSn+1[PCS])$$

一方、最初のODS(ODS1)のロード時刻は、最後のPDS(PDSlast)のロード時刻以前であるから、ENDのPTS(PTS(DSn[END]))は、DSnに属するPDSのロード時刻以降(PTS(DSn[PDSlast]))でなければならない。そのためENDのPTSは、以下の式の間を満たす必要がある。

$$PTS(DSn[PDSlast]) \leq PTS(DSn[END])$$

続いて再生装置のパイプラインにおいて、ENDのPTSが、どのような意味合いをなすのかについて説明する。図39は、再生装置のパイプライン動作時における、ENDの意味合いを示す図である。本図は、図28の第2段目以降をベースに作図しており、第2段以降の意味合いは図28と同一である。また本図では、DSn,DSn+1という2つのDisplay Setを描いている。DSnにおいてODSlastになるのは、A-ODSsの最後のODSnであるので、ENDのPTSは、このODSnのPTSを示すよう設定されている。そして、このENDのPTSに示される時点は、DSn+1のPCSのDTSにより示される時点より早いものになっている。

【0136】

このENDのPTSにより、再生時にあたっては、DSnについてのODSのロードが、どの時点で完了するのかを知得することができる。

尚、ENDにおいてDTSは再生時に参照されないが、MPEG2規格を満たすため、PDSのDTSは、そのPTSと同じ値に設定される。

以上のように本実施形態によれば、グラフィックプレーンの一部をグラフィックスの表示のためのウィンドウとして指定するので、再生装置は、プレーン全体のグラフィックス描画を行う必要はない。グラフィックプレーンの25%~33%など、ある限られた大きさのウィンドウに対してのみ、グラフィックス描画を行えばよい。グラフィックプレーンのうち、ウィンドウ以外の部分の描画を省くことができるので、再生装置側のソフトウェアの負担は遥かに軽くなる。

【0137】

グラフィックプレーンの1/4等、グラフィックスのアップデートがワーストケースで行われる場合であっても、ピクチャとの同期が保証されるような大きさにウィンドウを定めおけば、再生装置は、256Mbps等、ある決まった転送レートでグラフィックプレーンの書き込みを行いさえすれば、ピクチャとの同期を実現することができる。

同期保証が容易になるので、高い解像度での字幕表示を多くの再生装置において実現することができる。

【0138】

(第2実施形態)

第1実施形態では、ビデオフレーム毎にグラフィックスをアップデートできるように、ウィンドウの大きさをグラフィックスプレーン全体の1/4とし、グラフィックスプレーンへの書き込みレートを256Mbpsとした。またアップデートレートを、ビデオフレームレートの1/2,1/4とすることにより、大きなサイズのグラフィックスのアップデートの実現も可能とした。しかしアップデートレートをビデオフレームレートの1/2,1/4とした場合、グラフィックスプレーンへの書き込みは、2,4ビデオフレームを要する。そしてグラフィックスプレーンが1つであれば、この書き込みに費やされる2,4フレームの間、グラフィックスが書き込まれる過程がユーザに見えてしまう。そうすると、あるグラフィックスから大きなグラフィ

10

20

30

40

50

イクスへと瞬時に切り換わるような表示効果は実現できない。そこで第2実施形態ではグラフィックスプレーンを2つ設ける。図40は、第2実施形態に係る再生装置の内部構成を示す図である。本図が図26、図27と比較して新規なのは、グラフィックスプレーンが2つ有り(図中のグラフィックスプレーン81、グラフィックスプレーン82)、この2つのグラフィックスプレーンがダブルバッファを構成している点である。そのため、2つのグラフィックスプレーンのうち一方からの読み出しが行われている間、他方のグラフィックスプレーンへの書き込みは可能になる。そして第2実施形態に係るGraphical Controller 17は、この読出対象の切り換えをPCSに付与されているPTSの時点に実行する。

【0139】

図41は、ダブルバッファによるグラフィックスプレーン読み出し/書き込みを模式的に示す図である。本図の縦軸方向は、グラフィックスプレーン81、グラフィックスプレーン82の格納内容を、横軸方向は、1フレーム目、2フレーム目、3,4,5フレーム目のグラフィックスプレーンの格納内容を示している。そしてグラフィックスプレーン81、グラフィックスプレーン82の格納内容のうち、太い枠で囲まれているものが、読み出しの対象になっている。本図では、グラフィックスプレーン81に顔マークが格納されており、これを太陽マークに置き換えることを意図している。太陽マークはObject Buffer 15に存在しており、4MByteのサイズをもつ。この4MByteのサイズは、Object Buffer 15の目一杯のサイズである。

【0140】

太陽マークを、グラフィックスプレーンへの書込レート(Rc=256Mbps)で書き込むなら、グラフィックスプレーン82への書き込み完遂に4フレームがかかる。そしてこの書き込みに費やされる4フレームという期間において、1フレーム目では太陽マークのグラフィックスは1/4だけ、2フレーム目では2/4、3フレーム目では3/4だけ書き込みがなされる。しかし1フレームから4フレーム目までは、グラフィックスプレーン81、グラフィックスプレーン82のうち、読出対象がグラフィックスプレーン81なので、これら太陽マークがグラフィックスプレーンに書き込まれている過程は、ユーザには見えない。5フレーム目においてグラフィックスプレーンの読出対象がグラフィックスプレーン82に移った段階で、グラフィックスプレーン82の格納内容はユーザに見える。これにより顔マークから太陽マークへの切り換わり時になされる。

【0141】

以上のように本実施形態によれば、4フレームをかけて大きなサイズのグラフィックスをグラフィックスプレーンに書き込む場合でも、瞬時に画面切り換えを実現することができるので、画面一杯に映画制作者、制作スタッフ、出演者の氏名のクレジットを表示する場合や、映画のあらすじ、警告文を、瞬時に表示する場合に有意義である。

(第3実施形態)

本実施形態は、BD-ROMの製造工程に関する実施形態である。図42は、第3実施形態に係るBD-ROMの製造工程を示すフローチャートである。

【0142】

BD-ROMの制作工程は、動画収録、音声収録等の素材作成を行う素材制作工程S201、オーサリング装置を用いて、アプリケーションフォーマットを生成するオーサリング工程S202、BD-ROMの原盤を作成し、プレス・貼り合わせを行って、BD-ROMを完成させるプレス工程S203を含む。

これらの工程のうち、BD-ROMを対象としたオーサリング工程は、以下のステップS204～ステップS209を含む。

【0143】

ステップS204において、字幕の表示範囲となる windowを定義するようWDSを記述し、ステップS205では、windowが同じになる再生区間を Epochとし、個々のEpochにおけるPCSを記述する。

こうしてPCSが得られれば、ステップS206において字幕素材たるグラフィックスをODSに変換し、これをPCS、WDS、PDSを組み合わせることで Display Setを得る。その後、ス

10

20

30

40

50

ステップ S 2 0 7 では、Display Setにおける各機能セグメントを PESパケット化し、タイムスタンプを付与することによりグラフィクスストリームを得る。

【 0 1 4 4 】

最後に、ステップ S 2 0 8 では、グラフィクスストリームを別途生成されたビデオストリーム、オーディオストリームと多重することでAVClipを生成する。

AVClipが得られれば、AVClipをBD-ROMのフォーマットに適合させることにより、アプリケーションフォーマットが完成する。

(備考)

以上の説明は、本発明の全ての実施行為の形態を示している訳ではない。下記(A)(B)(C)(D)・・・の変更を施した実施行為の形態によっても、本発明の実施は可能となる。本願の請求項に係る各発明は、以上に記載した複数の実施形態及びそれらの変形形態を拡張した記載、ないし、一般化した記載としている。拡張ないし一般化の程度は、本発明の技術分野の、出願当時の技術水準の特性に基づく。しかし請求項に係る各発明は、従来技術の技術的課題を解決するための手段を反映したものであるから、請求項に係る各発明の技術範囲は、従来技術の技術的課題解決が当業者により認識される技術範囲を超えることはない。故に、本願の請求項に係る各発明は、詳細説明の記載と、実質的な対応関係を有する。

【 0 1 4 5 】

(A)全ての実施形態では、本発明に係る記録媒体をBD-ROMとして実施したが、本発明の記録媒体は、記録されるグラフィクスストリームに特徴があり、この特徴は、BD-ROMの物理的性質に依存するものではない。グラフィクスストリームを記録しうる記録媒体なら、どのような記録媒体であってもよい。例えば、DVD-ROM,DVD-RAM,DVD-RW,DVD-R,DVD+RW,DVD+R,CD-R,CD-RW等の光ディスク、PD,MO等の光磁気ディスクであってもよい。また、コンパクトフラッシュ(登録商標)カード、スマートメディア、メモリスティック、マルチメディアカード、PCM-CIAカード等の半導体メモリカードであってもよい。フレキシブルディスク、SuperDisk,Zip,Clik!等の磁気記録ディスク(i)、ORB,Jaz,SparQ,SyJet,EZFly,マイクロドライブ等のリムーバルハードディスクドライブ(ii)であってもよい。更に、機器内蔵型のハードディスクであってもよい。

