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

特許第4442564号
(P4442564)

(45)発行日　平成22年3月31日(2010.3.31)　(24)登録日　平成22年1月22日(2010.1.22)

(51)Int.Cl.		F I		
H04N 5/93	(2006.01)	H04N	5/93	Z
G11B 20/10	(2006.01)	G11B	20/10	321Z
G11B 27/34	(2006.01)	G11B	27/34	S
H04N 5/85	(2006.01)	H04N	5/85	Z
H04N 5/92	(2006.01)	H04N	5/92	H

請求項の数 87　(全 109 頁)　最終頁に続く

(21)出願番号　特願2005-510280 (P2005-510280)
(86)(22)出願日　平成15年11月14日 (2003.11.14)
(86)国際出願番号　PCT/JP2003/014511
(87)国際公開番号　WO2004/049710
(87)国際公開日　平成16年6月10日 (2004.6.10)
審査請求日　平成18年11月7日 (2006.11.7)
(31)優先権主張番号　特願2002-346133 (P2002-346133)
(32)優先日　平成14年11月28日 (2002.11.28)
(33)優先権主張国　日本国(JP)
(31)優先権主張番号　特願2003-22551 (P2003-22551)
(32)優先日　平成15年1月30日 (2003.1.30)
(33)優先権主張国　日本国(JP)
(31)優先権主張番号　特願2003-74441 (P2003-74441)
(32)優先日　平成15年3月18日 (2003.3.18)
(33)優先権主張国　日本国(JP)

(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
(74)代理人　100082762
弁理士　杉浦　正知
(74)代理人　100120640
弁理士　森　幸一
(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
(72)発明者　加藤　元樹
東京都品川区北品川６丁目７番３５号
ソニー株式会社内

審査官　鈴木　明

最終頁に続く

(54)【発明の名称】再生装置、再生方法、再生プログラムおよび記録媒体

(57)【特許請求の範囲】
【請求項１】
コンテンツデータを再生する再生装置において、
少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとが入力される入力手段と、
上記入力手段により入力された上記プログラムコードを格納するコード格納手段と、
上記入力手段により入力された上記画像データを格納する画像データ格納手段と、
上記入力手段により入力された上記動画データをデコードした復号動画データと、上記入力手段により入力された上記字幕データをデコードした復号字幕データとを合成する第１の合成手段と、
上記コード格納手段に格納された上記プログラムコードに基づき、上記画像データ格納手段に格納された上記画像データをデコードした復号画像データと、上記第１の合成手段により合成された動画字幕合成データとを合成する第２の合成手段と
を有することを特徴とする再生装置。
【請求項２】
請求の範囲１に記載の再生装置において、
上記画像データの上記デコードは、上記プログラムコードに基づきユーザからの入力に応じて行われることを特徴とする再生装置。
【請求項３】

請求の範囲１に記載の再生装置において、
上記第２の合成手段による上記合成は、上記プログラムコードに基づきユーザからの入力に応じて行われることを特徴とする再生装置。
【請求項４】
請求の範囲１に記載の再生装置において、
上記プログラムコードは、ＨＴＭＬまたはＸＨＴＭＬに基づき記述されていることを特徴とする再生装置。
【請求項５】
請求の範囲１に記載の再生装置において、
上記プログラムコードは、ＥＣＭＡスクリプトで記述されていることを特徴とする再生装置。
【請求項６】
請求の範囲１に記載の再生装置において、
上記復号動画データの解像度を変換する解像度変換手段をさらに有し、上記第１の合成手段は、上記解像度変換手段の出力と上記復号字幕データとを合成するようにしたことを特徴とする再生装置。
【請求項７】
請求の範囲１に記載の再生装置において、
上記第１の合成手段による上記合成の度合いは、上記復号字幕データに応じて制御されることを特徴とする再生装置。
【請求項８】
請求の範囲１に記載の再生装置において、
上記第２の合成手段による上記合成の度合いは、上記復号画像データに応じて制御されることを特徴とする再生装置。
【請求項９】
請求の範囲１に記載の再生装置において、
上記復号字幕データの色情報を上記画像字幕合成データに用いる色としてＹＣｂＣｒ形式の色情報として出力させる第１の色情報変換手段と、
上記復号画像データの色情報をＲＧＢ形式からＹＣｂＣｒ形式に変換する第２の色情報変換手段と
をさらに有し、
上記第１の合成手段は、上記第１の色情報変換手段で色情報が変換された上記復号字幕データと、上記復号動画データとを合成し、
上記第２の合成手段は、上記第２の色情報変換手段で色情報が変換された上記復号画像データと、上記動画字幕合成データとを合成するようにしたことを特徴とする再生装置。
【請求項１０】
請求の範囲１に記載の再生装置において、
上記リアルタイムストリームは、ＭＰＥＧ２　ＴＳのトランスポートパケットに格納されて上記記録媒体から再生されることを特徴とする再生装置。
【請求項１１】
請求の範囲１に記載の再生装置において、
上記リアルタイムストリームに上記画像データがさらに含まれることを特徴とする再生装置。
【請求項１２】
請求の範囲１に記載の再生装置において、
上記入力手段により入力された１または複数の上記字幕データを格納する字幕データ格納手段をさらに有し、
上記第１の合成手段は、上記字幕データ格納手段に格納された複数の上記字幕データをデコードした複数の上記復号字幕データと、上記復号動画データとを合成できるようにしたことを特徴とする再生装置。

【請求項１３】
請求の範囲１に記載の再生装置において、
上記画像データ格納手段は、１または複数の上記画像データを格納するようにされ、
上記第２の合成手段は、上記画像データ格納手段に格納された複数の上記画像データを復号した複数の上記復号画像データと、上記動画字幕合成データとを合成できるようにしたことを特徴とする再生装置。
【請求項１４】
請求の範囲１に記載の再生装置において、
上記画像データおよび上記字幕データは、上記動画データに対して同期的に表示制御され、共通のデータ構造に格納されることを特徴とする再生装置。
【請求項１５】
請求の範囲１４に記載の再生装置において、
上記データ構造は、少なくとも、表示制御命令と、該表示制御命令により表示制御される１または複数の上記画像データまたは上記字幕データとを有することを特徴とする再生装置。
【請求項１６】
請求の範囲１５に記載の再生装置において、
上記表示制御命令は、上記画像データまたは上記字幕データ自体を変更せずに、該画像データまたは該字幕データの表示属性を変更する命令を含むことを特徴とする再生装置。
【請求項１７】
請求の範囲１６に記載の再生装置において、
上記表示属性は、透明度であることを特徴とする再生装置。
【請求項１８】
請求の範囲１６に記載の再生装置において、
上記表示属性は、表示色であることを特徴とする再生装置。
【請求項１９】
請求の範囲１５に記載の再生装置において、
上記データ構造は、同時に実行される複数の上記表示制御命令をひとまとめにして表示制御命令群として格納し、それぞれ異なる時刻に実行される複数の上記表示制御命令群を格納できるようにしたことを特徴とする再生装置。
【請求項２０】
請求の範囲１４に記載の再生装置において、
上記データ構造に格納される上記画像データまたは上記字幕データは、上記画像データをデコードする際の時間管理情報または上記字幕データをデコードする際の時間管理情報に基づき有効期間が開始され、該データ構造に定義される有効期間終了時刻で有効期間が終了されることを特徴とする再生装置。
【請求項２１】
請求の範囲１に記載の再生装置において、
上記画像データはボタンを表示するボタン画像データであって、上記ボタンの３種類の状態にそれぞれ対応する第１、第２および第３のボタン画像データをユーザの上記操作に従い差し替えて表示するようにしたことを特徴とする再生装置。
【請求項２２】
請求の範囲１４に記載の再生装置において、
上記画像データはボタンを表示するボタン画像データであって、上記ボタンの３種類の状態にそれぞれ対応する第１、第２および第３のボタン画像データを同一の上記データ構造に格納するようにしたことを特徴とする再生装置。
【請求項２３】
請求の範囲１５に記載の再生装置において、
上記表示制御命令は、音声データの再生を制御する音声データ再生制御命令をさらに含めるようにしたことを特徴とする再生装置。

【請求項２４】
請求の範囲２３に記載の再生装置において、
上記音声データ再生制御命令に基づき上記字幕データの表示に対して同期的に音声データの再生を制御するようにしたことを特徴とする再生装置。
【請求項２５】
請求の範囲２３に記載の再生装置において、
上記画像データはボタンを表示するボタン画像データであって、上記音声データ再生制御命令に基づき上記ボタンの３種類の状態にそれぞれ対応する第１、第２および第３のボタン画像データの表示に応じて上記音声データを再生するようにしたことを特徴とする再生装置。
【請求項２６】
請求の範囲２３に記載の再生装置において、
上記音声データ再生制御命令により、上記画像データまたは上記字幕データに対して上記音声データを割り当てることができるようにしたことを特徴とする再生装置。
【請求項２７】
請求の範囲１に記載の再生装置において、
上記入力手段に対して、効果音を再生する音声データがさらに入力されることを特徴とする再生装置。
【請求項２８】
請求の範囲２７に記載の再生装置において、
上記音声データは、上記リアルタイムストリームに多重化されて入力されることを特徴とする再生装置。
【請求項２９】
請求の範囲２８に記載の再生装置において、
上記リアルタイムストリームは画像データがさらに多重化され、該リアルタイムストリームに含まれ多重化される画像データおよび上記音声データは、１のデータ構造にまとめて格納されて入力されることを特徴とする記録媒体。
【請求項３０】
請求の範囲２９に記載の再生装置において、
上記データ構造には、上記リアルタイムストリームに多重化される画像データの表示制御および上記音声データの再生制御を行う制御命令がさらに格納されていることを特徴とする再生装置。
【請求項３１】
請求の範囲２９に記載の再生装置において、
上記データ構造には、上記リアルタイムストリームに多重化される画像データと対応する上記音声データが一組とされて格納されていることを特徴とする再生装置。
【請求項３２】
請求の範囲３１に記載の再生装置において、
上記リアルタイムストリームに多重化される画像データの表示に応じて該画像データと対応する上記音声データが再生されるようにしたことを特徴とする再生装置。
【請求項３３】
請求の範囲３２に記載の再生装置において、
上記リアルタイムストリームに多重化される画像データは、ボタンを表示するボタン画像データであって、上記データ構造には、上記ボタンの状態に対応した上記ボタン画像データと上記音声データとが一組とされて格納されていることを特徴とする再生装置。
【請求項３４】
請求の範囲３３に記載の再生装置において、
上記ボタンは複数の状態を有し、該複数の状態に複数のボタン画像データがそれぞれ対応付けられると共に、該複数のボタン画像データの少なくとも１つに音声データが対応付けられて一組とされて、該複数のボタン画像データと該音声データとが同一の上記データ

構造に格納されていることを特徴とする再生装置。
【請求項３５】
請求の範囲３４に記載の再生装置において、
上記ボタンの上記状態の変化に伴う上記ボタン画像データの表示に応じて該ボタン画像データと一組をなす上記音声データを再生するようにしたことを特徴とする再生装置。
【請求項３６】
請求の範囲３５に記載の再生装置において、
上記データ構造は、同時に実行される複数の上記制御命令をひとまとめにして制御命令群として格納し、それぞれ異なる時刻に実行される複数の上記制御命令群を格納できるようにされたことを特徴とする再生装置。
【請求項３７】
請求の範囲２７に記載の再生装置において、
上記音声データは、ファイルに格納されて上記非リアルタイムストリームとして上記入力手段から入力されることを特徴とする再生装置。
【請求項３８】
請求の範囲３７に記載の再生装置において、
上記リアルタイムストリームに多重化される画像データと、上記音声データを示す情報とが、１のデータ構造にまとめて格納されて入力されるようにしたことを特徴とする再生装置。
【請求項３９】
請求の範囲３８に記載の再生装置において、
上記データ構造は、上記リアルタイムストリームに多重化される画像データと、該画像データと対応する上記音声データを示す情報とが一組とされて格納され、該画像データの表示に応じて該画像データに対応する上記音声データが再生されるようにしたことを特徴とする再生装置。
【請求項４０】
請求の範囲３９に記載の再生装置において、
上記リアルタイムストリームに多重化される画像データは、ボタンを表示するボタン画像データであって、上記データ構造には、上記ボタンの状態に対応した上記ボタン画像データと、該ボタン画像データと対応する上記音声データを示す情報とが一組とされて格納されていることを特徴とする再生装置。
【請求項４１】
請求の範囲４０に記載の再生装置において、
上記ボタンは複数の状態を有し、該複数の状態に複数のボタン画像データがそれぞれ対応付けられると共に、該複数のボタン画像データの少なくとも１つに音声データが対応付けられて一組とされ、該複数のボタン画像データと該音声データを示す情報とが同一の上記データ構造に格納されていることを特徴とする再生装置。
【請求項４２】
請求の範囲４１に記載の再生装置において、
上記データ構造は、上記ボタン画像データの表示に応じて該ボタン画像データと対になる上記音声データを再生するようにした制御命令がさらに格納され、上記制御命令に基づき上記音声データを再生するようにしたことを特徴とする再生装置。
【請求項４３】
請求の範囲４２に記載の再生装置において、
上記データ構造は、同時に実行される複数の上記制御命令をひとまとめにして制御命令群として格納し、それぞれ異なる時刻に実行される複数の上記制御命令群を格納できるようにされていることを特徴とする再生装置。
【請求項４４】
請求の範囲１に記載の再生装置において、
上記画像データをデコードする際のレートの上限を規定するようにしたことを特徴とす

る再生装置。
【請求項４５】
請求の範囲１に記載の再生装置において、
上記復号画像データをプレーンバッファに転送する際の転送レートの上限を規定するようにしたことを特徴とする再生装置。
【請求項４６】
請求の範囲４５に記載の再生装置において、
１または複数の上記復号画像データを完全に含みむ上記プレーンバッファ上の矩形領域を定義し、画像データの描画、更新および消去の際に上記矩形領域を書き換えることで上記転送レートの上限を規定するようにしたことを特徴とする再生装置。
【請求項４７】
請求の範囲１に記載の再生装置において、
上記非リアルタイムストリームは、円盤状記録媒体から再生されて上記入力手段に入力されることを特徴とする再生装置。
【請求項４８】
請求の範囲１に記載の再生装置において、
上記非リアルタイムストリームは、ネットワークから取得されて上記入力手段に入力されることを特徴とする再生装置。
【請求項４９】
請求の範囲１に記載の再生装置において、
上記リアルタイムストリームは、円盤状記録媒体から再生されて上記入力手段に入力されることを特徴とする再生装置。
【請求項５０】
請求の範囲１に記載の再生装置において、
上記リアルタイムストリームは、ネットワークから取得されて上記入力手段に入力されることを特徴とする再生装置。
【請求項５１】
コンテンツデータを再生する再生方法において、
少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとを入力する入力のステップと、
上記入力のステップにより入力された上記プログラムコードをコード格納手段に格納するステップと、
上記入力のステップにより入力された上記画像データを画像データ格納手段に格納するステップと、
上記入力のステップにより入力された上記動画データをデコードした復号動画データと、上記入力のステップにより入力された上記字幕データをデコードした復号字幕データとを合成する第１の合成のステップと、
上記コード格納手段に格納された上記プログラムコードに基づき、上記画像データ格納手段に格納された上記画像データをデコードした復号画像データと、上記第１の合成のステップにより合成された動画字幕合成データとを合成する第２の合成のステップと
を有することを特徴とする再生方法。
【請求項５２】
コンテンツデータを再生する再生方法をコンピュータ装置に実行させる再生プログラムにおいて、
上記再生方法は、
少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとを入力する入力のステップと、
上記入力のステップにより入力された上記プログラムコードをコード格納手段に格納す

るステップと、
上記入力のステップにより入力された上記画像データを画像データ格納手段に格納するステップと、
上記入力のステップにより入力された上記動画データをデコードした復号動画データと、上記入力のステップにより入力された上記字幕データをデコードした復号字幕データとを合成する第１の合成のステップと、
上記コード格納手段に格納された上記プログラムコードに基づき、上記画像データ格納手段に格納された上記画像データをデコードした復号画像データと、上記第１の合成のステップにより合成された動画字幕合成データとを合成する第２の合成のステップと
を有することを特徴とする再生プログラム。

【請求項５３】
コンテンツデータを再生する再生方法をコンピュータ装置に実行させる再生プログラムが記録された記録媒体において、
上記再生方法は、
少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとを入力する入力のステップと、
上記入力のステップにより入力された上記プログラムコードをコード格納手段に格納するステップと、
上記入力のステップにより入力された上記画像データを画像データ格納手段に格納するステップと、
上記入力のステップにより入力された上記動画データをデコードした復号動画データと、上記入力のステップにより入力された上記字幕データをデコードした復号字幕データとを合成する第１の合成のステップと、
上記コード格納手段に格納された上記プログラムコードに基づき、上記画像データ格納手段に格納された上記画像データをデコードした復号画像データと、上記第１の合成のステップにより合成された動画字幕合成データとを合成する第２の合成のステップと
を有することを特徴とする記録媒体。

【請求項５４】
コンテンツデータが記録された円盤状の形状を有する記録媒体において、
少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとが記録され、
上記プログラムコードに基づき、再生後に画像データ格納手段に格納された上記画像データをデコードした復号画像データと、再生された上記動画データをデコードした復号動画データおよび再生された上記字幕データをデコードした復号字幕データが合成された動画字幕合成データとが合成されることを特徴とする記録媒体。

【請求項５５】
請求の範囲５４に記載の記録媒体において、
上記画像データおよび上記字幕データは、上記動画データに対して同期的に表示制御され、共通のデータ構造に格納されて記録されることを特徴とする記録媒体。

【請求項５６】
請求の範囲５５に記載の記録媒体において、
上記データ構造は、少なくとも、表示制御命令と、該表示制御命令により表示制御される１または複数の上記画像データまたは上記字幕データとを有することを特徴とする記録媒体。

【請求項５７】
請求の範囲５６に記載の記録媒体において、
上記表示制御命令は、上記画像データまたは上記字幕データ自体を変更せずに、該画像データまたは該字幕データの表示属性を変更する命令を含むことを特徴とする記録媒体。

【請求項５８】
請求の範囲５７に記載の記録媒体において、
上記表示属性は、透明度であることを特徴とする記録媒体。
【請求項５９】
請求の範囲５７に記載の記録媒体において、
上記表示属性は、表示色であることを特徴とする記録媒体。
【請求項６０】
請求の範囲５６に記載の記録媒体において、
上記データ構造は、同時に実行される複数の上記表示制御命令をひとまとめにして表示制御命令群として格納し、それぞれ異なる時刻に実行される複数の上記表示制御命令群を格納できるようにして記録されることを特徴とする記録媒体。
【請求項６１】
請求の範囲５５に記載の記録媒体において、
上記画像データをデコードする際の時間管理情報または上記字幕データをデコードする際の時間管理情報に基づき有効期間が開始され、上記データ構造に定義される有効期間終了時刻で有効期間が終了されるようにした上記画像データまたは上記字幕データが上記データ構造に格納されて記録されることを特徴とする記録媒体。
【請求項６２】
請求の範囲５４に記載の記録媒体において、
上記画像データはボタンを表示するボタン画像データであって、上記ボタンの３種類の状態にそれぞれ対応する第１、第２および第３のボタン画像データをユーザの上記操作に従い差し替えて表示するようにしたことを特徴とする記録媒体。
【請求項６３】
請求の範囲５５に記載の記録媒体において、
上記画像データはボタンを表示するボタン画像データであって、上記ボタンの３種類の状態にそれぞれ対応する第１、第２および第３のボタン画像データを同一の上記データ構造に格納して記録することを特徴とする記録媒体。
【請求項６４】
請求の範囲５６に記載の記録媒体において、
上記表示制御命令は、音声データの再生を制御する音声データ再生制御命令をさらに含めて上記データ構造に格納され記録されることを特徴とする記録媒体。
【請求項６５】
請求の範囲６４に記載の記録媒体において、
上記音声データ再生制御命令に基づき上記字幕データの表示に対して同期的に音声データの再生を制御するようにした上記表示制御命令が上記データ構造に格納されて記録されることを特徴とする記録媒体。
【請求項６６】
請求の範囲６４に記載の記録媒体において、
上記画像データはボタンを表示するボタン画像データであって、上記音声データ再生制御命令に基づき上記ボタンの３種類の状態にそれぞれ対応する第１、第２および第３のボタン画像データの表示に応じて上記音声データを再生するようにした上記表示制御命令が上記データ構造に格納されて記録されることを特徴とする記録媒体。
【請求項６７】
請求の範囲６４に記載の記録媒体において、
上記音声データ再生制御命令により上記画像データまたは上記字幕データに対して上記音声データを割り当てることができるようにした上記表示制御命令が上記データ構造に格納されて記録されることを特徴とする記録媒体。
【請求項６８】
請求の範囲５４に記載の記録媒体において、
上記画像データがさらに上記リアルタイムストリームに多重化されて記録されることを

特徴とする記録媒体。
【請求項６９】
請求の範囲５４に記載の記録媒体において、
効果音を再生する音声データがさらに記録されることを特徴とする記録媒体。
【請求項７０】
請求の範囲６９に記載の記録媒体において、
上記音声データは、上記リアルタイムストリームに多重化されて記録されることを特徴とする記録媒体。
【請求項７１】
請求の範囲７０に記載の記録媒体において、
上記リアルタイムストリームは画像データがさらに多重化され、該リアルタイムストリームに含まれる画像データおよび上記音声データを、１のデータ構造にまとめて格納して記録するようにしたことを特徴とする記録媒体。
【請求項７２】
請求の範囲７１に記載の記録媒体において、
上記データ構造には、上記リアルタイムストリームに多重化される画像データの表示制御および上記音声データの再生制御を行う制御命令がさらに格納されることを特徴とする記録媒体。
【請求項７３】
請求の範囲７１に記載の記録媒体において、
上記データ構造には、上記リアルタイムストリームに多重化される画像データに対応する上記音声データが一組とされて格納されることを特徴とする記録媒体。
【請求項７４】
請求の範囲７３に記載の記録媒体において、
上記リアルタイムストリームに多重化される画像データは、ボタンを表示するボタン画像データであって、上記データ構造には、上記ボタンの状態に対応した上記ボタン画像データと上記音声データとが一組とされて格納されることを特徴とする記録媒体。
【請求項７５】
請求の範囲７４に記載の記録媒体において、
上記ボタンは複数の状態を有し、該複数の状態に複数のボタン画像データがそれぞれ対応付けられると共に、該複数のボタン画像データの少なくとも１つに音声データが対応付けられ一組とされて、該複数のボタン画像データと該音声データとを同一の上記データ構造に格納することを特徴とする記録媒体。
【請求項７６】
請求の範囲７５に記載の記録媒体において、
上記ボタンの上記状態の変化に伴う上記ボタン画像データの表示に応じて該ボタン画像データと一組をなす上記音声データを再生するようにされていることを特徴とする記録媒体。
【請求項７７】
請求の範囲７６に記載の記録媒体において、
上記データ構造は、同時に実行される複数の上記制御命令をひとまとめにして制御命令群として格納し、それぞれ異なる時刻に実行される複数の上記制御命令群を格納できるようにして記録されることを特徴とする記録媒体。
【請求項７８】
請求の範囲６９に記載の記録媒体において、
上記音声データは、ファイルに格納されて記録されることを特徴とする記録媒体。
【請求項７９】
請求の範囲７８に記載の記録媒体において、
上記リアルタイムストリームに多重化される画像データと、上記音声データを示す情報とを、１のデータ構造にまとめて格納して記録するようにしたことを特徴とする記録媒体

。
【請求項８０】
請求の範囲７９に記載の記録媒体において、
上記データ構造には、上記リアルタイムストリームに多重化される画像データと、該画像データと対応する上記音声データを示す情報とが一組とされて格納されることを特徴とする記録媒体。
【請求項８１】
請求の範囲８０に記載の記録媒体において、
上記リアルタイムストリームに多重化される画像データは、ボタンを表示するボタン画像データであって、上記データ構造には、上記ボタンの状態に対応した上記ボタン画像データと、該ボタン画像データと対応する上記音声データを示す情報とが一組とされて格納されることを特徴とする記録媒体。
【請求項８２】
請求の範囲８１に記載の記録媒体において、
上記ボタンは複数の状態を有し、該複数の状態に複数のボタン画像データがそれぞれ対応付けられると共に、該複数のボタン画像データの少なくとも１つに音声データが対応付けられ一組とされて、該複数のボタン画像データと該音声データを示す情報とが同一の上記データ構造に格納されることを特徴とする記録媒体。
【請求項８３】
請求の範囲８２に記載の記録媒体において、
上記ボタン画像データの表示に応じて該ボタン画像データと対になる上記音声データを再生するようにした制御命令が上記データ構造にさらに格納されることを特徴とする記録媒体。
【請求項８４】
請求の範囲８３に記載の記録媒体において、
上記データ構造は、同時に実行される複数の上記制御命令をひとまとめにして制御命令群として格納し、それぞれ異なる時刻に実行される複数の上記制御命令群を格納できるようにして記録されることを特徴とする記録媒体。
【請求項８５】
請求の範囲５４に記載の記録媒体において、
予め規定された、上記画像データをデコードする際のレートの下限を満足するようにして上記画像データが記録されることを特徴とする記録媒体。
【請求項８６】
請求の範囲５４に記載の記録媒体において、
予め規定された、上記復号画像データをプレーンバッファに転送する際の転送レートの下限を満足するようにして上記画像データが記録されることを特徴とする記録媒体。
【請求項８７】
請求の範囲８６に記載の記録媒体において、
１または複数の上記復号画像データを完全に含みむ上記プレーンバッファ上の矩形領域を定義し、画像データの描画、更新および消去の際に上記矩形領域を書き換えることで上記転送レートの下限を満足するようにしたことを特徴とする記録媒体。
【発明の詳細な説明】
【技術分野】
この発明は、ブルーレイディスク（Ｂｌｕ－ｒａｙ　Ｄｉｓｃ）といった大容量の記録媒体に記録されたプログラムに対するユーザによるインタラクティブな操作を可能とする再生装置、再生方法、再生プログラムおよび記録媒体に関する。
【背景技術】
近年、記録可能で記録再生装置から取り外し可能なディスク型記録媒体の規格として、Ｂｌｕ－ｒａｙ　Ｄｉｓｃ（ブルーレイディスク）規格が提案されている。Ｂｌｕ－ｒａｙ　Ｄｉｓｃ規格では、記録媒体として直径１２ｃｍ、カバー層０．１ｍｍのディスクを

用い、光学系として波長４０５ｎｍの青紫色レーザ、開口数０．８５の対物レンズを用いて、最大で２７ＧＢ（ギガバイト）の記録容量を実現している。これにより、日本のＢＳディジタルハイビジョン放送を、画質を劣化させることなく２時間以上記録することが可能である。

この記録可能光ディスクに記録するＡＶ（Ａｕｄｉｏ／Ｖｉｄｅｏ）信号のソース（供給源）としては、従来からの、例えばアナログテレビジョン放送によるアナログ信号によるものと、例えばＢＳディジタル放送をはじめとするディジタルテレビジョン放送によるディジタル信号によるものとが想定されている。Ｂｌｕ－ｒａｙ Ｄｉｓｃ規格では、これらの放送によるＡＶ信号を記録する方法を定めた規格は、既に作られている。

現状のＢｌｕ－ｒａｙ Ｄｉｓｃの派生規格として、映画や音楽などが予め記録された、再生専用の記録媒体を開発する動きが進んでいる。映画や音楽を記録するためのディスク状記録媒体としては、既にＤＶＤ（Ｄｉｇｉｔａｌ Ｖｅｒｓａｔｉｌｅ Ｄｉｓｃ）が広く普及しているが、このＢｌｕ－ｒａｙ Ｄｉｓｃの規格に基づいた再生専用光ディスクは、Ｂｌｕ－ｒａｙ Ｄｉｓｃの大容量および高速な転送速度などを活かし、ハイビジョン映像を高画質なままで２時間以上収録できる点が、既存のＤＶＤとは大きく異なり、優位である。

現状のＢｌｕ－ｒａｙ Ｄｉｓｃの規格では、ディスクに記録されている映像コンテンツの一覧を画面表示する方法や、その一覧表上にカーソルを表示させ、再生したい映像コンテンツをユーザに選択させるなどといったユーザインターフェイスに関する機能が定められていない。これらの機能は、Ｂｌｕ－ｒａｙ Ｄｉｓｃに対する記録再生を行う記録再生装置本体によって実現されている。そのため、同一の記録媒体を再生した場合でも、再生に用いた記録再生装置によってコンテンツ一覧画面のレイアウトが異なってしまい、ユーザインタフェースにも差が生じ、必ずしもユーザにとって使い易いものではない。再生専用ディスクとしては、再生機器によらず、ディスク（コンテンツ）制作者が意図した通りのメニュー画面などが表示され、意図通りのユーザインターフェイスが実現される必要がある。

映像コンテンツの再生中に選択画面が表示され、ユーザの選択によってストーリーが分岐していくマルチストーリーの機能は、一般にインタラクティブ機能とも呼ばれる。このインタラクティブ機能を実現するためには、ディスク制作者が再生順序や分岐を定めたシナリオを作り、そのシナリオをプログラム言語、スクリプト言語等を使って記述し、ディスクに記録しておく必要がある。再生装置側では、そのプログラムを読み込み、実行することで、制作者の意図に従った映像コンテンツの再生や、分岐のための選択画面提示を実現することになる。

現状のＢｌｕ－ｒａｙ Ｄｉｓｃ規格（Ｂｌｕ－ｒａｙ Ｄｉｓｃ Ｒｅｗｒｉｔａｂｌｅ Ｆｏｒｍａｔ Ｖｅｒ１．０）では、この制作者の意図通りのユーザインターフェイスを実現するための、メニュー画面や分岐選択画面の構成方法、ユーザ入力に対する処理を記述する方法が定められていない。現状では、Ｂｌｕ－ｒａｙ Ｄｉｓｃを用いて、制作者が意図したシナリオ通りの再生を、再生装置の製造メーカや機種に左右されることなく互換性を持たせた形で実現することが難しいという問題点があった。

再生専用ディスクにおいては、同一の被写体を複数のカメラで撮影し、ユーザが好みのアングルでの視聴を選択できるマルチアングル機能が制作者側より求められており、これを実現できる仕組みを用意する必要がある。

再生専用ディスクにおいては、字幕を表示する仕組みが不可欠である。しかしながら、この字幕表示についても、現状のＢｌｕ－ｒａｙ Ｄｉｓｃ規格では、定められていない。

従来から、例えばＤＶＤ（Ｄｉｇｉｔａｌ Ｖｅｒｓａｔｉｌｅ Ｄｉｓｃ）の規格においては、上述のようなインタラクティブな機能が既に実現されていた。ＤＶＤビデオでは、動画を再生中にリモートコントロールコマンダなどを用いてメニュー画面を呼び出し、メニュー画面上に配置されたボタンを選択するなどして、再生場面を変更するなどの処理が可能であった。字幕を表示する仕組みも規定されていた。字幕表示については、予め

用意されている日本語字幕と英語字幕とを切り換えて表示させることができた。マルチアングル機能も実現されていた。

ＤＶＤの場合、メニュー画面を固定的なサブピクチャデータにより構成し、メニュー画面が呼び出された際に、動画データにこのサブピクチャデータを合成して表示する。特開平１０－３０８９２４号公報に、このように動画データにサブピクチャデータを合成して記録可能なＤＶＤに記録する構成が記載されている。

従来技術によるメニュー画面の一例の表示について、概略的に説明する。例えば、映画が収録されたＤＶＤを再生装置で再生する際に、映画本編が再生されるのに先立って、メニュー画面が表示される。一般に、メニュー画面には、複数のボタンが配置される。各ボタンには、それぞれ特定の動作が割り当てられており、ボタンを選択し実行を指示すると、選択されたボタンに割り当てられた動作が実行される。「映画本編」と表示されたボタンを選択し実行を指示することで、当該ボタンに割り当てられた動作が実行され、当該ＤＶＤに収録された映画の本編が再生されることになる。

ユーザは、リモートコントロールコマンダ（以下、リモコンと略称する）に設けられた、上下左右の方向をそれぞれ指示するようにされたキー（方向キー）などを操作して、メニュー画面に表示されたボタンの一つを選択状態とし、そこで決定キーを操作することで、選択されたボタンに割り当てられた動作を実行状態とする。ボタンには、通常（非選択）状態、選択状態及び実行状態という３種類の状態が存在し、ユーザが区別し易いように、それぞれ画像や色などが異ならされている。一般に、選択状態および実行状態のボタンは、同時には一つしか存在しない。

ＤＶＤビデオの場合、ボタンの表示は、それぞれサブピクチャ、ハイライトと呼ばれる２種類のデータにより実現される。第１図は、従来技術による一例のＤＶＤメニュー画面３００を示す。メニュー画面３００は、「タイトルメニュー」と名付けられ、それぞれ「本編再生」、「映像特典」および「音声設定」と表示された３個のボタン３０１Ａ、３０１Ｂおよび３０１Ｃが配置される。この第１図の例では、「本編再生」ボタン３０１Ａの外枠の色がボタン３０１Ａの元の外枠の色と異なった色にされており、「本編再生」ボタン３０１Ａが選択状態となっていることが示される。

この状態で、ユーザがリモコンの方向キーを所定に操作することにより、第２図Ａ、第２図Ｂおよび第２図Ｃにそれぞれ例示されるように、選択状態のボタンを変更することができる。選択されたボタンは、上述の第１図と同様に、外枠の色が非選択状態の場合に対して異ならされる。第１図の状態で、ユーザがリモコンに設けられた決定ボタンを操作すると、その時点で選択状態にある「本編再生」ボタン３０１Ａが第３図に一例が示されるように、実行状態を表す色に変化される。その後、メニュー画面３００が消え、本編が再生されることになる。以上が、ＤＶＤビデオによるボタンの基本動作である。

ところで、第１図に例示されるメニュー画面３００は、第４図Ａ、第４図Ｂおよび第４図Ｃにそれぞれ一例が示されるように、背景画像３１０、サブピクチャ３１１およびハイライト３１２の３種類のデータから構成される。背景画像３１０は、静止画や、当該ＤＶＤに収録されるコンテンツ本体の動画などが表示される。

サブピクチャ３１１は、第５図に例示されるように、１ピクセル当たり２ビットの情報で表現された１枚のビットマップ画像と、４色の色情報（Ａ０，Ｂ０，Ｃ０，Ｄ０）と、当該サブピクチャ３１１の表示開始位置を示す座標（Ｘ，Ｙ）とを有する。色情報Ａ０、Ｂ０、Ｃ０およびＤ０は、それぞれ１組のＲ（赤）Ｇ（緑）Ｂ（青）データからなる１色の色情報データであって、Ｒ、Ｇ、Ｂ各色について各々８ビットの情報を有する。ビットマップ画像は、１ピクセルにつき２ビットの情報を有し、２ビットを用いて上述の４色の色情報（Ａ０，Ｂ０，Ｃ０，Ｄ０）の中から１色をピクセル毎に選択する。色情報は、透明度情報も含まれており、サブピクチャ３１１に対して背景画像３１０が透けて見える領域を設けることができる。サブピクチャ３１１の表示位置は、サブピクチャ３１１の左上隅の位置が背景画像３１０に対する相対的な座標（Ｘ，Ｙ）で示される。

サブピクチャ３１１は、表示開始および停止時刻を示す情報、当該サブピクチャ３１１に対してフェードイン、フェードアウトといった視覚効果を施すコマンドを有することが

できる。

ＤＶＤビデオにおいては、同時に複数のビットマップ画像を表示することができない。そのため、上述した第１図に例示されるような、複数のボタンが配置されたようなメニュー画面３００の表示は、第４図Ｂに例示されるような、３個のボタン画像を含む１枚の大きなビットマップ画像によりなされる。第４図Ｂのサブピクチャ３１１のビットマップ画像において、ボタン３０１Ａ、３０１Ｂおよび３０１Ｃの外側の領域を透明領域として設定することで、このサブピクチャ３１１と背景画像３１０とを合成した際に、ボタン３０１Ａ、３０１Ｂおよび３０１Ｃの表示領域以外に背景画像３１０を透過させて表示させることができる。

ハイライト３１２は、サブピクチャ３１１に用いられている４色を、他の４色に置き換えるための情報である。第５図に例示されるように、ハイライト３１２は、色情報として、選択状態の色情報（Ａ１，Ｂ１，Ｃ１，Ｄ１）および実行状態の色情報（Ａ２，Ｂ２，Ｃ２，Ｄ２）を有する。これらの色情報も、上述のサブピクチャ３１１と同様、ＲＧＢそれぞれ８ビットで表現される４色の色情報である。

また、ハイライト３１２は、位置情報として、色を置き換える領域の座標の組を有する。色を置き換える範囲は、サブピクチャ３１１の画像全体に限らず、サブピクチャ３１１内の一部の矩形領域とすることができる。このハイライト３１２により色を変化させるサブピクチャ３１１内の矩形領域の数がユーザが選択可能なボタンの数に相当する。それぞれの矩形領域の表示位置は、左上隅および右下隅の位置が座標（Ｘ，Ｙ）で示される。例えば、ボタン３０１Ａに対応するハイライト３１２Ａの位置は、座標（Ｘ１，Ｙ１）および（Ｘ１’，Ｙ１’）で示される。ボタン３０１Ｂおよび３０１Ｃにそれぞれ対応するハイライト３１２Ｂおよび３１２Ｃも、同様である。