【 0 1 4 6 】

(B)全ての実施形態における再生装置は、BD-ROMに記録されたAVClipをデコードした上でTVに出力していたが、再生装置をBD-ROMドライブのみとし、これ以外の構成要素をTVに具備させてもよい、この場合、再生装置と、TVとをIEEE1394で接続されたホームネットワークに組み入れることができる。また、実施形態における再生装置は、テレビと接続して利用されるタイプであったが、ディスプレイと一体型となった再生装置であってもよい。更に、各実施形態の再生装置において、処理の本質的部分をなすシステムLSI(集積回路)のみを、実施としてもよい。これらの再生装置及び集積回路は、何れも本願明細書に記載された発明であるから、これらの何れの態様であろうとも、第1実施形態に示した再生装置の内部構成を元に、再生装置を製造する行為は、本願の明細書に記載された発明の実施行為になる。第1実施形態に示した再生装置の有償・無償による譲渡(有償の場合は販売、無償の場合は贈与になる)、貸与、輸入する行為も、本発明の実施行為である。店頭展示、カタログ勧誘、パンフレット配布により、これらの譲渡や貸渡を、一般ユーザに申し出る行為も本再生装置の実施行為である。

【 0 1 4 7 】

(C)各フローチャートに示したプログラムによる情報処理は、ハードウェア資源を用いて具体的に実現されていることから、上記フローチャートに処理手順を示したプログラムは、単体で発明として成立する。全ての実施形態は、再生装置に組み込まれた態様で、本発明に係るプログラムの実施行為についての実施形態を示したが、再生装置から分離して、第1実施形態に示したプログラム単体を実施してもよい。プログラム単体の実施行為には、これらのプログラムを生産する行為(1)や、有償・無償によりプログラムを譲渡する

10

20

30

40

50

行為(2)、貸与する行為(3)、輸入する行為(4)、双方向の電子通信回線を介して公衆に提供する行為(5)、店頭、カタログ勧誘、パンフレット配布により、プログラムの譲渡や貸渡を、一般ユーザに申し出る行為(6)がある。

【0148】

(D)各フロ - チャ - トにおいて時系列に実行される各ステップの「時」の要素を、発明を特定するための必須の事項と考える。そうすると、これらのフロ - チャ - トによる処理手順は、再生方法の使用形態を開示していることがわかる。各ステップの処理を、時系列に行うことで、本発明の本来の目的を達成し、作用及び効果を奏するよう、これらのフロ - チャ - トの処理を行うのであれば、本発明に係る記録方法の実施行為に該当することはいうまでもない。

10

【0149】

(E)BD-ROMに記録するにあたって、AVClipを構成する各TSパケットには、拡張ヘッダを付与しておくことが望ましい。拡張ヘッダは、TP_extra_headerと呼ばれ、『Arribval_Time_Stamp』と、『copy_permission_indicator』とを含み4バイトのデータ長を有する。TP_extra_header付きTSパケット(以下EX付きTSパケットと略す)は、32個毎にグループ化されて、3つのセクタに書き込まれる。32個のEX付きTSパケットからなるグループは、6144バイト(=32×192)であり、これは3個のセクタサイズ6144バイト(=2048×3)と一致する。3個のセクタに収められた32個のEX付きTSパケットを"Aligned Unit"という。

【0150】

IEEE1394を介して接続されたホームネットワークでの利用時において、再生装置は、以下のような送信処理にてAligned Unitの送信を行う。つまり送り手側の機器は、Aligned Unitに含まれる32個のEX付きTSパケットのそれぞれからTP_extra_headerを取り外し、TSパケット本体をDTCP規格に基づき暗号化して出力する。TSパケットの出力にあたっては、TSパケット間の随所に、isochronousパケットを挿入する。この挿入箇所は、TP_extra_headerのArribval_Time_Stampに示される時刻に基づいた位置である。TSパケットの出力に伴い、再生装置はDTCP_Descriptorを出力する。DTCP_Descriptorは、TP_extra_headerにおけるコピー許否設定を示す。ここで「コピー禁止」を示すようDTCP_Descriptorを記述しておけば、IEEE1394を介して接続されたホームネットワークでの利用時においてTSパケットは、他の機器に記録されることはない。

20

30

(F)各実施形態におけるデジタルストリームは、BD-ROM規格のAVClipであったが、DVD-Video規格、DVD-Video Recording規格のVOB(Video Object)であってもよい。VOBは、ビデオストリーム、オーディオストリームを多重化することにより得られたISO/IEC13818-1規格準拠のプログラムストリームである。またAVClipにおけるビデオストリームは、MPEG4やWMV方式であってもよい。更にオーディオストリームは、Linear-PCM方式、Dolby-AC3方式、MP3方式、MPEG-AAC方式、dts方式であってもよい。

【0151】

(G)各実施形態における映画作品は、アナログ放送で放送されたアナログ映像信号をエンコードすることにより得られたものでもよい。デジタル放送で放送されたトランスポートストリームから構成されるストリームデータであってもよい。

40

またビデオテープに記録されているアナログ/デジタルの映像信号をエンコードしてコンテンツを得ても良い。更にビデオカメラから直接取り込んだアナログ/デジタルの映像信号をエンコードしてコンテンツを得ても良い。他にも、配信サーバにより配信されるデジタル著作物でもよい。

【0152】

(H)第1実施形態～第2実施形態に示したグラフィックスオブジェクトは、ランレングス符号化されたラスタデータである。グラフィックスオブジェクトの圧縮・符号化方式にランレングス符号方式を採用したのは、ランレングス符号化は字幕の圧縮・伸長に最も適しているためである。字幕には、同じ画素値の水平方向の連続長が比較的長くなるという特性があり、ランレングス符号化による圧縮を行えば、高い圧縮率を得ることができる。

50

また伸長のための負荷も軽く、復号処理のソフトウェア化に向いている。デコードを実現する装置構成を、字幕・グラフィックスオブジェクト間で共通化する目的で、字幕と同じ圧縮・伸長方式をグラフィックスオブジェクトに採用している。しかし、グラフィックスオブジェクトにランレングス符号化方式を採用したというのは、本発明の必須事項ではなく、グラフィックスオブジェクトはPNGデータであってもよい。またラスターデータではなくベクタデータであってもよい、更に透明な絵柄であってもよい。

【0153】

(H)PCSによる表示効果の対象は、装置側の言語設定に応じて選ばれた字幕のグラフィックスであってもよい。これにより、現状のDVDにおいて動画像本体で表現していたような文字を用いた表示効果を、装置側の言語設定に応じて表示された字幕グラフィックスで実現することができるので、実用上の価値は大きい。

10

(I)PCSによる表示効果の対象は、装置側のディスプレイ設定に応じて選ばれた字幕グラフィックスであってもよい。つまり、ワイドビジョン、パンスキャン、レターボックス用といった様々な表示モード用のグラフィックスがBD-ROMに記録されており、装置側は自身に接続されたテレビの設定に応じてこれらの何れかを選んで表示する。この場合、そうして表示された字幕グラフィックスに対し、PCSに基づく表示効果をほどこすので、見栄えがよくなる。これにより、動画像本体で表現していたような文字を用いた表示効果を、装置側のディスプレイ設定に応じて表示された字幕で実現することができるので、実用上の価値は大きい。

【0154】

20

(J)第1実施形態ではグラフィックスプレーンへの書込レート R_c は、1ビデオフレーム内にグラフィックスプレーンクリア及び再描画が可能になるよう、windowのサイズを全体の25%に定めたが、これらクリア・再描画が垂直帰線期間に完遂するよう、 R_c を定めても良い。垂直帰線期間は1/29.93秒の25%と仮定すると、 R_c は1Gbpsになる。 R_c をこのように設定することでグラフィックス表示はスムーズになされるので、実用上の効果は大きい。

【0155】

また垂直帰線期間での書き込みに加え、ラインスキャンに同期した書き込みを併用してもよい。これにより、 $R_c=256\text{Mbps}$ の書込レートであっても、スムーズな表示の実現が可能になる。

(K)各実施形態において再生装置には、グラフィックスプレーンを実装したが、このグラフィックスプレーンに代えて、一ライン分の非圧縮画素を格納するラインバッファを具備してもよい。映像信号への変換は水平行(ライン)毎に行われるので、このラインバッファさえ具備していれば、この映像信号への変換は行なえるからである。

30

【0156】

(L)グラフィックスたる字幕は、映画の台詞を表す文字列であるとして説明を進めたが、商標を構成するような図形、文字、色彩の組合せや、国の紋章、旗章、記章、国家が採用する監督/証明用の公の記号・印章、政府間国際機関の紋章、旗章、記章、特定商品の原産地表示を含んでいてもよい。

(M)第1実施形態では、字幕を画面の上側、下側に横書きで表示するものとして、ウィンドウをグラフィックスプレーンの上側、下側に定義したが、字幕を画面の右側、左側に表示するものとして、ウィンドウをグラフィックスプレーンの右側、左側に定義してもよい。こうすることにより、日本語字幕を縦書きで表示することができる。

40

【0157】

(N)各実施形態におけるAVClipは、映画作品を構成するものであったが、AVClipはカラオケを実現するものであってもよい、そしてこの場合、PCSは歌の進行に応じて、字幕の色を変えするという表示効果を実現してもよい。

【産業上の利用可能性】

【0158】

本発明に係る記録媒体、再生装置は、表示効果を伴う字幕表示を実現することができるので、より付加価値が高い映画作品を市場に供給することができ、映画市場や民生機器市

50

場を活性化させることができる。故に本発明に係る記録媒体、再生装置は、映画産業や民生機器産業において高い利用可能性をもつ。

【図面の簡単な説明】

【0159】

【図1】本発明に係る記録媒体の、使用行為についての形態を示す図である。

【図2】BD-ROMの構成を示す図である。

【図3】AVClipがどのように構成されているかを模式的に示す図である。

【図4】(a)プレゼンテーショングラフィクスストリームの構成を示す図である。(b)機能セグメントを変換することで得られるPESパケットを示す図である。

【図5】様々な種別の機能セグメントにて構成される論理構造を示す図である。

10

【図6】字幕の表示位置と、Epochとの関係を示す図である。

【図7】(a)ODSによるグラフィクスオブジェクトの定義を示す図である。(b)PDSのデータ構造を示す図である。

【図8】(a)WDSのデータ構造を示す図である。(b)PCSのデータ構造で構成される。

【図9】字幕表示を実現するためのDisplay Setの記述例である。

【図10】DS1におけるWDS、PCSの記述例を示す図である。

【図11】DS2におけるPCSの記述例を示す図である。

【図12】DS3におけるPCSの記述例を示す図である。

【図13】時間軸に沿った連続写真的な表記で、Cut-In/Outを実行する際のDisplay Setの記述例を示す図である。

20

【図14】時間軸に沿った連続写真的な表記で、Fade-In/Outを実行する際のDisplay Setの記述例を示す図である。

【図15】時間軸に沿った連続写真的な表記で、Scrollingがどのように行われるかを記述した図である。

【図16】時間軸に沿った連続写真的な表記で、Wipe-In/Outがどのように行われるかを記述した図である。

【図17】ウィンドウ内にグラフィクスオブジェクトが4つ存在する場合と、2つの存在する場合とを対比して示す図である。

【図18】decode_durationの計算アルゴリズムの一例を示す図である。

30

【図19】図18のプログラムのアルゴリズムを図式化したフローチャートである。

【図20】(a)(b)図18のプログラムのアルゴリズムを図式化したフローチャートである。

【図21】(a)1つのwindowに1つのODSが存在するケースを想定した図である。(b)(c)図18で引用した各数値の時間的な前後関係を示すタイミングチャートである。

【図22】(a)1つのwindowに2つのODSが存在するケースを想定した図である。(b)(c)図18で引用した各数値の時間的な前後関係を示すタイミングチャートである。

【図23】(a)2つのwindowのそれぞれに、ODSが1つずつ存在するケースを想定したタイミングチャートである。(b)デコード期間(2)がクリア期間(1)+書込期間(31)より長くなるケースを示すタイミングチャートである。(c)クリア期間(1)+書込期間(31)がデコード期間(2)より長くなるケースを示すタイミングチャートである。

40

【図24】アップデートの、時間的変遷の具体例を示す図である。

【図25】(a)図24に示したアップデートを実現するため、記述された4つのDisplay Setを示す図である。(b)図25(a)におけるDisplay Setに属する各機能セグメントのDTS、PTSがどのように設定されているかを示すタイミングチャートである。

【図26】本発明に係る再生装置の内部構成を示す図である。

【図27】書込レートRx,Rc,Rd、グラフィクスプレーン8、Coded Data Buffer13、Object Buffer15、Composition Buffer16のサイズを示す図である。

【図28】再生装置によるパイプライン処理を示すタイミングチャートである。

【図29】ODSのデコードが、グラフィックスプレーンのクリアより早く終わる場合を想

50

定したパイプライン処理を示すタイミングチャートである。

【図 3 0】機能セグメントのロード処理の処理手順を示すフローチャートである。

【図 3 1】多重化の一例を示す図である。

【図 3 2】DS10が再生装置のCoded Data Buffer 1 3 にロードされる様子を示す図である。

【図 3 3】通常再生が行われる場合を示す図である。

【図 3 4】図 3 3 のように通常再生が行われた場合のDS1,10,20のロードを示す図である。

【図 3 5】Graphical Controller 1 7 の処理手順を示すフローチャートである。

【図 3 6】Graphical Controller 1 7 の処理手順を示すフローチャートである。 10

【図 3 7】Graphical Controller 1 7 の処理手順を示すフローチャートである。

【図 3 8】PDSにおけるPTSに基づく、再生装置におけるパイプラインを示す図である。

【図 3 9】再生装置のパイプライン動作時における、ENDの意味合いを示す図である。

【図 4 0】第 2 実施形態に係る再生装置の内部構成を示す図である。

【図 4 1】ダブルバッファによるグラフィックスプレーン読み出し / 書き込みを模式的に示す図である。

【図 4 2】第 3 実施形態に係るBD-ROMの製造工程を示すフローチャートである。

【符号の説明】

【 0 1 6 0 】

1	BDドライブ	20
2	Read Buffer	
3	PIDフィルタ	
4	TBバッファ	
5	ビデオデコーダ	
6	ビデオプレーン	
7	オーディオデコーダ	
8	グラフィックスプレーン	
9	CLUT部	
1 0	加算器	
1 2	グラフィクスデコーダ	30
1 3	Coded Data Buffer	
1 4	Stream Graphics プロセッサ	
1 6	Composition バッファ	
1 7	Graphicsコントローラ	
1 8	U0コントローラ	
1 9	プレーヤレジスタ群	
2 0	制御部	
2 0 0	再生装置	
3 0 0	テレビ	
4 0 0	リモコン	40

【要約】

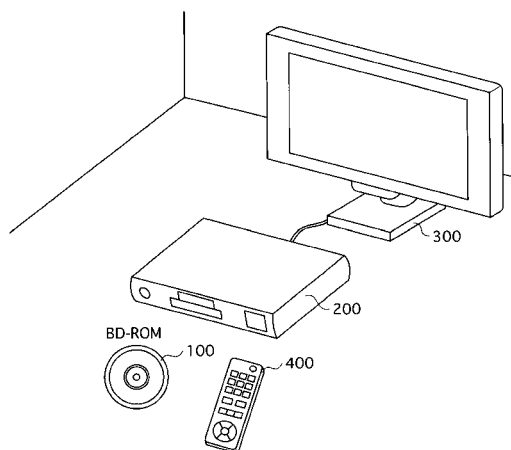
【課題】 グラフィックスプレーンに書き込むべきデータ量が多大であっても、ピクチャの表示と同期したグラフィックスのアップデートを実現することができる記録媒体を提供する。

【解決手段】 AVClipが記録された記録媒体であって、AVClipは、ビデオストリームと、グラフィックスストリームとを多重してなり、グラフィックスストリームは、動画像を構成する複数ピクチャに、グラフィックスを合成させるデータである。

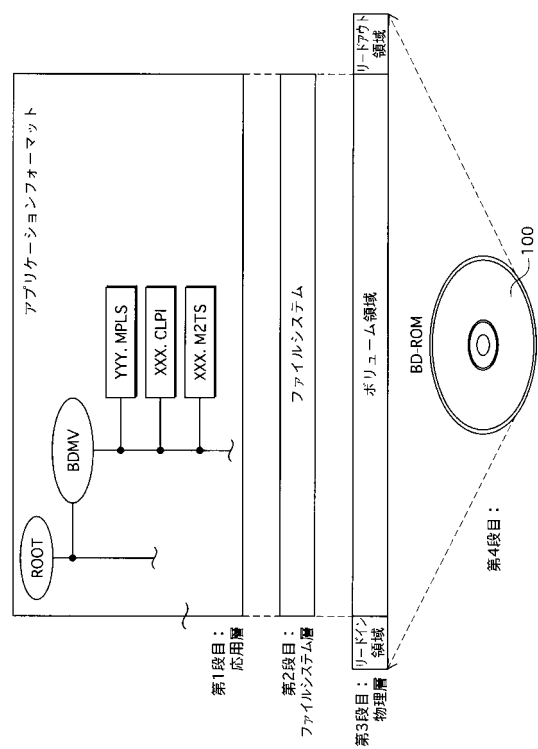
グラフィックスストリームはウィンドウ情報(WDS)を有し、ウィンドウ情報は、一ピクチャ全体に対応するプレーンメモリの一部を、グラフィックス表示用のウィンドウとして指定する情報であり、平面におけるウィンドウの位置、縦幅、横幅を示す。