一例として、ハイライト３１２Ａは、背景画像３１０の座標（Ｘ１，Ｙ１）および（Ｘ１’，Ｙ１’）で示される領域の色情報（Ａ０，Ｂ０，Ｃ０，Ｄ０）を、選択状態の色として指定される色情報（Ａ１，Ｂ１，Ｃ１，Ｄ１）に置き換える。このとき、背景画像３１０の色情報Ａ０がハイライト３１２Ａの色情報Ａ１に置き換えられる。同様に、背景画像３１０の色情報Ｂ０が色情報Ｂ１に、色情報Ｃ０が色情報Ｃ１に、色情報Ｄ０が色情報Ｄ１に、それぞれ置き換えられる。

このハイライト３１２による色の置き換えの例を、第１図、第２図Ａ、第２図Ｂおよび第２図Ｃ、ならびに、第３図を用いて上述した、メニュー画面３００におけるボタン３０１Ａの状態に応じた表示変化に対応させて説明する。ボタン３０１Ａが非選択状態のときには、ボタン３０１Ａの枠、表面および文字がそれぞれ色情報Ｂ０、色情報Ｃ０および色情報Ｄ０を用いて表示されているものとする。ボタン３０１Ａを選択状態とすると、ボタン３０１Ａの枠色Ｂ０がハイライト３１２Ａの選択状態の対応する色情報Ｂ１で置き換えられる。表面色Ｃ０および文字色Ｄ０は、変化しない。次いで、ボタン３０１Ａが実行状態とされると、選択状態の色に対して、ボタン３０１Ａの表面色Ｃ０が色情報Ｃ１に置き換えられる。枠色Ｂ１および文字色Ｄ０は、選択状態に対して変化しない。

ＤＶＤビデオにおいて、通常の映像再生時は、背景画像３１０に相当する画像だけが表示されている。字幕付きの映画などの場合は、映画本編が再生されている背景画像３１０と、字幕が表示されているサブピクチャ３１１とが合成された状態で表示がなされている。

しかしながら、サブピクチャ３１１、選択状態を表すハイライト３１２、実行状態を表すハイライト３１２は、それぞれ最大４色しか用いることができず、色数の多いサブピクチャを表示することができないという問題点があった。

ハイライト３１２は、サブピクチャ３１１の色を置き換え変更するだけであったため、選択状態や実行状態においてボタン中に表示されている文字を変えることや、ボタンの形状が変化するような効果を実現することができず、表現力に乏しいという問題点があった。

字幕表示とボタン表示とを、サブピクチャ３１１を用いた同一の仕組みで実現していたため、字幕とボタンとをそれぞれ独立して表示制御することや、字幕あるいはボタン表示

に透明度を設定してこれらを重ね合わせて表示させるといった、合成処理もできないという問題点があった。

メニュー画面を呼び出した際に、背景に表示される動画データの再生が中断されていた。このように、例えインタラクティブな機能が実現されていても、その機能を実現するためのユーザインターフェイスの自由度が低いという問題点があった。

字幕の表示や変化に同期させて効果音を発生させる仕組みが規格で定められていないため、字幕に同期させて効果音を発生させるようなことができないという問題点があった。

ボタンについても、ユーザがボタンを選択状態にした際に発せられる効果音や、ボタンの選択状態時に決定キーを操作するなどして、当該ボタンを実行状態にした際に発せられるクリック音のような効果音を発生させる仕組みが規格で定められていなかったため、表現豊かなユーザインターフェイスを実現することが困難であるという問題点があった。

自由度の高いユーザインタフェースを実現する上で、ボタンの描画速度や更新速度、ユーザ入力に対する応答性が問題となるが、それらを見積もるためのグラフィックスのデコーダモデルが必要であった。

ここでいう効果音とは、動画プレーン上に表示される動画や静止画に同期して再生される音声（例えば、映画などにおいて、映画映像と対になって収録されている音声）とは別途に用意された音声データが、字幕やボタンの表示制御に同期して再生される音声のことをいう。

【発明の開示】

この発明の目的は、大容量の再生専用光ディスクにおいて自由度の高いユーザインターフェイスを実現するための再生装置、再生方法、再生プログラムおよび記録媒体を提供することにある。

この発明の他の目的は、大容量の再生専用光ディスクにおいて、より表現豊かなユーザインターフェイスを実現する再生装置、再生方法、再生プログラムおよび記録媒体を提供することにある。

上述した課題を解決するために、第１の発明は、コンテンツデータを再生する再生装置において、少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとが入力される入力手段と、入力手段により入力されたプログラムコードを格納するコード格納手段と、入力手段により入力された画像データを格納する画像データ格納手段と、入力手段により入力された動画データをデコードした復号動画データと、入力手段により入力された字幕データをデコードした復号字幕データとを合成する第１の合成手段と、コード格納手段に格納されたプログラムコードに基づき、画像データ格納手段に格納された画像データをデコードした復号画像データと、第１の合成手段により合成された動画字幕合成データとを合成する第２の合成手段とを有することを特徴とする再生装置である。

第２の発明は、コンテンツデータを再生する再生方法において、少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとを入力する入力のステップと、入力のステップにより入力されたプログラムコードをコード格納手段に格納するステップと、入力のステップにより入力された画像データを画像データ格納手段に格納するステップと、入力のステップにより入力された動画データをデコードした復号動画データと、入力のステップにより入力された字幕データをデコードした復号字幕データとを合成する第１の合成のステップと、コード格納手段に格納されたプログラムコードに基づき、画像データ格納手段に格納された画像データをデコードした復号画像データと、第１の合成のステップにより合成された動画字幕合成データとを合成する第２の合成のステップとを有することを特徴とする再生方法である。

第３の発明は、コンテンツデータを再生する再生方法をコンピュータ装置に実行させる再生プログラムにおいて、再生方法は、少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なく

とも動画データおよび字幕データを含むリアルタイムストリームとを入力する入力のステップと、入力のステップにより入力されたプログラムコードをコード格納手段に格納するステップと、入力のステップにより入力された画像データを画像データ格納手段に格納するステップと、入力のステップにより入力された動画データをデコードした復号動画データと、入力のステップにより入力された字幕データをデコードした復号字幕データとを合成する第１の合成のステップと、コード格納手段に格納されたプログラムコードに基づき、画像データ格納手段に格納された画像データをデコードした復号画像データと、第１の合成のステップにより合成された動画字幕合成データとを合成する第２の合成のステップとを有することを特徴とする再生プログラムである。

第４の発明は、コンテンツデータを再生する再生方法をコンピュータ装置に実行させる再生プログラムが記録された記録媒体において、再生方法は、少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとを入力する入力のステップと、入力のステップにより入力されたプログラムコードをコード格納手段に格納するステップと、入力のステップにより入力された画像データを画像データ格納手段に格納するステップと、入力のステップにより入力された動画データをデコードした復号動画データと、入力のステップにより入力された字幕データをデコードした復号字幕データとを合成する第１の合成のステップと、コード格納手段に格納されたプログラムコードに基づき、画像データ格納手段に格納された画像データをデコードした復号画像データと、第１の合成のステップにより合成された動画字幕合成データとを合成する第２の合成のステップとを有することを特徴とする記録媒体である。

第５の発明は、コンテンツデータが記録された円盤状の形状を有する記録媒体において、少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとが記録され、プログラムコードに基づき、再生後に画像データ格納手段に格納された画像データをデコードした復号画像データと、再生された動画データをデコードした復号動画データおよび再生された字幕データをデコードした復号字幕データが合成された動画字幕合成データとが合成されることを特徴とする記録媒体である。

上述したように、第１、第２、第３および第４の発明は、少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとを入力し、入力されたプログラムコードおよび画像データをコード格納手段および画像データ格納手段にそれぞれ格納し、入力された動画データおよび字幕データをそれぞれデコードした復号動画データおよび復号字幕データとを合成して動画字幕合成データとし、コード格納手段に格納されたプログラムコードに基づき、画像データ格納手段に格納された画像データをデコードした復号画像データと動画字幕合成データとを合成するようにしているため、再生時に、同一の画像データを用いた操作画面を、異なるタイミングでそれぞれ表示させることが容易である。

第５の発明は、少なくともプログラムコードおよびユーザに対して操作を促す操作画面を構成する画像データを含む非リアルタイムストリームと、少なくとも動画データおよび字幕データを含むリアルタイムストリームとが記録され、プログラムコードに基づき、再生後に画像データ格納手段に格納された画像データをデコードした復号画像データと、再生された動画データをデコードした復号動画データおよび再生された字幕データをデコードした復号字幕データが合成された動画字幕合成データとが合成されるようにしているため、再生時に、同一の画像データを用いた操作画面を、異なるタイミングでそれぞれ表示させることが容易である。

【図面の簡単な説明】

第１図は、従来技術による一例のＤＶＤメニュー画面を示す略線図、第２図Ａ、第２図Ｂおよび第２図Ｃは、リモコン操作により選択状態のボタンを変更する様子を示す略線図

、第３図は、決定ボタンの操作によりボタンが実行状態を表す色に変更される様子を示す略線図、第４図Ａ、第４図Ｂおよび第４図Ｃは、従来技術によるメニュー画面の構成の例を示す略線図、第５図は、従来技術によるサブピクチャの一例のデータ構成を示す略線図、第６図は、ＡＶストリームファイルの再生順序指定の仕組みを模式的に示す略線図、第７図は、クリップＡＶストリーム、クリップ情報、クリップ、プレイアイテムおよびプレイリストの関係を示すＵＭＬ図、第８図は、複数のプレイリストから同一のクリップを参照する方法を説明するための略線図、第９図は、記録媒体に記録されるファイルの管理構造を説明するための略線図、第１０図は、ファイル「ｉｎｆｏ．ｂｄａｖ」の一例の構造を表すシンタクスを示す略線図、第１１図は、ブロックＵＩＡｐｐＩｎｆｏＢＤＡＶ（）の一例の構造を表すシンタクスを示す略線図、第１２図は、ブロックＴａｂｌｅＯｆＰｌａｙＬｉｓｔｓ（）の一例の構造を表すシンタクスを示す略線図、第１３図は、ファイル「ｘｘｘｘｘ．ｒｐｌｓ」および「ｙｙｙｙｙ．ｖｐｌｓ」の一例の構造を表すシンタクスを示す略線図、第１４図は、ブロックＵＩＡｐｐＩｎｆｏＰｌａｙＬｉｓｔ（）の一例の構造を表すシンタクスを示す略線図、第１５図は、ブロックＰｌａｙＬｉｓｔ（）の一例の構造を表すシンタクスを示す略線図、第１６図は、ブロックＰｌａｙＩｔｅｍ（）の一例の構造を表すシンタクスを示す略線図、第１７図は、ブリッジクリップを説明するための略線図、第１８図は、ブロックＰｌａｙＬｉｓｔＭａｒｋ（）の一例の構造を表すシンタクスを示す略線図、第１９図は、ファイル「ｚｚｚｚｚ．ｃｌｐｉ」の一例の構造を表すシンタクスを示す略線図、第２０図は、この発明の実施の一形態で画像の表示系として用いられるプレーン構造の一例を示す略線図、第２１図は、動画プレーン、字幕プレーンおよびグラフィクスプレーンの一例の解像度および表示可能色を示す略線図、第２２図は、動画プレーン、字幕プレーンおよびグラフィクスプレーンを合成する一例の構成を示すブロック図、第２３図は、パレットの入出力データの一例を示す略線図、第２４図は、パレットに格納される一例のパレットテーブルを示す略線図、第２５図は、グラフィクスプレーンに表示されるメニュー画面の一例を示す略線図、第２６図は、シナリオの一例の内部構造を示す略線図、第２７図は、シナリオの一例の内部構造を示す略線図、第２８図は、シナリオの構成の分類を説明するための略線図、第２９図Ａ、第２９図Ｂおよび第２９図Ｃは、プレイリストの構成の分類を説明するための略線図、第３０図は、タイトルおよびチャプタについて説明するための略線図、第３１図は、ＢＤ仮想プレーヤモデルについて説明するための略線図、第３２図Ａ、第３２図Ｂおよび第３２図Ｃは、ＢＤ仮想プレーヤにおいて独自に定義される一例のイベントを示す略線図、第３３図Ａ、第３３図Ｂ、第３３図Ｃ、第３３図Ｄ、第３３図Ｅ、第３３図Ｆ、第３３図Ｇおよび第３３図Ｈは、この発明の実施の一形態で定義される、ＢＤ仮想プレーヤが有するコマンドの一例を示す略線図、第３４図Ａ、第３４図Ｂ、第３４図Ｃ、第３４図Ｄ、第３４図Ｅ、第３４図Ｆ、第３４図Ｇ、第３４図Ｈ、第３４図Ｉおよび第３４図Ｊは、この発明の実施の一形態で定義される、ＢＤ仮想プレーヤが有するコマンドの一例を示す略線図、第３５図Ａおよび第３５図Ｂは、独自コマンドを記述言語として用いたシナリオで記述されるコマンドによるＢＤ仮想プレーヤの動作を概略的に示すフローチャート、第３６図Ａおよび第３６図Ｂは、プレイリストの再生動作を説明するためのフローチャート、第３７図Ａおよび第３７図Ｂは、シナリオの一例の階層構造を示す略線図、第３８図は、ディスクに記録されるシナリオ構成の一例を示す略線図、第３９図は、シナリオを構成する際に必要とされる一例のファイルを一覧して示す略線図、第４０図は、シナリオがディスクに記録される際の一例のディレクトリ構造を示す略線図、第４１図は、スクリプトファイルのより具体的な記述の例を示す略線図、第４２図は、スクリプトファイルのより具体的な記述の例を示す略線図、第４３図は、ＨＴＭＬファイルのより具体的な記述の例を示す略線図、第４４図は、スクリプトファイルのより具体的な記述の例を示す略線図、第４５図は、記録媒体に記録されるファイルの管理構造を説明するための略線図、第４６図は、シナリオを記述するためのシナリオファイル（ｓｃｅｎａｒｉｏ．ｈｄｍｖ）の一例の構造を表すシンタクスを示す略線図、第４７図は、ブロックＡｕｔｏｐｌａｙ（）の一例の構造を表すシンタクスを示す略線図、第４８図は、ブロックＳｃｅｎａｒｉｏ（）の一例の構造を表すシンタクス

を示す略線図、第４９図は、ブロックｅｎｔｒｙｌｉｓｔ．ｄａｔａの一例の構造を表すシンタクスを示す略線図、第５０図は、ブロックＡｐｐＩｎｆｏ（）の一例の構造を表すシンタクスを示す略線図、第５１図は、ブロックＳｃｅｎａｒｉｏＥｎｔｒｙ（）の一例の構造を表すシンタクスを示す略線図、第５２図は、ファイルｘｘｘｘｘ．ｍｐｌｓの一例の構造を表すシンタクスを示す略線図、第５３図は、ブロックＰＬＣｐｎｔｒｏｌＩｎｆｏ（）の一例の構造を表すシンタクスを示す略線図、第５４図は、フィールドＰＬ＿Ｐｌａｙｂａｃｋ＿ｔｙｐｅのの取り得る値と意味を示す略線図、第５５図は、フィールドＰＬ＿ｒａｎｄｏｍ＿ａｃｃｅｓｓ＿ｍｏｄｅの取り得る値と意味を示す略線図、第５６図は、ブロックＰｌａｙＬｉｓｔ（）の一例の構造を表すシンタクスを示す略線図、第５７図は、ブロックＰｌａｙＩｔｅｍ（）の一例の構造を表すシンタクスを示す略線図、第５８図は、フィールドＰＩ＿ｒａｎｄｏｍ＿ａｃｃｅｓｓ＿ｍｏｄｅの取り得る値と意味を示す略線図、第５９図は、フィールドｓｔｉｌｌ＿ｍｏｄｅの取り得る値と意味を示す略線図、第６０図は、フィールドｉｓ＿ｓｅａｍｌｅｓｓ＿ａｎｇｌｅ＿ｃｈａｎｇｅの取り得る値と意味を示す略線図、第６１図は、ブロックＳｕｂＰｌａｙＩｔｅｍ（）の一例の構造を表すシンタクスを示す略線図、第６２図は、フィールドｉｓ＿ｒｅｐｅａｔ＿ｆｌａｇの取り得る値と意味を示す略線図、第６３図は、メインパスとサブパスの時間軸の関係を示す略線図、第６４図は、ファイルｚｚｚｚｚ．ｃｌｐｉの一例の構造を表すシンタクスを示す略線図、第６５図は、ブロックＣｌｉｐＩｎｆｏ（）の一例の構造を表すシンタクスを示す略線図、第６６図は、フィールドａｐｐｌｉｃａｔｉｏｎ＿ｔｙｐｅの取り得る値と意味を示す略線図、第６７図は、ブロックＳｅｑｕｅｎｃｅＩｎｆｏ（）の一例の構造を表すシンタクスを示す略線図、第６８図は、ブロックＰｒｏｇｒａｍＩｎｆｏ（）の一例の構造を表すシンタクスを示す略線図、第６９図は、ブロックＳｔｒｅａｍＣｏｄｉｎｇＩｎｆｏ（）の一例の構造を表すシンタクスを示す略線図、第７０図は、ブロックＣＰＩ（）の一例の構造を表すシンタクスを示す略線図、第７１図は、フィールドＣＰＩ＿ｔｙｐｅの取り得る値と意味を示す略線図、第７２図は、ブロックＥＰ＿ｍａｐ＿ｆｏｒ＿ｏｎｅ＿ｓｔｒｅａｍ＿ＰＩＤ（）の一例の構造を表すシンタクスを示す略線図、第７３図Ａ、第７３図Ｂおよび第７３図Ｃは、プレーヤデコーダの一例の構成を示す機能ブロック図、第７４図は、ボタン表示の一例の状態遷移図、第７５図は、オブジェクトの種類の分類を説明するための図、第７６図Ａ、第７６図Ｂおよび第７６図Ｃは、この発明の実施の一形態によるグラフィクスオブジェクトのデータ構造の例を示す略線図、第７７図は、グラフィクスオブジェクトが分割されてＰＥＳパケットに格納される様子を示す略線図、第７８図は、グラフィクスオブジェクトのデコードを行うグラフィクスオブジェクトデコーダモデルの一例の構成を示す機能ブロック図、第７９図Ａ、第７９図Ｂ、第７９図Ｃおよび第７９図Ｄは、グラフィクスオブジェクト入力バッファ、ＰＮＧデコーダ、オブジェクトバッファおよびプレーンバッファにおける蓄積データ量の遷移の例を概略的に示す略線図、第８０図は、プレーンへのデータ転送速度について説明するための図、第８１図Ａおよび第８１図Ｂは、ウィンドウの定義について説明するための図、第８２図Ａおよび第８２図Ｂは、ウィンドウの定義について説明するための図、第８３図は、グラフィクスオブジェクトの一例の構造を表すシンタクスを示す略線図、第８４図は、ブロックＧｌｏｂａｌＰａｌｅｔｔｅＴａｂｌｅ（）の一例の構造を表すシンタクスを示す略線図、第８５図Ａおよび第８５図Ｂは、命令グループＤｉｓｐＣｍｄｓ（ｉ）に置かれる表示制御命令の例を一覧して示す略線図、第８６図Ａおよび第８６図Ｂは、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）および命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）を説明するための略線図、第８７図は、座標軸の定義を説明するための略線図、第８８図は、命令グループＤｉｓｐＣｍｄｓ（ｉ）の記述とグラフィクスオブジェクトの表示変化の一例を示す略線図、第８９図Ａ、第８９図Ｂ、第８９図Ｃおよび第８９図Ｄは、字幕表示が徐々に現れるフェードインの例を示す略線図、第９０図Ａおよび第９０図Ｂは、字幕のＰＮＧ画像がプレーン内を移動する例を示す略線図、第９１図Ａおよび第９１図Ｂは、字幕表示をスクロールさせる例を示す略線図、第９２図Ａおよび第９２図Ｂは、ＰＮＧ画像の一部を表示する枠を設定し、ＰＮＧ画像上でこ

の枠を移動させつつ、プレーン上での表示位置も移動させる例を示す略線図、第９３図は、ボタン画像に音声データを割り当てたグラフィクスオブジェクトの一例のデータ構造を示す略線図、第９４図は、グラフィクスオブジェクトに音声データを格納しない場合の一例のデータの管理構造を示す略線図、第９５図は、グラフィクスオブジェクトに音声データを格納する一例の方法を示す略線図、第９６図は、グラフィクスオブジェクトに音声データを格納しない場合の音声データの再生についてより具体的に説明するための略線図である。

【発明を実施するための最良の形態】

この発明の実施の一形態について説明する。この発明の実施の一形態では、記録再生規格であるＢｌｕ－ｒａｙ　Ｄｉｓｃ規格（Ｂｌｕ－ｒａｙ　Ｄｉｓｃ　Ｒｅｗｒｉｔａｂｌｅ　Ｆｏｒｍａｔ　Ｖｅｒ１．０）を基に、再生専用ディスクで必要となるインタラクティブ機能や、マルチアングル機能などの機能を実現できるようにした。

以下の説明は、下記の構成に従い行う。

１．ＢＤ－ＲＥフォーマットの概要

先ず、理解を容易とするために、Ｂｌｕ－ｒａｙ　Ｄｉｓｃに関し、”Ｂｌｕ－ｒａｙ　Ｄｉｓｃ　Ｒｅｗｒｉｔａｂｌｅ　Ｆｏｒｍａｔ　Ｖｅｒ１．０　ｐａｒｔ３　Ａｕｄｉｏ　Ｖｉｓｕａｌ　Ｓｐｅｃｉｆｉｃａｔｉｏｎｓ”で規定されている、コンテンツすなわちディスクに記録されたＡＶ（Ａｕｄｉｏ／Ｖｉｄｅｏ）データの管理構造について説明する。以下では、この管理構造をＢＤＡＶフォーマットと称する。

例えばＭＰＥＧ（Ｍｏｖｉｎｇ　Ｐｉｃｔｕｒｅｓ　Ｅｘｐｅｒｔｓ　Ｇｒｏｕｐ）ビデオやＭＰＥＧオーディオなどの符号化方式で符号化され、ＭＰＥＧ－２システムに従い多重化されたビットストリームは、クリップＡＶストリーム（またはＡＶストリーム）と称される。クリップＡＶストリームは、Ｂｌｕ－ｒａｙ　Ｄｉｓｃに関する規格の一つである”Ｂｌｕ－ｒａｙ　Ｄｉｓｃ　Ｒｅｗｒｉｔａｂｌｅ　Ｆｏｒｍａｔ　Ｖｅｒ１．０　ｐａｒｔ２”で定義されたファイルシステムにより、ファイルとしてディスクに記録される。このファイルを、クリップＡＶストリームファイル（またはＡＶストリームファイル）と称する。

クリップＡＶストリームファイルは、ファイルシステム上での管理単位であり、ユーザにとって必ずしも分かりやすい管理単位であるとは限らない。ユーザの利便性を考えた場合、複数のクリップＡＶストリームファイルに分割された映像コンテンツを一つにまとめて再生するための情報や、クリップＡＶストリームファイルの一部だけを再生するための情報、特殊再生や頭出し再生を滑らかに行うための情報などをデータベースとしてディスクに記録しておく必要がある。Ｂｌｕ－ｒａｙ　Ｄｉｓｃに関する規格の一つである”Ｂｌｕ－ｒａｙ　Ｄｉｓｃ　Ｒｅｗｒｉｔａｂｌｅ　Ｆｏｒｍａｔ　Ｖｅｒ１．０　ｐａｒ

ｔ３”で、このデータベースが規定される。

第６図は、ＡＶストリームファイルの一部または全部の範囲を指定して、必要な部分だけを並べて再生する再生順序指定の仕組みを模式的に示す。第６図において、プレイリスト（ＰｌａｙＬｉｓｔ）は、ＡＶストリームファイルの一部または全部の範囲を指定して、必要な部分だけを再生する指示を与える。ユーザがコンテンツの再生を行う際には、このプレイリストの単位で選択する。プレイリストは、ユーザからみてひとまとまりであり、再生が連続に行われることをユーザが暗黙に期待する、映像・音声の単位である。

最も簡単なプレイリストの構成は、記録が開始されてから記録が終了されるまでの一繋がりのＡＶストリームファイルからなり、編集をしなければ、これが一つのプレイリストになる。

プレイリストは、再生するＡＶストリームファイルの指定と、指定されたＡＶストリームファイルの再生箇所を指定する再生開始点と再生終了点の集まりとから構成される。この再生開始点と再生終了点の情報を一組としたものは、プレイアイテム（ＰｌａｙＩｔｅｍ）と称される。プレイリストは、プレイアイテムの集合で構成される。プレイアイテムを再生するということは、そのプレイアイテムに参照されるＡＶストリームファイルの一部分を再生するということになる。

クリップＡＶストリームは、上述したように、ビデオデータやオーディオデータがＭＰＥＧ２　ＴＳ（トランスポートストリーム）の形式などに多重化されたビットストリームである。このクリップＡＶストリームに関する情報がクリップ情報（Ｃｌｉｐ　Ｉｎｆｏｒｍａｔｉｏｎ）としてファイルに記録される。

クリップＡＶストリームファイルと、対応するクリップ情報が記録されたクリップ情報ファイルとをひとまとまりのオブジェクトと見なし、クリップ（Ｃｌｉｐ）と称する。クリップは、クリップＡＶストリームとクリップ情報とから構成される、一つのオブジェクトである。

ファイルは、一般的に、バイト列として扱われる。クリップＡＶストリームファイルのコンテンツは、時間軸上に展開され、クリップ中のエントリーポイントは、主に時間ベースで指定される。所定のクリップへのアクセスポイントのタイムスタンプが与えられた場合、クリップＡＶストリームファイルの中でデータの読み出しを開始すべきアドレス情報を見つけるために、クリップ情報ファイルを用いることができる。

１のディスク中に記録された全てのプレイリストおよびクリップは、ボリュームインフォメーション（Ｖｏｌｕｍｅ　Ｉｎｆｏｒｍａｔｉｏｎ）で管理される。

第７図は、上述のようなクリップＡＶストリーム、クリップ情報（Ｓｔｒｅａｍ　Ａｔｔｒｉｂｕｔｅｓ）、クリップ、プレイアイテムおよびプレイリストの関係を示すＵＭＬ（Ｕｎｉｆｉｅｄ　Ｍｏｄｅｌｉｎｇ　Ｌａｎｇｕａｇｅ）図である。プレイリストは、１または複数のプレイアイテムに対応付けられ、プレイアイテムは、１のクリップに対応付けられる。１のクリップに対して、それぞれ開始点および／または終了点が異なる複数のプレイアイテムを対応付けることができる。１のクリップから１のクリップＡＶストリームファイルが参照される。１のクリップから１のクリップ情報ファイルが参照される。クリップＡＶ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の対応関係を有する。このような構造を定義することにより、クリップＡＶストリームファイルを変更することなく、任意の部分だけを再生する、非破壊の再生順序指定を行うことが可能となる。

第８図のように、複数のプレイリストから同一のクリップを参照することもできる。第８図の例では、クリップ１が２つのプレイリスト２および３から参照されている。第８図において、クリップ１は、横方向が時間軸を表す。プレイリスト２により、コマーシャル区間ｂ－ｃとシーンｅとを含むクリップ１の区間ａ～ｆが参照される。プレイリスト３により、シーンｅを含むクリップ１の区間ｄ－ｇが参照される。プレイリスト２を指定することで、クリップ１の区間ａ～ｆを再生することができ、プレイリスト３を指定することで、クリップ１の区間ｄ－ｇを再生することができる。

”Ｂｌｕ－ｒａｙ　Ｄｉｓｃ　Ｒｅｗｒｉｔａｂｌｅ　Ｆｏｒｍａｔ　Ｖｅｒ１．０　ｐａｒｔ３”で規定された、記録媒体に記録されるファイルの管理構造について、第９図

を用いて説明する。ファイルは、ディレクトリ構造により階層的に管理される。記録媒体上には、先ず、１つのディレクトリ（第９図の例ではルートディレクトリ）が作成される。このディレクトリの下が、１つの記録再生システムで管理される範囲とする。

ルートディレクトリの配下に、ディレクトリＢＤＡＶが置かれる。ディレクトリＢＤＡＶは、第９図のように、ディレクトリＢＤＡＶ、ＢＤＡＶ１、ＢＤＡＶ２、・・・、ＢＤＡＶｎの如く、ルートディレクトリの配下に複数を置くことができる。以下では、複数のディレクトリＢＤＡＶ、ＢＤＡＶ１、ＢＤＡＶ２、・・・、ＢＤＡＶｎをディレクトリＢＤＡＶで代表させて説明する。

ディレクトリＢＤＡＶの配下には、以下の６種類のファイルが置かれる。

（１）ｉｎｆｏ．ｂｄａｖ
（２）ｍｅｎｕ．ｔｉｄｘ、ｍａｒｋ．ｔｉｄｘ
（３）ｍｅｎｕ．ｔｄｔ１、ｍｅｎｕ．ｔｄｔ２、ｍａｒｋ．ｔｄｔ１、ｍａｒｋ．ｔｄｔ２
（４）＃＃＃＃＃．ｒｐｌｓ、＃＃＃＃＃．ｖｐｌｓ
（５）％％％％％．ｃｌｐｉ
（６）＊＊＊＊＊．ｍ２ｔｓ

（４）のファイル「＃＃＃＃＃．ｒｐｌｓ」および「＃＃＃＃＃．ｖｐｌｓ」において、「＃＃＃＃＃」は、任意の番号を示す。（５）のファイル「％％％％％．ｃｌｐｉ」において、「％％％％％」は任意の番号を示す。（６）のファイル「＊＊＊＊＊．ｍ２ｔｓ」において、「＊＊＊＊＊」は、ファイル「＊＊＊＊＊．ｍ２ｔｓ」がファイル「％％％％％．ｃｌｐｉ」と一対一で対応するような番号である。番号「＊＊＊＊＊」は、番号「％％％％％」と同一とすることができる。

（１）のファイル「ｉｎｆｏ．ｂｄａｖ」は、ディレクトリＢＤＡＶ全体の情報が格納されるファイルである。（２）のファイル「ｍｅｎｕ．ｔｉｄｘ」および「ｍａｒｋ．ｔｉｄｘ」は、サムネイル画像の情報が格納されるファイルである。（３）のファイル「ｍｅｎｕ．ｔｄｔ１」、「ｍｅｎｕ．ｔｄｔ２」、「ｍａｒｋ．ｔｄｔ１」および「ｍａｒｋ．ｔｄｔ２」は、サムネイル画像が格納されるファイルである。各ファイルの拡張子「ｔｄｔ１」および「ｔｄｔ２」は、当該ファイルに格納されているサムネイル画像データが暗号化されているか否かを示す。

（４）のファイル「＃＃＃＃＃．ｒｐｌｓ」、「＃＃＃＃＃．ｖｐｌｓ」は、プレイリストの情報が格納されるファイルである。ファイル「＃＃＃＃＃．ｒｐｌｓ」、「＃＃＃＃＃．ｖｐｌｓ」は、ディレクトリＢＤＡＶの配下に設けられたディレクトリＰＬＡＹＬＩＳＴのさらに配下に置かれている。

（５）のファイル「％％％％％．ｃｌｐｉ」は、クリップ情報が格納されるファイルである。ファイル「％％％％％．ｃｌｐｉ」は、ディレクトリＢＤＡＶの配下に設けられたディレクトリＣＬＩＰＩＮＦのさらに配下に置かれている。（６）のファイル「＊＊＊＊＊．ｍ２ｔｓ」は、クリップＡＶストリームが格納される、クリップＡＶストリームファイルである。このクリップＡＶストリームファイルは、ファイル名の番号「＊＊＊＊＊」により、１つのクリップ情報ファイル「％％％％％．ｃｌｐｉ」に対応付けられる。ファイル「＊＊＊＊＊．ｍ２ｔｓ」は、ディレクトリＢＤＡＶの配下に設けられたディレクトリＳＴＲＥＡＭのさらに配下に置かれる。

各ファイルについて、より詳細に説明する。（１）のファイル「ｉｎｆｏ．ｂｄａｖ」は、ディレクトリＢＤＡＶの配下に唯一、設けられる。第１０図は、ファイル「ｉｎｆｏ．ｂｄａｖ」の一例の構造を表すシンタクスを示す。シンタクスをコンピュータ装置などのプログラムの記述言語として用いられるＣ言語の記述法に基づき示す。他のシンタクスを表す図において、同様である。

第１０図において、ファイル「ｉｎｆｏ．ｂｄａｖ」の内部は、機能別の情報毎にブロックが構成される。フィールドｔｙｐｅ＿ｉｎｄｉｃａｔｏｒには、文字列「ＢＤＡＶ」が格納され、このファイルがファイル「ｉｎｆｏ．ｂｄａｖ」であることが示される。フィールドｖｅｒｓｉｏｎ＿ｎｕｍｂｅｒには、このファイル「ｉｎｆｏ．ｂｄａｖ」のバ

ージョンが示される。ブロックＵＩＡｐｐＩｎｆｏＢＤＡＶ（）には、ディレクトリＢＤＡＶの配下に関する情報が格納される。ブロックＴａｂｌｅＯｆＰｌａｙＬｉｓｔｓ（）には、プレイリストの並び順に関する情報が格納される。ブロックＭａｋｅｒｓＰｒｉｖａｔｅＤａｔａ（）には、記録再生装置のメーカ固有の情報が格納される。

ファイル「ｉｎｆｏ．ｂｄａｖ」の先頭には、各ブロックの先頭を表すアドレスが記述される。例えば、フィールドＴａｂｌｅＯｆＰｌａｙＬｉｓｔｓ＿Ｓｔａｒｔ＿ａｄｄｒｅｓｓには、ブロック「ＴａｂｌｅＯｆＰｌａｙＬｉｓｔｓ（）」の開始する位置がファイル内の相対バイト数で示される。

第１１図は、ブロックＵＩＡｐｐＩｎｆｏＢＤＡＶ（）の一例の構造を表すシンタクスを示す。フィールドｌｅｎｇｔｈには、フィールドｌｅｎｇｔｈ直後のフィールドからこのブロックＵＩＡｐｐＩｎｆｏＢＤＡＶ（）の最後までの長さがバイトで示される。フィールドＢＤＡＶ＿ｃｈａｒａｃｔｅｒ＿ｓｅｔには、このブロックＵＩＡｐｐＩｎｆｏＢＤＡＶ（）内のフィールドＢＤＡＶ＿ｎａｍｅに記録されている文字列の文字セットが示される。文字セットとしては、ＡＳＣＩＩ、Ｕｎｉｃｏｄｅなどが選択できる。

フラグＢＤＡＶ＿ｐｒｏｔｅｃｔ＿ｆｌａｇは、ディレクトリＢＤＡＶの配下に置かれるコンテンツを、ユーザに制限無しに見せて良いか否かを示す。例えば、このフラグが値「１」にセットされており、且つ、ユーザが正しく暗証番号ＰＩＮ（Ｐｅｒｓｏｎａｌ Ｉｄｅｎｔｉｆｉｃａｔｉｏｎ Ｎｕｍｂｅｒ）を入力できた場合に、ディレクトリＢＤＡＶの配下に置かれたコンテンツをユーザに見せることができるものとする。一方、フラグＢＤＡＶ＿ｐｒｏｔｅｃｔ＿ｆｌａｇが値「０」にセットされている場合には、ユーザによる暗証番号ＰＩＮの入力が無くても、ディレクトリＢＤＡＶの配下に置かれたコンテンツをユーザに見せることができる。

暗証番号ＰＩＮは、フィールドＰＩＮに記録される。暗証番号ＰＩＮは、例えば４桁の０～９の数字からなり、上述のように再生制御を有効にした場合に必要となる暗証番号を表す。暗証番号ＰＩＮのそれぞれの数字は、例えばＩＳＯ（Ｉｎｔｅｒｎａｔｉｏｎａｌ Ｏｒｇａｎｉｚａｔｉｏｎ ｆｏｒ Ｓｔａｎｄａｒｉｚａｔｉｏｎ）／ＩＥＣ（Ｉｎｔｅｒｎａｔｉｏｎａｌ Ｅｌｅｃｔｒｏｔｅｃｈｎｉｃａｌ Ｃｏｍｍｉｓｓｉｏｎ）６４６の規定に従って符号化される。

このブロックＵＩＡｐｐＩｎｆｏＢＤＡＶ（）に示される以上の情報により、ディレクトリＢＤＡＶに対する再生制限が規定される。なお、詳細は後述するが、個々のプレイリストに対する再生制限については、ファイル「＃＃＃＃＃．ｒｐｌｓ」、「＃＃＃＃＃．ｖｐｌｓ」内に示されるブロックＵＩＡｐｐＩｎｆｏＰｌａｙＬｉｓｔ（）の中で定義されるフラグｐｌａｙｂａｃｋ＿ｃｏｎｔｒｏｌ＿ｆｌａｇによって規定される。

この例では、ディレクトリＢＤＡＶの配下のコンテンツの再生を再開する際に、優先して再生すべきプレイリストを指定する、レジューム機能を用いることができる。このレジューム機能は、前回視聴を中断した箇所から再生を再開したい場合などに使うことを想定している。

第１１図において、フラグｒｅｓｕｍｅ＿ｖａｌｉｄ＿ｆｌａｇは、このレジューム機能の有効／無効を示す。例えば、このフラグが値「０」にセットされている場合には、レジューム機能は無効とされる。このフラグが値「１」にセットされている場合には、レジューム機能が有効とされ、フィールドｒｅｓｕｍｅ＿ＰｌａｙＬｉｓｔ＿ｆｉｌｅ＿ｎａｍｅで指定されるプレイリストを、優先して再生すべきプレイリストとして扱う。