【選択図】 図8

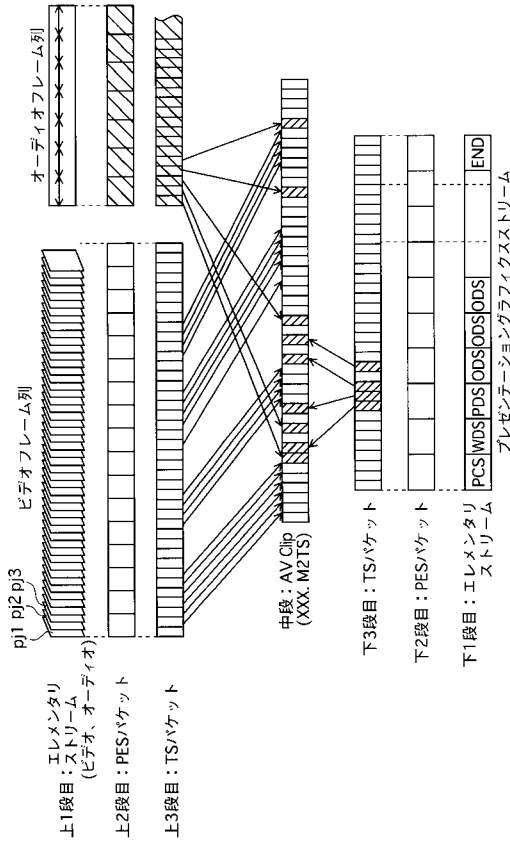
【図1】



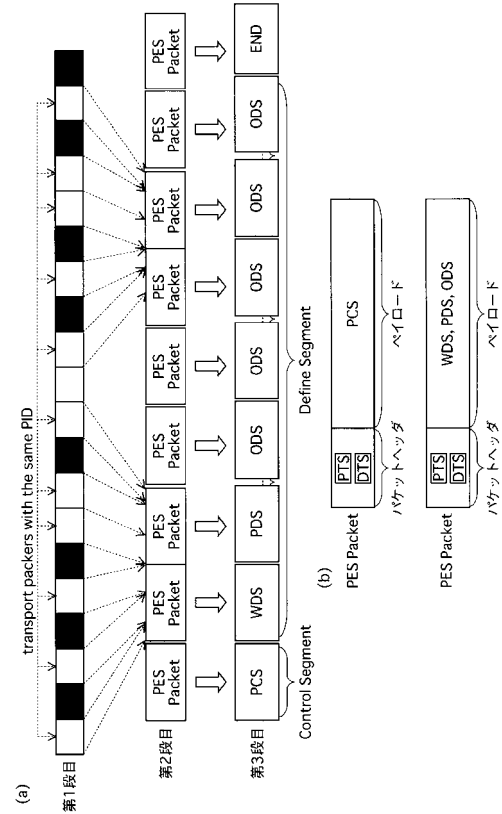
【図2】



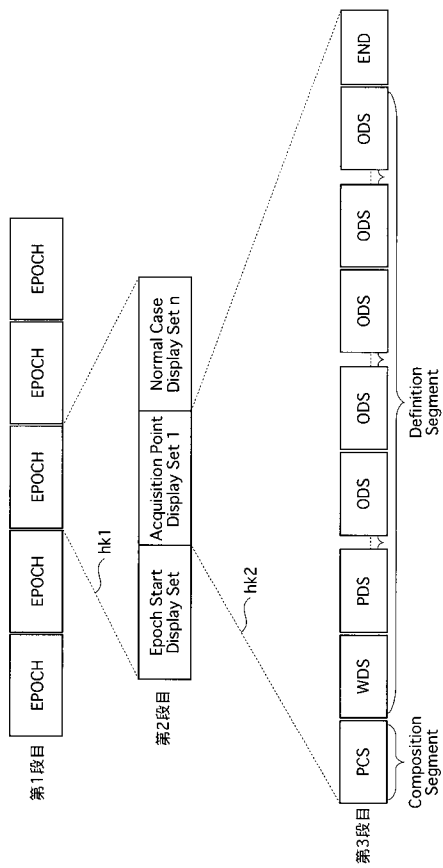
【図 3】



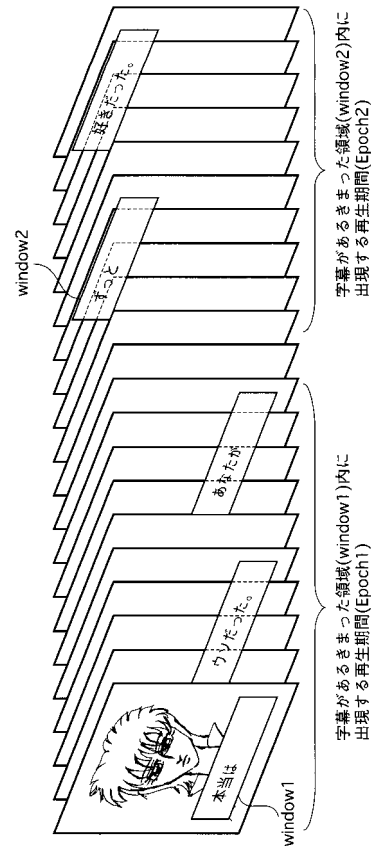
【図 4】



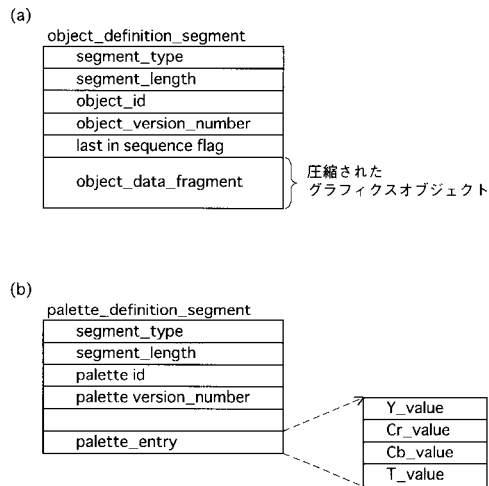
【図 5】



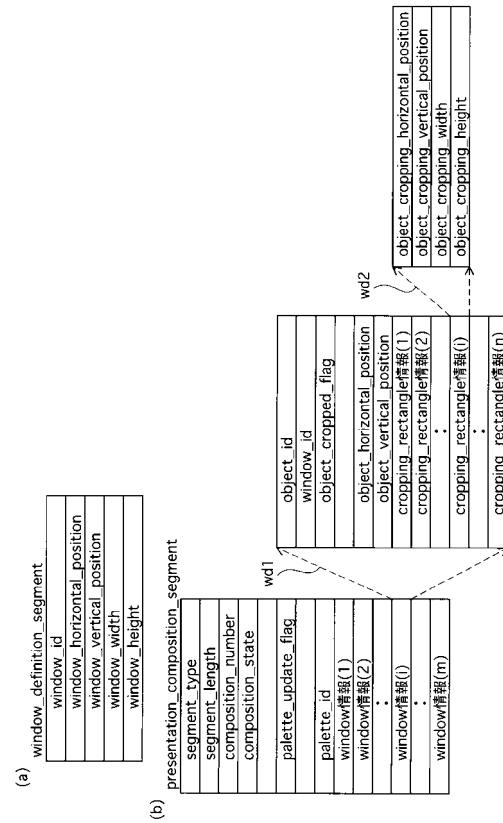
【図 6】



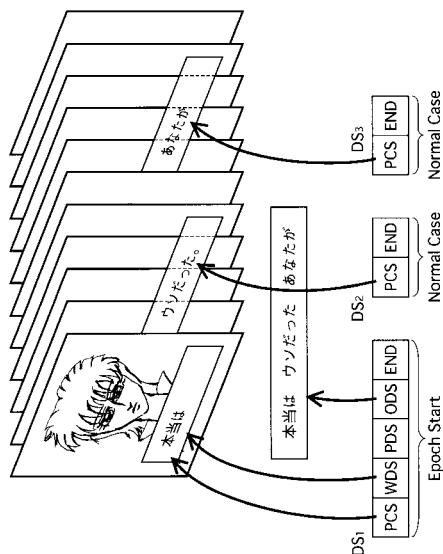
【図 7】



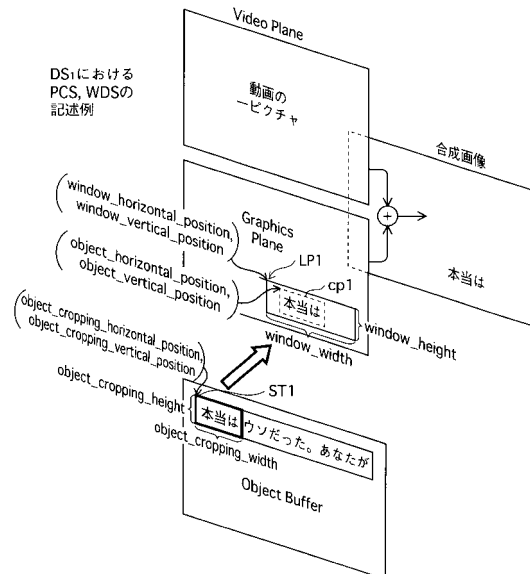
【図 8】



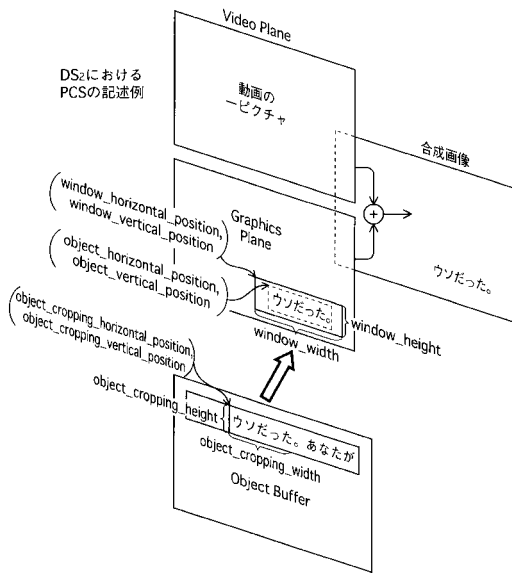
【図 9】