フィールドｒｅｆ＿ｔｏ＿ｍｅｎｕ＿ｔｈｕｍｂｎａｉｌ＿ｉｎｄｅｘは、このディレクトリＢＤＡＶを代表するサムネイル画像が存在する場合、そのサムネイル画像を特定するサムネイル番号が格納される領域である。Ｂｌｕ－ｒａｙ Ｄｉｓｃの規格では、ディレクトリＢＤＡＶを代表する静止画を特にメニューサムネイルと呼ぶ。フィールドｒｅｆ＿ｔｏ＿ｍｅｎｕ＿ｔｈｕｍｂｎａｉｌ＿ｉｎｄｅｘで指定されるインデックスｔｈｕｍｂｎａｉｌ＿ｉｎｄｅｘを持つサムネイル画像がこのディレクトリＢＤＡＶのメニューサムネイルとなる。

フィールドＢＤＡＶ＿ｎａｍｅ＿ｌｅｎｇｔｈは、フィールドＢＤＡＶ＿ｎａｍｅに示

されるディレクトリＢＤＡＶ名のバイト長を示す。このフィールドＢＤＡＶ＿ｎａｍｅにおいて、左から、フィールドＢＤＡＶ＿ｎａｍｅ＿ｌｅｎｇｔｈで示されるだけのバイト数が有効な文字列であり、それがこのディレクトリＢＤＡＶの名前を示す。なお、フィールドＢＤＡＶ＿ｎａｍｅの中で、フィールドＢＤＡＶ＿ｎａｍｅ＿ｌｅｎｇｔｈで示された有効な文字列の後のバイト列には、どのような値が入っていても良い。

第１２図は、ブロックＴａｂｌｅＯｆＰｌａｙＬｉｓｔｓ（）の一例の構造を表すシンタクスを示す。フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＬｉｓｔｓには、このディレクトリＢＤＡＶの配下に置かれたプレイリストの数が示される。このプレイリスト数をループ変数とする、次に続く「ｆｏｒ」文のループにおいてフィールドＰｌａｙＬｉｓｔ＿ｆｉｌｅ＿ｎａｍｅにより示されるプレイリストの並びが、プレイリスト一覧表示画面などに表示される際のプレイリストの順序を示す。プレイリストは、フィールドＰｌａｙＬｉｓｔ＿ｆｉｌｅ＿ｎａｍｅにより、「＃＃＃＃＃．ｒｐｌｓ」や「＃＃＃＃＃．ｖｐｌｓ」といったファイル名で指定される。

ファイル「＃＃＃＃＃．ｒｐｌｓ」および「＃＃＃＃＃．ｖｐｌｓ」は、上述したように、ディレクトリＰＬＡＹＬＩＳＴの配下に置かれる。これらのファイルは、１つのプレイリストと一対一に対応している。

第１３図は、ファイル「＃＃＃＃＃．ｒｐｌｓ」および「＃＃＃＃＃．ｖｐｌｓ」の一例の構造を表すシンタクスを示す。第１３図において、ファイル「＃＃＃＃＃．ｒｐｌｓ」および「＃＃＃＃＃．ｖｐｌｓ」の内部は、機能別の情報毎にブロックが構成される。フィールドｔｙｐｅ＿ｉｎｄｉｃａｔｏｒにこのファイルを示す文字列が格納され、フィールドｖｅｒｓｉｏｎ＿ｎｕｍｂｅｒにこのファイルのバージョンが示される。

ブロックＵＩＡｐｐＩｎｆｏＰｌａｙＬｉｓｔ（）には、このプレイリストに関する属性情報が格納される。ブロックＰｌａｙＬｉｓｔ（）には、このプレイリストを構成するプレイアイテムに関する情報が格納される。ブロックＰｌａｙＬｉｓｔＭａｒｋ（）には、このプレイリストに付されるマークの情報が格納される。ブロックＭａｋｅｒｓＰｒｉｖａｔｅＤａｔａ（）には、このプレイリストファイルを記録した装置の、メーカ固有の情報が格納される。ファイル「＃＃＃＃＃．ｒｐｌｓ」および「＃＃＃＃＃．ｖｐｌｓ」の先頭に設けられるフィールドＰｌａｙＬｉｓｔ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓ、ＰｌａｙＬｉｓｔＭａｒｋ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓおよびＭａｋｅｒｓＰｒｉｖａｔｅＤａｔａ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓには、それぞれ対応するブロックの先頭アドレスが例えば３２ビットのアドレス情報として示される。

各ブロックの先頭アドレスがファイル「＃＃＃＃＃．ｒｐｌｓ」および「＃＃＃＃＃．ｖｐｌｓ」の先頭に示されるために、ブロックの前および／または後ろに、任意の長さのデータｐａｄｄｉｎｇ＿ｗｏｒｄを挿入することができる。但し、ファイル「＃＃＃＃＃．ｒｐｌｓ」および「＃＃＃＃＃．ｖｐｌｓ」の最初のブロックであるブロックＵＩＡｐｐＩｎｆｏＰｌａｙＬｉｓｔ（）の開始位置は、ファイルの先頭から３２０バイト目に固定される。

第１４図は、ブロックＵＩＡｐｐＩｎｆｏＰｌａｙＬｉｓｔ（）の一例の構造を表すシンタクスを示す。ブロックＵＩＡｐｐＩｎｆｏＰｌａｙＬｉｓｔ（）は、プレイリストの再生において直接的に必要とされない、プレイリストに関する各種の属性情報が格納される。フィールドＰｌａｙＬｉｓｔ＿ｃｈａｒａｃｔｅｒ＿ｓｅｔは、プレイリストに関する文字列情報の文字セットが指定される。

フラグｐｌａｙｂａｃｋ＿ｃｏｎｔｒｏｌ＿ｆｌａｇには、情報表示やプレイリストの再生を暗証番号ＰＩＮに基づき制限するか否かが指定される。例えばフラグｐｌａｙｂａｃｋ＿ｃｏｎｔｒｏｌ＿ｆｌａｇが値「１」であれば、ユーザにより正しい暗証番号ＰＩＮが入力されなければ、プレイリストのサムネイル画像などの情報表示や、プレイリストの再生ができないとされる。フラグｗｒｉｔｅ＿ｐｒｏｔｅｃｔ＿ｆｌａｇは、消去禁止フラグである。例えばこのフラグｗｒｉｔｅ＿ｐｒｏｔｅｃｔ＿ｆｌａｇが値「１」であれば、ユーザが容易にプレイリストを消去できないようなユーザインタフェースにする必要がある。フラグｉｓ＿ｐｌａｙｅｄ＿ｆｌａｇは、このプレイリストが再生済みである

ことを示す。フラグｉｓ＿ｅｄｉｔｅｄ＿ｆｌａｇは、このプレイリストが編集されたことを示す。

フィールドｔｉｍｅ＿ｚｏｎｅは、このプレイリストが記録された地点のタイムゾーンを示す。フィールドｒｅｃｏｒｄ＿ｔｉｍｅ＿ａｎｄ＿ｄａｔｅは、プレイリストの記録日時を示す。フィールドＰｌａｙＬｉｓｔ＿ｄｕｒａｔｒｏｎは、プレイリストの再生時間を示す。

フィールドｍａｋｅｒ＿ＩＤおよびｍａｋｅｒ＿ｍｏｄｅｌ＿ｃｏｄｅには、このプレイリストを最終更新した記録機のメーカおよび記録機をそれぞれ特定する情報が例えば番号で示される。フィールドｃｈａｎｎｅｌ＿ｎｕｍｂｅｒには、記録したクリップＡＶストリームのチャンネル番号が示され、フィールドｃｈａｎｎｅｌ＿ｎａｍｅには、当該チャンネル名が示される。フィールドｃｈａｎｎｅｌ＿ｎａｍｅに示されるチャンネル名の長さがフィールドｃｈａｎｎｅｌ＿ｎａｍｅ＿ｌｅｎｇｔｈに示される。フィールドｃｈａｎｎｅｌ＿ｎａｍｅの領域のうち、フィールドｃｈａｎｎｅｌ＿ｎａｍｅ＿ｌｅｎｇｔｈで示される長さの文字列が有効となる。フィールドＰｌａｙＬｉｓｔ＿ｎａｍｅには、フィールドＰｌａｙＬｉｓｔ＿ｎａｍｅ＿ｌｅｎｇｔｈに示される値を有効長としてプレイリスト名が示される。フィールドＰｌａｙＬｉｓｔ＿ｄｅｔａｉｌには、フィールドＰｌａｙＬｉｓｔ＿ｄｅｔａｉｌ＿ｌｅｎｇｔｈに示される値を有効長としてプレイリストの詳細情報が示される。

第１５図は、ブロックＰｌａｙＬｉｓｔ（）の一例の構造を表すシンタクスを示す。フィールドｌｅｎｇｔｈは、フィールドｌｅｎｇｔｈの直後のフィールドからこのブロックＰｌａｙＬｉｓｔ（）の終端までのバイト長を示す。フィールドＰＬ＿ＣＰＩ＿ｔｙｐｅは、このプレイリストが持つＣＰＩ（Ｃｈａｒａｃｔｅｒｉｓｔｉｃ　Ｐｏｉｎｔ　Ｉｎｆｏｒｍａｔｉｏｎ：特徴点情報）の種類を示す。フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＩｔｅｍｓは、このプレイリストを構成するプレイアイテムの数を示す。フィールドｎｕｍｂｅｒ＿ｏｆ＿ＳｕｂＰｌａｙＩｔｅｍｓは、このプレイリストに付けられているアフレコオーディオ用のプレイアイテム（サブプレイアイテム）の数を示す。詳細は省略するが、プレイリストは、所定の条件を満たす場合にだけ、サブプレイアイテムを持つことができる。

ブロックＰｌａｙＩｔｅｍ（）は、プレイアイテムの情報が格納される。ブロックＳｕｂＰｌａｙＩｔｅｍ（）は、サブプレイアイテムの情報が格納される。

第１６図は、ブロックＰｌａｙＩｔｅｍ（）の一例の構造を表すシンタクスを示す。フィールドＣｌｉｐ＿Ｉｎｆｏｒｍａｔｉｏｎ＿ｆｉｌｅ＿ｎａｍｅには、このプレイアイテムが参照しているクリップと一対一に対応するクリップ情報ファイル（拡張子がｃｌｐｉであるファイル）のファイル名が文字列で示される。クリップ情報ファイルは、拡張子が「ｃｌｐｉ」とされているファイルである。

フィールドＣｌｉｐ＿ｃｏｄｅｃ＿ｉｄｅｎｔｉｆｉｅｒは、このプレイアイテムにより参照されるクリップの符号化方式を示す。この例では、フィールドＣｌｉｐ＿ｃｏｄｅｃ＿Ｉｄｅｎｔｉｆｉｅｒは、値「Ｍ２ＴＳ」に固定的とされる。フィールドｃｏｎｎｅｃｔｉｏｎ＿ｃｏｎｄｉｔｉｏｎは、このプレイアイテムが次のプレイアイテムとの間でどのような接続がされているかを示す情報である。フィールドｃｏｎｎｅｃｔｉｏｎ＿ｃｏｎｄｉｔｉｏｎにより、プレイアイテムとプレイアイテムとの間が継ぎ目なくシームレスに再生できるか否かが示される。

フィールドｒｅｆ＿ｔｏ＿ＳＴＣ＿ｉｄは、このプレイアイテムにより参照されるクリップ内のシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅを指定する。シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅは、ＭＰＥＧ２　ＴＳ（Ｔｒａｎｓｐｏｒｔ　Ｓｔｒｅａｍ）における時間軸の基準であるＰＣＲ（Ｐｒｏｇｒａｍ　Ｃｌｏｃｋ　Ｒｅｆｅｒｅｎｃｅ）が連続な範囲を表すＢｌｕ－ｒａｙ　Ｄｉｓｃ規格独自の構造である。シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅには、クリップ内で一意な番号ＳＴＣ＿ｉｄが割り当てられる。このシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅ内では、不連続の無い一貫した時間軸を定義できるので、プレイアイテムの開始時刻および終了時刻を一意に定めることができる。各プレイアイテムの開始

点と終了点は、同一のシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅに存在していなければならない。フィールドｒｅｆ＿ｔｏ＿ＳＴＣ＿ｉｄでは、番号ＳＴＣ＿ｉｄによりシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅが指定される。

フィールドＩＮ＿ｔｉｍｅおよびＯＵＴ＿Ｔｉｍｅは、このプレイアイテムにおける開始点および終了点の、シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅ上でのタイムスタンプｐｔｓ（ｐｒｅｓｅｎｔａｔｉｏｎ＿ｔｉｍｅ＿ｓｔａｍｐ）をそれぞれ示す。

ブロックＢｒｉｄｇｅＳｅｑｕｎｃｅＩｎｆｏ（）には、ブリッジクリップ（Ｂｒｉｄｇｅ－Ｃｌｉｐ）に関する情報が格納される。ブリッジクリップは、第１７図に一例が示されるように、プレイアイテム間を継ぎ目無く、シームレスに再生する機能を実現する際に作成されるビットストリームである。前のプレイアイテムと現在のプレイアイテムとの継ぎ目の部分で、本来再生すべきビットストリームとは異なるビットストリーム、すなわちブリッジクリップを代わりに再生することで、２つのプレイアイテムをシームレスに再生することができる。

第１８図は、ブロックＰｌａｙＬｉｓｔＭａｒｋ（）の一例の構造を表すシンタクスを示す。ブロックＰｌａｙＬｉｓｔＭａｒｋ（）は、マークの情報が格納されるデータ構造である。マークは、プレイリスト上の時刻を保持するための構造であって、マークを用いることにより、プレイリスト中への頭出し点の設定、プレイリストを細分化するチャプタの構造を設ける、などの機能が実現される。後述するグラフィクスプレーン上に描画された画像の表示開始／停止のタイミングを、マークを用いて指定することができる。

フィールドｌｅｎｇｔｈは、フィールドｌｅｎｇｔｈの直後のフィールドからブロックＰｌａｙＬｉｓｔＭａｒｋ（）の終端までのバイト長を示す。フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＬｉｓｔ＿ｍａｒｋｓは、プレイリスト中のマークの数を表す。「ｆｏｒ」文での１回のループが１つのマークの情報を示す。フラグｍａｒｋ＿ｉｎｖａｌｉｄ＿ｆｌａｇは、このマークが有効であるか否かを示す。例えば、フラグｍａｒｋ＿ｉｎｖａｌｉｄ＿ｆｌａｇが値「０」でそのマークが有効であることが示され、値「１」であれば、そのマークの情報がデータベース上に存在するが、当該マークがユーザには見えない無効マークとされていることを示す。

フィールドｍａｒｋ＿ｔｙｐｅは、このマークの種類を表す。マークの種類としては、プレイリストのサムネイル画像（代表画）とする映像の位置を示すマーク、どこまで再生したかを示すレジュームマーク、頭出し点を示すチャプタマーク、ある区間の再生を飛ばして先に進むことを示すスキップマーク、グラフィクス画像の読み込み開始タイミングを示すマーク、グラフィクス画像の表示開始のタイミングを示すマーク、グラフィクス画像の表示停止のタイミングを示すマークなどがある。

フィールドｍａｒｋ＿ｎａｍｅ＿ｌｅｎｇｔｈは、後述するフィールドｍａｒｋ＿ｎａｍｅのデータ長を示す。フィールドｍａｋｅｒ＿ＩＤは、このマークを作成した記録機の製造者を示す。これは、製造者独自のマークを識別する際に用いられる。フィールドｒｅｆ＿ｔｏ＿ＰｌａｙＩｔｅｍ＿ｉｄには、このマークにより指示される時刻がどのプレイアイテム上に存在するかが示される。フィールドｍａｒｋ＿ｔｉｍｅ＿ｓｔａｍｐは、このマークにより指示される時刻が示される。

フィールドｅｎｔｒｙ＿ＥＳ＿ＰＩＤには、このマークがどのエレメンタリストリームすなわち映像データおよび／または音声データが符号化されたストリームに対して付与されたものかが示される。フィールドｒｅｆ＿ｔｏ＿ｍｅｎｕ＿ｔｈｕｍｂｎａｉｌ＿ｉｎｄｅｘおよびフィールドｒｅｆ＿ｔｏ＿ｍａｒｋ＿ｔｈｕｍｂｎａｉｌ＿ｉｎｄｅｘは、マークを視覚的に表すサムネイル画像を示す。サムネイル画像としては、例えばそのマークにより指定される時刻の映像を静止画として抜き出した画像が考えられる。

フィールドｄｕｒａｔｉｏｎには、マークが時間軸上で長さを持つ場合に使われる。例えばスキップマークであれば、どのくらいの時間ジャンプをするかがフィールドｄｕｒａｔｉｏｎにより指定される。

フィールドｍａｋｅｒｓ＿ｉｎｆｏｒｍａｔｉｏｎは、製造者独自の情報を記録するための領域である。フィールドｍａｒｋ＿ｎａｍｅには、マークに名前をつけた際、その名

前を格納するための領域であり、サイズが上述のフィールドｍａｒｋ＿ｎａｍｅ＿ｌｅｎｇｔｈで指定される。

第１９図は、ファイル「％％％％％．ｃｌｐｉ」の一例の構造を表すシンタクスを示す。上述したように、ファイル「％％％％％．ｃｌｐｉ」は、ディレクトリＣＬＩＰＩＮＦの配下に置かれ、各ＡＶストリームファイル（ファイル「＊＊＊＊＊．ｍ２ｔｓ」）毎に作られる。ファイル「％％％％％．ｃｌｐｉ」の内部は、機能別の情報毎にブロックが構成される。フィールドｔｙｐｅ＿ｉｎｄｉｃａｔｏｒにこのファイルを示す文字列が格納され、フィールドｖｅｒｓｉｏｎ＿ｎｕｍｂｅｒにこのファイルのバージョンが示される。

ブロックＣｌｉｐＩｎｆｏ（）には、クリップに関する情報が格納される。ブロックＳｅｑｕｅｎｃｅＩｎｆｏ（）には、ＭＰＥＧ２システムにおけるトランスポートストリームの時刻基準を表すＰＣＲの不連続点に関する情報が格納される。ブロックＰｒｏｇｒａｍＩｎｆｏ（）には、ＭＰＥＧ２システムのプログラムに関する情報が格納される。ブロックＣＰＩ（）には、ランダムアクセス開始可能点などの、ＡＶストリーム中の特徴的な箇所を表す特徴点情報ＣＰＩに関する情報が格納される。ブロックＣｌｉｐＭａｒｋ（）には、クリップに付された頭出しのためのインデックス点やコマーシャルの開始および／または終了点などのマーク情報が格納される。ブロックＭａｋｅｒｓＰｒｉｖａｔｅＤａｔａ（）には、記録機の製造者独自の情報が格納される。

これら各ブロックのファイル「％％％％％．Ｃｌｐｉ」中での先頭を表すアドレス情報が、当該ファイルの先頭部分にフィールドＳｅｑｕｅｎｃｅＩｎｆｏ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓ、ＰｒｏｇｒａｍＩｎｆｏ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓ、ＣＰＩ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓ、ＣｌｉｐＭａｒｋ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓおよびＭａｋｅｒｓＰｒｉｖａｔｅＤａｔａ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓとして示される。

ＢＤＡＶフォーマットは、以上に概略的に説明したようなデータ構造を持つことにより、クリップＡＶストリーム中の再生したい部分を開始および終了点の組で指定したプレイアイテムの並びでプレイリストを構築し、ユーザが認識するひとまとまりの再生単位を管理することが可能となる。

２．ＢＤ－ＲＯＭフォーマットの概要説明

次に、この発明の実施の一形態について説明する。この発明では、上述したＢＤＡＶフォーマットを拡張し、再生専用ディスク（ＢＤ－ＲＯＭ：Ｂｌｕ－ｒａｙ　Ｄｉｓｃ－Ｒｅａｄ　Ｏｎｌｙ　Ｍｅｍｏｒｙ）に適したフォーマットを構築する。拡張されたＢＤＡＶフォーマットを、ＢＤＭＶフォーマットと称する。

ディスクの内容を提示するメニュー画面を実現するためのプレーン構成を説明する。コンテンツ制作者側がプレイリストの再生順序を指定できるようにするためのシナリオ構造を追加する。そのシナリオ構造において、再生専用ディスクに特徴的なスチル（一時停止）、ランダム・シャッフル再生、マルチアングルなどの機能を実現するために必要なデータと、その格納方法について説明する。

２－１．プレーンについて

この発明の実施の一形態では、画像の表示系について、第２０図に一例が示されるようなプレーン構成を取る。動画プレーン１０は、最も後ろ側（ボトム）に表示され、プレイリストで指定された画像（主に動画データ）が扱われる。字幕プレーン１１は、動画プレーン１０の上に表示され、動画再生中に表示される字幕データが扱われる。グラフィクスプレーン１２は、最も前面に表示され、メニュー画面を表示するための文字データやボタンを表すビットマップデータなどのグラフィクスデータが扱われる。１つの表示画面は、これら３つのプレーンが合成されて表示される。

従来のＤＶＤビデオと異なる特徴的な点は、従来の技術で説明した字幕やメニュー画面、ボタンなどが表示されるサブピクチャを、字幕プレーン１１およびグラフィクスプレーン１２とに分離し、字幕とボタンとをそれぞれ独立的に制御できるようにしたことである。従来のＤＶＤビデオでは、メニュー画面やボタンなどのグラフィクス表示と、字幕表示とを、同じ仕組みで制御しており、これらを同一プレーン上に表示していた。同時に表示

できるビットマップ画像は、１枚に限られていた。そのため、ＤＶＤビデオでは、複数のビットマップ画像を同時に表示することができなかった。この発明では、字幕を表示する字幕プレーン１１と、グラフィクス表示を行うグラフィクスプレーン１２とをそれぞれ独立的に設けることで、ＤＶＤビデオの持つ問題点を解消している。

字幕プレーン１１およびグラフィクスプレーン１２が、“Ｂｌｕ－ｒａｙ Ｄｉｓｃ Ｒｅｗｒｉｔａｂｌｅ Ｆｏｒｍａｔ Ｖｅｒ１．０ ｐａｒｔ３”に対する拡張部分であると考えることができる。

動画プレーン１０、字幕プレーン１１およびグラフィクスプレーン１２は、それぞれ独立して表示が可能とされ、第２１図に一例が示されるような解像度および表示可能色を有する。動画プレーン１０は、解像度が１９２０画素×１０８０ラインで１画素当たりに換算したデータ長が１６ビットであって、輝度信号Ｙ、色差信号Ｃｂ、Ｃｒが４：２：２のシステム（以下、ＹＣｂＣｒ（４：２：２））とされる。ＹＣｂＣｒ（４：２：２）は、各画素当たり輝度信号Ｙが８ビット、色差信号Ｃｂ、Ｃｒがそれぞれ８ビットで、色差信号Ｃｂ、Ｃｒが水平２画素で一つの色データを構成すると見なすカラーシステムである。

字幕プレーン１１は、１９２０画素×１０８０ラインで各画素のサンプリング深さが８ビットとされ、カラーシステムは、２５６色のパレットを用いた８ビットカラーマップアドレスとされる。

グラフィクスプレーン１２は、解像度が１９２０画素×１０８０ラインで各画素のサンプリング深さが８ビットとされ、カラーシステムは、２５６色のパレットを用いた８ビットカラーマップアドレスとされる。

上述では、字幕プレーン１１とグラフィクスプレーン１２のカラーシステムを、２５６色のパレットを用いた８ビットカラーマップアドレスとしたが、この例に限定されない。色数については、サンプリング深さを変えてパレットの色数を増やせばよい。例えばサンプリング深さを１２ビットとすれば、パレットで使用可能な色数を４０９６色とすることができる。サンプリング深さを２４ビットとして、パレットを持たずに各画素が色情報を持つようにしたＹＣｂＣｒ（４：４：４）およびＲＧＢ（４：４：４）も、同様の仕組みで可能である。

グラフィクスプレーン１２および字幕プレーン１１は、２５６段階のアルファブレンディングが可能とされており、他のプレーンとの合成の際に、透明度を２５６段階で設定することが可能とされている。透明度の設定は、画素毎に行うことができる。以下では、透明度αが（０≦α≦１）の範囲で表され、透明度α＝０で完全に透明、透明度α＝１で完全に不透明であるものとする。

字幕プレーン１１では、例えばＰＮＧ（Ｐｏｒｔａｂｌｅ Ｎｅｔｗｏｒｋ Ｇｒａｐｈｉｃｓ）形式の画像データが扱われる。グラフィクスプレーン１２でも、ＰＮＧ形式の画像データを扱うことができる。ＰＮＧ形式は、１画素のサンプリング深さが１ビット～１６ビットとされ、サンプリング深さが８ビットまたは１６ビットの場合に、アルファチャンネル、すなわち、それぞれの画素成分の透明度情報（アルファデータと称する）を付加することができる。サンプリング深さが８ビットの場合には、２５６段階で透明度を指定することができる。このアルファチャンネルによる透明度情報を用いてアルファブレンディングが行われる。２５６色までのパレットイメージを用いることができ、予め用意されたパレットの何番目の要素（インデックス）であるかがインデックス番号により表現される。

字幕プレーン１１およびグラフィクスプレーン１２で扱われる画像データは、ＰＮＧ形式に限定されない。ＪＰＥＧ方式など他の圧縮符号化方式で圧縮符号化された画像データや、ランレングス圧縮された画像データ、圧縮符号化がなされていないビットマップデータなどを扱うようにしてもよい。

第２２図は、上述の第２０図および第２１図に従い３つのプレーンを合成する一例の構成を示す。動画プレーン１０の動画データが４２２／４４４変換回路２０に供給される。動画データは、４２２／４４４変換回路２０でカラーシステムがＹＣｂＣｒ（４：２：２）からＹＣｂＣｒ（４：４：４）に変換され、乗算器２１に入力される。４２２／４４４

変換回路２０と乗算器２１との間に解像度変換回路を挿入し、動画データの解像度を変換するようにしてもよい。

字幕プレーン１１の画像データがパレット２２に入力され、ＲＧＢ（４：４：４）の画像データとして出力される。この画像データに対してアルファブレンディングによる透明度が指定されている場合には、指定された透明度α１（０≦α１≦１）がパレット２２から出力される。

第２３図は、パレット２２の入出力データの一例を示す。パレット２２には、例えばＰＮＧ形式のファイルに対応したパレット情報がテーブルとして格納される。パレット２２からは、入力された８ビットの画素データをアドレスとして、インデックス番号が参照される。このインデックス番号に基づき、それぞれ８ビットのデータからなるＲＧＢ（４：４：４）のデータが出力される。それと共に、パレット２２からは、透明度を表すアルファチャンネルのデータが取り出される。

第２４図は、パレット２２に格納される一例のパレットテーブルを示す。２５６個のカラーインデックス値〔０ｘ００〕～〔０ｘＦＦ〕（〔０ｘ〕は１６進表記であることを示す）のそれぞれに対して、各々８ビットで表現される三原色の値Ｒ、ＧおよびＢと、透明度αとが割り当てられる。パレット２２は、入力されたＰＮＧ形式の画像データに基づきパレットテーブルが参照され、画像データにより指定されたインデックス値に対応する、それぞれ８ビットのデータからなるＲ、ＧおよびＢ各色のデータ（ＲＧＢデータ）と、透明度αとを画素毎に出力する。後述するパレット２６にも、同様のパレットテーブルが格納される。

パレット２２から出力されたＲＧＢデータは、ＲＧＢ／ＹＣｂＣｒ変換回路２９に供給され、各データ長が８ビットの輝度信号Ｙと色信号Ｃｂ、Ｃｒのデータに変換される（以下、まとめてＹＣｂＣｒデータと称する）。これは、以降のプレーン間合成を共通のデータ形式で行う必要があるためで、動画データのデータ形式であるＹＣｂＣｒデータに統一している。

ＲＧＢ／ＹＣｂＣｒ変換回路２９から出力されたＹＣｂＣｒデータおよび透明度データα１とがそれぞれ乗算器２３に入力される。ＲＧＢ／ＹＣｂＣｒ変換回路２９と乗算器２３との間に解像度変換回路を挿入し、ＹＣｂＣｒデータの解像度を変換するようにしてもよい。乗算器２３では、入力されたＹＣｂＣｒデータに透明度データα１が乗ぜられる。乗算結果は、加算器２４の一方の入力端に入力される。乗算器２３では、ＹＣｂＣｒデータにおける輝度信号Ｙ、色差信号Ｃｂ、Ｃｒのそれぞれについて、透明度データα１との乗算が行われる。透明度データα１の補数（１－α１）が乗算器２１に供給される。

乗算器２１では、４２２／４４４変換回路２０から入力された動画データに透明度データα１の補数（１－α１）が乗ぜられる。乗算結果は、加算器２４の他方の入力端に入力される。加算器２４において、乗算器２１および２３の乗算結果が加算される。これにより、動画プレーン１０と字幕プレーン１１とが合成される。加算器２４の加算結果が乗算器２５に入力される。

グラフィクスプレーン１２の画像データも字幕プレーン１１と同様に、パレットテーブル２６によりＲＧＢ（４：４：４）のデータが出力され、ＲＧＢ／ＹＣｂＣｒ変換回路２７に入力される。グラフィクスプレーン１２の画像データのカラーシステムがＲＧＢ（４：４：４）である場合には、カラーシステムがＹＣｂＣｒ（４：４：４）に変換されてＲＧＢ／ＹＣｂＣｒ変換回路２７から出力される。ＲＧＢ／ＹＣｂＣｒ変換回路２７から出力されたＹＣｂＣｒデータが乗算器２８に入力される。ＲＧＢ／ＹＣｂＣｒ変換回路２７と乗算器２８との間に解像度変換回路を挿入し、ＹＣｂＣｒデータの解像度を変換するようにしてもよい。

パレット２６において、インデックス値に対してアルファブレンディングによる透明度が指定されている場合には、指定された透明度α２（０≦α２≦１）がパレット２６から出力される。透明度データα２は、乗算器２８に供給される。乗算器２８では、ＲＧＢ／ＹＣｂＣｒ変換回路２７から入力されたＹＣｂＣｒデータに対し、輝度信号Ｙ、色差信号Ｃｂ、Ｃｒのそれぞれについて、透明度データα２との乗算が行われる。乗算器２８によ

る乗算結果が加算器２９の一方の入力端に入力される。透明度データα２の補数（１－α２）が乗算器２５に供給される。

乗算器２５では、加算器２４の加算結果に対して透明度データα２の補数（１－α２）が乗ぜられる。乗算器２５の乗算結果は、加算器２７の他方の入力端に入力され、上述した乗算器２８による乗算結果と加算される。これにより、動画プレーン１０と字幕プレーン１１との合成結果に対して、グラフィクスプレーン１２が合成される。

字幕プレーン１１およびグラフィクスプレーン１２において、例えば、表示すべき画像の無い領域の透明度α＝０と設定することで、そのプレーンの下に表示されるプレーンを透過表示させることができる。そうすることにより、動画プレーン１０に表示されている動画データを、字幕プレーン１１やグラフィクスプレーン１２の背景として表示することができる。

なお、この第２２図に示される構成は、ハードウェアおよびソフトウェアの何れでも実現可能なものである。

上述では、グラフィクスプレーン１２の解像度を１９２０画素×１０８０ライン、カラーシステムを２５６色のカラーパレットを用いた８ビットカラーマップアドレスとしたが、グラフィクスプレーン１２の解像度および色数は、上述に限定されない。

例えば、解像度を９６０画素×５４０ラインとし、サンプリング深さを２４ビットとして各画素にＲＧＢ各色にそれぞれ８ビットの色情報を持たせ、さらに各画素に８ビットのアルファデータを持たせるようにできる。使用可能な色数が上述の２５６色に比べて格段に増大し、表現力が増す。画素数が減るのでプレーンの書き換え速度が低下することもない。グラフィクスプレーン１２の用途を、将来的に自然画や高速のアニメーションの描画に拡張した際に、有用であると考えられる。

この例では、上述の１９２０画素×１０８０ラインの例に対して全体の画素数が１／４になる一方、各画素毎のデータ量が８ビットから３２ビットと４倍になるので、グラフィクスプレーン１２全体のデータ量に変化がない。そのため、追加のメモリは必要なく、フレームメモリの使用方法を変えるだけで容易に実現可能である。

また、ＲＧＢ各色にそれぞれ８ビットずつの色情報を持たせており、表示可能色数が十分とされるため、第２２図におけるパレット２２を省略できる。図示は省略するが、グラフィクスプレーン１２の画像データは、直接的にＲＧＢ／ＹＣｂＣｒ変換回路２７に入力されることになる。アルファデータα２は、ＲＧＢ／ＹＣｂＣｒ変換回路２７の前で抜き出されて、乗算器２８に供給される。

グラフィクスプレーン１２と合成される動画プレーン１０および字幕プレーン１１の解像度は、１９２０画素×１０８０ラインのままである。実際の表示においては、動画プレーン１０または字幕プレーン１１の例えば２画素×２ラインからなる４画素に対して、解像度を１／４にしたグラフィクスプレーン１２の１画素を繰り返し表示させ、見かけ上の解像度を合わせた後に、動画プレーン１０および字幕プレーン１１との合成を行う。

以上のようなプレーンを設定することで、再生専用規格に必要なメニュー画面とボタンの表示を可能とする。メニュー画面上のボタンが選択されると、そのボタンに対応したプレイリストが再生されることになる。このとき、プレイリスト間がどのように結び付けられているかという情報がディスク上に記録されていることが必要となる。節２－２でメニュー画面について説明し、節２－３および節２－４でプレイリスト間の結びつき（リンク）を定義するシナリオについて説明する。

２－２．メニュー画面について

グラフィクスプレーン１２には、ユーザに対して操作を促す画面、例えばメニュー画面を表示させることができる。第２５図は、グラフィクスプレーン１２に表示されるメニュー画面６０の一例を示す。メニュー画面６０は、文字や画像を特定の位置に表示させ、ユーザが選択や実行を指示することで新たな動作が開始される「リンク」や「ボタン」を配置することができる。

「リンク」は、例えば、文字列や画像データに対して所定のファイルへのアクセス方法を示す記述を設定し、ユーザがポインティングデバイスなどを用いて画面に表示された当

該文字列や画像データ上を指定することで、当該文字列や画像データに設定されたアクセス方法に従って当該ファイルにアクセスできるようにしたものである。「ボタン」は、例えば、「リンク」において通常の状態、選択された状態および押された状態の３種類の画像データを用意し、ユーザにより当該ボタン画像が指定された際に、３種類の画像データの表示を切り換えて、ユーザが行った操作をユーザ自身に認識しやすくしたものである。

「リンク」や「ボタン」に対する指定動作は、マウスを用いて画面上のカーソル表示を移動させ、「リンク」による文字列や画像、「ボタン」による画像上で、マウスボタンを１または複数回押すクリック動作を行うことでなされる。マウス以外のポインティングデバイスによっても、同様の動作を行うことができる。これに限らず、リモートコントロールコマンダやキーボードのキー操作により、「リンク」や「ボタン」の指定を行うこともできる。方向キーなど所定のキーで指定を行いたい「リンク」や「ボタン」を選択し、決定キーなどにより選択された「リンク」や「ボタン」を指定する。

第２５図の例では、グラフィクスプレーン１２上に表示されたメニュー画面６０の上部に、画像データとしてタイトル６１が表示される。タイトル６１に続けて、リンクとしての選択項目６２Ａ、６２Ｂ、６２Ｃおよび６２Ｄが表示される。例えばリモートコントロールコマンダのキー操作により、選択項目６２Ａ、６２Ｂ、６２Ｃおよび６２Ｄのうち１つを選択し指定することで、指定された選択項目にリンクとして設定されたファイルがアクセスされる。

メニュー画面６０の下部に、字幕の表示の有無や、出力される音声を例えば英語および日本語から選択するためのボタン６４および６５が表示される。これらボタン６４や６５を上述のように操作することで、それぞれの設定を行う画面を表示するためのファイルがアクセスされ、所定の画面表示がなされる。

メニュー画面６０の左下部分には、項目の選択方法を示す文字列６３が表示される。この文字列６３も、グラフィクスプレーン１２上の表示である。

第２５図に一例が示されるようなメニュー画面６０を表示するためには、画面の表示方法やリンク情報などを記述可能な、何らかの記述言語が必要とされる。Ｂｌｕ－ｒａｙ Ｄｉｓｃによる、このようなメニュー画面表示を可能とする記述言語としては、様々な方式のものが考えられるが、以下の説明では、下記の２種類の記述言語を用いた例について、それぞれ説明する。

（１）ＤＶＤビデオのナビゲーションコマンドを基に変更・拡張を加えたコマンド体系に、字幕・ボタン用の独自の表示制御コマンドを加えた命令群を追加した記述言語。以下、独自コマンドと称する。

（２）インターネット上のWWW（Ｗｏｒｌｄ Ｗｉｄｅ Ｗｅｂ）などで広く普及している記述言語である、ＨＴＭＬ（Ｈｙｐｅｒ Ｔｅｘｔ Ｍａｒｋｕｐ Ｌａｎｇｕａｇｅ）と、ＨＴＭＬと親和性の高いスクリプト言語であるＥＣＭＡスクリプト。

２－３．シナリオについて

さて、前節で述べたようなＢｌｕ－ｒａｙ Ｄｉｓｃのメニュー画面６０では、例えばプレイリストの一覧が画像データや文字列、ボタンなどで表示され、あるプレイリストを指定することで、指定されたプレイリストがディスクから読み出され再生されることが期待される。

第２５図の例では、メニュー画面６０に対してプレイリストの一覧が表示される。このメニュー画面６０およびメニュー画面６０上の選択項目を選択した際に再生される映像・音声は、実際には、複数のプレイリストが並んだ構造によって構成されている。メニューの一項目を構成する複数のプレイリスト内で、プレイリストの関連付けを行うことで、ストーリーを分岐させる仕組みを実現することができる。ストーリーを分岐できるようにすることで、ユーザの選択によってストーリーが変化していくマルチストーリー機能や、プレーヤの設定言語に従って自動的に適切な言語を再生する機能や、ユーザの年齢に応じてシーンを差し替えるパレンタル機能を実現することができる。

このような機能は、記録済みのディスクにおいて特に有用であって、テレビジョン放送の記録／再生が主目的である現行のＢｌｕ－ｒａｙ Ｄｉｓｃ規格では想定されていない

。
以下では、この複数のプレイリストが並べられた構造を、シナリオと称する。第２６図は、独自コマンドを用いた場合のシナリオ７０の一例の内部構造を示す。このシナリオ７０は、複数のプレイリスト７３Ａ～７３Ｍを含む。また、シナリオ７０は、グラフィクスプレーン１２を用いて分岐選択画面を表示する箇所が２箇所（画面８０Ａおよび８０Ｂ）設けられている。画面８０Ａは、分岐選択画面を表示するためのグラフィクスデータ７４Ａとプレイリスト７３Ｃを有する。同様に、画面８０Ｂは、分岐選択画面を表示するためのグラフィクスデータ７４Ｂとプレイリスト７３Ｊを有する。
シナリオは、プレイリストの並びを決めると共に、グラフィクスプレーン１２に対する表示のタイミングを規定する。グラフィクスプレーン１２への表示のタイミングは、グラフィクスプレーンに表示する画像に対して付加された表示制御命令によって規定することができる。
第２６図の例では、メニュー画面６０はシナリオ７０における画面８０Ａに対応する。メニュー画面６０における選択項目（例えば選択項目６２Ａ）は、グラフィクス７４Ａによって構成されている。メニュー画面６０において選択項目６２Ａが指定されると、選択項目に対応したプレイリスト７３Ｄが再生されることになる。
第２６図のシナリオ７０では、先ず、ディスクがプレーヤに挿入されると、プレイリスト７３Ａが再生される。プレイリスト７３Ａの再生が終了されると、続けてプレイリスト７３Ｂが再生され、プレイリスト７３Ｂの再生が終了されると、プレイリスト７３Ｃが再生され、グラフィクスデータ７４Ａが読み出されて、ユーザにストーリーの分岐選択を促す画面８０Ａが表示される。
画面８０Ａが表示された以降は、ユーザの選択に基づきストーリーが分岐される。この第２６図の例では、第１の選択により、画面８０Ａの後、プレイリスト７３Ｄ、７３Ｅ、７３Ｆの順に再生され、全体の再生が終了される。プレイリスト７３Ｆの再生が終了された後、メインのメニュー画面（例えば上述したメニュー画面６０）に戻るようにしてもよい。
画面８０Ａにおける第２の選択では、画面８０Ａの後、プレイリスト７３Ｇが再生される。プレイリスト７３Ｇは、所定のタイミングにマークが設定されていてもよい。プレイリスト７３Ｇが再生されると、再生装置自身の設定やユーザによる他のシナリオや分岐選択画面などでの選択に基づき、プレイリスト７３Ｇのマーク位置で分岐するか、プレイリスト７３Ｇを最後まで再生するかが制御される。プレイリスト７３Ｇが最後まで再生される場合には、プレイリスト７３Ｇの再生終了後、プレイリスト７３Ｍ、７３Ｉと再生され、プレイリスト７３Ｊがさらに再生される。
プレイリスト７３Ｇにおいてマーク位置で分岐された場合には、次にプレイリスト７３Ｋ、７３Ｌと再生され、プレイリスト７３Ｌの再生が終了されたら、次に、プレイリスト７３Ｉに設定されたマーク位置から再生が継続される。
プレイリスト７３Ｊは、グラフィクスデータ７４Ｂが読み出されて、ユーザにストーリーの分岐選択を促す画面８０Ｂが表される。画面８０Ｂにおける第１の選択では、プレイリスト７３Ｆが再生される。画面８０Ｂの第２の選択では、プレイリスト７３Ｋに設定されているマーク位置からの再生が行われる。
シナリオの再生において、マークの検出、ユーザからの入力、プレーヤの動作変更の検出がされた時の動作は、プレイリスト毎にコマンド列（プログラム）が用意されており、そのプログラムをプレーヤが実行することにより行われる。
次に、記述言語としてＨＴＭＬおよびＥＣＭＡスクリプトを用いた場合のシナリオ構造について、第２７図を用いて説明する。第２７図において、第２６図と共通する部分には同一の符号を付し、繁雑さを避けるために詳細な説明を省略する。
第２７図に示されるシナリオ７０’は、第２６図におけるシナリオ７０と同様のメニューや分岐選択を有し、シナリオ７０と同様に進行可能である。シナリオ７０’は、複数のプレイリスト７３Ａ～７３Ｍを含み、グラフィクスプレーン１２を用いて分岐選択画面を表示する箇所が２箇所（画面８０Ａおよび８０Ｂ）設けられる。

画面８０Ａは、分岐選択画面を表示するためのグラフィクスデータ７４Ａとプレイリスト７３Ｃを有する。画面８０Ｂは、分岐選択画面を表示するためのグラフィクスデータ７４Ｂとプレイリスト７３Ｊを有する。シナリオ７０’は、プレイリストの並びを決めると共に、グラフィクスプレーン１２に対する表示のタイミングを規定する。グラフィクスプレーン１２への表示タイミングは、プレイリスト中のマークを用いて規定することができる。

マークの検出、ユーザからの入力、プレーヤの動作変更の検出は、イベントドリブンモデルに基づきなされる。プレイリストの再生が開始された、プレイリストの再生が終了された、プレイリストの再生中にマークが検出された、リモートコントロールコマンダのキー操作などによりユーザからの入力があった、などによりイベントが発生される。特定のイベントがきっかけになって実行されるイベントハンドラをプログラム中に用意しておくことで、プレーヤにおいて、当該イベントにより期待される動作が実行される。

第２７図に一例が示されるシナリオ７０’には、２つのイベントハンドラ７１および７２が設定されている。これらのうち、イベントハンドラ７１は、シナリオ７０の全体にわたって有効なイベントハンドラが記述される、グローバルなイベントハンドラ定義である。

例えば、シナリオ７０’内のプレイリスト７３Ａ～７３Ｍの何れが再生中であっても、リモートコントロールコマンダに設けられたメニューボタンが押されたら、シナリオの一覧が表示されるメニュー画面６０が表示される。この、メニュー画面６０を表示するためのプレイリストに再生処理が移行される動作を実現したい場合について考える。この場合、リモートコントロールコマンダのメニューボタンが押されたときに発生するイベント（メニューボタン押下イベント）に対応するイベントハンドラであり、メニュー画面６０を表示させるプレイリストに処理が移行される命令を、グローバルなイベントハンドラ７１として記述する。

イベントハンドラ７２は、特定のプレイリストの再生中にのみ、あるいは、特定のユーザ入力画面が表示されている間のみに実行され得るイベントハンドラが記述される、ローカルなイベントハンドラ定義である。例えば、分岐選択画面である画面８０Ａに表示されているリンクがユーザにより指定されたときに、別のプレイリストを再生するという動作は、リンクが指定されたというイベントに対して、プレイリストの再生を開始する命令をローカルなイベントハンドラ７２として記述するすることで、実現することができる。

このようなイベントハンドラ定義は、ＥＣＭＡスクリプトを用いて記述される。ＥＣＭＡスクリプトは、ＥＣＭＡ（Ｅｕｒｏｐｅａｎ Ｃｏｍｐｕｔｅｒ Ｍａｎｕｆａｃｔｕｒｅｒｓ Ａｓｓｏｃｉａｔｉｏｎ）により定められた、ＪａｖａＳｃｒｉｐｔ（登録商標）に基づいたクロスプラットフォーム用のスクリプト言語である。ＥＣＭＡスクリプトは、ＨＴＭＬ文書との親和性が高いことと、独自のオブジェクトの定義が可能である。

２－４．シナリオの分類

シナリオは、後述するように、ＢＤＭＶディレクトリに一つ定義され、１または複数のプレイリストから構成される。ここで、シナリオの構成の分類について第２８図および第２９図Ａ、第２９図Ｂおよび第２９図Ｃを用いて説明する。シナリオの構成は、プレイリスト同士の接続を基に考えると、第２８図に示されるように、（１）シングルプレイリスト、（２）シーケンシャルプレイリスト、（３）マルチプレイリスト、の３種類に大別することができる。

（１）のシングルプレイリストは、第２９図Ａに一例が示されるように、単一のプレイリストで構成されるシナリオである。タイムラインが定義可能で、シナリオ再生中の割り込みが無い。コンテンツが映画であれば、ディスクのローディング後に映画本編だけが再生されるようなシナリオである。

（２）のシーケンシャルプレイリストは、第２９図Ｂに一例が示されるように、複数のプレイリストが分岐無しに直列的に並べられて構成されるシナリオである。各プレイリストは、プレイリストの末尾と次のプレイリストの先頭とが接続されるように並べられる。各プレイリストを通じてのタイムラインが定義可能である。コンテンツが映画であれば、

メニュー画面と映画本編との組み合わせからなるようなシナリオである。ディスクのローディング後にメニュー画面を表示させるプレイリストが実行され、メニュー画面に対して映画本編の再生が指示されると、次のプレイリストが実行され、映画の本編が再生される。

（３）のマルチプレイリストは、第２９図Ｃに一例が示されるように、プレイリストの分岐や収束による接続を含むシナリオである。プレイリストを通じてのタイムラインを定義することができず、例えば各プレイリスト内にタイムラインが定義される。マルチプレイリストによれば、ユーザ入力によって再生内容が変化するようなインタラクティブ性やゲーム性を実現することができる。コンテンツが映画の場合には、同一シーンを様々なアングルで収録し、再生時にユーザの指示により選択して鑑賞できるようにしたマルチアングルを実現するようなシナリオである。

再生専用メディアの場合、シナリオは、ＢＤＭＶディレクトリに一つ定義されるが、ユーザに対しては、シナリオをより細分化した単位を見せる必要がある。但し、プレイリストの単位がユーザの認識する単位と一致するとは限らない。例えば、一つのプレイリストに３本の映画が収録されているときには、各映画の頭出し点をユーザに見せる必要がある。プレイリストの構造とは独立した頭出し点（エントリポイント）を、タイトルおよび／またはチャプタと呼ぶことにする。

第３０図を用いてタイトルおよびチャプタについて説明する。タイトルは、シナリオ中の任意の再生開始点を表す。この第３０図の例では、プレイリスト４７０Ａの先頭にタイトル１に設定されている。タイトル２は、プレイリスト４７０Ｄの途中に設定されている。プレイリスト４７０Ａの先頭からタイトル２の直前までがタイトル１とされる。チャプタは、タイトルをさらに細分化した単位で、これもユーザが再生開始点として認識できる単位となる。タイトル１がチャプタによりさらに細分化される。第３０図の例では、タイトル１に対してチャプタ１、２および３が設定され、タイトル１が３つに細分化されている。また、第３０図に示されるように、タイトルおよびチャプタの位置は、プレイリストの途中を指定することもできる。

２－５．仮想プレーヤモデルについて

次に、シナリオ記述に従って動作する再生装置のモデルを考える。このモデル化された再生装置をＢＤ（Ｂｌｕ－ｒａｙ　ｄｉｓｃ）仮想プレーヤと称し、このＢＤ仮想プレーヤの構成の定義をＢＤ仮想プレーヤモデルと称する。

第３１図を用いて、ＢＤ仮想プレーヤモデルについて説明する。ＢＤ仮想プレーヤ３０は、ディスクのローディング後、この発明で定義したシナリオ記述言語により記述されディスクに記録されたシナリオを、ＰＢＣプログラム４０として読み込み、シナリオの記述に従い動作する。

ＢＤ仮想プレーヤ３０は、この発明の実施の一形態により定義されるディスク状記録媒体を再生するもので、パーソナルコンピュータなどのコンピュータ環境上のオブジェクトである。コンピュータ環境は、汎用的なパーソナルコンピュータに限られず、この発明の実施の一形態により定義されるディスク状記録媒体などを再生するように専用的に構成された再生装置および／または記録再生装置に組み込まれたソフトウェア環境も含む。以下、この発明の実施の一形態により定義されるディスク状記録媒体を、ディスクと略称する。

ＢＤ仮想プレーヤ３０の状態は、概略的には、プレイリストおよびグラフィクスを再生する状態Ａと、再生が停止された状態Ｂの２つの状態に分類できる。ある状態から別の状態への遷移や、ある状態の中での次の動作の指定は、ＢＤ仮想プレーヤ３０のオブジェクトに対するコマンドにより行われる。

状態Ａは、複数の動作を含む。状態Ａに含まれる動作としては、高速再生、逆転再生といった変速再生や、ディスク上の任意の時刻から再生する飛び込み再生といった特殊再生などが考えられる。ＢＤ仮想プレーヤ３０においてグラフィクスプレーン１２による表示がなされているときは、これら変速再生や特殊再生が制限されることがある。

ＰＢＣ（Ｐｌａｙ　Ｂａｃｋ　Ｃｏｎｔｒｏｌ）プログラム４０は、例えばディスクに

記録されているシナリオに対応する。詳細は後述するが、シナリオは、ディスクに記録されたプレイリストの再生方法や、メニュー画面の表示を指示する。ＰＢＣプログラム４０とＢＤ仮想プレーヤ３０とは、ＡＰＩ（Ａｐｐｌｉｃａｔｉｏｎ Ｐｒｏｇｒａｍｍｉｎｇ Ｉｎｔｅｒｆａｃｅ）４１を介してコマンドのやりとりを行い、ディスクに記録されたプレイリストを再生する。

より具体的には、ＢＤ仮想プレーヤ３０の状態遷移の際には、ＰＢＣプログラム４０により、ＡＰＩ４１を介して、ＢＤ仮想プレーヤ３０において使用目的が定められたメモリとして定義された共通パラメータ３２に対して、必要な情報が伝達される。この共通パラメータ３２に対する値のセットは、ＰＢＣプログラム４０とＢＤ仮想プレーヤ３０との間でＡＰＩ４１を介してやりとりされるコマンドによって直接、あるいは、ＡＰＩ４１を介して実行されるプレーヤコマンド３１から間接的に行うことができる。

この発明の実施の一形態では、ＢＤ仮想プレーヤ３０は、イベントドリブンモデルに基づき制御される。ＢＤ仮想プレーヤ３０の動作中には、様々なイベントが発生する。イベントは、ユーザによるキー入力やリモートコントロールコマンダによる操作、タイマによる割り込みなどにより、ハードウェア／ＯＳ（Ｏｐｅｒａｔｉｎｇ Ｓｙｓｔｅｍ）５０において発生され、ＢＤ仮想プレーヤ３０に渡される。これに限らず、再生されたプレイリスト中のマークの検出や、プレーヤの動作変更の検出など、ＢＤ仮想プレーヤ３０自身によりイベントが発生される場合もある。

発生されるイベントの種類は、ＢＤ仮想プレーヤモデルで定義されている。イベントが発生すると、発生したイベントに対応するイベントハンドラが実行される。ＢＤ仮想プレーヤに組み込まれているイベントハンドラが実行され、規格で規定されているプレーヤ動作が実行される。

ＢＤ仮想プレーヤ３０における割り込みイベントは、（１）再生中のコンテンツから発せられるイベント、（２）ユーザからの割り込みにより発生されるイベント、（３）プレーヤの状態の変化により発生されるイベント、の３種類に大別される。

（１）の、再生中のコンテンツから発せられるイベントは、予め設定された割り込みであって、当該コンテンツの再生の度に、同一のタイミングで発生される。例えば、ＢＤ仮想プレーヤ３０においてプレイリストを再生中に、プレイリスト中に記述されたマークで指し示したディスク上の時刻に到達すると、ＢＤ仮想プレーヤ３０においてマーク検出の割り込みが発生される。スクリプトにおいてタイマが設定された場合に、設定された時刻や、当該スクリプトによりタイマ設定がなされた１０秒後などにタイマ割り込みのイベントが発生される。

（２）の、ユーザからの割り込みによるイベントは、発生および発生タイミングの予測ができないイベントである。例えば、ユーザによるリモートコントロールコマンダのキー操作により、この割り込みイベントが発生される。この場合、ユーザが何時キー操作を行うかが分からないため、事前にそのタイミングを知ることができない。

（３）の、ＢＤ仮想プレーヤ３０の状態変化によるイベントは、例えば、音声や字幕のストリーム変更を通知するためのイベントである。コンテンツの再生状態から停止状態へ、停止状態から再生状態への状態遷移でも、このイベントが発生される。このプレーヤの状態変化によるイベントは、上述した（１）の再生中のコンテンツから発せられるイベントや、（２）のユーザによる割り込みイベントに伴い引き起こされる場合も多い。上述の音声や字幕のストリーム変更の通知のイベントの例では、ユーザによるリモートコントロールコマンダなどのキー操作による割り込みイベントで、音声や字幕のストリームが変更され、この変更に伴うＢＤ仮想プレーヤ３０の状態遷移に基づき、変更通知のイベントが発生される。

記述言語としてＨＴＭＬおよびＥＣＭＡスクリプトを用いた場合、グラフィクスプレーン１２を用いた表示制御がＨＴＭＬ（Ｈｙｐｅｒ Ｔｅｘｔ Ｍａｒｋｕｐ Ｌａｎｇｕａｇｅ）４．０またはＸＨＴＭＬ（ｅＸｔｅｎｓｉｂｌｅ ＨＴＭＬ）文書として記述され、グラフィクスプレーン１２による表示画面に対するイベントは、ＨＴＭＬ４．０の組み込みイベントが用いられる。また、ＨＴＭＬ４．０の組み込みイベントでは不十分なイ

ベントは、ＥＣＭＡスクリプトを用いて記述される。
ＨＴＭＬ形式とＥＣＭＡスクリプトを組み合わせて用いる場合、イベントが発生すると、先ず、そのイベントが発生した要素に属性で指定されているイベントハンドラが文書中に存在するか否かで、処理が異なる。イベントハンドラが存在する場合は、そのイベントハンドラが実行される。イベントハンドラが存在しない場合は、文書中にグローバルなイベントハンドラが用意されているか否かが探される。その結果、若し、グローバルなイベントハンドラが当該文書中に存在すれば、そのイベントハンドラが実行される。文書中にスクリプト言語によるイベントハンドラの記述が無い場合には、ＢＤ仮想プレーヤ３０において当該イベントに対して用意されたデフォルトのイベント処理が行われる。
イベントハンドラの記述方法としては、ＨＴＭＬ文書における要素の属性として記述する方法と、ＥＣＭＡスクリプトにおいてメソッドｃａｐｔｕｒｅＥｖｅｎｔｓを使って記述する方法とを用いることができる。
ＨＴＭＬ文書を用いてイベントハンドラを記述する方法について説明する。この発明の実施の第１の形態によるＢＤ仮想プレーヤ３０では、ＨＴＭＬ４．０に規定される組み込みイベントのうち、例えばイベントｏｎｌｏａｄ、イベントｏｎｕｎｌｏａｄ、イベントｏｎｃｌｉｃｋおよびイベントｏｎｋｅｙｐｒｅｓｓを用いることができる。これらのイベントは、タグの要素中に属性として記述される。
イベントｏｎｌｏａｄは、利用者エージェントが１つのウィンドウまたは１対のタグ〈ＦＲＡＭＥＳＥＴ〉〈／ＦＲＡＭＥＳＥＴ〉で定義される全てのフレームを終了する際に発生される。例えば、メニュー画面の表示の際にイベントｏｎｌｏａｄが発生される。
ウィンドウは、ＨＴＭＬの規定に基づきＨＴＭＬファイルを表示するブラウザアプリケーションにおける表示単位であり、フレームは、１つのウィンドウを分割して複数のＨＴＭＬファイルを表示させる際に用いられる枠組みである。フレーム内に表示されるＨＴＭＬファイルも含めてフレームと称する。このイベントｏｎｌｏａｄ属性は、要素ＢＯＤＹおよび要素ＦＲＡＭＥＳＥＴと共に使用することができる。
イベントｏｎｕｎｌｏａｄは、利用者エージェントが１つのウィンドウまたは１つのフレームから１つのＨＴＭＬ文書を除去する際に発生される。このイベントｏｎｕｎｌｏａｄ属性は、要素ＢＯＤＹおよび要素ＦＲＡＭＥＳＥＴと共に使用することができる。
イベントｏｎｃｌｉｃｋは、ポインティングデバイスなどにより要素が指定された場合に発生される。例えば、マウスボタンのクリック操作を行った場合に、イベントｏｎｃｌｉｃｋが発生される。このイベントｏｎｃｌｉｃｋ属性は、ＨＴＭＬ４．０における殆どの要素と共に用いることができる。
イベントｏｎｋｅｙｐｒｅｓｓは、キーが要素上で押されたり解放されたりする際に発生される。例えば、ある要素により定義される画面上の領域が選択状態のときに、キーボード上の所定のキーやリモートコントロールコマンダのキーが押された際に、イベントｏｎｋｅｙｐｒｅｓｓが発生される。このイベントｏｎｋｅｙｐｒｅｓｓ属性は、ＨＴＭＬ４．０における殆どの要素と共に用いることができる。
ＢＤ仮想プレーヤ３０の動作を制御するためには、上述のＨＴＭＬによるイベントでは不十分であるため、独自のイベントを定義する必要がある。第３２図Ａ、第３２図Ｂおよび第３２図Ｃは、ＢＤ仮想プレーヤ３０において独自に定義される一例のイベントを示す。このイベントは、ＥＣＭＡスクリプトにより、ＨＴＭＬ文書に埋め込まれて記述される。イベントハンドラを指定する属性名は、イベント名の先頭に「ｏｎ」を付して示す。
イベントＴｉｍｅｒＦｉｒｅｄは、カウントダウンタイマの値が「０」になるか、カウントアップタイマの値が指定された値になったときに発生される。イベントＰｌａｙＳｔｏｐｐｅｄおよびイベントＰｌａｙＳｔｉｌｌｅｄは、それぞれ再生の停止および一時停止により発生される。イベントＳｔｉｌｌＲｅｌｅａｓｅｄは、一時停止が解除された際に発生される。イベントＰｌａｙＰａｕｓｅｄおよびイベントＰａｕｓｅＲｅｌｅａｓｅｄは、それぞれユーザによる再生の一時停止および一時停止の解除の際に発生される。イベントＰｌａｙＳｔａｒｔｅｄは、再生が開始された際に発生される。イベントＰｌａｙＲｅｐｅａｔｅｄは、繰り返し再生時の先頭が検出された際に発生される。

イベントＳＰＤｉｓｐｌａｙＳｔａｔｕｓＣｈａｎｇｅｄは、サブピクチャ（字幕）ストリームの表示／非表示状態が変化した際に発生される。イベントＳｅｌｅｃｔｅｄＡｕｄｉｏＣｈａｎｇｅｄおよびイベントＶｉｄｅｏＳｔｏｐｐｅｄは、それぞれ再生するオーディオストリームおよびビデオストリームが変わった際に発生される。

イベントＳｃｅｎａｒｉｏＳｔａｒｔｅｄおよびイベントＳｃｅｎａｒｉｏＥｎｄｅｄは、それぞれシナリオの先頭および終端が検出された際に発生される。イベントＰｌａｙＬｉｓｔＳｔａｒｔｅｄおよびイベントＰｌａｙＬｉｓｔＥｎｄｅｄは、それぞれプレイリストの先頭および終端が検出された際に発生される。イベントＰｌａｙＩｔｅｍＳｔａｒｔｅｄおよびイベントＰｌａｙＩｔｅｍＥｎｄｅｄは、それぞれプレイアイテムの先頭および終端が検出された際に発生される。

イベントＭａｒｋＥｎｃｏｕｎｔｅｒｅｄは、プレイリストの再生中にマークが検出された際に発生される。このイベントは、例えばグラフィクスプレーン１２に画像データを表示する際に用いられる。共通パラメータ３２に、検出されたマークの種類と番号とが格納される。

イベントＢｕｔｔｏｎＰｒｅｓｓｅｄは、画面上に配置されたボタンが押された際に発生される。例えば、グラフィクスプレーン１２上に表示されたボタンを、キー操作やマウスのクリック操作などにより仮想的に押した場合に、イベントＢｕｔｔｏｎＰｒｅｓｓｅｄが発生される。

イベントＶａｌｉｄＰｅｒｉｏｄＳｔａｒｔｅｄは、有効期間が開始された際に発生される。これは、リンクの選択有効期間を設定する際に用いることができる。イベントＶａｌｉｄＰｅｒｉｏｄＥｎｄｅｄは、有効期間が終了した際に発生される。これは、リンクの強制実行のときに用いることができる。

イベントＫｅｙＰｒｅｓｓｅｄは、リモートコントロールコマンダのキーが押されたときに発生される。押されたキーの種類は、イベントハンドラ内の「ｓｗｉｔｃｈ」文などで識別される。

シナリオ記述言語として独自コマンドを用いた場合、この独自コマンドとしてシナリオを実行するに当たって必要なイベントを予め言語として定義することができる。そのため、独自コマンドを用いてシナリオ記述を行う場合には、汎用なＥＣＭＡスクリプトを用いる場合のように、シナリオの実行に特に適合したイベントをプログラム内で定義する必要は、無い。

２－６．コマンドについて

ＢＤ仮想プレーヤ３０は、一連のコマンドを有し、これらのコマンドにより、ＢＤ仮想プレーヤ３０における動作および状態、ＢＤ仮想プレーヤ３０で扱うビデオストリーム、オーディオストリームおよびサブピクチャ（字幕プレーン１１の画像データ）に関する情報取得や制御、共通パラメータ３２に対する操作、タイマやキー入力の割り込みに応じた処理、グラフィクスプレーン１２で扱う画像データの制御がそれぞれ定義される。

これらのコマンドは、第３１図で既に説明した、ＢＤ仮想プレーヤ３０におけるＡＰＩ４１に実装される。ＰＢＣプログラム４０の記述に基づきこれらのコマンドがＡＰＩ４１を介して呼び出され、ＢＤ仮想プレーヤ３０におけるディスク再生が制御される。ＰＢＣプログラム４０の具体的な例については、後述する。

ＢＤ仮想プレーヤ３０が有するコマンドは、シナリオ記述言語として独自コマンドを用いる場合と、ＨＴＭＬおよびＥＣＭＡスクリプトを用いる場合とで、多少異なる。先ず、シナリオ記述言語として独自コマンドを用いる場合のコマンドについて、第３３図Ａ、第３３図Ｂ、第３３図Ｃ、第３３図Ｄ、第３３図Ｅ、第３３図Ｆ、第３３図Ｇおよび第３３図Ｈを用いて説明する。

再生開始位置指定に関するコマンドについて説明する。コマンドＬｉｎｋＰｌａｙＬｉｓｔ（ｐｌａｙＬｉｓｔＮｕｍｂｅｒ）で、”ｐｌａｙＬｉｓｔＮｕｍｂｅｒ”で指定されたプレイリストの再生開始が指示される。コマンドＬｉｎｋＰｌａｙＩｔｅｍ（ｐｌａｙＬｉｓｔＮｕｍｂｅｒ，ｐｌａｙＩｔｅｍＮｕｍｂｅｒ）で、指定のプレイリストの指定のプレイアイテムからの再生開始が指示される。”ｐｌａｙＩｔｅｍＮｕｍｂｅｒ”は

、”ＰｌａｙＩｔｅｍ＿ｉｄ”であり、値が「０」から始まる。”ｐｌａｙＩｔｅｍＮｕｍｂｅｒ”に値「０」を指定することで、当該プレイアイテムが属するプレイリストの先頭から再生される。

コマンドＬｉｎｋ（ｐｏｓｉｔｉｏｎ）（ｏｂｊｅｃｔ）で、シナリオ内での移動が指示される。現在移動中の箇所から前後のプレイリスト、プレイアイテムおよびチャプタに移動することがこのコマンドで指示される。なお、パラメータ”ｐｏｓｉｔｉｏｎ”は、”ｐｒｅｖ”、”ｎｅｘｔ”、”ｔｏｐ”、”Ｐａｒｅｎｔ”または”ｔａｉｌ”の何れかの値を取り、パラメータ”ｏｂｊｅｃｔ”で示される移動対象（プレイリスト、プレイアイテムまたはチャプタ）に対する移動方法が指示される。

コマンドＥｘｉｔで、シナリオ再生の停止が指示される。この場合、標準レジスタの値は、保持されない。コマンドＲＳＭで、プレーヤ内のメモリに保存されているレジューム情報を呼び出し、レジスタにセットして再生を開始する。

プレーヤ状態の取得に関するコマンドについて説明する。コマンドｇｅｔＭｅｎｕＤｅｓｃｒｉｐｔｉｏｎＬａｎｇｕａｇｅ（）で、メニュー表示の際に用いられる言語が取得される。コマンドｇｅｔＳｃｅｎａｒｉｏＮｕｍｂｅｒ（）、コマンドｇｅｔＰｌａｙＬｉｓｔＮｕｍｂｅｒ（）およびコマンドｇｅｔＣｈａｐｔｅｒＮｕｍｂｅｒ（）で、再生中のシナリオ番号、プレイリスト番号およびチャプタ番号がそれぞれ取得される。コマンドｇｅｔＰｌａｙｅｒＳｕｐｐｏｒｔ（）で、プレーヤのバージョン情報が取得される。

ビデオストリームに関するコマンドについて説明する。コマンドｇｅｔＶｉｄｅｏＳｔｒｅａｍＡｖａｉｌａｂｉｌｉｔｙ（）で、指定のビデオストリームが含まれているか否かが取得される。コマンドｓｅｔＶｉｄｅｏＳｔｒｅａｍＮｕｍｂｅｒ（）で、デコードするビデオストリームが指定される。コマンドｇｅｔＶｉｄｅｏＳｔｒｅａｍＮｕｍｂｅｒ（）で、選択中のビデオストリームの番号が取得される。コマンドｇｅｔＶｉｄｅｏＳｔｒｅａｍＡｔｔｒｉｂｕｔｅ（）で、選択中のビデオストリームの属性が取得される。ビデオストリームの属性は、例えば、そのビデオストリームの符号化方式、解像度、アスペクト比、アスペクト比が４：３のときのディスプレイモード、クローズドキャプションの有無である。コマンドｓｅｔＡｎｇｌｅＮｕｍｂｅｒ（）で、アングル番号が指定される。コマンドｇｅｔＡｎｇｌｅＮｕｍｂｅｒ（）で、選択中のアングル番号が取得される。コマンドｇｅｔＭａｘＶｉｄｅｏＳｔｒｅａｍｓ（）で、ビデオストリームの最大数が取得される。

オーディオストリームに関するコマンドについて説明する。コマンドｇｅｔＡｕｄｉｏＳｔｒｅａｍＡｖａｉｌａｂｉｌｉｔｙ（）で、指定のオーディオストリームが含まれているか否かが取得される。コマンドｇｅｔＡｕｄｉｏＳｔｒｅａｍＬａｎｇｕａｇｅ（）で、指定のオーディオストリームの言語に関する情報が取得される。コマンドｇｅｔＡｕｄｉｏＳｔｒｅａｍＳｔａｔｕｓ（）で、指定のオーディオストリームの状態が取得され、コマンドｓｅｔＡｕｄｉｏＳｔｒｅａｍＳｔａｔｕｓ（）で、指定のオーディオストリームに対する状態のセットが指示される。オーディオストリームの状態は、例えば再生または非再生である。コマンドｇｅｔＡｕｄｉｏＳｔｒｅａｍＡｔｔｒｉｂｕｔｅ（）で、指定のオーディオストリームの属性が取得される。

サブピクチャストリーム（字幕データ）ストリームに関するコマンドについて説明する。コマンドｇｅｔＳＰＳｔｒｅａｍＡｖａｉｌａｂｉｌｉｔｙ（）で、指定のサブピクチャストリームが含まれているか否かが取得される。コマンドｇｅｔＳＰＳｔｒｅａｍＬａｎｇｕａｇｅ（）で、指定のサブピクチャストリームで用いられている言語が取得される。コマンドｇｅｔＳＰＤｉｓｐｌａｙＳｔａｔｕｓ（）で、サブピクチャストリームの表示状態が取得され、コマンドｓｅｔＳＰＤｉｓｐｌａｙＳｔａｔｕｓ（）で、サブピクチャストリームの表示状態がセットされる。サブピクチャストリームの表示状態は、例えばサブピクチャストリームの表示のオン／オフである。コマンドｇｅｔＳＰＳｔｒｅａｍＡｔｔｒｉｂｕｔｅ（）で、指定のサブピクチャストリームの属性が取得される。サブピクチャストリームの属性は、例えばサブピクチャストリームの表示がアスペクト比４：３またはワイド画面の何れでなされるかである。

共通パラメータ３２に関するコマンドについて説明する。図中では、レジスタリード／ライトとして記されている。コマンドｃｌｅａｒＲｅｇ（）で、ＢＤ仮想プレーヤ３０が内蔵しているメモリ領域すなわちレジスタについて、全レジスタの初期化が指示される。コマンドｓｅｔＲｅｇ（）で、指定のレジスタへの値のセットが指示される。コマンドｇｅｔＲｅｇ（）で、指定のレジスタからの値の読み出しが指示される。

タイマに関するコマンドについて説明する。コマンドｓｌｅｅｐ（）で、指定された時間の処理の停止が指示される。コマンドｓｅｔＴｉｍｅｏｕｔ（）で、指定された時間後に関数や処理を実行することが指示される。コマンドｓｅｔＩｎｔｅｒｖａｌ（）で、指定された時間毎に処理を実行することが指示される。タイマに関するコマンドにおいて、時間はミリ秒単位で指定することができる。コマンドｃｌｅａｒＴｉｍｅｒ（）で、指定された登録タイマＩＤの処理の中止が指示される。コマンドｐａｕｓｅＴｉｍｅｒ（）で、指定された登録タイマＩＤのタイマの一時停止が指示される。コマンドｒｅｓｕｍｅＴｉｍｅｒ（）で、指定された登録タイマＩＤのタイマを一時停止から再開することが指示される。

効果音に関するコマンドでは、コマンドｐｌａｙＳｏｕｎｄＥｆｆｅｃｔ（ｓｏｕｎｄ＿ｉｄ）で、指定の効果音を再生する。

この第３３図Ａ、第３３図Ｂ、第３３図Ｃ、第３３図Ｄ、第３３図Ｅ、第３３図Ｆ、第３３図Ｇおよび第３３図Ｈに例示されるようなコマンドを、後述するポストコマンド領域やボタンコマンド領域に書くことにより、特定のプレイリストへのジャンプを実現することができる。コマンドは、この第３３図Ａ、第３３図Ｂ、第３３図Ｃ、第３３図Ｄ、第３３図Ｅ、第３３図Ｆ、第３３図Ｇおよび第３３図Ｈに例示されるに止まらず、さらに他のコマンドを定義することも可能である。

シナリオ記述言語としてＨＴＭＬおよびＥＣＭＡスクリプトを用いる場合のコマンドについて、第３４図Ａ、第３４図Ｂ、第３４図Ｃ、第３４図Ｄ、第３４図Ｅ、第３４図Ｆ、第３４図Ｇ、第３４図Ｈ、第３４図Ｉおよび第３４図Ｊを用いて説明する。

プレーヤ動作に関するコマンドについて説明する。コマンドｐｌａｙＳｃｅｎａｒｉｏ（ｓｃｅｎａｒｉｏＮｕｍｂｅｒ，［ｓｃｅｎａｒｉｏＴｉｍｅ］）で、”ｓｃｅｎａｒｉｏＮｕｍｂｅｒ”で指定されるシナリオの再生が指示される。”ｓｃｅｎａｒｉｏＮｕｍｂｅｒ”は、実際には、シナリオ構成を記述したファイルなどの場所を示すＵＲＩ（Ｕｎｉｖｅｒｓａｌ　Ｒｅｓｏｕｒｃｅ　Ｉｄｅｎｔｉｆｉｅｒ）になる。コマンドｐｌａｙＰｌａｙＬｉｓｔ（ｐｌａｙＬｉｓｔＮｕｍｂｅｒ）で、”ｐｌａｙＬｉｓｔＮｕｍｂｅｒ”で指定されたプレイリストの再生が指示される。コマンドｐｌａｙＣｈａｐｔｅｒＭａｒｋ（ｐｌａｙＬｉｓｔＮｕｍｂｅｒ，ｃｈａｐｔｅｒＮｕｍｂｅｒ）で、”ｐｌａｙＬｉｓｔＮｕｍｂｅｒ”で示されるプレイリスト中の、”ｃｈａｐｔｅｒＮｕｍｂｅｒ”指定されるチャプタからの再生が指示される。コマンドｐｌａｙＰｌａｙＩｔｅｍ（ｐｌａｙＬｉｓｔＮｕｍｂｅｒ，ｐｌａｙＩｔｅｍＮｕｍｂｅｒ）で、”ｐｌａｙＬｉｓｔＮｕｍｂｅｒ”および”ｐｌａｙＩｔｅｍＮｕｍｂｅｒ”で示されるプレイアイテムからの再生が指示される。”ｐｌａｙＩｔｅｍＮｕｍｂｅｒ”は、”ｐｌａｙＩｔｅｍ＿ｉｄ”であり、値「０」を指定することで、当該プレイアイテムが属するプレイリストの先頭から再生される。

コマンドｐｌａｙ（ｐｏｓｉｔｉｏｎ）（ｏｂｊｅｃｔ）で、シナリオ内での移動が指示される。現在移動中の箇所から前後のプレイリストやプレイアイテムに移動することがこのコマンドで指示される。パラメータ”ｐｏｓｉｔｉｏｎ”は、”ｐｒｅｖ”、”ｎｅｘｔ”、”ｔｏｐ”、”ｇｏＵｐ”または”ｔａｉｌ”の何れかの値を取り、パラメータ”ｏｂｊｅｃｔ”で示される移動対象（プレイリスト、プレイアイテムまたはチャプタ）に対する移動方法が指示される。