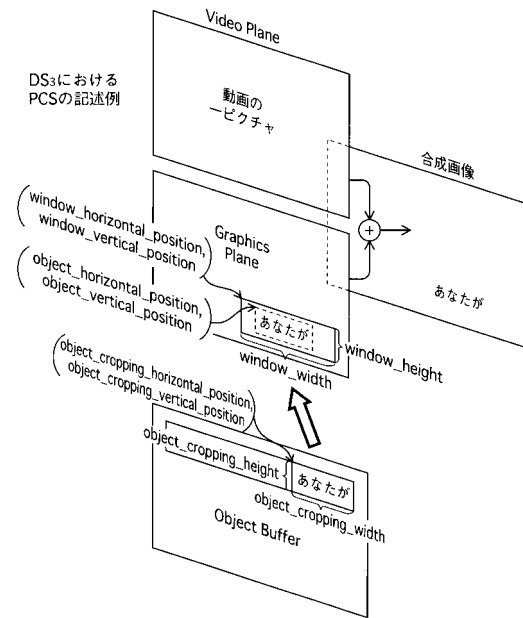
【図 10】



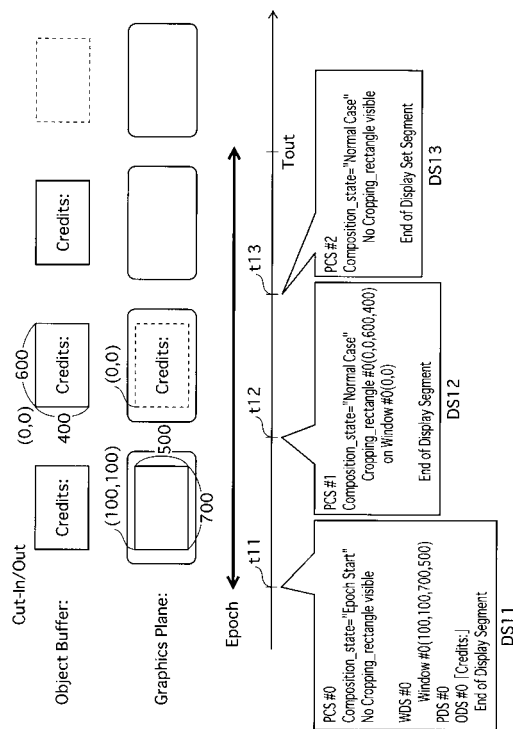
【図 1 1】



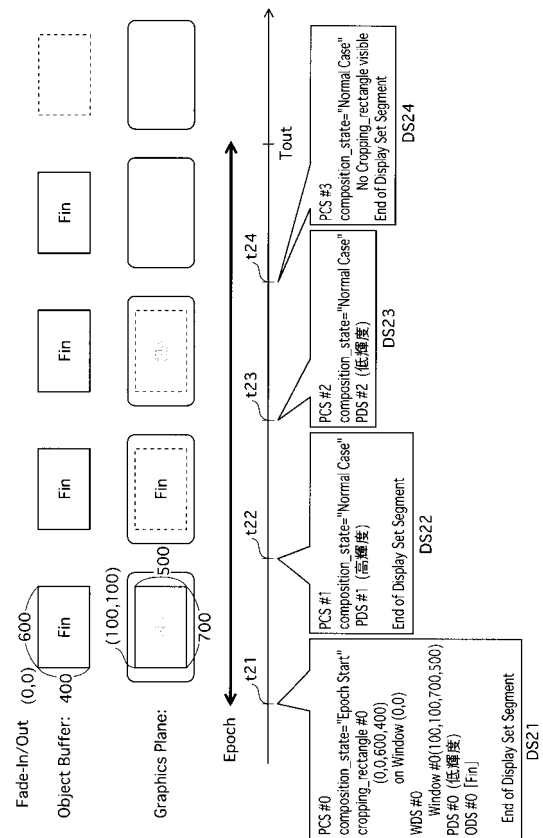
【図 1 2】



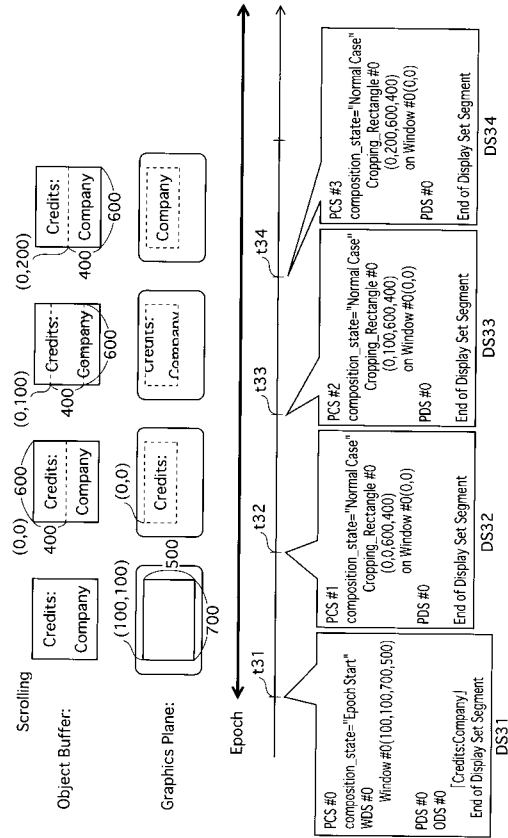
【図 1 3】



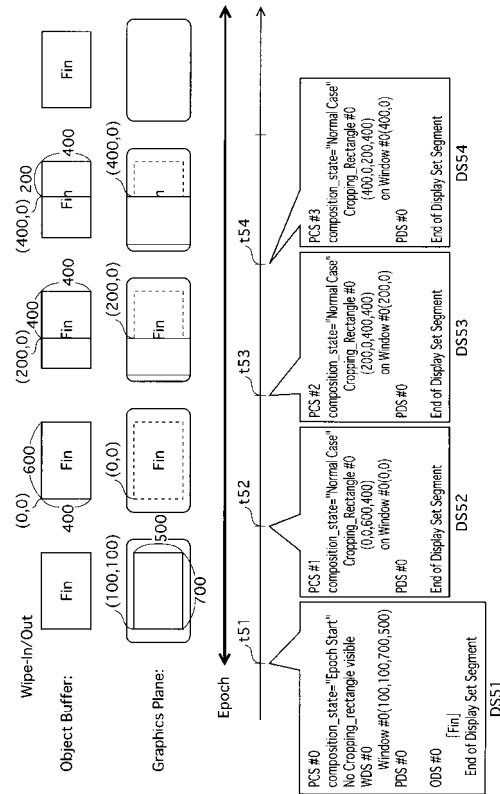
【図 1 4】



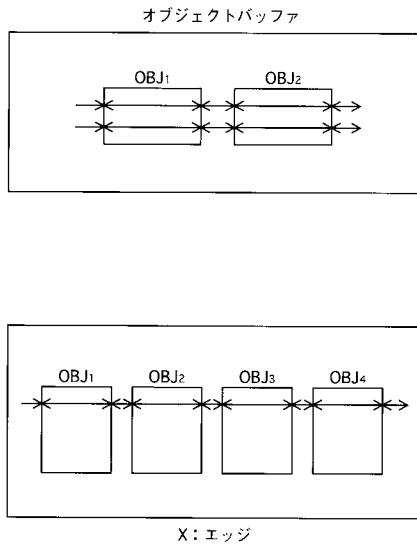
【図 15】



【図 16】



【図 17】



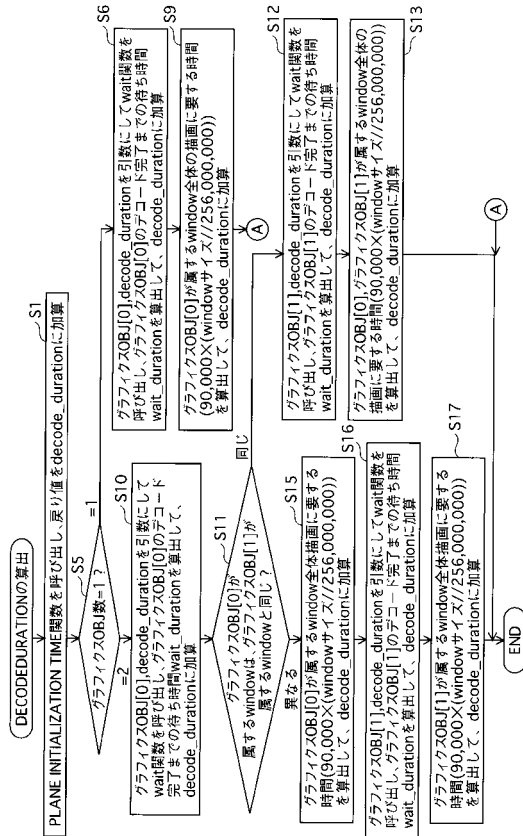
【図 18】

PTS(DSn[PCS]) >= DTS(DSn[PCS]) + DECODEDURATION(DSn)

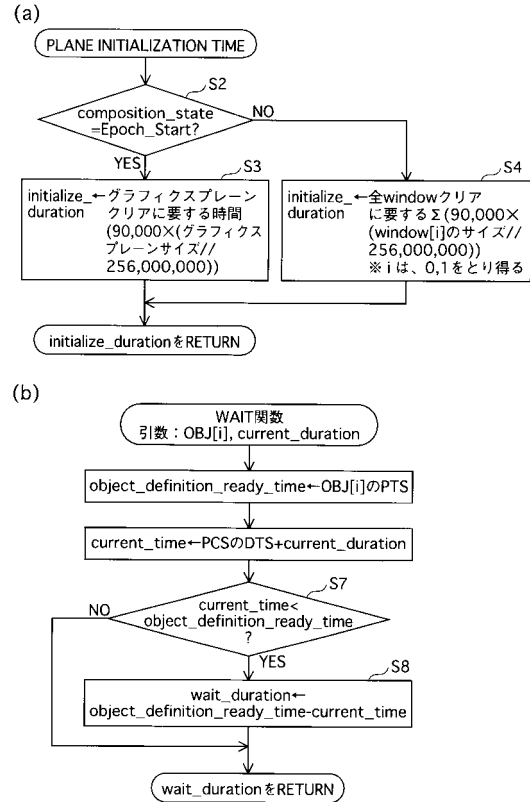
Where:

- DECODEDURATION(DSn) is calculated as follows:
 $\text{decode_duration} = 0;$
 $\text{decode_duration} += \text{PLANEINITIALIZATIONTIME}(\text{DSn});$
 if(DSn.PCS.num_of_objects == 2)
 {
 $\text{decode_duration} += \text{WAIT}(\text{DSn}, \text{DSn.PCS.OBJ}[0], \text{decode_duration});$
 $\text{decode_duration} += \text{WAIT}(\text{DSn}, \text{DSn.PCS.OBJ}[1], \text{decode_duration});$
 $\text{decode_duration} += 90000 * \text{SIZE}(\text{DSn.PCS.OBJ}[0].\text{window_id}) / (256 * 10^6);$
 $\text{decode_duration} += 90000 * \text{SIZE}(\text{DSn.PCS.OBJ}[1].\text{window_id}) / (256 * 10^6);$
 }
 else if(DSn.PCS.num_of_objects == 1)
 {
 $\text{decode_duration} += \text{WAIT}(\text{DSn}, \text{DSn.PCS.OBJ}[0], \text{decode_duration});$
 $\text{decode_duration} += 90000 * \text{SIZE}(\text{DSn.PCS.OBJ}[0].\text{window_id}) / (256 * 10^6);$
 }
 return decode_duration;
- PLANEINITIALIZATIONTIME(DSn) is calculated as follows:
 $\text{initialize_duration} = 0;$
 if(DSn.PCS.composition_state == EPOCH_START)
 {
 $\text{initialize_duration} = 90000 * (8 * \text{video_width} * \text{video_height}) / (256 * 10^6);$
 }
 else
 {
 for(i=0; i < WDS.num_windows; i++)
 {
 if(EMPTY(DSn.WDS.WIN[i], DSn))
 {
 $\text{initialize_duration} += 90000 * \text{SIZE}(\text{DSn.WDS.WIN}[i].\text{window_id}) / (256 * 10^6);$
 }
 }
 }
 return initialize_duration;
- WAIT(DSn, OBJ, current_duration) is calculated as follows:
 $\text{wait_duration} = 0;$
 if(EXISTS(OBJ, object_id, DSn))
 {
 $\text{object_definition_ready_time} = \text{PTS}(\text{GET}(\text{OBJ}, \text{object_id}, \text{DSn}));$
 $\text{current_time} = \text{DTS}(\text{DSn.PCS}) + \text{current_duration};$
 if(current_time < object_definition_ready_time)
 {
 $\text{wait_duration} += \text{object_definition_ready_time} - \text{current_time};$
 }
 }
 return wait_duration;