コマンドｓｔｏｐ（）で、シナリオ再生の停止が指示される。この場合、標準レジスタの値は、保持されない。コマンドｒｅｓｕｍｅ（）で、前回再生を停止した箇所からの再生の再開が指示される。コマンドｐｌａｙＳｏｕｎｄＥｆｆｅｃｔ（）で、選択された効果音の再生が指示される。

プレーヤ状態に関するコマンドについて説明する。コマンドｇｅｔＭｅｎｕＤｅｓｃｒｉｐｔｉｏｎＬａｎｇｕａｇｅ（）で、メニュー表示の際に用いられる言語が取得される。コマンドｇｅｔＳｃｅｎａｒｉｏＮｕｍｂｅｒ（）、コマンドｇｅｔＰｌａｙＬｉｓｔＮｕｍｂｅｒ（）およびコマンドｇｅｔＣｈａｐｔｅｒＮｕｍｂｅｒ（）で、再生中のシナリオ番号、プレイリスト番号およびチャプタ番号がそれぞれ取得される。コマンドｇｅｔＰｌａｙｅｒＳｕｐｐｏｒｔ（）で、プレーヤのバージョン情報が取得される。

ビデオストリームに関するコマンドについて説明する。コマンドｓｅｔＶｉｄｅｏＳｔｒｅａｍＮｕｍｂｅｒ（）で、デコードするビデオストリームが指定される。コマンドｇｅｔＶｉｄｅｏＳｔｒｅａｍＮｕｍｂｅｒ（）、コマンドｇｅｔＶｉｄｅｏＳｔｒｅａｍＳｔａｔｕｓ（）およびコマンドｇｅｔＶｉｄｅｏＳｔｒｅａｍＡｔｔｒ（）で、再生中のビデオストリームの番号、状態および属性がそれぞれ取得される。なお、ビデオストリームの属性は、例えば、そのビデオストリームの符号化方式、解像度、アスペクト比、アスペクト比が４：３のときのディスプレイモード、クローズドキャプションの有無である。コマンドｓｅｔＡｎｇｌｅＮｕｍｂｅｒ（）で、アングル番号が指定される。コマンドｇｅｔＡｎｇｌｅＮｕｍｂｅｒ（）で、選択中のアングル番号が取得される。コマンドｇｅｔＭａｘＶｉｄｅｏＳｔｒｅａｍ（）で、ビデオストリームの最大数が取得される。

オーディオストリームに関するコマンドについて説明する。コマンドｇｅｔＡｕｄｉｏＳｔｒｅａｍＡｖａｉｌａｂｉｌｉｔｙ（）で、指定のオーディオストリームが含まれているか否かが取得される。コマンドｇｅｔＡｕｄｉｏＳｔｒｅａｍＬａｎｇｕａｇｅ（）で、指定のオーディオストリームの言語に関する情報が取得される。コマンドｇｅｔＡｕｄｉｏＳｔｒｅａｍＳｔａｔｕｓ（）で、指定のオーディオストリームの状態が取得され、コマンドｓｅｔＡｕｄｉｏＳｔｒｅａｍＳｔａｔｕｓ（）で、指定のオーディオストリームに対する状態のセットが指示される。オーディオストリームの状態は、例えば再生または非再生である。コマンドｇｅｔＡｕｄｉｏＳｔｒｅａｍＡｔｔｒｉｂｕｔｅ（）で、指定のオーディオストリームの属性が取得される。

サブピクチャストリーム（字幕データ）ストリームに関するコマンドについて説明する。コマンドｇｅｔＳＰＳｔｒｅａｍＡｖａｉｌａｂｉｌｉｔｙ（）で、指定のサブピクチャストリームが含まれているか否かが取得される。コマンドｇｅｔＳＰＳｔｒｅａｍＬａｎｇｕａｇｅ（）で、指定のサブピクチャストリームで用いられている言語が取得される。コマンドｇｅｔＳＰＤｉｓｐｌａｙＳｔａｔｕｓ（）で、サブピクチャストリームの表示状態が取得され、コマンドｓｅｔＳＰＤｉｓｐｌａｙＳｔａｔｕｓ（）で、サブピクチャストリームの表示状態がセットされる。サブピクチャストリームの表示状態は、例えばサブピクチャストリームの表示のオン／オフである。コマンドｇｅｔＳｐＳｔｒｅａｍＡｔｔｒｉｂｕｔｅ（）で、指定のサブピクチャストリームの属性が取得される。サブピクチャストリームの属性は、例えばサブピクチャストリームの表示がアスペクト比４：３またはワイド画面の何れでなされるかである。

共通パラメータ３２に関するコマンドについて説明する。第３４図Ａ、第３４図Ｂ、第３４図Ｃ、第３４図Ｄ、第３４図Ｅ、第３４図Ｆ、第３４図Ｇ、第３４図Ｈ、第３４図Ｉおよび第３４図Ｊでは、レジスタリード／ライトとして記されている。コマンドｃｌｅａｒＲｅｇ（）で、ＢＤ仮想プレーヤ３０が内蔵しているメモリ領域すなわちレジスタについて、全レジスタの初期化が指示される。コマンドｓｅｔＲｅｇ（）で、指定のレジスタへの値のセットが指示される。コマンドｇｅｔＲｅｇ（）で、指定のレジスタからの値の読み出しが指示される。

タイマに関するコマンドについて説明する。コマンドｓｌｅｅｐ（）で、指定された時間の処理の停止が指示される。コマンドｓｅｔＴｉｍｅｏｕｔ（）で、指定された時間後に関数や処理を実行することが指示される。コマンドｓｅｔＩｎｔｅｒｖａｌ（）で、指定された時間毎に処理を実行することが指示される。タイマに関するコマンドにおいて、時間はミリ秒単位で指定することができる。コマンドｃｌｅａｒＴｉｍｅｒ（）で、指定された登録タイマＩＤの処理の中止が指示される。コマンドｐａｕｓｅＴｉｍｅｒ（）で、指定された登録タイマＩＤのタイマの一時停止が指示される。コマンドｒｅｓｕｍｅＴ

ｉｍｅｒ（）で、指定された登録タイマＩＤのタイマを一時停止から再開することが指示される。

キー入力に関するコマンドでは、コマンドｇｅｔＰｒｅｓｓｅｄＫｅｙ（）で、入力された（押された）キーの種類が取得される。

グラフィクスに関するコマンドについて説明する。コマンドｌｏａｄＧｒａｐｈｉｃｓ（ｈｔｍｌｆｉｌｅ，ＩＤ）で、”ｈｔｍｌｆｉｌｅ”で示されるファイルを読み込み、グラフィクスプレーン１２に非表示で展開することが指示される。展開されたグラフィクス画像には”ＩＤ”が割り当てられ、後述のコマンドで参照される。コマンドｓｈｏｗＧｒａｐｈｉｃｓ（ＩＤ）は、上述のコマンドｌｏａｄＧｒａｐｈｉｃｓ（ｈｔｍｌｆｉｌｅ，ＩＤ）により既にグラフィクスプレーン１２上に展開されている画像に対して、表示が指示される。コマンドｈｉｄｅＧｒａｐｈｉｃｓ（ＩＤ）で、”ＩＤ”で示される画像の表示を消去することが指示される。

その他のコマンドについて説明する。コマンドｒａｎｄｏｍ（ｉｎｐｕｔ　Ｎｕｍｂｅｒ　ｎｕｍ）で、１から”ｎｕｍ”までの乱数の発生が指示される。乱数の発生は、独自の定義で行われる。コマンドｃａｔｃｈＥｖｅｎｔ（ｅｖｅｎｔｎａｍｅ，ｅｖｅｎｔｈａｎｄｌｅｒ）で、”ｅｖｅｎｔｎａｍｅ”で示されたイベントが発生した際、”ｅｖｅｎｔｈａｎｄｌｅｒ”で示された関数を実行することが指示される。

２－７．コマンドの実行について

上述のように定義されたコマンドの実行について説明する。先ず、シナリオ記述言語として独自コマンドを用いた場合の例について説明する。シナリオ記述言語として独自コマンドを用いた場合、シナリオには、プレーヤの動作を指示するコマンドが並んだプログラムを含むコマンドを配置する領域として、グローバルコマンド領域と、ローカルコマンド領域の二つがある。

グローバルコマンド領域は、シナリオ全体に有効なプログラムが格納されており、例えば、ディスクがプレーヤに挿入されたときに最初に自動的にプレーヤによって実行されて、パラメータの初期化を行い、メニュー画面を構成するプレイリストにジャンプするようなプログラムを記述する領域である。ローカルコマンド領域は、各プレイリストに関係するプログラムを記述する領域である。ローカルコマンドは、さらにプレコマンド、プレイアイテムコマンド、ポストコマンド、ボタンコマンドの４種類に分類される。

第３５図Ａおよび第３５図Ｂは、独自コマンドを記述言語として用いたシナリオで記述されるコマンドによるＢＤ仮想プレーヤ３０の動作を概略的に示す。第３５図Ａは、ディスクのローディング時の動作の例を示す。既に述べたように、シナリオは、後述するＢＤＭＶディレクトリに対して一つが作られる。ディスクがプレーヤに挿入されディスクに対するイニシャルアクセスがなされると（ステップＳ３０）、レジスタすなわち共通パラメータ３２が初期化される（ステップＳ３１）。そして、次のステップＳ３２で、プログラムがディスクから読み込まれて実行される。イニシャルアクセスとは、ディスク挿入時のように、ディスクの再生が初めて行われることをいう。

この、ディスクローディング時に最初に読み込まれ、実行されるコマンド群（プログラム）を、グローバルコマンドと称する。グローバルコマンドは、例えば宣伝用映像（トレーラー）やメニュー画面を構成するプレイリストへのジャンプ命令が記述されており、その命令通りにプレーヤがプレイリストを再生していくことになる。

第３５図Ｂは、プレーヤ３０が停止状態からユーザにより例えばプレイキーが押下され再生が指示された場合の動作の例を示す。これは、第３１図で説明したＢＤ仮想プレーヤ３０における状態Ｂから状態Ａへの状態遷移に相当する。最初の停止状態（ステップＳ４０）に対して、ユーザにより、例えばリモコンなどを用いて再生が指示される（ＵＯＰ：Ｕｓｅｒ　Ｏｐｅｒａｔｉｏｎ）。再生が指示されると、先ず、レジスタすなわち共通パラメータ３２が初期化され（ステップＳ４１）、次のステップＳ４２で、プレイリストの再生フェイズに移行する。

プレイリストの再生フェイズにおけるプレイリストの再生について、第３６図Ａおよび第３６図Ｂを用いて説明する。第３６図Ａは、プレイリストが単一のプレイアイテムから

なる場合の例である。プレイリストは、プレコマンド領域、プレイアイテムコマンド領域およびポストコマンド領域の３箇所にプログラムが配される。プレイリストの再生フェイズに移行すると、先ずプレコマンド領域のプレコマンドが実行され（ステップＳ１０）、プレコマンドの実行が終了されると、プレイリストを構成するプレイアイテムの再生フェイズに移行する（ステップＳ１１）。プレイアイテムの再生フェイズでは、プレイアイテムにおいて開始点および終了点で指定されたストリームが再生される（ステップＳ１１０）。終了点までの再生が終了した時点で、プレイアイテムコマンドが実行される（ステップＳ１１１）。プレイアイテムコマンドが実行されると、次にポストコマンド領域のポストコマンドが実行され（ステップＳ１２）、プレイリストの再生が終了される。

ポストコマンドは、一般的な使用においては、次に再生すべきプレイリストへのジャンプ命令や、メニュー画面を構成するプレイリストへのジャンプ命令が記述される。ジャンプ命令が無い場合には、そこで再生が停止し、プレーヤは、停止状態（第３１図の状態Ｂ）に遷移される。

第３６図Ｂは、プレイリストが複数のプレイアイテムを含む場合の例である。この場合でも、プレイリスト中にプレコマンド領域、プレイアイテムコマンド領域およびポストコマンド領域が配され、それぞれプログラムが格納される。複数のプレイアイテムを含む場合、プレイアイテムコマンド領域では、プレイアイテム数分のプレイアイテムストリームおよびプレイアイテムコマンドが、時系列に従って配置される。

プレイリストが複数のプレイアイテムを含む場合でも、プレイリストの再生フェイズに移行すると、先ず、プレコマンドが実行される（ステップＳ１０）。次のプレイアイテムの再生フェイズでは、プレイアイテムの開始点から終了点までのストリームの再生と、プレイアイテムコマンドの実行とが、プレイリストが含むプレイアイテムの数だけ実行される。第３６図Ｂの例では、最初のプレイアイテムストリームが再生され（ステップＳ１１０－１）、対応するプレイアイテムコマンドが実行される（ステップＳ１１１－１）。次に、図示は省略するが、２番目のプレイアイテムストリームが再生され（ステップＳ１１０－２）、対応するプレイアイテムコマンドが実行される（ステップＳ１１１－２）。これをプレイアイテムの数だけ行い、最後のプレイアイテムストリームが再生され（ステップＳ１１０－ｎ）、対応するプレイアイテムコマンドが実行されると（ステップＳ１１１－ｎ）、プレイアイテムの再生フェイズが終了する。プレイアイテムの再生フェイズが終了すると、ポストコマンドが実行され（ステップＳ１２）、このプレイリストの再生フェイズが終了される。

この発明の実施の一形態では、ＢＤ仮想プレーヤ３０、ならびに、ＢＤ仮想プレーヤ３０上で実行されるシナリオ、プレイリストおよびプレイアイテムを、階層的に考えることができる。すなわち、第３７図Ａで示されるように、ＢＤ仮想プレーヤ層６００の下に１つのシナリオ層６０１があり、シナリオ層６０１の下に、１または複数のプレイリストを含むプレイリスト層６０２がある。プレイリスト層６０２の下には、プレイアイテム（ＰＩ）層６０３があり、プレイリスト毎に１または複数のプレイアイテムを持つことができる。

このような階層的な構造とすることで、プレイリストおよびプレイアイテムは、シナリオ層６０１を介してＢＤ仮想プレーヤにより実行されることになる。したがって、シナリオにプレイリストに対する制御コマンドを記述することで、プレイリストの分岐などの制御が容易に実現される。これは、プレイアイテムに関しても同様である。この様子を第３７図Ｂに示す。

シナリオ記述言語としてＨＴＭＬおよびＥＣＭＡスクリプトを用いた場合の例について説明する。以下の説明は、シナリオ記述言語としてＨＴＭＬおよびＥＣＭＡスクリプトを用いた場合の、より具体的なＰＢＣプログラム４０の例である。

ＨＴＭＬおよびＥＣＭＡスクリプトを用いてシナリオを記述した場合、一つのシナリオに対して、一つのスクリプトファイルが作成される。グラフィクスプレーン１２を用いてメニュー画面６０などを表示する場合には、１画面に対して一つのＨＴＭＬファイルが作成される。スクリプトファイルおよびＨＴＭＬファイルは、ファイル名の拡張子がそれぞ

れ「ｊｓ」および「ｈｔｍｌ」とされ、拡張子で以てこれら２種類のファイルが識別可能される。ディスクがドライブ装置に挿入されて最初に実行されるスクリプトプログラムが格納されるファイルは、ファイル名を例えば「ｓｔａｒｔｕｐ．ｊｓ」に固定的とする。

一例として、第３８図に示されるようなシナリオ構成のディスクを考える。このディスクは、シナリオＳｃｅｎａｒｉｏ０００およびＳｃｅｎａｒｉｏ００１の２つのシナリオを有し、シナリオＳｃｅｎａｒｉｏ０００は、シナリオＳｃｅｎａｒｉｏ００１へのリンクボタン９２を表示するメニュー画面９１を表示させる。ディスクが再生装置に装填されると、メニュー画面９１が表示され、メニュー画面９１に表示されるボタン９２が例えばクリック操作により押されると、シナリオＳｃｅｎａｒｉｏ００１が再生される。シナリオＳｃｅｎａｒｉｏ００１の再生が終了されると、再びメニュー画面９１が表示される。

第３９図は、第３８図の構成に対して必要とされる一例のファイルを一覧して示す。この例では、ファイル「ｓｔａｒｔｕｐ．ｊｓ」、ファイル「ｓｃｅｎａｒｉｏ０００．ｊｓ」、ファイル「０００．ｈｔｍｌ」、ファイル「００００．ｒｐｌｓ」、ファイル「ｓｃｅｎａｒｉｏ００１．ｊｓ」およびファイル「００００１．ｒｐｌｓ」の６ファイルが必要とされる。

これらのうち、ファイル「ｓｃｅｎａｒｉｏ０００．ｊｓ」は、シナリオＳｃｅｎａｒｉｏ０００の構成情報が記述されるスクリプトファイルであって、メニュー画面９１すなわちシナリオ一覧画面の構成情報が記述される。ファイル「０００．ｈｔｍｌ」は、メニュー画面９１のレイアウト情報が記述されるＨＴＭＬファイルである。ファイル「００００．ｒｐｌｓ」は、メニュー画面９１において背景として表示されるプレイリストファイルである。ファイル「ｓｃｅｎａｒｉｏ００１．ｊｓ」は、シナリオＳｃｅｎａｒｉｏ００１の構成情報が記述されるスクリプトファイルである。ファイル「００００１．ｒｐｌｓ」は、シナリオＳｃｅｎａｒｉｏ００１で再生されるプレイリストの情報が記述されたプレイリストファイルである。

第３９図では、プレイリストファイル「００００．ｒｐｌｓ」および「０００００１．ｒｐｌｓ」で実際に再生されるコンテンツファイルなど（クリップ情報ファイル「％％％％％．ｃｌｉｐ」、ＡＶストリームファイル「＊＊＊＊＊．ｍ２ｔｓ」）については省略されている。

第３９図に示される各ファイルは、第４０図に一例が示されるディレクトリ構造に従い、ディスクに記録される。ディレクトリＢＤＡＶの直下にファイル「ｓｔａｒｔｕｐ．ｊｓ」が置かれると共に、ディレクトリＢＤＡＶの配下にディレクトリＳＣＲＩＰＴが設けられる。このディレクトリＳＣＲＩＰＴの下に、シナリオの構成情報が記述されるスクリプトファイル（ファイル「ｓｃｅｎａｒｉｏ０００．ｊｓ」など）や、メニュー画面のレイアウト情報などが記述されるＨＴＭＬファイル（ファイル「０００．ｈｔｍｌ」など）が置かれる。

第４１図～第４４図は、スクリプトファイルおよびＨＴＭＬファイルのより具体的な記述の例を示す。第４１図は、ファイル「ｓｔａｒｔｕｐ．ｊｓ」の記述例である。ファイル「ｓｔａｒｔｕｐ．ｊｓ」では、当該ディスクに格納されているシナリオの数と名前とが定義される。そして、メソッドｐｌａｙＳｃｅｎａｒｉｏ（”ｓｃｅｎａｒｉｏ０００”）の記述により、ファイル「ｓｃｅｎａｒｉｏ０００．ｊｓ」が呼び出され、シナリオＳｃｅｎａｒｉｏ０００の再生が指示される。シナリオＳｃｅｎａｒｉｏ０００が再生されることにより、メニュー画面９１が表示される。

第４２図は、ファイル「ｓｃｅｎａｒｉｏ０００．ｊｓ」の記述例である。関数ｆｕｎｃｔｉｏｎが定義され、メソッドｃａｔｃｈＥｖｅｎｔ（）に「ｏｎ」が付されて記述されたイベント（第３２図Ａ、第３２図Ｂおよび第３２図Ｃ参照）が発生すると、対応する関数ｆｕｎｃｔｉｏｎで定義された内容が実行される。第４２図の例では、メニュー画面９１のレイアウト情報が記述されたＨＴＭＬファイル「０００．ｈｔｍｌ」が指定され、グラフィクスプレーン１２へのメニュー画面９１の読み込みおよび表示のタイミングが制御される。それと共に、メソッドｐｌａｙＰｌａｙＬｉｓｔ（”００００．ｒｐｌｓ”）が記述され、プレイリスト「００００．ｒｐｌｓ」の再生が指示される。

このファイル「ｓｃｅｎａｒｉｏ０００．ｊｓ」により、プレイリスト「００００．ｒｐｌｓ」による動画の動画プレーン１０に対する表示が指示されると共に、プレイリスト「００００．ｒｐｌｓ」の再生中に検出されたマークのタイミングで、メニュー画面９１をグラフィクスプレーン１２に表示する旨が指示される。

第４３図は、ファイル「０００．ｈｔｍｌ」の記述例である。タグ〈ｓｔｙｌｅ　ｔｙｐｅ＝”ｔｅｘｔ／ｃｓｓ”〉および〈／ｓｔｙｌｅ〉で囲まれた部分に、「ｍｅｎｕ」および「ｓｃｅｎａｒｉｏ００１」で参照される画像の、メニュー画面９１におけるレイアウト情報が記述される。この第４３図の例では、レイアウト情報が画面上の絶対座標で記述され、画像名「ｍｅｎｕ」で参照される画像データが画面の上端から２００ピクセル、左端から８００ピクセルの位置に、幅２００ピクセル、高さ５０ピクセルの画像として表示されることが示される。同様に、画像名「ｓｃｅｎａｒｉｏ」で参照される画像データが画面の上端から７００ピクセル、左端から７００ピクセルの位置に、幅４００ピクセル、高さ１００ピクセルの画像として表示されることが示される。

タグ〈ｓｃｒｉｐｔ　ｔｙｐｅ＝”ｔｅｘｔ／ｊａｖａｓｃｒｉｐｔ”〉および〈／ｓｃｒｉｐｔ〉で囲まれた部分に、関数ｆｕｎｃｔｉｏｎで、マウス操作に対するイベントハンドラｏｎＭｏｖｅｒｈａｎｄｌｅｒ（ｆ）、ｏｎＭｏｕｔｈａｎｄｌｅｒ（ｆ）およびｏｎＭｃｌｉｃｋｈａｎｄｌｅｒ（ｆ）がそれぞれ定義される。この第４３図の例では、イベントハンドラｏｎＭｏｖｅｒｈａｎｄｌｅｒ（ｆ）、ｏｎＭｏｕｔｈａｎｄｌｅｒ（ｆ）およびｏｎＭｃｌｉｃｋｈａｎｄｌｅｒ（ｆ）に対してボタン画像を表示する画像データである画像データ「２０１．ｐｎｇ」、「２００．ｐｎｇ」および「２０２．ｐｎｇ」がそれぞれ対応付けられると共に、イベントハンドラｏｎＭｃｌｉｃｋｈａｎｄｌｅｒ（ｆ）においては、シナリオファイル「ｓｃｅｎａｒｉｏ００１．ｊｓ」の再生が指示される。

タグ〈ｂｏｄｙ〉および〈／ｂｏｄｙ〉で囲まれた部分に、メニュー画面９１においてグラフィクスプレーン１２に表示される画像データが指定される。上述のタグ〈ｓｔｙｌｅ　ｔｙｐｅ＝”ｔｅｘｔ／ｃｓｓ”〉および〈／ｓｔｙｌｅ〉で囲まれた部分に記述される画像名に対応する画像データのファイル名（「１００．ｐｎｇ」、「２００．ｐｎｇ」）がそれぞれ記述される。画像名「ｓｃｅｎａｒｉｏ００１」で参照される画像データに対しては、マウスなどポインティングデバイスに対する操作に応じて発生するイベントｏｎＭｏｕｓｅｏｖｅｒ、ｏｎＭｏｕｓｅｏｕｔおよびｏｎｃｌｉｃｋに対して、上述の関数ｆｕｎｃｔｉｏｎで定義されたイベントハンドラｏｎＭｏｖｅｒｈａｎｄｌｅｒ（ｆ）、ｏｎＭｏｕｔｈａｎｄｌｅｒ（ｆ）およびｏｎＭｃｌｉｃｋｈａｎｄｌｅｒ（ｆ）がそれぞれ実行されることが指示される。

イベントｏｎＭｏｕｓｅｏｖｅｒは、指定された領域などにカーソル表示が重なった際に発生されるイベントである。イベントｏｎＭｏｕｓｅｏｕｔは、指定された領域などからカーソル表示が離れた際に発生されるイベントである。イベントｏｎｃｌｉｃｋは、指定された領域にカーソル表示があるときにポインティングデバイスに対する所定の操作、例えばマウスボタンのクリックが行われた際に発生されるイベントである。

第４４図は、ファイル「ｓｃｅｎａｒｉｏ００１．ｊｓ」の記述例である。関数ｆｕｎｃｔｉｏｎ　ＵＯＰＣｏｎｔｒｏｌ（）において、このシナリオファイル「ｓｃｅｎａｒｉｏ００１．ｊｓ」の再生中に、例えばリモートコントロールコマンダに設けられたメニューキーが押されたときに、メニュー画面９１を表示するためのシナリオＳｃｅｎａｒｉｏ０００を再生することが定義される。また、関数ｆｕｎｃｔｉｏｎ　ｐｌａｙＬｉｓｔ　Ｅｎｄｅｄ（）により、このシナリオファイル「ｓｃｅｎａｒｉｏ００１．ｊｓ」によるプレイリストの再生が終了したら、メニュー画面９１を表示するためのシナリオＳｃｅｎａｒｉｏ０００を再生することが定義される。このシナリオファイル「ｓｃｅｎａｒｉｏ００１．ｊｓ」でプレイリスト「００００１．ｒｐｌｓ」が再生されることが指示される。

第４１図～第４４図の記述に基づく動作について説明する。ディスクが再生装置に装填されると、ファイル「ｓｔａｒｔｕｐ．ｊｓ」が読み込まれ、このファイル中の記述によ

りファイル「ｓｃｅｎａｒｉｏ０００．ｊｓ」が呼び出される。このファイルに記述されたシナリオ「ｓｃｅｎａｒｉｏ０００」が実行されると、第４２図の記述に基づき、プレイリスト「００００．ｒｐｌｓ」による動画が動画プレーン１０に表示される。

プレイリスト「００００．ｒｐｌｓ」中に指定されたマークに対応するタイミングで、ファイル「０００．ｈｔｍｌ」が呼び出され、ファイル「０００．ｈｔｍｌ」の記述に基づきシナリオ一覧を表示するメニュー画面９１がグラフィクスプレーン１２上に展開され、表示される。このメニュー画面９１自体も、シナリオ「ｓｃｅｎａｒｉｏ０００」という一つのシナリオで構成されたものである。

メニュー画面９１には、例えば「Ｍｅｎｕ」という文字列が描画された画像ファイル「１００．ｐｎｇ」と、「Ｓｃｅｎａｒｉｏ００１」という文字列が描画された画像ファイル「２００．ｐｎｇ」とが配置されている。これらの画像ファイルは、グラフィクスプレーン１２上に配置され表示される。グラフィクスプレーン１２の背景に表示される動画プレーン１０上では、プレイリスト「００００．ｒｐｌｓ」による動画が表示される。動画プレーン１０上のプレイリスト「００００．ｒｐｌｓ」による動画と、グラフィクスプレーン１２上のファイル「０００．ｈｔｍｌ」によるメニュー画面９１とが重ね合わされて同一画面上に表示される。動画を背景にメニュー画面９１が表示される。

このとき、グラフィクスプレーン１２上に表示されている画面（メニュー画面９１）に所定に透明度を設定し、メニュー画面９１を動画プレーン１０上の動画に対して透過させて表示させることができる。この例では、プレイリスト「００００．ｒｐｌｓ」の先頭および末尾にマークが設定され、メニュー画面９１は、プレイリスト「００００．ｒｐｌｓ」の再生と共に表示が開始され、プレイリスト「００００．ｒｐｌｓ」の再生終了と共に表示が消去されるようになっている。

メニュー画面９１上には、ユーザのリモートコントロールコマンダのキー操作などに応じて移動可能なカーソルが表示される。画像ファイル「２００．ｐｎｇ」の表示にカーソル表示が重なると、ファイル「０００．ｈｔｍｌ」において定義されるイベントＭｏｕｓｅｏｖｅｒが発生される。このイベントＭｏｕｓｅｏｖｅｒが発生されると、当該画像ファイル「２００．ｐｎｇ」にフォーカスが当たっていることを表現するために、イベントｏｎＭｏｕｓｅｏｖｅｒに対応するイベントハンドラｏｎＭｏｖｅｒｈａｎｄｌｅｒ（）が実行される。イベントハンドラｏｎＭｏｖｅｒｈａｎｄｌｅｒ（）の実行により、画像ファイル「２００．ｐｎｇ」が画像ファイル「２０１．ｐｎｇ」に置き換えられる。画像ファイル「２０１．ｐｎｇ」は、画像ファイル「２００．ｐｎｇ」に対して配色が変えられたボタン画像などである。

カーソル表示が画像ファイル「２０１．ｐｎｇ」上にあるときに、ユーザにより、リモートコントロールコマンダの所定のキー操作によりクリック操作が行われると、イベントｏｎｃｌｉｃｋに対応するイベントハンドラｏｎＭｃｌｉｃｋｈａｎｄｌｅｒ（）が実行され、画像ファイル「２０１．ｐｎｇ」が、選択されたことを表現する画像ファイル「２０２．ｐｎｇ」に置き換えられる。画像ファイル「２０２．ｐｎｇ」は、例えばボタンが押されたことを仮想的に表現するようなボタン画像である。

このように、「フォーカスされた」、「クリックされた」というイベント発生に対応するイベントハンドラをファイル「０００．ｈｔｍｌ」中に記述することで、ユーザの入力に反応するインタラクティブな機能を有するメニュー画面が実現される。

メニュー画面９１において、「Ｓｃｅｎａｒｉｏ００１」と描画されたボタン画像がクリックされると、シナリオ「Ｓｃｅｎａｒｉｏ００１」の再生処理に移行される。ファイル「ｓｃｅｎａｒｉｏ００１．ｊｓ」が実行されることで、シナリオ「Ｓｃｅｎａｒｉｏ００１」が再生される。第４４図に示したように、ファイル「ｓｃｅｎａｒｉｏ００１．ｊｓ」中に記述されたメソッドｐｌａｙＰｌａｙＬｉｓｔ（”００００１．ｒｐｌｓ”）が呼び出され、プレイリスト「００００１．ｒｐｌｓ」が再生される。

プレイリスト「００００１．ｒｐｌｓ」の再生が終了されると、プレイリスト再生終了のイベントＰｌａｙＬｉｓｔＥｎｄｅｄ（）が発生され、このイベントに対応したイベントハンドラｐｌａｙＳｃｅｎａｒｉｏ（”ｓｃｅｎａｒｉｏ０００．ｊｓ”）により、シ

ナリオ「Ｓｃｅｎａｒｉｏ０００．ｊｓ」を再生するように指示される。この例では、シナリオ「Ｓｃｅｎａｒｉｏ００１」の再生が終了されると、再びメニュー画面９１が表示される。

シナリオ「Ｓｃｅｎａｒｉｏ００１」の再生中に、「ｋｅｙＩＤ」で指定されるキーが操作されても、シナリオ「Ｓｃｅｎａｒｉｏ０００．ｊｓ」の再生が指示され、再びメニュー画面９１が表示される。

第４１図～第４４図で示したＨＴＭＬおよびＥＣＭＡスクリプトの記述は、一例であって、これはこの例に限定されるものではない。ＨＴＭＬやＥＣＭＡスクリプトの記述には自由度があり、一部を異なる記述にしても同等の動作を実現するようにできる。

２－８．シンタクスについて

シナリオ記述言語として独自コマンドを用いた場合の、各ファイルのシンタクスについて説明する。先ず、シナリオを記述するコマンドやデータベースのディスクへの格納方法について説明する。第４５図は、シナリオ記述言語として独自コマンドを用いた場合の一例のファイルの管理構造を示す。

ディスク上には、先ず、１つのルートディレクトリが作成される。このルートディレクトリの下が、１つの再生システムで管理される範囲とする。ルートディレクトリの下に、ディレクトリＢＤＭＶが置かれる。図示はされていないが、ディレクトリＢＤＭＶは、第９図の如く、ルートディレクトリの下に複数を置くことができる。

ディレクトリＢＤＭＶの下には、２つのファイルすなわちファイル「ｓｃｅｎａｒｉｏ．ｈｄｍｖ」およびファイル「ｅｎｔｒｙｌｉｓｔ．ｄａｔａ」が置かれると共に、ディレクトリ「ＰＬＡＹＬＩＳＴ」、ディレクトリ「ＣＬＩＰＩＮＦ」、ディレクトリ「ＳＴＲＥＡＭ」といった複数のディレクトリが置かれる。

第４６図は、ファイル「ｓｃｅｎａｒｉｏ．ｈｄｍｖ」の一例の構造を表すシンタクスを示す。このファイル「ｓｃｅｎａｒｉｏ．ｈｄｍｖ」は、ディスク挿入時などのイニシャルアクセス時に最初に読み込まれ実行されるファイルである。ファイル「ｓｃｅｎａｒｉｏ．ｈｄｍｖ」は、最初にファイル識別記号（フィールドｔｙｐｅ＿ｉｎｄｉｃａｔｏｒ）、バージョン番号（フィールドｖｅｒｓｉｏｎ＿ｎｕｍｂｅｒ）が記述され、その後に、機能毎のデータを集めたブロックが並んでいる。

フィールドｔｙｐｅ＿ｉｎｄｉｃａｔｏｒは、３２ビットのデータ長を有し、このファイルがファイル「ｓｃｅｎａｒｉｏ．ｈｄｍｖ」であることを表す特定の文字列が格納されている。フィールドｖｅｒｓｉｏｎ＿ｎｕｍｂｅｒは、３２ビットのデータ長を有し、バージョン番号が入る。フィールドＳｃｅｎａｒｉｏ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓは、３２ビットのデータ長を有する符号無しの整数で、ブロックＳｃｅｎａｒｉｏ（）が開始される位置をファイル「ｓｃｅｎａｒｉｏ．ｈｄｍｖ」の先頭からの相対バイト数で表した値が格納される。

ブロックＡｕｔｏｐｌａｙ（）は、ファイルの先頭から４１バイト目の固定位置から始まる。ブロックＡｕｔｏｐｌａｙ（）には、イニシャルアクセス時（ディスク挿入時のようにディスクの再生が始めて行われる時）に実行されるプログラムが記述される。Ａｕｔｏｐｌａｙ（）の後には、任意の数のパディングワード（ｐａｄｄｉｎｇ＿ｗｏｒｄ）が入り、ブロックの後に空きを持たせることができるようになっている。

第４７図は、第４６図におけるブロックＡｕｔｏｐｌａｙ（）の一例のデータ構造を表すシンタクスを示す。フィールドｌｅｎｇｔｈは、データ長が３２ビットの符号無しの整数で、このフィールドｌｅｎｇｔｈの直後からブロックＡｕｔｏｐｌａｙ（）の終わりまでのデータ長を、バイトで表したものが記述される。フィールドｎｕｍｂｅｒ＿ｏｆ＿ｃｏｍｍａｎｄｓは、その後に続くフィールドｃｏｍｍａｎｄ（ｉ）の数を表す。フィールドｃｏｍｍａｎｄ（ｉ）は、データ長が３２ビットの値で、上述の第３３図Ａ、第３３図Ｂ、第３３図Ｃ、第３３図Ｄ、第３３図Ｅ、第３３図Ｆ、第３３図Ｇおよび第３３図Ｈに一例が記されている、プレーヤのパラメータのセット、指定のプレイリストの再生開始、演算などの命令が記述されている。

ブロックＳｃｅｎａｒｉｏ（）は、これまで説明してきた「シナリオ」が記述される。

プレイリストの再生順序の情報およびプレイリスト毎に持つローカルコマンド領域などが、このブロックＳｃｅｎａｒｉｏ（）に記述される。

第４８図は、ブロックＳｃｅｎａｒｉｏ（）の一例の構造を表すシンタクスを示す。ブロックＳｃｅｎａｒｉｏ（）は、シナリオの情報、すなわちプレイリスト間のリンクを定義するブロックであって、上述したプレコマンド、ポストコマンド、プレイアイテムコマンドなどの情報およびコマンドそのものが格納される。ブロックＳｃｅｎａｒｉｏ（）は、ブロックＳｃｅｎａｒｉｏ（）内に格納されるコマンドの情報を示すフィールドが記述される領域と、実際のプレコマンド、ポストコマンド、プレイアイテムコマンドが列挙される領域とから構成される。

フィールドｌｅｎｇｔｈは、このフィールドｌｅｎｇｔｈの直後からブロックＳｃｅｎａｒｉｏ（）の終わりまでの長さをバイトで表現した値が記述される。フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＬｉｓｔｓは、シナリオを構成するプレイリストの数を表す。このフィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＬｉｓｔｓ以降は、プレイリスト毎に持つデータである。すなわち、フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＬｉｓｔｓの次行からのｆｏｒループにより、フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＬｉｓｔｓで示される値をループカウンタｉの最大値として、プレイリスト毎のデータが記述される。

フィールドＰｒｅ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄは、プレイリストを再生する前に実行されるプログラム（プレコマンド）の、コマンドテーブル中での開始番号を表す。フィールドＰｒｅ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄにより示される番号は、後述するフィールドＰＬ＿Ｃｏｍｍａｎｄ（ｉ）が列挙されるｆｏｒループ内でのループカウンタｊを表す。同様に、フィールドＰｏｓｔ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄは、プレイリストを再生した後に実行されるプログラム（ポストコマンド」）の、コマンドテーブル中での開始番号を表す。フィールドＰｏｓｔ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄにより示される番号は、後述するフィールドＰＬ＿Ｃｏｍｍａｎｄ（ｊ）が列挙されるｆｏｒループ内でのループカウンタｊを表す。

フィールドｎｕｍｂｅｒ＿ｏｆ＿Ｐｒｅ＿Ｃｏｍｍａｎｄｓは、このプレイリストを再生する前に実行されるプログラム（プレコマンド）を構成するコマンドの数を表す。同様に、フィールドｎｕｍｂｅｒ＿ｏｆ＿Ｐｏｓｔ＿Ｃｏｎｍｍａｎｄｓは、このプレイリストを再生した後に実行されるプログラム（ポストコマンド）を構成するコマンドの数を表す。これらのプログラムは、後述するコマンドテーブル内に記述される。

フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＩｔｅｍｓは、このプレイリストを構成するプレイアイテムの数を表す。フィールドＰＩ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄは、プレイアイテムの再生後に実行するコマンドの、コマンドテーブル中の開始番号を表す。フィールドＰＩ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄにより示される番号は、後述するコマンドテーブル内でのループカウンタｊを表す。フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰＩ＿Ｃｏｍｍａｎｄｓは、プレイアイテムの再生後に実行するコマンド（プレイアイテムコマンド）の数を表す。後述するコマンドテーブル内において、フィールドＰＩ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄで示される位置から始まり、フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰＩ＿Ｃｏｍｍａｎｄｓで表される数のコマンドを、プレイアイテムの再生後に実行する。

フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰＬ＿Ｃｏｍｍａｎｄｓは、次行からのコマンドテーブル中のコマンドの数を表す。コマンドテーブルは、フィールドＰＬ＿Ｃｏｍｍａｎｄ（ｊ）が列挙されるｆｏｒループにより構成される。コマンドテーブル内のコマンドには、順に番号ｊが割り当てられている。番号ｊは、コマンドテーブルを記述するためのｆｏｒループのループカウンタｊに対応する。フィールドＰＬ＿Ｃｏｍｍａｎｄ（ｊ）は、一つのコマンドを表し、番号ｊは、上述したフィールドＰｒｅ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄ、フィールドＰｏｓｔ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄおよびフィールドＰＩ＿Ｃｏｍｍａｎｄ＿ｓｔａｒｔ＿ｉｄから参照される。

第４９図は、ファイル「ｅｎｔｒｙｌｉｓｔ．ｄａｔａ」の一例のデータ構造を表したシンタクスを示す。ファイル「ｅｎｔｒｙｌｉｓｔ．ｄａｔａ」には、最初にファイル識別記号（フィールドｔｙｐｅ＿ｉｎｄｉｃａｔｏｒ）、バージョン番号（フィールドｖｅ

ｒｓｉｏｎ＿ｎｕｍｂｅｒ）、ブロックの開始アドレス（フィールドＳｃｅｎａｒｉｏＥｎｔｒｙ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓ）が記述され、その後に、機能毎のデータを集めたブロックが並んでいる。

フィールドｔｙｐｅ＿ｉｎｄｉｃａｔｏｒには、データ長が３２ビットを有し、このファイルがタイトルやメニューのエントリポイントを記述したファイルであることを表す特定の文字列が格納されている。フィールドｖｅｒｓｉｏｎ＿ｎｕｍｂｅｒには、データ長が３２ビットを有し、バージョン番号が格納される。フィールドＳｃｅｎａｒｉｏＥｎｔｒｙ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓには、データ長が３２ビットの符号無しの整数で、ブロックＳｃｅｎａｒｉｏＥｎｔｒｙ（）が開始される位置を、ファイル「ｅｎｔｒｙｌｉｓｔ．ｄａｔａ」の先頭からの相対バイト数で表した値が格納される。

第５０図は、ブロックＡｐｐＩｎｆｏ（）の一例の構造を表すシンタクスを示す。フィールドｌｅｎｇｔｈには、データ長が３２ビットの符号無しの整数で、このフィールドｌｅｎｇｔｈの直後からブロックＡｐｐＩｎｆｏ（）の終わりまでの長さをバイトで表したものが記述される。フィールドＢＤＭＶ＿ｎａｍｅ＿ｃｈａｒａｃｔｅｒ＿ｓｅｔは、後述するフィールドＢＤＭＶ＿ｎａｍｅを記述する文字セットを表す。フィールドＰＩＮ＿ｖａｌｉｄ＿ｆｌａｇは、再生時に暗証番号を設定するか否かを表し、設定有効の場合は、直後のフィールドＰＩＮが暗証番号を表す。フィールドＢＤＭＶ＿ｎａｍｅ＿ｌｅｎｇｔｈは、次のフィールドＢＤＭＶ＿ｎａｍｅの有効部分の長さを表す。フィールドＢＤＭＶ＿ｎａｍｅは、このファイル「ｅｎｔｒｙｌｉｓｔ．ｄａｔａ」が置かれるディレクトリＢＤＭＶに付けられた名称がテキスト形式で格納されている領域である。フィールドＢＤＭＶ＿ｎａｍｅは、データ長が２５５バイトの固定長とされる。実際に名称が格納される部分は、フィールドＢＤＭＶ＿ｎａｍｅの先頭からフィールドＢＤＭＶ＿ｎａｍｅ＿ｌｅｎｇｔｈで指定されている長さまでとされる。

第５１図は、ブロックＳｃｅｎａｒｉｏＥｎｔｒｙ（）の一例の構造を表すシンタクスを示す。ブロックＳｃｅｎａｒｉｏＥｎｔｒｙ（）は、シナリオ内の頭出し点が羅列される。シナリオは、上述したようにディレクトリＢＤＭＶに対して一つ作られ、ディレクトリＢＤＭＶの下に置かれる複数のプレイリストを結び付けて再生順序を定義する。ユーザから見た場合、シナリオは、必ずしも一つの映像・音声単位として見える訳ではなく、複数の「タイトル」から構成されているように見える。

例えば、一枚のディスクに映画が３本収録されている場合、再生順序を定義するシナリオは、ディスクに一つのみ存在するが、ユーザには当該ディスクに３つのタイトルが収録されているように見える。あるいは、その３つのタイトルのリストを表示し、ユーザに選択させるタイトルメニュー画面も含めて、４つのタイトルに分割されているように見える。メニュー画面もユーザにとっては一まとまりの映像、音声単位であるので、この発明の実施の一形態では、タイトルの一種としてみなすことにする。

このように、プレイリストの結びつきを定義するシナリオという単位と、ユーザが一まとまりの映像、音声として認識する単位には違いがあるため、シナリオ内に頭出し点を定義しておく必要がある。このシナリオ内の頭出し点を、タイトルエントリと称する。このブロックＳｃｅｎａｒｉｏＥｎｔｒｙ（）に、タイトルエントリの情報が記述される。

第５１図の説明に戻り、フィールドｌｅｎｇｔｈは、データ長が３２ビットの符号無しの整数で、このフィールドｌｅｎｇｔｈの直後からブロックＳｃｅｎａｒｉｏＥｎｔｒｙ（）の終わりまでの長さをバイトで表したものが格納される。フィールドｎａｍｅ＿ｃｈａｒａｃｔｅｒ＿ｓｅｔは、後述するフィールドＴｏｐＭｅｎｕ＿ｎａｍｅとフィールドＴｉｔｌｅ＿ｎａｍｅとを表現する文字セットを表す。

次のブロックＴｏｐ　Ｍｅｎｕ　ＰＬ（）は、ユーザがリモコンのタイトルメニューキーを押したときに表示されるメニューを構成しているプレイリストあるいはプレイリスト群へのエントリポイントを指定する。トップメニューは、シナリオに一つあり、例えばタイトルをユーザに提示するためのメニューとして利用される。トップメニューに対して、さらにオーディオや字幕を設定するサブメニューを下層メニューとして設けることができる。サブメニューは、ストリーム設定メニューとも称される。

フィールドｆｌａｇｓは、詳細は省略するが、トップメニューに関する属性情報を表す領域である。フィールドＴｏｐＭｅｎｕ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＬｉｓｔ＿ｆｉｌｅ＿ｎａｍｅは、トップメニューを構成するプレイリストあるいはプレイリスト群の入り口となるプレイリストを指定する。フィールドＴｏｐＭｅｎｕ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＩｔｅｍ＿ｉｄは、フィールドＴｏｐＭｅｎｕ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＬｉｓｔ＿ｆｉｌｅ＿ｎａｍｅで指定したプレイリスト中の特定のプレイアイテムからトップメニューが開始されている場合、そのプレイアイテムの番号を指定する。プレイリストの先頭からの再生であれば、フィールドＴｏｐＭｅｎｕ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＩｔｅｍ＿ｉｄの値は、「０」となる。フィールドＴｏｐＭｅｎｕ＿ｎａｍｅ＿ｌｅｎｇｔｈは、トップメニューに付された名前の長さを表す。フィールドＴｏｐＭｅｎｕ＿ｎａｍｅは、トップメニューに付された名前の文字列が格納される。

ブロックＴｏｐ　Ｍｅｎｕ　ＰＬ（）の次に、タイトルに関する情報が記述される。フィールドｎｕｍｂｅｒ＿ｏｆ＿Ｔｉｔｌｅｓは、その直後のｆｏｒループ内のタイトル頭出し点（タイトルエントリ）の数を表す。フィールドｆｌａｇｓは、詳細は省略するが、タイトルに関する属性情報を表す領域である。フィールドＴｉｔｌｅ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＬｉｓｔ＿ｆｉｌｅ＿ｎａｍｅは、タイトルエントリを含むプレイリストのファイル名を表す。フィールドＴｉｔｌｅ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＩｔｅｍ＿ｉｄは、フィールドＴｉｔｌｅ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＬｉｓｔ＿ｆｉｌｅ＿ｎａｍｅで指定したプレイリストの中の特定のプレイアイテムからタイトルが始まる場合に用いる。フィールドＴｉｔｌｅ＿ｎａｍｅ＿ｌｅｎｇｔｈは、タイトルに付された名前の長さを表す。フィールドＴｉｔｌｅ＿ｎａｍｅは、タイトルに付された名前の文字列が格納される。

サブメニューに関する情報が記述される。「Ｓｔｒｅａｍ　Ｓｅｔｕｐ　Ｍｅｎｕ」以下に、プレイアイテム毎に持つことができるストリーム設定メニュー（すなわちサブメニュー）を構成するプレイリストあるいはプレイリスト群へのエントリポイントが記述されている。ストリーム設定メニューは、オーディオ、字幕、アングルなどの切り替えなど、プレイリスト毎に個別にメニューを用意したい場合に利用することができる。例えば、上述の第２５図におけるボタン６４、６５を押すことで表示される画面がサブメニューである。

フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＬｉｓｔｓは、ストリーム設定メニューに用いられるプレイリストの数を表す。このフィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＬｉｓｔｓの値が直後のｆｏｒループのループ回数として用いられる。フィールドＳＳＭｅｎｕ＿ｆｌａｇｓは、詳細は省略するが、ストリーム設定メニューに関連する属性情報を格納する領域である。フィールドＳＳＭｅｎｕ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＬｉｓｔ＿ｆｉｌｅ＿ｎａｍｅは、ストリーム設定メニューを構成するプレイリストあるいはプレイリスト群の入り口となるプレイリストを指定する。フィールドＳＳＭｅｎｕ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＩｔｅｍ＿ｉｄは、フィールドＳＳＭｅｎｕ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＬｉｓｔ＿ｆｉｌｅ＿ｎａｍｅで指定したプレイリスト中の特定のプレイアイテムからストリーム設定メニューが開始されている場合、そのプレイアイテムの番号を指定する。プレイリストの先頭からの再生であれば、フィールドＳＳＭｅｎｕ＿ｒｅｆ＿ｔｏ＿ＰｌａｙＩｔｅｍ＿ｉｄの値は、「０」となる。

第５２図は、ファイル「ｘｘｘｘｘ．ｍｐｌｓ」の一例の構造を表すシンタクスを示す。第５２図において、ファイル「ｘｘｘｘｘ．ｍｐｌｓ」の内部は、機能別の情報毎にブロックが構成される。フィールドｔｙｐｅ＿ｉｎｄｉｃａｔｏｒにこのファイルを示す文字列が格納され、フィールドｖｅｒｓｉｏｎ＿ｎｕｍｂｅｒにこのファイルのバージョンが示される。フィールドＰｌａｙＬｉｓｔ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓおよびＰｌａｙＬｉｓｔＭａｒｋ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓには、それぞれ対応するブロックの先頭アドレスが例えばデータ長が３２ビットのアドレス情報として示される。

ブロックＰＬＣｏｎｔｒｏｌＩｎｆｏ（）は、このプレイリストに関する属性情報が格納される。ブロックＰｌａｙＬｉｓｔ（）は、このプレイリストを構成するプレイアイテムに関する情報が格納される。ブロックＰｌａｙＬｉｓｔＭａｒｋ（）は、このプレイリ

ストに付されるマークの情報が格納される。

ファイル「ｘｘｘｘｘ．ｍｐｌｓ」において、ブロックＰＬＣｏｎｔｒｏｌＩｎｆｏ（）、ＰｌａｙＬｉｓｔ（）およびＰｌａｙＬｉｓｔＭａｒｋ（）の先頭アドレスがこれらのブロックより前に示されるために、各ブロックの前および／または後ろに、任意の長さのパディングデータｐａｄｄｉｎｇ＿ｗｏｒｄを挿入することができる。ファイル「ｘｘｘｘｘ．ｍｐｌｓ」の最初のブロックであるブロックＰＬＣｏｎｔｒｏｌＩｎｆｏ（）の開始位置は、ファイルの先頭から４１バイト目に固定される。

第５３図は、ブロックＰＬＣｏｎｔｒｏｌＩｎｆｏ（）の一例の構造を表すシンタクスを示す。このブロックＰＬＣｏｎｔｒｏｌＩｎｆｏ（）は、プレイリストの再生において直接的に必要とされない、プレイリストに関する各種の属性情報が格納される。フィールドＰｌａｙＬｉｓｔ＿ｃｈａｒａｃｔｅｒ＿ｓｅｔは、プレイリストに関する文字列情報の文字セットが指定される。

フィールドＰＬ＿ｐｌａｙｂａｃｋ＿ｔｙｐｅは、第５４図に一例が示されるような値を取り、このプレイリストがシーケンシャルに再生される通常のプレイリストか、プレイアイテムをランダム再生するプレイリストか、プレイアイテムをシャッフル再生するプレイリストであるかを表す。ランダムシャッフルは、プレイリストの単位で指定する。一つのプレイリストに通常再生のプレイアイテムとランダムシャッフルのプレイアイテムブロックを混在させない。再生専用の記録媒体の場合には、制作者の意図により、ランダム再生やシャッフル再生を指定されることがあるため、このような情報が必要とされる。

フィールドｐｌａｙｂａｃｋ＿ｃｏｕｎｔは、このプレイリストがランダム再生あるいはシャッフル再生のプレイリストであるとき、プレイアイテムの再生回数を指定する。このフィールドｐｌａｙｂａｃｋ＿ｃｏｕｎｔにより、ランダム再生あるいはシャッフル再生に用いるプレイアイテムの数を指定できる。

フィールドＰＬ＿ＵＯＰ＿ｍａｓｋ＿ｔａｂｌｅ（）は、ユーザ操作を制限する情報が格納される。再生、早送り、早戻しなどのユーザ操作を、このプレイリストの再生中に禁止したい場合、この領域で適切な指定をする。例えば、このフィールドＰＬ＿ＵＯＰ＿ｍａｓｋ＿ｔａｂｌｅ（）内の値を所定に設定することで、警告表示や著作権の表示を早送りなどで飛ばして再生されないようにできる。

フィールドＰＬ＿ｒａｎｄｏｍ＿ａｃｃｅｓｓ＿ｍｏｄｅは、第５５図に一例が示される値を取り、他プレイリストからこのプレイリスト中の任意の箇所に飛び込み再生をするランダムアクセスが可能か否かを示す。例えば、必ずユーザに見せたいプレイリストがある場合に、フィールドＰＬ＿ｒａｎｄｏｍ＿ａｃｃｅｓｓ＿ｍｏｄｅの値を〔０ｘ１〕に設定する。このプレイリストに飛び込んで再生する際には、早送り、早戻し、任意時刻からの再生などが不可能になり、必ずプレイリストの先頭からの再生となる。再生専用の記録媒体においては、制作会社のロゴ表示や注意事項など、ユーザに必ず見せたいシーンが収録されている場合がある。このフィールドＰＬ＿ｒａｎｄｏｍ＿ａｃｃｅｓｓ＿ｍｏｄｅは、そのようなシーンを変速再生等で飛ばされないようにするために必要な情報である。フィールドＰＬ＿ＵＯＰ＿ｍａｓｋ＿ｔａｂｌｅ（）が当該プレイリストの再生中の操作の制限であるのに対し、このＰＬ＿ｒａｎｄｏｍ＿ａｃｃｅｓｓ＿ｍｏｄｅは他プレイリストから当該プレイリストに飛び込んで再生を開始する際の制限を定めているという違いがある。

フィールドＰｌａｙＬｉｓｔ＿ｄｕｒａｔｉｏｎは、プレイリストの再生時間を示す。フィールドＰｌａｙＬｉｓｔ＿ｎａｍｅは、フィールドＰｌａｙＬｉｓｔ＿ｎａｍｅ＿ｌｅｎｇｔｈに示される値を有効長としてプレイリスト名が示される。フィールドＰｌａｙＬｉｓｔ＿ｄｅｔａｉｌは、フィールドＰｌａｙＬｉｓｔ＿ｄｅｔａｉｌ＿ｌｅｎｇｔｈに示される値を有効長としてプレイリストの詳細情報が示される。

第５６図は、ブロックＰｌａｙＬｉｓｔ（）の一例の構造を表すシンタクスを示す。フィールドｌｅｎｇｔｈは、フィールドｌｅｎｇｔｈの直後のフィールドからこのブロックＰｌａｙＬｉｓｔ（）の終端までのバイト長を示す。フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰｌａｙＩｔｅｍｓは、このプレイリストを構成するプレイアイテムの数を示す。フィール

ドｎｕｍｂｅｒ＿ｏｆ＿ＳｕｂＰｌａｙＩｔｅｍｓは、このメインのプレイアイテムと同時に再生される補助的なプレイアイテム（サブプレイアイテム）の数を示す。

ブロックＰｌａｙＩｔｅｍ（）は、プレイアイテムの情報が格納される。ブロックＳｕｂＰｌａｙＩｔｅｍ（）は、サブプレイアイテムの情報が格納される。

第５７図は、ブロックＰｌａｙＩｔｅｍ（）の一例の構造を表すシンタクスを示す。フィールドＣｌｉｐ＿Ｉｎｆｏｒｍａｔｉｏｎ＿ｆｉｌｅ＿ｎａｍｅは、このプレイアイテムが参照しているクリップと１対１に対応するクリップ情報ファイル（拡張子が「ｃｌｐｉ」であるファイル）のファイル名が文字列で示される。

フィールドＣｌｉｐ＿ｃｏｄｅｃ＿ｉｄｅｎｔｉｆｉｅｒは、このプレイアイテムにより参照されるクリップの符号化方式を示す。この実施の一形態では、フィールドＣｌｉｐ＿ｃｏｄｅｃ＿Ｉｄｅｎｔｉｆｉｅｒは、値「Ｍ２ＴＳ」に固定的とされる。すなわち、この実施の一形態では、プレイアイテムにより参照されるクリップの符号化方式が値「Ｍ２ＴＳ」により示される方式に固定的とされる。

フラグｉｓ＿ｍｕｌｔｉ＿ａｎｇｌｅは、このプレイアイテムがマルチアングル構造であるかどうかを表す。

フィールドｃｏｎｎｅｃｔｉｏｎ＿ｃｏｎｄｉｔｉｏｎは、このプレイアイテムと次のプレイアイテムとがどのように接続されているかを示す情報である。フィールドｃｏｎｎｅｃｔｉｏｎ＿ｃｏｎｄｉｔｉｏｎにより、プレイアイテムとプレイアイテムとの間が継ぎ目なくシームレスに再生できるか否かが示される。

フィールドｒｅｆ＿ｔｏ＿ＳＴＣ＿ｉｄは、このプレイアイテムにより参照されるクリップ内のシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅを指定する。シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅは、ＭＰＥＧ２　ＴＳ（Ｔｒａｎｓｐｏｒｔ　Ｓｔｒｅａｍ）における時間軸の基準であるＰＣＲ（Ｐｒｏｇｒａｍ　Ｃｌｏｃｋ　Ｒｅｆｅｒｅｎｃｅ）が連続な範囲を表すＢｌｕ－ｒａｙ　Ｄｉｓｃ規格独自の構造である。シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅには、クリップ内で一意な番号ＳＴＣ＿ｉｄが割り当てられる。このシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅ内では、不連続の無い一貫した時間軸を定義できるので、プレイアイテムの開始時刻および終了時刻を一意に定めることができる。つまり、各プレイアイテムの開始点と終了点は、同一のシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅに存在していなければならない。フィールドｒｅｆ＿ｔｏ＿ＳＴＣ＿ｉｄでは、番号ＳＴＣ＿ｉｄによりシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅが指定される。

フィールドＩＮ＿ｔｉｍｅおよびＯＵＴ＿Ｔｉｍｅは、このプレイアイテムにおける開始点および終了点の、シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅ上でのタイムスタンプｐｔｓ（ｐｒｅｓｅｎｔａｔｉｏｎ＿ｔｉｍｅ＿ｓｔａｍｐ）をそれぞれ示す。

フィールドＰＩ＿ＵＯＰ＿ｍａｓｋ＿ｔａｂｌｅ（）は、ユーザ操作を制限するためのデータが格納される。ここで制限されたユーザ操作は、仮にユーザがその操作を行ったとしても、プレーヤは反応してはならない。例えばメニュー画面で早送りが実行されないようにする場合に、ここで早送りのユーザ操作を無効にするような設定をしておく。

このプレイアイテム毎に設けられるフィールドＰＩ＿ＵＯＰ＿ｍａｓｋ＿ｔａｂｌｅ（）は、上述したプレイリストの再生に関するブロックＰＬＣｏｎｔｒｏｌＩｎｆｏ（）に設けられるフィールドＰＬ＿ＵＯＰ＿ｍａｓｋ＿ｔａｂｌｅ（）と同様の目的の情報が格納される。プレイリストとプレイアイテムとの何方かで禁止となっていれば、そのユーザ操作は禁止となる。プレイリストの情報と、プレイアイテムの情報とで和演算がなされ、あるプレイアイテム再生中でのユーザ操作の制限が決まることになる。

フィールドＰＩＤ＿ｆｉｌｔｅｒ（）は、詳細は省略するが、このプレイアイテムで再生するストリームの優先順位を決めるテーブルである。

フィールドＰＩ＿ｒａｎｄｏｍ＿ａｃｃｅｓｓ＿ｍｏｄｅは、第５８図に一例が示されるような値を取り、このプレイアイテム中の任意の箇所に飛び込み再生をするランダムアクセスが可能か否かを示す。例えば、必ずユーザに見せたいプレイリストがある場合に、フィールドＰＩ＿ｒａｎｄｏｍ＿ａｃｃｅｓｓ＿ｍｏｄｅの値を〔０ｘ１〕に設定しておく。これにより、このプレイアイテムを再生開始する際に、早送り、早戻し、任意時刻か

らの再生などを不可能にできる。
フィールドｓｔｉｌｌ＿ｍｏｄｅは、このプレイアイテム再生後に一時停止を行うかを決めるフィールドである。フィールドｓｔｉｌｌ＿ｍｏｄｅは、第５９図に一例が示されるような値を取り、例えばフィールドｓｔｉｌｌ＿ｍｏｄｅの値が〔０ｘ１〕のときは一時停止することを表し、次のフィールドｓｔｉｌｌ＿ｔｉｍｅのフィールドで指定された時間、一時停止を行う。これにより、スライドショーなどのような、静止画像を一定間隔で次々と表示するような再生を実現することができる。この場合には、静止画１枚ずつがそれぞれプレイアイテムとなっている。フィールドｓｔｉｌｌ＿ｔｉｍｅは、有限の時間だけでなく、ユーザが入力するまで一時停止し続けるという設定（ポーズ設定）も可能となっている。例えばフィールドｓｔｉｌｌ＿ｍｏｄｅの値を〔０ｘ２〕とすることで、ポーズ設定が可能とされる。
上述したフラグｉｓ＿ｍｕｌｔｉ＿ａｎｇｌｅの値が例えば「１」とされていれば、このプレイアイテムがマルチアングルであるとされ、「Ａｎｇｌｅ」以下に、マルチアングルのための情報が追加される。
フィールドｎｕｍｂｅｒ＿ｏｆ＿ａｎｇｌｅｓは、アングル数を表す。フィールドｉｓ＿ｓｅａｍｌｅｓｓ＿ａｎｇｌｅ＿ｃｈａｎｇｅは、第６０図に一例が示されるような値を取り、アングルがシームレスに切り替え可能なように、すなわち、アングル間をなめらかに切り替えることができる状態に各アングルがディスク上配置されているか否かを表す。
次のｆｏｒループは、アングルを構成するクリップの情報が記述される。ｆｏｒループ中のフィールドＣｌｉｐ＿Ｉｎｆｏｒｍａｔｉｏｎ＿ｆｉｌｅ＿ｎａｍｅは、このプレイアイテムが参照しているクリップと１対１に対応するクリップ情報ファイル（拡張子が「ｃｌｐｉ」であるファイル）のファイル名が文字列で示される。フィールドｒｅｆ＿ｔｏ＿ＳＴＣ＿ｉｄは、このプレイアイテムにより参照されるクリップ内のシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅを指定する。
値ａｎｇｌｅ＿ｉｄ＝０に相当するアングルは、アングルを構成しない通常のプレイアイテムと同じように、このブロックＰｌａｙＩｔｅｍ（）の前半部分で既に定義されている。値ａｎｇｌｅ＿ｉｄ＝１以降のアングルがこのｆｏｒループの中で定義されている。このｆｏｒループの中には、値ａｎｇｌｅ＝０に相当するアングルは、含まれていない。
第６１図は、ブロックＳｕｂＰｌａｙＩｔｅｍ（）の一例の構造を表すシンタクスを示す。フィールドｌｅｎｇｔｈは、この直後のフィールドからＳｕｂＰｌａｙＩｔｅｍ（）の終わりまでの長さをバイトで表したものである。フィールドＣｌｉｐ＿Ｉｎｆｏｒｍａｔｉｏｎ＿ｆｉｌｅ＿ｎａｍｅは、このサブプレイアイテムが参照しているクリップと１対１に対応するクリップ情報ファイル（拡張子が「ｃｌｐｉ」であるファイル）のファイル名が文字列で示される。
フィールドＣｌｉｐ＿ｃｏｄｅｃ＿ｉｄｅｎｔｉｆｉｅｒは、このサブプレイアイテムにより参照されるクリップの符号化方式を示す。この実施の一形態では、フィールドＣｌｉｐ＿ｃｏｄｅｃ＿Ｉｄｅｎｔｉｆｉｅｒは、値「Ｍ２ＴＳ」に固定的とされる。
フィールドｉｓ＿ｒｅｐｅａｔ＿ｆｌａｇは、第６２図に一例が示されるような値を取り、このサブプレイアイテムをメインのプレイアイテム（メインパス）と非同期に繰り返し再生するか否かを表すフラグである。例えば、このフィールドｉｓ＿ｒｅｐｅａｔ＿ｆｌａｇの値が「１」の場合は、メインのプレイアイテムの再生が終了するまで、メインのプレイアイテムとは非同期に、このサブプレイアイテムが繰り返し再生される。このフィールドｉｓ＿ｒｅｐｅａｔ＿ｆｌａｇの値が「０」の場合は、このサブプレイアイテムは、メインのプレイアイテムと同期して、一度だけ再生される。
例えば、このサブプレイアイテムがオーディオのみのサブプレイアイテムである場合、フィールドｉｓ＿ｒｅｐｅａｔ＿ｆｌａｇの値を「１」とすることで、ＢＧＭ（Ｂａｃｋ Ｇｒｏｕｎｄ Ｍｕｓｉｃ）再生を行うことができる。
フィールドＳｕｂＰｌａｙＩｔｅｍ＿ｔｙｐｅは、このサブプレイアイテムがどのような性質を持ったサブプレイアイテムであるかを表す。例えば、フィールドＳｕｂＰｌａｙ

Ｉｔｅｍ＿ｔｙｐｅの値が「１」であれば、オーディオのみのサブプレイアイテムであることを表す。

フィールドｒｅｆ＿ｔｏ＿ＳＴＣ＿ｉｄは、このプレイアイテムにより参照されるクリップ内のシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅを指定する。フィールドＳｕｂＰｌａｙＩｔｅｍ＿ＩＮ＿ｔｉｍｅおよびＳｕｂＰｌａｙＩｔｅｍ＿ＯＵＴ＿Ｔｉｍｅは、このサブプレイアイテムにおける開始点および終了点の、シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅ上でのタイムスタンプｐｔｓ（ｐｒｅｓｅｎｔａｔｉｏｎ＿ｔｉｍｅ＿ｓｔａｍｐ）をそれぞれ示す。

上述のフィールドｉｓ＿ｒｅｐｅａｔ＿ｆｌａｇの値が例えば「０」であって、このサブプレイアイテムがメインのプレイアイテムと同期して再生されることを示す場合、フィールドｓｙｎｃ＿ＰｌａｙＩｔｅｍ＿ｉｄとフィールドｓｙｎｃ＿ｓｔａｒｔ＿ＰＴＳ＿ｏｆ＿ＰｌａｙＩｔｅｍにより、サブプレイアイテムがメインのプレイアイテムのどの時刻から同期再生されるかが指定される。

第６３図に一例が示されるように、フィールドｓｙｎｃ＿ＰｌａｙＩｔｅｍ＿ｉｄで、メインパスのプレイアイテムを指定し（ＰｌａｙＩｔｅｍ＝１）、フィールドｓｙｎｃ＿ｓｔａｒｔ＿ＰＴＳ＿ｏｆ＿ＰｌａｙＩｔｅｍで、サブプレイアイテムが再生開始されるメインのプレイアイテム上の時刻を表す（ｔ１）。サブプレイアイテムとして、対応するクリップがフィールドＳｕｂＰｌａｙＩｔｅｍ＿ＩＮ＿ｔｉｍｅおよびフィールドＳｕｂＰｌａｙＩｔｅｍ＿ＯＵＴ＿ｔｉｍｅで指定される期間、再生されることが示されている。

第６４図は、ファイル「ｚｚｚｚｚ．ｃｌｐｉ」の一例の構造を表すシンタクスを示す。第５３図において、ファイル「ｚｚｚｚｚ．ｃｌｐｉ」の内部は、機能別の情報毎にブロックが構成される。フィールドｔｙｐｅ＿ｉｎｄｉｃａｔｏｒにこのファイルを示す文字列が格納され、フィールドｖｅｒｓｉｏｎ＿ｎｕｍｂｅｒにこのファイルのバージョンが示される。フィールドＳｅｑｕｅｎｃｅＩｎｆｏ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓ、フィールドＰｒｏｇｒａｍＩｎｆｏ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓ、フィールドＣＰＩ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓおよびフィールドＣｌｉｐＭａｒｋ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓで、それぞれ対応するブロックの開始位置が示される。

第６５図は、ブロックＣｌｉｐＩｎｆｏ（）の一例の構造を表すシンタクスを示す。フィールドｌｅｎｇｔｈは、この直後のフィールドからブロックＣｌｉｐＩｎｆｏ（）の終わりまでの長さをバイトで表す。

フィールドａｐｐｌｉｃａｔｉｏｎ＿ｔｙｐｅは、クリップＡＶストリーム（拡張子が「ｍ２ｔｓ」のファイル）がどのような多重化によって作られているかを示す。フィールドａｐｐｌｉｃａｔｉｏｎ＿ｔｙｐｅは、第６６図に一例が示されるような値を取り、このクリップＡＶストリームが通常のビデオストリームであるか、静止画の表示に適した多重化がなされているストリームであるかを表す。

より具体的には、この例によれば、フィールドａｐｐｌｉｃａｔｉｏｎ＿ｔｙｐｅの値が「１」で、対応するクリップＡＶストリームのファイルがこの実施の一形態によるＢＤＭＶトランスポートストリームのルールに従っていることを示す。当該クリップＡＶストリームは、通常の動画が再生される。

フィールドａｐｐｌｉｃａｔｉｏｎ＿ｔｙｐｅの値が「２」で、対応するクリップＡＶストリームのファイルがオーディオ再生に同期する静止画用の、ＢＤＭＶトランスポートストリームのルールに従っていることを示す。当該クリップＡＶストリームは、例えばＭＰＥＧ２形式のファイルであって、ビデオデータおよびオーディオデータがマルチプレクスされている。ビデオデータは、例えばＭＰＥＧ２におけるＩピクチャが静止画として並んでいるような構成とされる。これにより、例えばオーディオの時間軸に同期して表示されるスライドショーのような再生が可能とされる。このような再生を、タイムベーススライドショーと称する。

フィールドａｐｐｌｉｃａｔｉｏｎ＿ｔｙｐｅの値が「３」で、対応するクリップＡＶストリームのファイルがオーディオとは非同期に再生される静止画用の、ＢＤＭＶトラン

スポートストリームのルールに従っていることを示す。オーディオデータとビデオデータとは、別ファイルとされ、例えば、オーディオデータが再生されている間、ビデオデータは、静止画が任意の間隔またはユーザの指定に基づき切り換えられて表示される。ビデオデータは、例えばＭＰＥＧ２におけるＩピクチャが静止画として並んでいるような構成とすることができる。このような再生を、ブラウザブルスライドショーと称する。

フィールドａｐｐｌｉｃａｔｉｏｎ＿ｔｙｐｅの値が「０」のときは、対応するクリップＡＶストリームがＢＤＭＶトランスポートストリームのルールに従っていない場合である。

静止画の表示に適した多重化とは、主として静止画スライドショーのようなアプリケーションの実現を容易にすることを想定している。このようなアプリケーションにおいては、静止画１枚と、その上に重ねる字幕やグラフィクスデータとをカプセル化して多重化すると、読み込みが容易となる。

通常の動画像と同様の多重化をしてしまうと、ある静止画と同時に表示されるべき字幕が、前の静止画の画像データ付近に多重化されているような状況（いわゆる多重化位相差）が発生し、より広範囲のストリームデータを読みださないと字幕とグラフィクスとが重ねられた静止画像１枚を表示することができない。

この発明の実施の一形態においては、ビデオデータや字幕を表示するためのグラフィクスデータは、ＭＰＥＧ－２システム規格におけるＴＳ（Ｔｒａｎｓｐｏｒｔ　Ｓｔｒｅａｍ）パケットに格納されて伝送される。１つのＴＳパケットは、１８８バイトからなり、上述のビデオデータやグラフィクスデータは、それぞれＴＳパケットに収まるように分割されてＴＳパケットに格納される。ある静止画像データ（画像Ｐ１とする）に対応した字幕データのパケットが、次以降の静止画像データ（画像Ｐ２とする）のパケットの後ろに配置されてしまうと、画像Ｐ１に対応した字幕を表示するために、画像Ｐ２の後ろまでのデータを読み込んでおかなければならない。

ある静止画一枚とそれに付随する字幕およびグラフィクスだけを対象として多重化を行えば（カプセル化）、他のデータが混入しないストリームを作ることができる。それを静止画ごとに繰り返してストリームをつなげていくと、静止画（および当該静止画に付随する字幕、グラフィクスデータ）毎のデータが直列につながった一本のストリームが出来る。このような多重化を行ったストリームを、静止画用のＢＤＭＶストリームとしている。

静止画用のＢＤＭＶストリームには、タイムベーススライドショーと、ブラウザブルスライドショーとの２種類がある。この実施の一形態では、この２種類を、フィールドａｐｐｌｉｃａｔｉｏｎ＿ｔｙｐｅで別の番号を割り当てて、区別している。

静止画とそれに付随する字幕やグラフィクスとをカプセル化して記録することで、静止画を切り替えながら再生する際のアクセス性が向上する。

第６５図の説明に戻り、フィールドＣｌｉｐ＿ｓｔｒｅａｍ＿ｔｙｐｅは、クリップＡＶストリームの種別を表す。再生専用の規格においては、通常のクリップであることを示す値、例えば値「１」に固定的としてよい。フイールドＴＳ＿ｒｅｃｏｒｄｉｎｇ＿ｒａｔｅは、クリップＡＶストリームファイルの記録レートをバイト／秒で表したものである。フィールドｎｕｍ＿ｏｆ＿ｓｏｕｒｃｅ＿ｐａｃｋｅｔｓは、クリップＡＶストリームに含まれるパケット数を表す。フィールドＢＤ＿ｓｙｓｔｅｍ＿ｕｓｅおよびブロックＴＳ＿ｔｙｐｅ＿ｉｎｆｏ＿ｂｌｏｃｋ（）は、この発明と関連が低いので説明を省略する。

第６７図は、ブロックＳｅｑｕｅｎｃｅＩｎｆｏ（）の一例の構造を表すシンタクスを示す。フィールドｌｅｎｇｔｈは、この直後のフィールドからブロックＳｅｑｕｅｎｃｅＩｎｆｏ（）の終わりまでの長さをバイトで表したものである。フィールドｎｕｍ＿ｏｆ＿ＡＴＣ＿ｓｅｑｕｅｎｃｅｓは、連続した時間に記録されたことを表すシーケンスＡＴＣ＿ｓｅｑｕｅｎｃｅの数を示す。再生専用の媒体の場合、シーケンスＡＴＣ＿ｓｅｑｕｅｎｃｅの数は、「１」にすることができるため、詳細は省略する。フィールドＳＰＮ＿ＡＴＣ＿ｓｔａｒｔは、シーケンスＡＴＣ＿ｓｅｑｕｅｎｃｅの開始をパケット番号で表したものであり、シーケンスＡＴＣ＿ｓｅｑｕｅｎｃｅが１つの場合には、クリップＡＶ