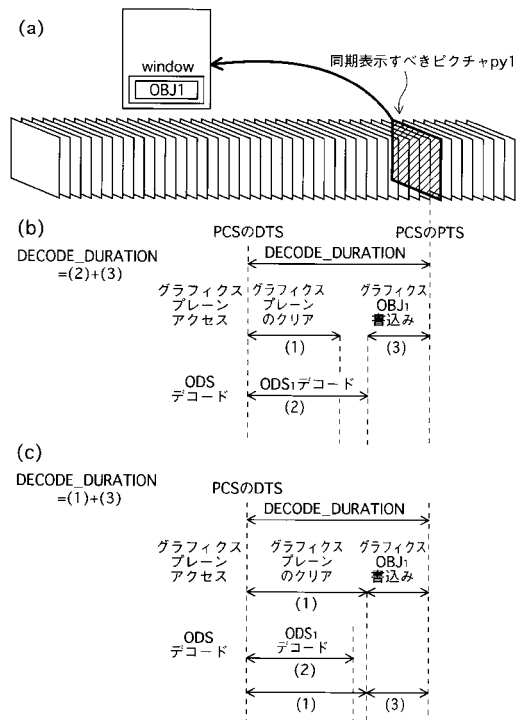
【図 19】



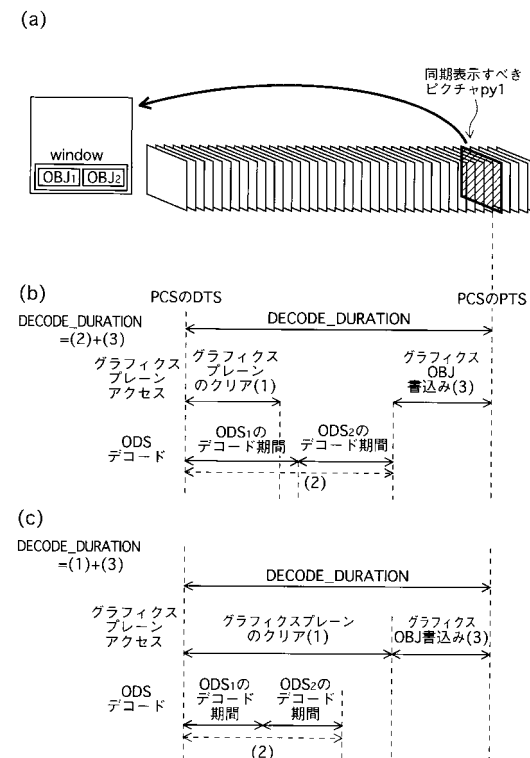
【図 20】



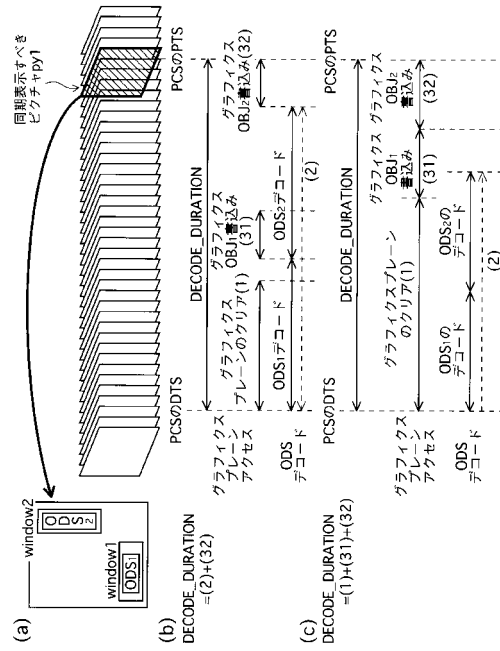
【図 21】



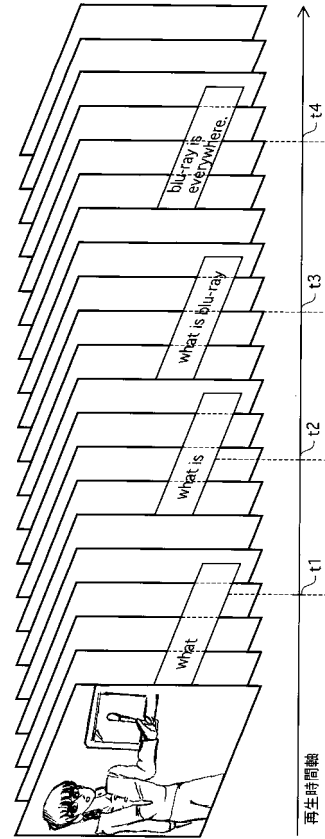
【図 22】



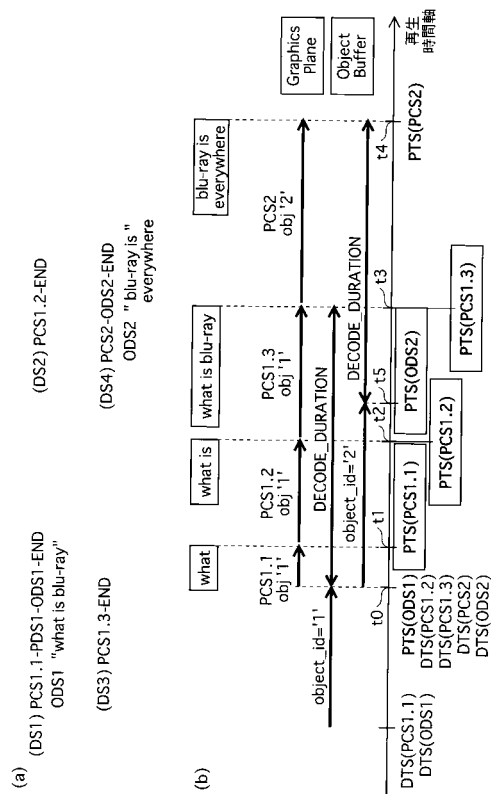
【図 23】



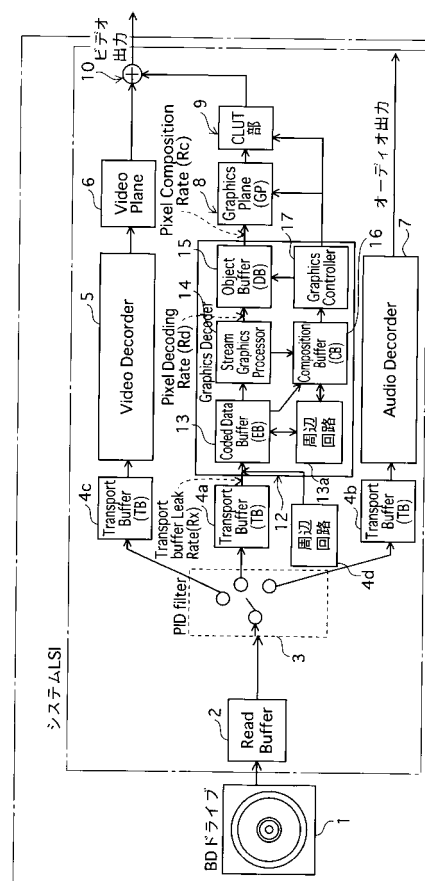
【図 24】



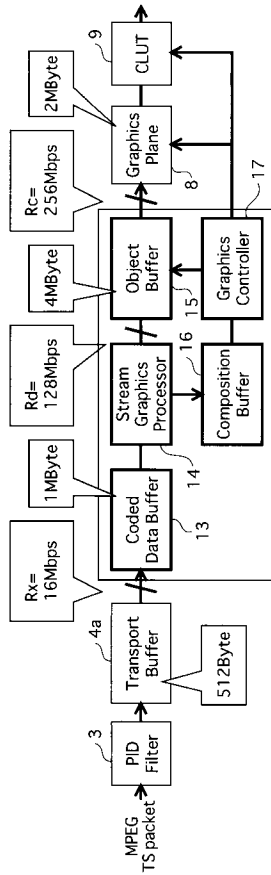
【図 25】



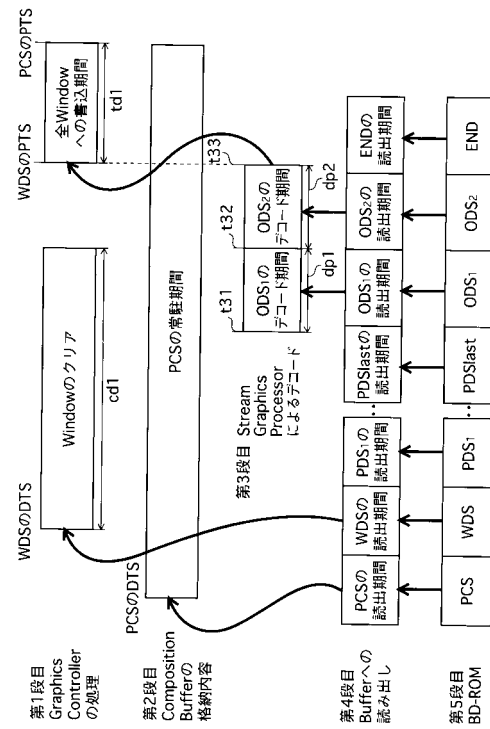
【図 26】



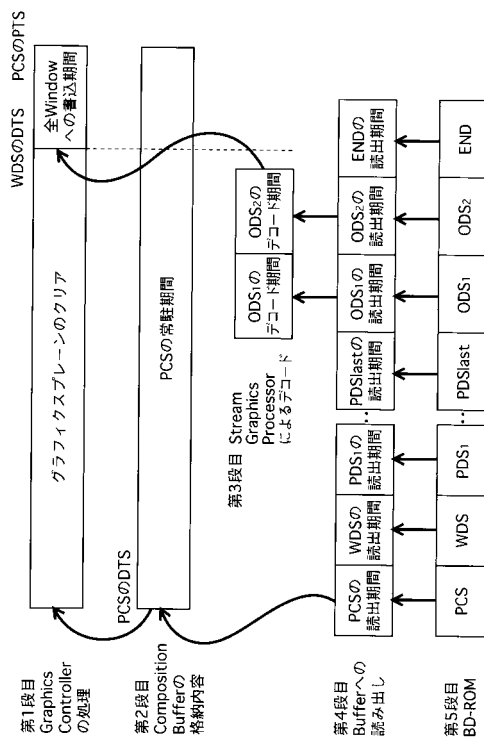
【 図 2 7 】



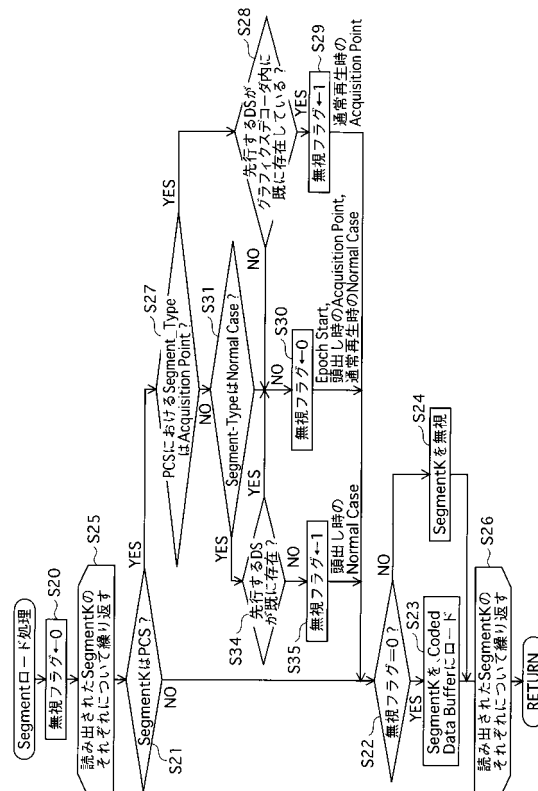
【 図 2 8 】



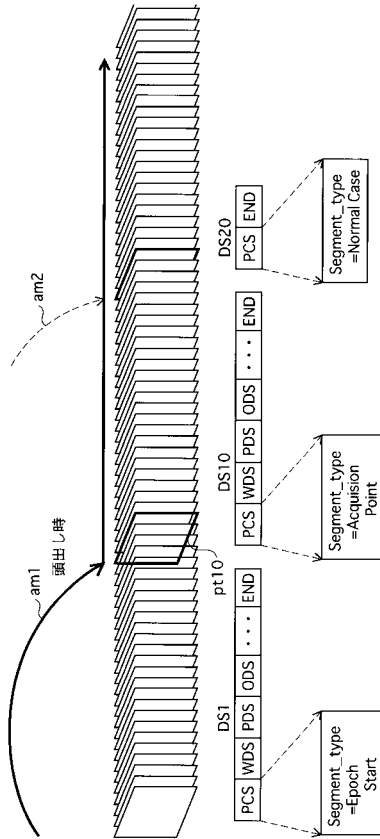
【 図 2 9 】



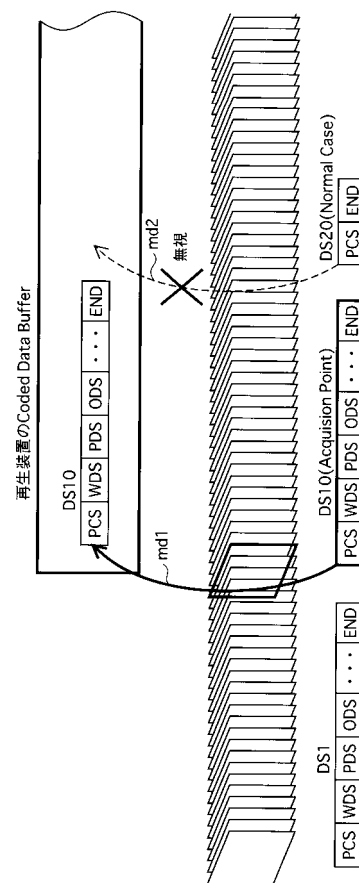
【 図 3 0 】



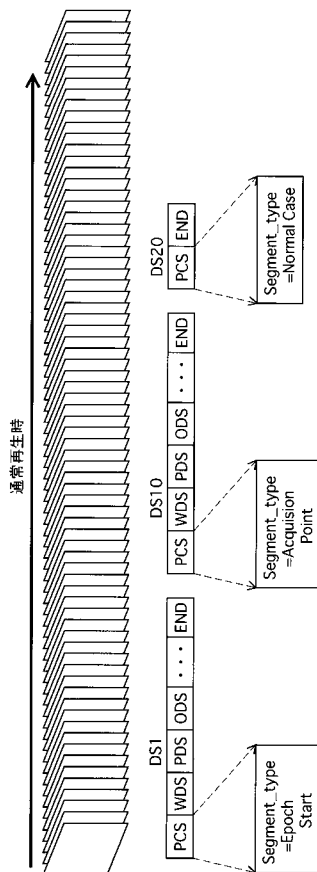
【図 3 1】