ストリームファイルの先頭と一致し、値が「０」となる。

フィールドｎｕｍ＿ｏｆ＿ＳＴＣ＿ｓｅｑｕｅｎｃｅｓは、シーケンスＡＴＣ＿ｓｅｑｕｅｎｃｅ上のシーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅの数を表す。再生専用の媒体の場合、シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅの数は、「１」にすることができるため、詳細は省略する。フィールドｏｆｆｓｅｔ＿ＳＴＣ＿ｉｄは、値が「０」に固定的とされる。フィールドＰＣＲ＿ＰＩＤは、ＭＰＥＧ２　ＴＳにおいて、ＰＣＲ（Ｐｒｏｇｒａｍ　Ｃｌｏｃｋ　Ｒｅｆｅｒｅｎｃｅ）が含まれるＴＳパケットのＰＩＤを表す。フィールドＳＰＮ＿ＳＴＣ＿ｓｔａｒｔは、シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅの開始をパケット番号で表したものであり、シーケンスＳＴＣ＿ｓｅｑｕｅｎｃｅが１つの場合には、クリップＡＶストリームファイルの先頭と一致し、値が「０」となる。フィールドｐｒｅｓｅｎｔａｔｉｏｎ＿ｓｔａｒｔ＿ｔｉｍｅおよびフィールドｐｒｅｓｅｎｔａｔｉｏｎ＿ｅｎｄ＿ｔｉｍｅは、クリップＡＶストリーム中の有効な範囲を表す。フィールドｐｒｅｓｅｎｔａｔｉｏｎ＿ｓｔａｒｔ＿ｔｉｍｅおよびフィールドｐｒｅｓｅｎｔａｔｉｏｎ＿ｅｎｄ＿ｔｉｍｅで示される範囲がプレイアイテムから参照できる範囲となる。

第６８図は、ブロックＰｒｏｇｒａｍＩｎｆｏ（）の一例の構造を表すシンタクスを示す。記録再生媒体用のシンタックス構造を再生専用媒体に適用することができ、新規の構造は無いので詳細な説明は省略する。フィールドｎｕｍ＿ｏｆ＿ｐｒｏｇｒａｍ＿ｓｅｑｕｅｎｃｅｓの値が「１」であること、フィールドｎｕｍ＿ｏｆ＿ｇｒｏｕｐｓの値が「１」であるなどの制限を加えることができる。

第６９図は、ブロックＳｔｒｅａｍＣｏｄｉｎｇＩｎｆｏ（）一例の構造を表すシンタクスを示す。ブロックＰｒｏｇｒａｍＩｎｆｏ（）と同様、記録再生媒体用のシンタックス構造と略同じ構造を有し、ビデオデータに関しては、ビデオデータのフォーマット、フレームレート、アスペクト比などの属性情報、オーディオデータに関しては、サンプリング周波数などの属性情報が記述される。再生専用媒体に適用する際には、この第６９図に示されるように、字幕、オーディオストリームの言語を表すフィールドｌａｎｇｕａｇｅ＿ｃｏｄｅを追加する必要がある。この情報は、プレーヤの言語設定に従って最適な音声・字幕の言語を選択する際に有用である。

第７０図は、ブロックＣＰＩ（）の一例の構造を表すシンタクスを示す。一般に、ＭＰＥＧストリームのような、フレーム間圧縮を行っている符号化ストリームにおいては、デコード開始可能な箇所は、ＧＯＰ（Ｇｒｏｕｐ　Ｏｆ　Ｐｉｃｔｕｒｅ）の先頭など一部の箇所に限定されている。ＣＰＩ（Ｃｈａｒａｃｔｅｒｉｓｔｉｃ　Ｐｏｉｎｔ　Ｉｎｆｏｒｍａｔｉｏｎ）とは、そのデコード可能な開始点の位置の情報を集めたデータベースで、再生時刻とファイル内アドレスとが対応付けられたテーブルになっている。ＣＰＩは、デコード単位の先頭位置を示す情報がテーブル化されている。

このようにデータベースを定めることで、例えば、任意の時刻から再生したい場合、再生時刻を元にＣＰＩを参照することによって再生位置のファイル内アドレスがわかる。このアドレスは、デコード単位の先頭となっているため、プレーヤは、そこからデータを読み出してデコードし、素早く画像を表示することができる。

このＣＰＩに格納される、デコード単位の先頭位置（この例ではＧＯＰの先頭位置）を、ＥＰ（Ｅｎｔｒｙ　Ｐｏｉｎｔ）エントリと称する。

フィールドＣＰＩ＿ｔｙｐｅは、ＣＰＩの種類を表し、第７１図に一例が示されるような値を取る。この発明の場合は、再生専用媒体用のＣＰＩであることを示す値となる。具体的には、値が「８」とされ、ＢＤＭＶ用のＥＰエントリのマップ（ＥＰ＿ｍａｐ＿ｔｙｐｅ＿ｆｏｒ＿ＢＤＭＶ）が格納されることが示される。

第７２図は、再生専用媒体用のＥＰエントリのマップＥＰ＿ｍａｐ、すなわち、上述のフィールドＣＰＩ＿ｔｙｐｅ内のブロックＥＰ＿ｍａｐ＿ｆｏｒ＿ＢＤＭＶ（）の一例のデータ構造を表すシンタクスを示す。マップＥＰ＿ｍａｐは、ＧＯＰの先頭位置について、再生時刻とファイル内アドレスを対応付けたテーブルである。第７２図の例では、ＧＯＰの先頭位置について、ＭＰＥＧのＰＴＳ（Ｐｒｅｓｅｎｔａｔｉｏｎ　Ｔｉｍｅ　Ｓｔａｍｐ）とＳＰＮ（Ｓｏｕｒｃｅ　Ｐａｃｋｅｔ　Ｎｕｍｂｅｒ）とが対応付けられたテ

ーブルとしてデータベースが作られている。ＳＰＮは、ソースパケット番号を示し、ファイルの先頭からのバイトアドレスに相当する。

記録再生用のマップＥＰ＿ｍａｐの構造と、この再生専用媒体用のマップＥＰ＿ｍａｐの構造は、略同一とされている。この実施の一形態では、データ量の削減および検索の高速化のために、各値をｃｏａｒｓｅとｆｉｎｅに分割し、大まかな単位での検索（ｃｏａｒｓｅ）と、より精密な単位での検索（ｆｉｎｅ）とを分けて行うことが可能なようにされている。そのため、内部構造がｃｏａｒｓｅおよびｆｉｎｅのそれぞれに対応した二つのｆｏｒループに分かれており、「ＧＯＰの最初のＩピクチャのＰＴＳ対ファイル内アドレス」という単純なテーブルと比べて、多少複雑になっている。

フィールドＥＰ＿ｆｉｎｅ＿ｔａｂｌｅ＿ｓｔａｒｔ＿ａｄｄｒｅｓｓは、より精密な検索を行うためのテーブルの位置が示される。次のｆｏｒループは、フィールドＰＴＳ＿ＥＰ＿ｃｏａｒｓｅおよびＳＰＮ＿ＥＰ＿ｃｏａｒｓｅにより、大まかな単位で検索を行う（ｃｏａｒｓｅ）ためのテーブルが格納され、フィールドｒｅｆ＿ｔｏ＿ＥＰ＿ｆｉｎｅ＿ｉｄにより、大まかな単位から参照される精密な検索（ｆｉｎｅ）のためのテーブルの番号が記述される。フィールドＰＴＳ＿ＥＰ＿ｃｏａｒｓｅおよびＳＰＮ＿ＥＰ＿ｃｏａｒｓｅは、それぞれＰＴＳおよびＳＰＮの上位側のビットが示される。

次に、パディングワードを挟んで、精密な検索を行うためのフィールドＰＴＳ＿ＥＰ＿ｆｉｎｅおよびＳＰＮ＿ＥＰ＿ｆｉｎｅが格納されるｆｏｒループが配される。それと共に、フラグｉｓ＿ａｎｇｌｅ＿ｃｈａｎｇｅ＿ｐｏｉｎｔおよびフィールドＩ＿ｅｎｄ＿ｐｏｓｉｔｉｏｎ＿ｏｆｆｓｅｔが格納される。フラグｉｓ＿ａｎｇｌｅ＿ｃｈａｎｇｅ＿ｐｏｉｎｔは、クリップＡＶストリームがマルチアングルを構成している場合に、各ＥＰエントリがアングル切り替え可能点に該当しているか否かを示す。

２－９．デコーダモデル

第７３図Ａ、第７３図Ｂおよび第７３図Ｃは、この発明の実施の一形態に適用できるプレーヤデコーダ１００の一例の構成を示す機能ブロック図である。このプレーヤデコーダ１００は、図示されないドライブ装置に装填されたディスクから再生されたデータを解釈し、ＡＶストリームを出力すると共に、出力されたＡＶストリームに対するユーザによるインタラクティブな操作を可能とする。

プレーヤデコーダ１００は、図示されないＣＰＵ（Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）により全体の動作が制御される。例えば、プレーヤデコーダ１００の各部におけるストリームやデータの流れは、ＣＰＵにより監視され、制御される。

以下の説明では、特に記載がない限り、独自コマンドを用いて記述されたシナリオがプレーヤデコーダ１００で実行されるものとする。

図示されないドライブ装置にディスクが装填されると、上述したように、先ずファイル「ｓｃｅｎａｒｉｏ．ｈｄｍｖ」およびファイル「ｅｎｔｒｙｌｉｓｔ．ｄａｔａ」が再生され、このファイル「ｓｃｅｎａｒｉｏ．ｈｄｍｖ」およびファイル「ｅｎｔｒｙｌｉｓｔ．ｄａｔａ」の記述に基づき、必要な他のファイルが読み出され、ディスクに記録されたコンテンツが再生される。例えば、ファイル「ｓｃｅｎａｒｉｏ．ｈｄｍｖ」およびファイル「ｅｎｔｒｙｌｉｓｔ．ｄａｔａ」の記述に基づき、動画プレーン１０に表示するための動画データ、字幕プレーン１１やグラフィクスプレーン１２に表示するための画像データ、プレイリストファイルなどがディスクから読み出される。

以下では、ディスクから読み出されるこれらのデータのうち、動画データ、サブピクチャ（字幕データ）や音声データといった、連続的に処理する必要があるストリームをリアルタイムストリームと称する。そして、シナリオファイル、プレイリストファイルといった、連続的な処理を要求されない非リアルタイムなデータを、ストアオブジェクトと称する。ストアオブジェクトは、メモリ上などに蓄積、展開され、必要に応じて処理される。

プレーヤデコーダ１００は、チャンネル（１）および（２）の２系統の入力チャンネルを有し、入力チャンネル（１）の入力端１０１に、ストアオブジェクトが入力される。入力チャンネル（２）の入力端２０２に、リアルタイムストリームが入力される。入力端２０２に、ストアオブジェクトを入力することも可能である。この実施の一形態では、入力

端２０２に入力されるリアルタイムストリームおよび一部のストアオブジェクトは、例えばＭＰＥＧ２　ＴＳである。

入力端１０１に入力されるストアオブジェクトは、ディスクから読み出されたデータに限られない。例えば、プレーヤデコーダ１００に対してインターネットなどのネットワーク接続機能を持たせ、ネットワークを介して取得されたストアオブジェクトを入力端１０１に対して入力するようにしてもよい。ボタン画像を表示させるための画像データや新たなシナリオデータなどを、ネットワークを介して取得し、入力端１０１から入力することが考えられる。これに限らず、字幕データなどリアルタイムストリームとして扱われるデータをネットワークを介して取得し、入力端１０１から入力することも可能である。

入力端２０２に入力されるリアルタイムストリームは、ＭＰＥＧ２　ＴＳに限られない。パケット単位で伝送され、ビデオデータ、オーディオデータ、静止画像データなどを多重化可能であれば、他の形式のストリームを入力するようにしてもよい。このときには、後述するＰＩＤフィルタ１１０は、そのストリーム形式に適合したデマルチプレクサとして用いられ、ビデオデータ、オーディオデータ、静止画像データなどを分離する。

ドライブ装置においてディスクの回転速度を２倍速などの高速回転としてディスクからの読み出し転送レートを上げ、時分割で動作させることにより、ディスクからの、チャンネル（１）および（２）の２系統の読み出しが実現可能である。

入力チャンネル（１）の系統について説明する。入力端１０１に入力されたストアオブジェクトは、スイッチ回路１０２に入力される。ストアオブジェクトとしてＥＣＭＡスクリプトやＨＴＭＬファイルなどによるプログラムコードが入力された場合、スイッチ回路１０２において出力端１０２Ａが選択され、入力されたプログラムコードがコードバッファ１０４に蓄えられる。

ストアオブジェクトとして画像データが入力された場合、スイッチ回路１０２において出力端１０２Ｂが選択され、入力された画像データがスイッチ回路１０３に入力される。入力端２０２に入力されたリアルタイムストリームに、字幕プレーン１１やグラフィクスプレーン１２に表示するための画像データが含まれていない場合には、スイッチ回路１０３で入力端１０３Ａが選択され、スイッチ回路１０２から入力された画像データがコンテンツバッファ１０５に蓄えられる。

同様にして、入力端２０２に入力されたリアルタイムストリームに、字幕プレーン１１やグラフィクスプレーン１２に表示するための画像データが含まれている場合には、スイッチ回路１０３において入力端１０３Ｂが選択され、当該画像データがコンテンツバッファ１０５に蓄えられる。コードバッファ１０４およびコンテンツバッファ１０５に蓄えられたストアオブジェクトは、必要に応じて読み出され、マルチメディアエンジン１０６に供給される。

コンテンツバッファ１０５に蓄えられたストアオブジェクトのうち画像データは、スイッチ回路１０７および１０８をそれぞれ介して、グラフィクスデコーダＡ１１６およびグラフィクスデコーダＢ１１７にも供給される。

マルチメディアエンジン１０６は、ＸＭＬパーサ１０６Ａ、スクリプトインタプリタ１０６Ｂおよびグラフィクスレンダラ１０６Ｃを含む。マルチメディアエンジン１０６は、独立的なハードウェアで構成してもよいし、上述した図示されないＣＰＵの、所定のプログラムに基づく処理で実現することも可能である。

ＸＭＬパーサ１０６Ａは、ＸＭＬ（Ｅｘｔｅｎｓｉｂｌｅ　Ｍａｒｋｕｐ　Ｌａｎｇｕａｇｅ）文書を解析する機能を有し、ＨＴＭＬ文書の解析も可能である。ＸＭＬパーサ１０６Ａで解釈されたＨＴＭＬ文書は、このプレーヤデコーダ１００で実行可能な形式に変換される。スクリプトインタプリタ１０６Ｂは、ＥＣＭＡスクリプトを解析し、このプレーヤデコーダ１００で実行可能な形式に変換される。グラフィクスレンダラ１０６Ｃは、画像データを、字幕プレーン１１およびグラフィクスプレーン１２に展開可能な形式にデコードする。

マルチメディアエンジン１０６において、バッファ１０９をワークメモリとして、これらＸＭＬパーサ１０６Ａ、スクリプトインタプリタ１０６Ｂおよびグラフィクスレンダラ

１０６Ｃの処理が行われる。例えば、ＸＭＬパーサ１０６Ａおよびスクリプトインタプリタ１０６Ｂにより、バッファ１０９のうちコードバッファ１０９Ａが用いられる。グラフィクスレンダラ１０６Ｃにより、バッファ１０９のうちグラフィクスバッファ１０９Ｄが用いられる。バッファ１０９は、上述のコードバッファ１０９Ａおよびグラフィクスバッファ１０９Ｄの他に、文字列の表示に用いるフォントデータが格納されるフォントバッファ１０９Ｂ、ＸＭＬパーサ１０６ＡでＨＴＭＬ文書を解析した結果を階層化された木構造で保持するためのツリーバッファ１０９Ｃなどが含まれる。

マルチメディアエンジン１０６では、例えば、シナリオ記述言語としてＨＴＭＬとＥＣＭＡスクリプトとを組み合わせて用いている場合コードバッファ１０４に蓄えられたＥＣＭＡスクリプトを読み出し、読み出されたＥＣＭＡスクリプトの記述に基づいて用いられる。さらに必要に応じて、マルチメディアエンジン１０６は、コードバッファ１０４からの他のＥＣＭＡスクリプトやＨＴＭＬ文書の読み出し、コンテンツバッファ１０５からの画像データの読み出しなどを行う。コードバッファ１０４およびコンテンツバッファ１０５に格納されたデータは、当該データが不要になるまで、コードバッファ１０４やコンテンツバッファ１０５に保持しておくことができる。したがって、これらコードバッファ１０４やコンテンツバッファ１０５に格納されたデータは、必要に応じて何度でも読み出して使うことができる。

マルチメディアエンジン１０６では、上述の他にも、入力された複数種類のデータのデマルチプレクス処理、ＪａｖａＶＭ（Ｊａｖａ（登録商標）仮想マシン）機能などが行われる。さらに、マルチメディアエンジン１０６により、ユーザからの、リモートコントロールコマンダやポインティングデバイスなどによる入力が受け取られ、それらの入力に基づく処理が行われる。ユーザ入力は、後述するグラフィクスデコーダＡ１１６、グラフィクスデコーダＢ１１７、オーディオデコーダ１１８、ＭＰＥＧビデオデコーダ１２０およびシステムデコーダ１２１にも供給される。

グラフィクスレンダラ１０６Ｃで処理された画像データは、スイッチ回路１３０および１３１をそれぞれ介して字幕プレーン１３２およびグラフィクスプレーン１３３に供給される。この例では、字幕プレーン１３２およびグラフィクスプレーン１３３に供給される画像データは、ＰＮＧ形式であるものとする。これらの各プレーン１３２、１３３に画像データが供給されるタイミングは、マルチメディアエンジン１０６により制御される。

字幕プレーン１３２およびグラフィクスプレーン１３３は、それぞれ上述した字幕プレーン１１およびグラフィクスプレーン１２に対応する。動画像プレーン１３４は、上述した動画プレーン１０に対応する。字幕プレーン１３２、グラフィクスプレーン１３３および動画像プレーン１３４は、例えばフレームメモリからなる。

マルチメディアエンジン１０６は、後述するプレゼンテーションプロセッサ１５５に対して、動画像プレーン１３４、字幕プレーン１３２、ならびに、グラフィクスプレーン１３３を切り換える制御信号を供給する。同様に、マルチメディアエンジン１０６は、後述するプレゼンテーションプロセッサ１５７に対して、オーディオストリーム出力を制御するような制御信号を供給する。

次に、入力チャンネル（２）の系統について説明する。入力端２０２にＭＰＥＧ２　ＴＳで入力されたリアルタイムストリームは、ＰＩＤフィルタ１１０に供給され、ＭＰＥＧ２　ＴＳのトランスポートパケットに格納されるＰＩＤ（Ｐａｃｋｅｔ　Ｉｄｅｎｔｉｆｉｃａｔｉｏｎ）が抽出され、当該トランスポートパケットに格納されるストリームの属性が検出される。ＰＩＤフィルタ１１０では、このストリーム属性に基づき、入力されたリアルタイムストリームが、トランスポートパケット毎に対応する系統に振り分けられる。

ＰＩＤに基づき、トランスポートパケットがストアオブジェクトに属する画像データが格納されているパケットであるとされれば、当該トランスポートパケットは、バッファＴＢｎ１１１Ａに一旦溜め込まれ、所定のタイミングで読み出されて入力端１０３Ｂが選択されたスイッチ回路１０３に入力され、スイッチ回路１０３を介してコンテンツバッファ１０５に格納される。

ＰＩＤフィルタ１１０において、ＰＩＤに基づき、トランスポートパケットがサブピクチャデータが格納されているパケットであるとされれば、当該トランスポートパケットは、バッファＴＢｎ１１１ＢおよびバッファＢｎ１１２Ｂに一旦溜め込まれ、所定のタイミングで読み出されて入力端１０７Ｂが選択されたスイッチ回路１０７に入力され、スイッチ回路１０７を介してグラフィクスデコーダＡ１１６に供給される。

グラフィクスデコーダＡ１１６では、供給されたトランスポートパケットのヘッダ情報を除去すると共に、当該トランスポートパケットに格納されたサブピクチャデータがデコードされて字幕等を表示するための画像データとされる。この画像データは、所定のタイミングでスイッチ回路１３０の入力端１３０Ｂに入力され、スイッチ回路１３０を介して字幕プレーン１３２に展開される。

ネットワークを介して取得されるなどの方法により、入力端１０１に字幕データが入力された場合、当該字幕データは、スイッチ回路１０２およびスイッチ回路１０３を介してコンテンツバッファ１０５に蓄えられ、さらに、スイッチ回路１０７で入力端１０７Ａが選択されることで、コンテンツバッファ１０５からグラフィクスデコーダＡ１１６に供給される。

ＰＩＤフィルタ１１０において、ＰＩＤに基づき、トランスポートパケットがグラフィクスデータが格納されているパケットであるとされれば、当該トランスポートパケットは、バッファＴＢｎ１１１ＣおよびバッファＢｎ１１２Ｃに一旦溜め込まれ、所定のタイミングで読み出されて入力端１０８Ｂが選択されたスイッチ回路１０８に入力され、スイッチ回路１０８を介してグラフィクスデコーダＢ１１７に供給される。

グラフィクスデコーダＢ１１７では、供給されたトランスポートパケットのヘッダ情報を除去すると共に、当該トランスポートパケットに格納されたグラフィクスデータがデコードされ、グラフィクスデータとされる。この画像データは、所定のタイミングでスイッチ回路１３１の入力端１３１Ｂに入力され、スイッチ回路１３１を介してグラフィクスプレーン１３３に展開される。

グラフィクスデコーダＡ１１６とグラフィクスデコーダＢ１１７には機能的な違いはない。モデル上、独立して動作するグラフィクスデコーダが２系統あることを表している。すなわち、字幕データとグラフィクスデータを独立にデコードできることを想定している。現実の実装においては、高速なグラフィクスデコーダを時分割で使用し、仮想的に２系統のグラフィクスデコーダが存在しているとみなす方法もある。

ＰＩＤフィルタ１１０において、ＰＩＤに基づき、トランスポートパケットがオーディオデータが格納されているパケットであるとされれば、当該トランスポートパケットは、バッファＴＢｎ１１１ＤおよびバッファＢｎ１１２Ｄに一旦溜め込まれ、所定のタイミングで読み出されてオーディオデコーダ１１８に供給される。このトランスポートパケットに格納されるオーディオデータは、例えばＭＰＥＧに準拠した方式で圧縮符号化されている。

オーディオデコーダ１１８は、例えばリニアＰＣＭ（Ｐｕｌｓｅ Ｃｏｄｅ Ｍｏｄｕｌａｔｉｏｎ）オーディオデコーダ１１９も有する。オーディオデコーダ１１８は、入力されたトランスポートストリームのヘッダ情報を除去すると共に、当該トランスポートパケットに格納された圧縮符号化されたオーディオデータをリニアＰＣＭオーディオデータにデコードする。

オーディオデコーダ１１８から出力されたリニアＰＣＭオーディオデータは、オーディオ用のプレゼンテーションプロセッサ１５７に入力され、マルチメディアエンジン１０６の制御に基づき所定の音響効果などが付加されて、出力端１５８に導出される。

ＰＩＤフィルタ１１０において、ＰＩＤに基づき、トランスポートパケットが動画像データが格納されているパケットであるとされれば、当該トランスポートパケットは、バッファＴＢｎ１１１Ｅ、バッファＭＢｎ１１３およびバッファＥＢｎ１１４に一旦溜め込まれ、所定のタイミングで読み出されてＭＰＥＧビデオデコーダ１２０に供給される。このトランスポートパケットに格納される動画像データは、ＭＰＥＧ２方式により圧縮符号化されている。

ＭＰＥＧビデオデコーダ１２０では、供給されたトランスポートパケットのヘッダ情報を除去すると共に、当該トランスポートパケットに格納された、ＭＰＥＧ２方式で圧縮符号化された動画像データをベースバンドの動画像データにデコードする。

ＭＰＥＧデコーダ１２０から出力された動画像データは、スイッチ回路１２４の入力端１２４Ａに入力されると共に、バッファ１２３を介してスイッチ回路１２４の入力端１２４Ｂに入力される。スイッチ回路１２４において、所定のタイミングで入力端１２４Ａおよび１２４Ｂが選択され、出力された動画像データが動画像プレーン１３４に展開される。

ＰＩＤフィルタ１１０において、ＰＩＤに基づき、トランスポートパケットがシステム情報が格納されているパケットであるとされれば、当該トランスポートパケットは、バッファＴＢｎ１１１ＦおよびＢｓｙｓ１１５を介してシステムデコーダ１２１に供給される。システムデコーダ１２１では、供給されたトランスポートパケットのヘッダ情報が除去され、格納されているシステム情報が取り出される。システム情報は、例えば図示されないＣＰＵに渡される。

字幕プレーン１３２上の画像データは、上述のパレット２２に対応するパレット１５０に供給され、２５６色からなるパレットに対してインデックスによる参照がなされ、ＲＧＢデータが出力されると共に、透明度データα１が抜き出される。ＲＧＢデータは上述のＲＧＢ／ＹＣｂＣｒ変換回路２９に対応するＲＧＢ／ＹＣｂＣｒ変換回路１５１によりＹＣｂＣｒデータに変換され、ＹＣｂＣｒデータおよび透明度データα１は、プレゼンテーションプロセッサ１５５に供給される。

グラフィクスプレーン１３３上の画像データは、上述のパレット２６に対応するパレット１５２に供給され、２５６色からなるパレットに対してインデックスによる参照がなされ、ＲＧＢデータが出力されると共に、透明度データα２が抜き出される。ＲＧＢデータは上述のＲＧＢ／ＹＣｂＣｒ変換回路２７に対応するＲＧＢ／ＹＣｂＣｒ変換回路１５３によりＹＣｂＣｒデータに変換され、ＹＣｂＣｒデータおよび透明度データα２は、プレゼンテーションプロセッサ１５５に供給される。

動画像プレーン１３４の出力は、アップ／ダウンコンバータ１５４を介してプレゼンテーションプロセッサ１５５に供給される。

アップ／ダウンコンバータ１５４は、画像の解像度を変換する回路であって、例えば高解像度のＨＤ（Ｈｉｇｈ　Ｄｅｆｉｎｉｔｉｏｎ）画像から通常の解像度を有するＳＤ（Ｓｔａｎｄａｒｄ　Ｄｅｆｉｎｉｔｉｏｎ）画像への変換を行う。

プレゼンテーションプロセッサ１５５は、第２２図を用いて説明した、字幕プレーン１１（字幕プレーン１３２）の画像データによる透明度α１と、グラフィクスプレーン１２（グラフィクスプレーン１３３）による透明度α２とを用いたアルファブレンディング処理を行う。

プレゼンテーションプロセッサ１５５では、動画像プレーン１３４の画像データに対して、字幕プレーン１３２の画像データに設定された透明度α１に基づき、字幕プレーン１３２の画像データが合成される。動画像プレーン１３４および字幕プレーン１３２が合成された画像データに対して、グラフィクスプレーン１３３の画像データに設定された透明度α２に基づき、グラフィクスプレーン１３３の画像データが合成される。この、グラフィクスプレーン１３３の画像データ、字幕プレーン１３２の画像データ（字幕データ）、ならびに、動画像プレーン１３４の画像データが合成された画像データが出力端１５６に導出される。

プレゼンテーションプロセッサ１５５は、画像データに対してリアルタイムでエフェクト処理を行うこともできる。

字幕がサブピクチャデータとしてグラフィクスデコーダ１１６Ａでデコードされ、字幕プレーン１１に供給されるとして説明したが、字幕の表示は、この例に限定されない。例えば、字幕をテキストデータのような文字コードで供給することもできる。文字コードは、フォントバッファ１０９Ｂに記憶されたフォントデータを参照することで、文字列を表示するためのビットマップデータに変換される。

フォントデータは、例えばディスクから再生されてストアオブジェクトとして入力端１０１に入力され、スイッチ回路１０２を介してコードバッファ１０４に溜め込まれ、コードバッファ１０４からマルチメディアエンジン１０６を介してフォントバッファ１０９Ｂに供給される。

字幕を表示するための文字コードは、例えば、ディスクから再生されてストアオブジェクトとして入力端１０１から入力され、スイッチ回路１０２および１０３を介してコンテンツバッファ１０５に溜め込まれる。リアルタイムストリームとして入力端２０２から入力し、ＰＩＤフィルタ１１０およびバッファＴＢｎ１１１Ａを介してスイッチ回路１０３に供給され、コンテンツバッファ１０５に溜め込まれるようにしてもよい。文字コードは、コンテンツバッファ１０５から読み出され、マルチメディアエンジン１０６に供給される。

文字コードの表示タイミングは、プログラムにより制御される。マルチメディアエンジン１０６は、文字コードの表示タイミングで、表示される文字コードに基づきフォントバッファ１０９Ｂを参照し、対応するフォントデータを選択する。例えば、文字コードが〔０ｘ４１〕、〔０ｘ４２〕、〔０ｘ４３〕、・・・（〔０ｘ〕は、続く数値が１６進表記されていることを示す）となっている場合、それぞれ対応する文字「Ａ」、「Ｂ」、「Ｃ」、・・・を表示するためのフォントデータが選択される。そして、フォントデータを基に文字の飾りやグリフ形状を変化させ、プログラムで指定された大きさのビットマップデータを生成する（レンダリングと称する）。

生成されたビットマップデータは、スイッチ回路１３０を介して字幕プレーン１３２に供給される。これは、字幕表示は、動画像プレーン１３４上の動画と同期を取る必要があるためである。

文字コードのレンダリングは、マルチメディアエンジン１０６やシステムのＣＰＵで行うのに限らず、専用のハードウェアを設けて行ってもよい。文字コードが参照するフォントデータは、ディスクから再生されたものに限らず、例えばネットワークを介して取得するようにしてもよいし、プレーヤのハードウェアが有するＲＯＭ（Ｒｅａｄ　Ｏｎｌｙ　Ｍｅｍｏｒｙ）などに予め記憶させておいてもよい。ユーザによりフォントデータの種類を選択させるようにすることも可能である。

字幕を文字コードで供給することで、予め画像データで供給される字幕データに比べ、字幕表示に要するデータ量が非常に少なくて済むメリットがある。

プレーヤデコーダ１００の各部がハードウェアで構成されるように説明したが、これはこの例に限られない。例えば、プレーヤデコーダ１００をソフトウェア上の処理として実現することも可能である。この場合、プレーヤデコーダ１００をコンピュータ装置上で動作させることができる。プレーヤデコーダ１００をハードウェアおよびソフトウェアが混合された構成で実現することもできる。例えば、オーディオデコーダ１１８やＭＰＥＧビデオデコーダ１２０をハードウェアで構成し、その他をソフトウェアで構成することが考えられる。

プレーヤデコーダ１００をソフトウェアのみ、または、ハードウェアおよびソフトウェアの混合により構成し、コンピュータ装置で実行させるためのプログラムは、例えばＣＤ－ＲＯＭ（Ｃｏｍｐａｃｔ　Ｄｉｓｃ－Ｒｅａｄ　Ｏｎｌｙ　Ｍｅｍｏｒｙ）といった記録媒体に記録されて提供される。このＣＤ－ＲＯＭをコンピュータ装置のＣＤ－ＲＯＭドライブに装填し、ＣＤ－ＲＯＭに記録されたプログラムを所定のコンピュータ装置にインストールすることで、上述の処理をコンピュータ装置上で実行可能な状態とすることができる。コンピュータ装置の構成は、極めて周知であるため、説明は省略する。

２－１０．ボタンについて

この発明の実施の一形態によるユーザインターフェイスについて、より詳細に説明する。第７４図は、グラフィクスプレーン１２上に表示されるボタン表示の一例の状態遷移図である。ボタン表示の状態は、画面にボタンが表示されるボタン表示状態と、画面にボタンが表示されていないボタン非表示状態とに大別される。ボタン非表示状態からボタン表示が開始されボタン表示状態に遷移する。ボタン表示状態からボタン表示が終了され、ボ

タン非表示状態に遷移する。ボタン表示状態は、さらに、通常状態、選択状態および実行状態の３種類の状態を有する。ボタン表示は、これら３種類の状態間をそれぞれ互いに遷移することができる。遷移方向を一方向に限定することも可能である。

上述の第２５図を用いて、より具体的に説明する。先ず、ディスク挿入時や、ユーザがリモコンのメニューキーを押したときに、メニュー画面６０が表示される。メニュー画面６０の表示に伴い、ボタン６２Ａ、６２Ｂ、６２Ｃ、６３Ｄ、６４および６５が非表示状態から表示状態へと遷移する。ボタン表示開始時は、ボタン６２Ａ、６２Ｂ、６２Ｃ、６３Ｄ、６４および６５のうち一つが予め選択状態にされていることが多い。例えばボタン６２Ａが予め選択状態とされ、他のボタンが通常状態とされているものとする。

この状態で、ユーザがリモコンの例えば矢印キーを操作すると、通常状態のボタンの一つ（例えばボタン６２Ｂ）が選択状態に遷移すると共に、選択状態だったボタン６２Ａが通常状態へと遷移する。このようにして、ユーザ操作によってカーソルが移動していく。ユーザがリモコンの決定キーを操作すると、そのときに選択状態にあったボタン６２Ｂが選択状態から実行状態へと遷移し、当該ボタン６２Ｂに予め割り当てられているプレーヤ動作が実行される。

上述したように、プレーヤ動作は、独自コマンドによるプログラミング言語やＥＣＭＡスクリプトなどのスクリプト言語により記述され、ディスク上に記録され用意されている。このプレーヤ動作に関する記述は、独立したファイルとしてディスク上に記録してもよいし、後述するグラフィックオブジェクトのように、クリップＡＶストリームファイルに多重化することも可能である。プレーヤ動作に関する記述をネットワークを介してプレーヤ内部のメモリやストレージ装置にダウンロードすることも考えられる。

このようなメニュー画面を構成するボタンなどの画像データおよび当該画像データに付随する制御情報のデータ構造について説明する。ディスクに収録されるコンテンツ本体をなす動画以外に表示される映像要素として、字幕およびグラフィクス（静止画）を考える。画面に表示される字幕やグラフィクスといった要素をオブジェクトとして考え、オブジェクトの種類を、第７５図に示されるように、字幕、同期型グラフィクスおよび非同期型グラフィクスの３種類に分類する。

字幕とは、例えば映画の字幕のように、動画と同期して表示／非表示の切り換えが行われ、リモコンなどによるユーザ入力とは無関係な映像要素を指す。グラフィクスとは、メニュー画面上のボタンなどのように、ユーザ入力を受け付けることができる映像要素とする。グラフィクスは、同期型グラフィクスおよび非同期型グラフィクスの２種類に分類される。同期型グラフィクスとは、例えば本編再生中のあるタイミングに表示される分岐選択画面のように、動画と同期して表示される映像要素とする。非同期型グラフィクスとは、ディスク挿入時に最初に表示されるメニュー画面や、ユーザ入力に応じて表示される画面など、本編の再生とは非同期に表示される映像要素を指す。ＪａｖａＶＭ上で動作するＪａｖａアプリケーションにより表示される映像要素や、ブラウザソフトウェア上でＨＴＭＬファイルの記述に基づき表示される映像要素は、非同期グラフィクスである。

各映像要素と動画プレーン１０に表示される本編映像との関係からいえば、字幕および同期型グラフィクスは、本編映像に同期して表示されるため、共に同期型である。一方、非同期型グラフィクスは、上述のように本編映像と非同期で表示されるので、その名の示す通り、非同期型である。

字幕とグラフィクスとは、表示先のプレーンの違いから分類することができる。字幕は、字幕プレーン１１に表示される。同期型および非同期型グラフィクスは、グラフィクスプレーン１２に表示される。

字幕および同期型グラフィクスは、本編の動画再生中に表示するという共通の性質を持つことから、共通のデータ構造とすることが好ましい。以下では、これら共通のデータ構造とされる字幕および同期型グラフィクスを、グラフィクスオブジェクトと称する。グラフィクスオブジェクトは、常に動画再生と同期して表示されるため、動画と同一のストリームに多重化しておくと、扱いが容易になる。

第７６図Ａ、第７６図Ｂおよび第７６図Ｃは、この発明の実施形態におけるグラフィク

スオブジェクト２００のデータ構造の例を示す。グラフィクスオブジェクト２００は、第７６図Ａに一例が示されるように、グラフィクスオブジェクトヘッダ２０１、表示制御命令テーブル２０２およびＰＮＧデータ領域２０３、音声データ領域２０４の４つの領域からなる。

以下の例では、グラフィクスオブジェクト２００で扱われる画像データの形式をＰＮＧ形式とし、画像データをＰＮＧ画像データとしているが、ＪＰＥＧ形式なビットマップデータ、ランレングス圧縮された画像データ、圧縮符号化がなされていないビットマップデータなど、他の形式の画像データをグラフィクスオブジェクト２００で扱うことも可能である。便宜上、画像データをＰＮＧ画像、ＰＮＧ画像データなどと表現する。

第７６図Ａ、第７６図Ｂおよび第７６図Ｃにおいて、グラフィクスオブジェクトヘッダ２０１は、当該グラフィクスオブジェクト２００の属性を表す情報が格納される。グラフィクスオブジェクト２００の属性は、例えば、当該グラフィクスオブジェクト２００のデータサイズ、当該グラフィクスオブジェクト２００が有するＰＮＧ画像数、当該グラフィクスオブジェクト２００が有するＰＮＧ画像データが共通に使用するパレットデータ、ならびに、当該グラフィクスオブジェクト２００を識別する識別情報からなる。識別情報は、例えばグラフィクスオブジェクト２００毎に一意に割り当てられた番号である。グラフィクスオブジェクトヘッダ２０１に、さらに他の情報を格納してもよい。