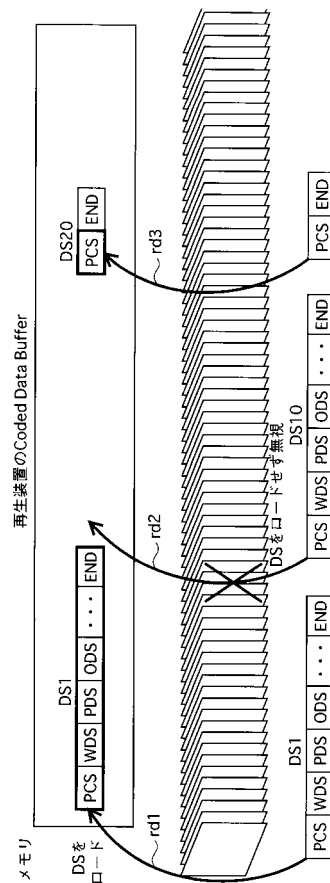
【図 3 2】



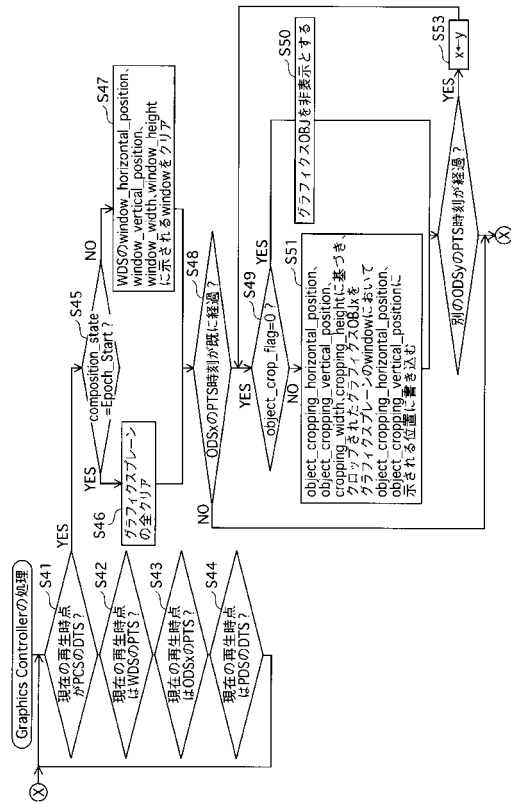
【図 3 3】



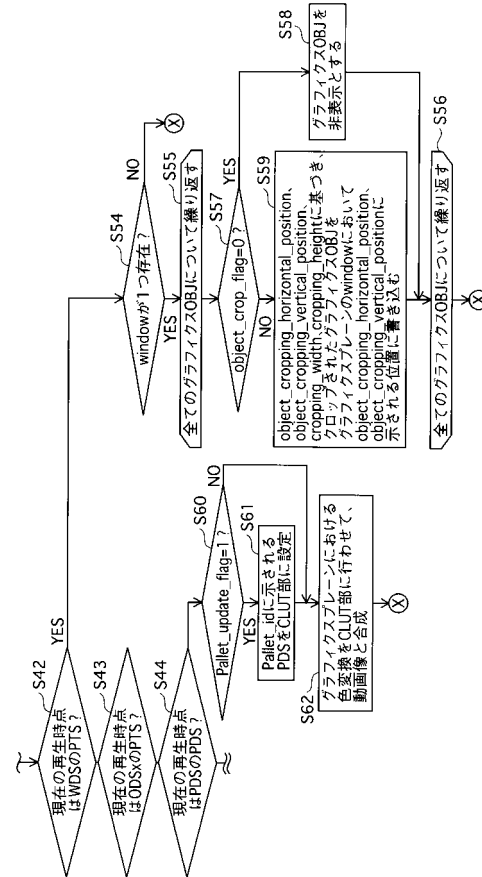
【図 3 4】



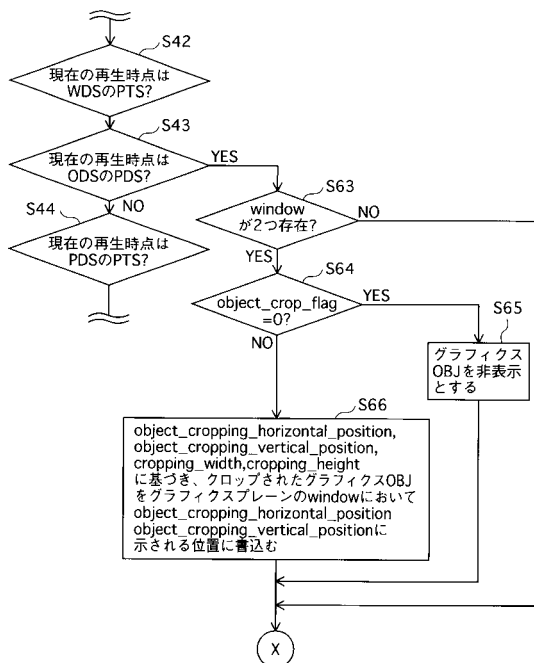
【図 35】



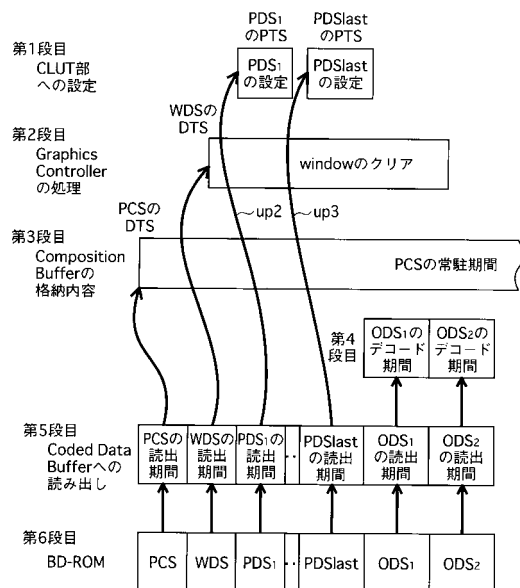
【図 36】



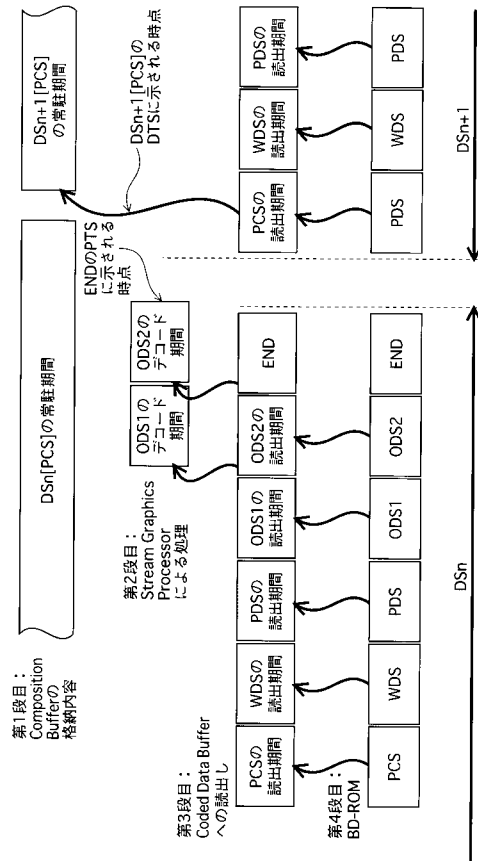
【図 37】



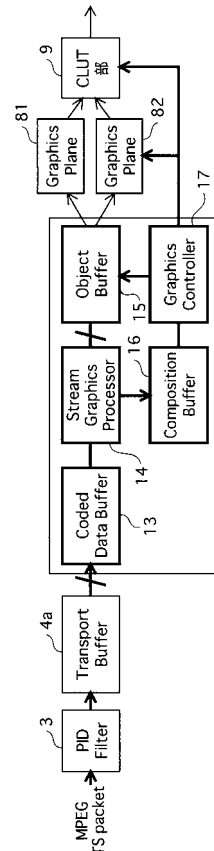
【図 38】



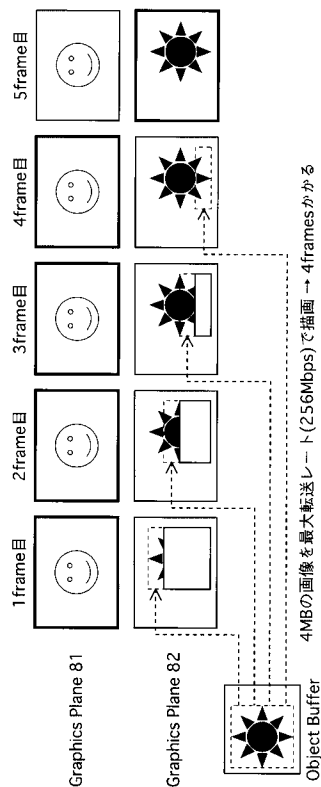
【図 39】



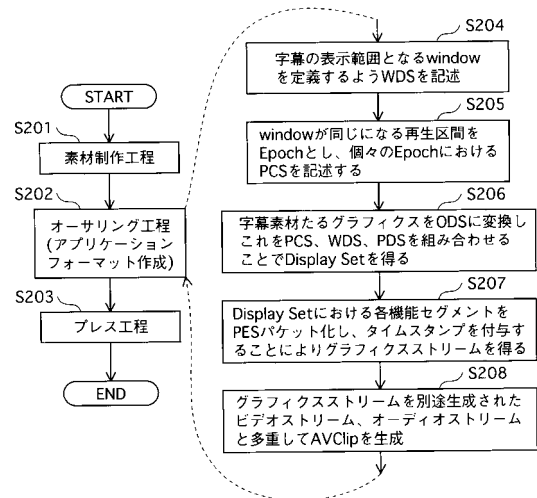
【図 40】



【図 41】



【図 42】



フロントページの続き

(51) Int.Cl.	F I		
	H 0 4 N	5/92	C
	H 0 4 N	5/85	Z

審査官 梅岡 信幸

(56) 参考文献 特開 2 0 0 2 - 2 0 9 1 7 7 (J P , A)
特開 2 0 0 3 - 2 1 9 3 7 2 (J P , A)

(58) 調査した分野(Int.Cl. , D B 名)
H04N 5/76- 5/956
G11B 20/10-20/16