表示制御命令テーブル２０２は、当該グラフィクスオブジェクト２００が有するＰＮＧ画像の表示位置、表示開始時刻および表示終了時刻といった、ＰＮＧ画像の表示を制御するための情報が格納されるテーブルである。

ＰＮＧデータ領域２０３は、ＰＮＧ形式で圧縮符号化された画像データ（以下、ＰＮＧデータと称する）が格納される。ＰＮＧデータ領域２０３には、複数のＰＮＧデータ２０３Ａ、２０３Ｂ、・・・、２０３ｎを格納することができる。ＰＮＧデータ領域２０３に格納されるＰＮＧデータ２０３の数は、グラフィクスオブジェクトヘッダ２０１に記述されている。

ＰＮＧデータ領域２０３に格納される複数のＰＮＧデータ２０３Ａ、２０３Ｂ、・・・、２０３ｎは、アニメーションを構成する複数枚の静止画の組や、上述したボタン表示の３状態を表す画像といった、相互に関連の強い画像であることを想定している。これらを、一つのグラフィクスオブジェクト２００に纏めることで、ＰＮＧ画像の扱いを容易にすることができる。

音声データ領域２０４については、後述する。

グラフィクスオブジェクト２００は、表示が可能になる時刻を示す時刻情報を有する。リアルタイムストリームがＭＰＥＧ２　ＴＳで伝送されるこの例では、この時刻情報として、ＭＰＥＧ２（Ｍｏｖｉｎｇ　Ｐｉｃｔｕｒｅｓ　Ｅｘｐｅｒｔｓ　Ｇｒｏｕｐ　２）により定義された、ｐｔｓ（Ｐｒｅｓｅｎｔａｔｉｏｎ　Ｔｉｍｅ　Ｓｔａｍｐ）を用いる。ｐｔｓは、再生出力の時刻管理情報で、９０ｋＨｚのクロックで計測した値を３３ビット長で表す。ＭＰＥＧシステムの基準復号器内部のＳＴＣ（Ｓｙｓｔｅｍ　Ｔｉｍｅ　Ｃｌｏｃｋ）がｐｔｓに一致したときに、対象となるアクセスユニットを再生出力する。一つのグラフィクスオブジェクト２００は、ｐｔｓで表される時刻から表示が可能になり、その時刻以降に、表示制御命令により実際に表示のＯＮ／ＯＦＦが行われる。表示制御命令により表示の管理を行うため、表示を停止した後に再度、同じグラフィクスオブジェクト２００を表示させるような制御が可能である。

第７６図Ｂは、字幕を格納するグラフィクスオブジェクト２００の例を示す。ＰＮＧデータ領域２０３に、字幕として表示する画像がＰＮＧデータ（１）２０３Ａ－１として格納される。通常、字幕であれば、一つのグラフィクスオブジェクト２００に含まれるデータは、ＰＮＧデータ（１）２０３Ａ－１一つで十分である。表示制御命令テーブル２０２に対して、ＰＮＧデータ（１）Ａ－１の表示属性（透明度、表示色など）を変化させる命令を加えることで、ＰＮＧデータ（１）Ａ－１自体を変更せずに、ＰＮＧデータ（１）Ａ－１の表示を変化させることができる。

例えば、字幕に対して、フェードイン／フェードアウトのような、画像の内容が変化し

ない特殊効果を施すときは、表示制御命令テーブル２０２に対してＰＮＧデータ（１）Ａ－１の透明度を変化させる表示制御命令を加えることで、実現できる。フェードイン／フェードアウトを行う際に、ＰＮＧデータ（１）Ａ－１自体を変更する必要が無い。同様に、表示制御命令テーブル２０２に対して、ＰＮＧデータ（１）Ａ－１が参照するパレットデータを変更する表示制御命令を加えることで、ＰＮＧデータ（１）Ａ－１自体を変更せずに、表示色だけを変更させることができる。

字幕に対して、アニメーションなどのような、画像そのものが変化するような効果を施す場合には、第７６図Ｂに点線で示されるように、アニメーションの各動作に対応する複数のＰＮＧデータ（２）Ｂ－１、ＰＮＧデータ（３）Ｃ－１、ＰＮＧデータ（４）Ｄ－１、・・・を、一つのグラフィクスオブジェクト２００に格納すればよい。これに限らず、日本語字幕、英語字幕など、言語が異なる字幕を表示するＰＮＧデータをＰＮＧデータ（１）Ａ－１、ＰＮＧデータ（２）Ｂ－１、・・・として、一つのグラフィクスオブジェクト２００に格納することもできる。

第７６図Ｃは、ボタンを構成するグラフィクスオブジェクト２００の例を示す。ボタンは、上述したように、通常状態、選択状態および実行状態の３種類の状態を有し、これら３種類の状態のそれぞれでボタン表示を異ならせることができる。３種類の状態でボタン表示を異ならせる場合、３枚のボタン画像データを用意しておく必要がある。これら３枚のボタン画像データを、ひとまとまりのグラフィクスオブジェクト２００として扱う。グラフィクスオブジェクト２００のＰＮＧデータ領域２０３に、通常状態、選択状態および実行状態それぞれのボタン表示に用いるＰＮＧデータ２０３Ａ－２、２０３Ｂ－２および２０３Ｃ－２を格納する。

第７６図Ｂに実線で示される通常の字幕のように、グラフィクスオブジェクト２００内に一つだけ字幕表示のＰＮＧデータ２０３Ａ－１が存在する場合には、当該グラフィクスオブジェクト２００内の表示制御命令テーブル２０２に格納される表示制御命令は、そのＰＮＧデータ２０３Ａ－１に対するものとなる。第７６図Ｃに一例が示されるような、グラフィクスオブジェクト２００内に、複数のＰＮＧデータ２０３Ａ－２、２０３Ｂ－２および２０３Ｃ－２が存在する場合、表示制御命令テーブル２０２に格納される表示制御命令が、複数のＰＮＧデータ２０３Ａ－２、２０３Ｂ－２および２０３Ｃ－２のうちどのデータに対する表示制御命令であるかを特定する必要がある。

例えば、第７６図Ｃに例示される、ボタンのグラフィクスオブジェクト２００においては、このボタンの表示開始時の初期状態が選択状態と定められている場合は、最初に表示するボタン画像は、ＰＮＧデータ領域２０３の先頭に配置される、通常状態時のＰＮＧデータ２０３Ａ－２ではなく、選択状態時のＰＮＧデータ２０３Ｂ－２としなければならない。この発明の実施の第１の形態では、このような表示制御は、グラフィクスオブジェクト２００の外で行うようにしている。

例えば、各ボタンの初期状態や、表示開始および表示停止、ボタンが実行状態に遷移された際に実行されるプログラムなどは、グラフィクスオブジェクト２００外部のスクリプトプログラム、例えば上述のＥＣＭＡスクリプトやＪａｖａＳｃｒｉｐｔによって指定する方法が考えられる。また、ボタンの表示中における、ボタン表示を行うＰＮＧデータの変更は、例えばユーザがリモコンの矢印キーを操作し、カーソルを移動させた際に発生する。この場合には、プレーヤがユーザ入力に応じて各ボタンのＰＮＧデータを入れ替えることになる。

この発明の実施の一形態では、グラフィクスオブジェクト２００は、ＭＰＥＧ２に規定されるパケットに分割され、クリップＡＶストリームに多重化され、クリップＡＶストリームファイルとしてディスクに記録される。第７７図に一例が示されるように、グラフィクスオブジェクト２００は、分割され、ＭＰＥＧ２に規定されるＰＥＳ（Ｐａｃｋｅｔｉｚｅｄ　Ｅｌｅｍｅｎｔａｒｙ　Ｓｔｒｅａｍ）パケット２１０、２１０、・・・に格納される。このとき、グラフィクスオブジェクトヘッダ２０１および表示制御命令テーブル２０２、ならびに、各ＰＮＧデータ２０３Ａ、２０３Ｂ、・・・、２０３ｎそれぞれの先頭が、ＰＥＳパケット２１０、２１０、・・・のペイロードの先頭に合致するように、デ

ータが配置される。こうすることで、再生時に、グラフィクスオブジェクト２００内の各データの検索が容易となる。

こうして分割されてＰＥＳパケット２１０、２１０、・・・に格納されたグラフィクスオブジェクト２００は、さらに、データサイズが１８８バイトに固定されたＴＳパケットに分割され（図示しない）、クリップＡＶストリームなどの動画データや音声データによるストリームに多重化される。

第７８図は、グラフィクスオブジェクト２００のデコードを行うグラフィクスオブジェクトデコーダモデル２４０の一例の構成を示す機能ブロック図である。このグラフィクスオブジェクトデコーダモデル２４０は、第７３図Ａ、第７３図Ｂおよび第７３図Ｃを用いて説明したプレーヤデコーダ１００におけるマルチメディアエンジン１０６およびグラフィクスデコーダＡ１１６を中心に構成される。グラフィクスデコーダＢ１１７の機能は、グラフィクスデコーダＡ１１６と同一であるので、ここでの説明では、グラフィクスデコーダＡ１１６を想定することにする。第７８図において、第７３図Ａ、第７３図Ｂおよび第７３図Ｃと共通する部分には同一の符号を付して詳細な説明を省略する。この第７８図は、より機能面に注目して表現されており、上述の第７３図Ａ、第７３図Ｂおよび第７３図Ｃとは異なる表現がなされている場合があるので、必要に応じて第７３図Ａ、第７３図Ｂおよび第７３図Ｃの構成と対応付けながら説明する。

端子２０２からＭＰＥＧ　ＴＳとして入力されたクリップＡＶストリームがＰＩＤフィルタ１１０に供給される。ＰＩＤフィルタ１１０は、ＭＰＥＧ２　ＴＳ（トランスポートストリーム）に対するデマルチプレクサとして機能し、供給されたＭＰＥＧ　ＴＳのＰＩＤに基づき、動画像データ、オーディオデータおよびグラフィクスオブジェクト２００をそれぞれ抜き出す。動画像データは、ビデオバッファとしてのバッファＴＢｎ１１１Ｅに供給される。オーディオデータは、オーディオバッファとしてのバッファ１１１Ｄに供給される。グラフィクスオブジェクト２００は、グラフィクスオブジェクト（第７８図では「ＧＯＢＪ」と表記）の入力バッファであるバッファＴＢｎ１１１Ｂに供給される。

グラフィクスオブジェクト２００は、バッファＴＢｎ１１１Ｂから読み出され、ＧＯＢＪパーサ２２４に供給される。ＧＯＢＪパーサ２２４は、例えば第７３図におけるグラフィクスデコーダＡ１１６の機能の一つである。ＧＯＢＪパーサ２２４では、供給されたグラフィクスオブジェクト２００のグラフィクスオブジェクトヘッダ２０１を読み取り、パレットデータを抽出すると共に、表示制御命令テーブル２０２、ＰＮＧデータ領域２０３および音声データ領域２０４を分離する。パレットデータおよび表示制御命令テーブル２０２は、コマンドプロセッサ／グラフィックレンダラ２２５に供給される。ＰＮＧデータ領域２０３のＰＮＧデータ２０３Ａ、２０３Ｂ、・・・は、ＰＮＧデコーダバッファ２２６に一時的に格納される。ＰＮＧデコーダバッファ２２６は、例えば第７３図Ａ、第７３図Ｂおよび第７３図Ｃにおけるバッファ Ｂｎ１１２Ｂに対応する。

音声データ領域２０４の音声データ２０４Ａ、２０４Ｂ、・・・、２０４ｎはコマンドプロセッサ／グラフィックレンダラ２２５に供給され、図示されないバッファにそれぞれ溜め込まれる。

ＰＮＧデコーダバッファ２２６に格納されたＰＮＧデータ２０３は、グラフィクスデコーダＡ１１６の機能の一つであるＰＮＧデコーダ２２７によりデコードされ、ビットマップデータとされる。このビットマップデータは、オブジェクトバッファ２２８に蓄積される。オブジェクトバッファ２２８は、第７３図Ａ、第７３図Ｂおよび第７３図Ｃにおいては、デコーダ１１６が内部で持つバッファメモリに対応する。ソフトウェアによるデコードにおいては、第７３図Ａ、第７３図Ｂおよび第７３図Ｃにおけるグラフィクスバッファ１０９Ｄに対応する。

コマンドプロセッサ／グラフィックレンダラ２２５は、ＧＯＢＪパーサ２２４から供給された表示制御命令テーブル２０２に記述される表示制御命令に従い、オブジェクトバッファ２２８に蓄積されているビットマップデータを読み出して指定された時刻にプレーンバッファ２２９に転送する。プレーンバッファ２２９は、例えば第７３図Ａ、第７３図Ｂおよび第７３図Ｃにおける字幕プレーン１３２およびグラフィクスプレーン１３３に対応

させることができる。例えば、字幕と、字幕以外のグラフィクスオブジェクトとに対してそれぞれプレーンバッファ２２９Ａ、２２９Ｂ（図示しない）を設けることができる。これに限らず、字幕プレーン１３２およびグラフィクスプレーン１３３を、プレーンバッファ２２９上の異なる領域とすることもできる。

コマンドプロセッサ／グラフィックレンダラ２２５は、ＧＯＢＪパーサ２２４から供給されたパレットデータを、第７３図Ａ、第７３図Ｂおよび第７３図Ｃのパレット１５０に対応する共通パレットテーブル２３０に供給する。コマンドプロセッサ／グラフィックレンダラ２２５は、第７３図Ａ、第７３図Ｂおよび第７３図Ｃのマルチメディアエンジン１０６の一部の機能およびグラフィクスデコーダＡ１１６の一部の機能を併せ持つ。

コマンドプロセッサ／グラフィックレンダラ２２５は、ＧＯＢＪパーサ２２４から供給された表示制御命令テーブル２０２に記述される表示制御命令に従い、バッファから音声データを読み出し、出力する。グラフィクスオブジェクト２００に格納された音声データ２０４Ａ、２０４Ｂ、・・・、２０４ｎが圧縮符号化されている場合には、コマンドプロセッサ／グラフィックレンダラ２２５において復号化されて出力される。

コマンドプロセッサ／グラフィックレンダラ２２５から出力された音声データは、オーディオミキサ２３１に供給され、プレゼンテーションプロセッサ１５７に対して出力される。オーディオミキサ２３１に他の音声データも入力されている場合には、これらの音声データが所定の割合で混合されて出力される。

グラフィクスオブジェクト２００がボタンを構成するものである場合、ボタンの３種類の状態にそれぞれ対応したＰＮＧデータ２０３Ａ、２０３Ｂおよび２０３Ｃがグラフィクスオブジェクト２００に格納される。ＰＮＧデータ２０３Ａ、２０３Ｂおよび２０３Ｃは、ＰＮＧデコーダ２２７でデコードされ、オブジェクトバッファ２２８にそれぞれ蓄積される。

ユーザのリモコンなどによる入力が、コマンドプロセッサ／グラフィックレンダラ２２５により受け取られる。コマンドプロセッサ／グラフィックレンダラ２２５は、このユーザ入力に応じて、オブジェクトバッファ２２８から該当するビットマップを読み出して、プレーンバッファ２２９に転送する。例えば、ユーザ入力があるボタンに対して選択状態から実行状態に状態遷移させるものであれば、オブジェクトバッファ２２８から、実行状態のボタン画像に対応するビットマップデータが選択的に読み出されて、プレーンバッファ２２９に転送される。

コマンドプロセッサ／グラフィックレンダラ２２５は、オブジェクトバッファ２２８から読み出したビットマップデータに対して、表示制御命令に従い部分切り出しなどの特殊効果処理を施すこともできる。

この実施の一形態では、ＰＮＧデータは、１画素のサンプリング深さが８ビットとされているため、プレーンバッファ２２９は、１画素当たり８ビットのデータが並んでいることになる。プレーンバッファ２２９は、ディスプレイ装置などに実際に画像を表示させる処理を行う表示系の走査周期毎に読み出される。プレーンバッファ２２９から読み出されたビットマップデータは、例えば第７３図Ａ、第７３図Ｂおよび第７３図Ｃのパレット１５０に対応できる共通パレットテーブル２３０に供給され、パレットインデックス値から実際のＲＧＢ（４：４：４）の色情報に変換されると共に、透明度データα１、α２が抜き出される。ＲＧＢ（４：４：４）の色情報は、図示されない変換回路でＹＣｂＣｒ（４：４：４）の色情報に変換され、透明度データα１、α２と共に、第７３図Ａ、第７３図Ｂおよび第７３図Ｃのプレゼンテーションプロセッサ１５５に供給される。

例えばフェードイン／フェードアウトなどの、パレットや透明度を変える処理が必要な特殊効果は、コマンドプロセッサ／グラフィックレンダラ２２５が表示制御命令に従い共通パレットテーブル２３０のデータを変化させることで実現される。字幕と、字幕以外のグラフィクスオブジェクト２００に対して、それぞれ共通パレットテーブル２３０Ａ、２３０Ｂ（図示しない）を設けてもよい。

第７９図Ａ、第７９図Ｂ、第７９図Ｃおよび第７９図Ｄは、グラフィクスオブジェクト入力バッファ（バッファＴＢｎ１１１Ｂ、以下ＧＯＢＪ入力バッファ）、ＰＮＧデコーダ

２２７、オブジェクトバッファ２２８およびプレーンバッファ２２９における蓄積データ量の遷移の例を概略的に示す。ＰＮＧデコーダ２２７は、ＰＮＧデコーダ２２７がＰＮＧデータのデコード時に用いるバッファにおける蓄積データ量が示されている。

第７９図Ａ、第７９図Ｂ、第７９図Ｃおよび第７９図Ｄの例では、３つのグラフィクスオブジェクトＧＯＢＪ＃１、ＧＯＢＪ＃２およびＧＯＢＪ＃３によるデータ量の遷移が時間の経過に従い示される。グラフィクスオブジェクトのデコード開始時刻は、ＭＰＥＧ２システムにおけるｄｔｓ（Ｄｅｃｏｄｉｎｇ　Ｔｉｍｅ　Ｓｔａｍｐ）で表される。オブジェクトの有効期間は、開始時刻がｐｔｓで表され、グラフィクスオブジェクトヘッダ２０１内に記述される時刻ｐｒｅｓｅｎｔａｔｉｏｎ＿ｅｎｄで終了される。この有効期間内に、表示制御命令によって画像の表示開始および終了が指示される。

第７９図Ｄを参照し、ＧＯＢＪ入力バッファにグラフィクスオブジェクトＧＯＢＪ＃１のＰＮＧデータが入力され、時刻ｄｔｓ　ｏｆ　ＧＯＢＪ＃１に、このＰＮＧデータのデコードが開始される。第７９図Ｃを参照し、ＰＮＧデータがＧＯＢＪ入力バッファからＰＮＧデコーダ２２７に転送され、ＰＮＧデータがビットマップデータにデコードされる。実際には、ＰＮＧデータは、ＧＯＢＪ入力バッファから一旦ＰＮＧデコーダバッファ２２６に溜め込まれ、ＰＮＧデコーダ２２７は、ＰＮＧデコーダバッファ２２６に溜め込まれたデータを用いてデコード処理を行う。

ＰＮＧデコーダ２２７は、デコード速度に上限があるため、ＧＯＢＪ入力バッファからＰＮＧデコーダバッファ２２６に対して、ＰＮＧデコーダ２２７のデコード速度を超えないような転送速度でデータが供給される。そのため、ＰＮＧデコーダバッファ２２６に対して、ＰＮＧデコーダ２２７への転送時間を０とした概念モデル上の場合の垂直線に対し、ある傾きＡに対応するデータ転送レートで以てＰＮＧデータが入力される。

ＰＮＧデータのＰＮＧデコーダ２２７への入力が完全に終了していなくても、ＰＮＧデータのデコードを開始することが可能である。この第７９図Ａ、第７９図Ｂ、第７９図Ｃおよび第７９図Ｄの例では、ＧＯＢＪ入力バッファに格納されたオブジェクトＧＯＢＪ＃１が全てＰＮＧデコーダ２２７に転送された後、次のオブジェクトＧＯＢＪ＃２のＰＮＧデータの、ＧＯＢＪバッファへの入力が開始されている。

オブジェクトＧＯＢＪ＃２およびオブジェクトＧＯＢＪ＃３についても、ある傾きＢおよびＣにそれぞれ対応する転送レートで以て、ＰＮＧデコーダバッファ２２６に対してＰＮＧデータが入力される。傾きＢは、実際には、複数の区間でそれぞれ異なる傾きとなっている。

時刻ｐｔｓ　ｏｆ　ＧＯＢＪ＃１によりオブジェクトＧＯＢＪ＃１の有効期間が開始されると、デコードされＰＮＧデコーダバッファに蓄積された、オブジェクトＧＯＢＪ＃１のビットマップデータがオブジェクトバッファ２２８に転送される（第７９図Ｂ）。オブジェクトバッファ２２８に転送されたオブジェクトＧＯＢＪ＃１は、このオブジェクトＧＯＢＪ＃１に対して時刻ｐｒｅｓｅｎｔａｔｉｏｎ　ｅｎｄ　ｏｆ　ＧＯＢＪ＃１（ＧＯＢＪ＃１の提示終了）で示される時刻まで、有効期間が持続される。

オブジェクトＧＯＢＪ＃１の有効期間内に、命令Ｄｉｓｐｌａｙ　ＯＮ　ｃｍｄ．ｏｆ　ＧＯＢＪ＃１（ＧＯＢＪ＃１の表示開始指示）によりオブジェクトＧＯＢＪ＃１の表示命令が出されたら、オブジェクトバッファ２２８に蓄積されたオブジェクトＧＯＢＪ＃１のビットマップデータがプレーンバッファ２２９に転送され、表示される（第７９図Ａ）。詳細は後述するが、ビットマップデータをプレーンバッファ２２９へ転送する際の転送レートは、バス幅などの影響により上限を有する。そのため、プレーンバッファ２２９に対するビットマップデータの書き込みは、例えばある傾きＤに対応する転送レートで以て行われる。

他のオブジェクトＧＯＢＪ＃２やオブジェクトＧＯＢＪ＃３についても同様に、ある傾きＥ、Ｆ、Ｇに対応する転送レートで以てビットマップデータが転送され、プレーンバッファ２２９に書き込まれる。

表示は、オブジェクトＧＯＢＪ＃１に対して表示停止を命令するコマンドＤｉｓｐｌａｙ　ＯＦＦ　ｃｍｄ．ｏｆ　ＧＯＢＪ＃１（ＧＯＢＪ＃１の表示終了指示）が出されるま

で、持続される。コマンドＤｉｓｐｌａｙ　ＯＦＦ　ｃｍｄ．ｏｆ　ＧＯＢＪ＃１が出されると、プレーンバッファ２２９に格納された当該オブジェクトＧＯＢＪ＃１のビットマップデータが破棄され、表示が停止される。

ＧＯＢＪバッファには、オブジェクトＧＯＢＪ＃２およびＧＯＢＪ＃３のＰＮＧデータも、順次、入力される。そして、オブジェクトＧＯＢＪ＃１の場合と同様に、時刻ｄｔｓ　ｏｆ　ＧＯＢＪ＃２、時刻ｄｔｓ　ｏｆ　ＧＯＢＪ＃３からそれぞれデコード開始が指示され、ＰＮＧデータがＰＮＧデコーダ２２７に供給され、ＰＮＧデコーダバッファを用いてビットマップデータへのデコードが行われる。そして、オブジェクトＧＯＢＪ＃２に対して時刻ｐｔｓ　ｏｆ　ＧＯＢＪ＃２により有効期間開始が指示され、第７９図では省略されているが、コマンドＤｉｓｐｌａｙ　ＯＮ　ｃｍｄ．ｏｆ　ＧＯＢＪ＃２によりオブジェクトＧＯＢＪ＃２の表示命令が出され、オブジェクトバッファ２２８からプレーンバッファ２２９にビットマップデータが転送され、コマンドＤｉｓｐｌａｙ　ＯＦＦ　ｃｍｄ．ｏｆ　ＧＯＢＪ＃２が出されるまで、オブジェクトＧＯＢＪ＃２が表示される。

この第７９図Ａ、第７９図Ｂ、第７９図Ｃおよび第７９図Ｄの例では、オブジェクトＧＯＢＪ＃２は、一旦、図示されないコマンドＤｉｓｐｌａｙ　ＯＦＦ　ｃｍｄ．ｏｆ　ＧＯＢＪ＃２により表示が停止された後、再びコマンドＤｉｓｐｌａｙ　ＯＮ　ｃｍｄ．ｏｆ　ＧＯＢＪ＃２による表示が再開されている。オブジェクトバッファ２２８に格納されたオブジェクトＧＯＢＪ＃２のビットマップデータは、オブジェクトＧＯＢＪ＃２に対して有効期間終了時刻ｐｒｅｓｅｎｔａｓｉｏｎ　ｅｎｄ　ｏｆ　ＧＯＢＪ＃１を指定するまで保持されるため、コマンドＤｉｓｐｌａｙ　ＯＮ　ｃｍｄ．ｏｆ　ＧＯＢＪ＃２を与えることにより、繰り返しオブジェクトＧＯＢＪ＃２を表示させることができる。

オブジェクトＧＯＢＪ＃３は、オブジェクトＧＯＢＪ＃２の有効期間中に重複して、有効期間が指定されている。この場合、オブジェクトバッファ２２８は、空き容量に応じて、複数のビットマップデータを互いに異なる領域に格納する。例えば、オブジェクトバッファ２２８からオブジェクトＧＯＢＪ＃２のビットマップデータをプレーンバッファ２２９に転送して表示中に、オブジェクトバッファ２２８の異なる領域からオブジェクトＧＯＢＪ＃３のビットマップデータをプレーンバッファ２２９に転送することで、２枚のビットマップデータを同時に表示させることができる。

２－１１．グラフィクスの転送速度について

このグラフィクスオブジェクトデコーダモデル２４０（以下、デコーダモデル２４０と略称する）をプレーヤに実装する場合について考える。同一ディスクを異なるプレーヤで再生した際の、プレーヤ同士の再生互換性を保つためには、デコーダモデル２４０に対して所定の制限を設けることが必要となる場合がある。例えば、デコーダモデル２４０のグラフィクス処理能力には上限があり、その能力を超えるグラフィクスデータが入力された場合には、グラフィクスデータのデコードを完全に行うことができなくなり、正常な表示を行うことができなくなる。

プレーヤ側では、プレーヤが最小限備える必要のあるグラフィクス処理能力を規格により定める。一方、ディスク側では、ディスクに記録するコンテンツを制作する側において、規格により定められたプレーヤの能力の範囲内で処理可能なグラフィクスを用意する。このようにしてプレーヤ側とディスク制作側とでグラフィクス処理能力の整合性を図ることで、再生互換性を保つことができる。

この発明の実施の一形態では、上述の第７８図における、ＧＯＢＪパーサ２２４からＰＮＧデコーダバッファ２２６へのデータ転送速度Ｒ（１）と、コマンドプロセッサ２２５からプレーンバッファ２２９へのデータ転送速度Ｒ（２）とを規定する。

データ転送速度Ｒ（１）は、ＰＮＧデコーダバッファ２２６に入力されるデータの、単位時間当たりのデータ転送量を規定する。すなわち、上述の第７９図Ｃに示される傾きＡ、ＢおよびＣがデータ転送速度Ｒ（１）に対応する。これは、ＰＮＧデコーダバッファ２２６の後に接続されるＰＮＧデコーダ２２７の、圧縮符号化されたグラフィクスデータを単位時間当たりにデコードできる量を示すデコード能力を規定する。したがって、データ転送速度Ｒ（１）を制限することで、入力された圧縮符号化グラフィクスデータに対して

デコードが間に合わなくなり、表示などが破綻状態になる状況が防止される。

データ転送速度Ｒ（２）は、画像の更新速度を規定する。プレーンバッファ２２９は、実際に表示装置に表示される画面に対応している。プレーンバッファ２２９へのデータの書き込み速度によって、ユーザが見るグラフィクスの更新速度が決まる。データ転送速度Ｒ（２）の単位は、［バイト／秒］であって、プレーン全体すなわち全画面を更新する際の最小更新時間間隔を規定する。上述の第７９図Ａに示される傾きＤ、Ｅ、ＦおよびＧがデータ転送速度Ｒ（２）に対応する。

プレーンの一部だけを更新する場合には、更新される画像データが少ないためデータ転送速度Ｒ（２）で規定される最小更新時間間隔よりも短い周期で更新される。更新間隔は、更新される画像データのデータ量に反比例するとは限らず、当該画像データのプレーン上の配置により大きく影響される。

プレーンの更新速度について、第８０図を用いてより具体的に説明する。オブジェクトバッファ２２８には、２個のグラフィクスオブジェクト４６０および４６１が格納されているものとする。これら２個のグラフィクスオブジェクト４６０および４６１を、同時にプレーンバッファ２２９に書き込み表示させることを考える。

グラフィクスオブジェクト４６０および４６１は、オブジェクトバッファ２２８から読み出され、コマンドプロセッサ／グラフィックレンダラ２２５に供給される。このコマンドプロセッサ／グラフィックレンダラ２２５からの出力が上述のデータ転送速度Ｒ（２）により制限され、画面上の更新速度（更新間隔）が制限されることになる。

しかしながら、書き換えデータ量が同一のオブジェクトであっても、画面上の更新速度は、オブジェクトのプレーン上の位置、さらにオブジェクトの変形や移動などによって変化してしまい、見積もることが難しい。第８０図の例では、更新速度は、グラフィクスオブジェクト４６０および４６１の合計のデータ量ではなく、グラフィクスオブジェクト４６０および４６１を、プレーン上にどのように配置するかによって決まる。

そこで、プレーンバッファへの書き換えデータ量について、ウィンドウと称する矩形の更新領域を定義した。これにより、以下に説明するように、最小更新時間間隔を見積もることが可能となり、実装の実現性と再生互換性を高めることができるようになる。定義される領域が矩形であるため、グラフィクス処理を行う一般的なグラフィクスプロセッサに対して容易に適用可能で実現性が高い考え方である。以下、このウィンドウの定義に基づくモデルをウィンドウモデルと称する。

例えば、第８０図においては、プレーン上に配置されるグラフィクスオブジェクト４６０および４６１を全て含む矩形領域４６２で以て、プレーンの更新がなされる。コマンドプロセッサ／グラフィックレンダラ２２５において、グラフィクスオブジェクト４６０および４６１の配置情報に基づき矩形領域４６２の画像データが形成される。この矩形領域４６２の画像データが例えば転送バスを介してプレーンバッファ２２９に供給される。プレーンバッファ２２９では、例えば表示位置の指定に基づき、矩形領域４６２に対応する領域のデータが、新たに供給された矩形領域４６２のデータと置き換えられる。

コマンドプロセッサ／グラフィックレンダラ２２５から出力される画像データは、ビットマップデータであるので、画像の内容によらず、画像の面積に応じたデータ量を有することになる。第８０図の例の場合、２個のグラフィクスオブジェクト４６０および４６１を含む矩形領域４６２の画像データ量は、例えば幅（ｗｉｄｔｈ）および高さ（Ｈｅｉｇｈｔ）をそれぞれ画素数で表すとき〔幅（ｗｉｄｔｈ）×高さ（Ｈｅｉｇｈｔ）〕バイトで表すことができる。この矩形領域４６２をウィンドウと呼ぶことにする。ウィンドウは、１または複数個のグラフィクスオブジェクトを完全に含むと共に、面積が最小となるようにすると、転送データ量が最小になり好ましい。

プレーンバッファ２２９へのデータ転送速度が速度Ｒ（２）〔バイト／秒〕とされていることから、これら２個のグラフィクスオブジェクト４６０および４６１は、｛速度Ｒ（２）／（幅×高さ）｝秒の時間で更新できることが分かる。ある幅および高さのウィンドウがプレーンバッファ２２９に転送されてから少なくとも｛速度Ｒ（２）／（幅×高さ）｝秒後には、次のグラフィクスオブジェクトの描画が可能となる。ディスク制作側では、

次のグラフィクスオブジェクトの描画を、少なくともこの時間以上の間隔を空けて行うようにプログラムすることで、どのプレーヤでも同一のグラフィクス表示がなされ、再生互換性を保つことができる。

以上のように、同時に表示制御される複数オブジェクトを囲む矩形領域をウィンドウと定義し、ウィンドウのデータ量を転送速度Ｒ（２）で割ることにより、ウィンドウの最短更新間隔を見積もることができる。

オブジェクトを消去する際にも、プレーンへの書き込みが必要になる。上述したウィンドウモデルにおいては、ウィンドウ全体を透明色で書き換えればよい。そのときの時間も、ウィンドウの最短更新間隔と同じである。

オブジェクトが変形する、あるいはオブジェクトがプレーン上を移動するグラフィクスを制作したい場合でも、上述したウィンドウモデルは、動作速度を見積もることができる。例えば、第８１図Ａのようにオブジェクトが時間と共に変形する場合は、第８１図Ｂのように、時間により変化するオブジェクトを全て含む矩形領域（x_{min}，y_{min}），（x_{max}，y_{max}）をウィンドウと定義すればよい。例えば、第８２図Ａの例のようにオブジェクトが時間と共に移動する場合は、第８２図Ｂのように、オブジェクトが移動する軌跡を全て含む矩形領域（x_{min}，y_{min}），（x_{max}，y_{max}）をウィンドウと定義すればよい。

ウィンドウは、プレーンに複数個（例えば２個）作ることが許される。このとき、同一プレーン上の複数個のウィンドウは、互いに重なり合わないものとする。

この発明の実施の一形態では、以上のようなウィンドウモデルを定義することによって、表示するオブジェクトの数、形状、大きさ、時間経過による変形、表示位置など、多くのパラメータで決まるために算出が難しかったグラフィクスの表示速度（最小更新間隔）を簡単に導出することができるようになる。そのため、ディスク制作側がグラフィクスの動作速度を予め見積もることができるようになり、プレーヤでの動作の互換性を高めることが可能となる。

後述する動きのある字幕表示についても、上述のようにしてデータ転送速度Ｒ（２）を見積もることで、再生互換性を保つようにアニメーション速度を決めることができる。

２－１２．グラフィクスオブジェクトについて

グラフィクスオブジェクト２００の構造について、より詳細に説明する。第８３図は、グラフィクスオブジェクト２００の一例の構造を表すシンタクスを示す。第７６図Ａなどにおけるグラフィクスオブジェクトヘッダ２０１がブロックＧｒａｐｈｉｃｓＯｂｊｅｃｔＨｅａｄｅｒ（）、表示命令制御テーブル２０２がブロックＧＯＢＪＣｏｍｍａｎｄＴａｂｌｅ（）、ＰＮＧデータ領域２０３がブロックＰＮＧＩｍａｇｅＲｅｇｉｏｎ（）、音声データ領域２０４がブロックＳｏｕｎｄＤａｔａＲｅｇｉｏｎ（）に相当する。

ブロックＧｒａｐｈｉｃｓＯｂｊｅｃｔＨｅａｄｅｒ（）の先頭にフィールドｌｅｎｇｔｈが配置される。フィールドｌｅｎｇｔｈは、データ長が８ビットの０以上の整数で、フィールドｌｅｎｇｔｈの直後からブロックＧｒａｐｈｉｃｓＯｂｊｅｃｔＨｅａｄｅｒ（）の終わりまでの長さをバイトで示す。フィールドｐｒｅｓｅｎｔａｔｉｏｎ＿ｅｎｄ＿ｔｉｍｅ＿ｓｔａｍｐは、データ長が３３ビットの０以上の整数で、このグラフィクスオブジェクト２００の有効期間終了時刻を表す。グラフィクスオブジェクトの有効期間は、ＰＥＳパケットヘッダのｐｔｓからこのフィールドｐｒｅｓｅｎｔａｔｉｏｎ＿ｅｎｄ＿ｔｉｍｅ＿ｓｔａｍｐで示される有効期間終了時刻までである。フィールドｎｕｍｂｅｒ＿ｏｆ＿ＤｉｓｐＣｍｄｓは、データ長が８ビットの０以上の整数で、ブロックＧＯＢＪＣｏｍｍａｎｄＴａｂｌｅ（）に格納されている表示制御命令の数を表す。フィールドｎｕｍｂｅｒ＿ｏｆ＿ＰＮＧ＿ｉｍａｇｅｓは、データ長が８ビットの０以上の整数で、ブロックＰＮＧＩｍａｇｅＲｅｇｉｏｎ（）に格納されているＰＮＧ画像の数を表す。フィールドｎｕｍｂｅｒ＿ｏｆ＿ｓｏｕｎｄ＿ｄａｔａが配置され、ブロックＳｏｕｎｄＤａｔａＲｅｇｉｏｎ（）に格納されている音声データの数がデータ長が８ビットの０以上の整数で示される。

ブロックＧｒａｐｈｉｃｓＯｂｊｅｃｔＨｅａｄｅｒ（）内のブロックＧｌｏｂａｌＰ

ａｌｅｔｔｅＴａｂｌｅ（）は、このグラフィクスオブジェクト２００で共通に使用されるパレットテーブルの情報が格納される。例えばこのブロックＧｌｏｂａｌＰａｌｅｔｔｅＴａｂｌｅ（）に記述されるパレットテーブルの情報が、例えば共通パレットテーブル２３０の内容としてセットされる。フィールドｓｔａｒｔ＿ａｄｄｒｅｓｓ＿ｏｆ＿ＰＮＧ＿ｉｍａｇｅ（ｉ）は、データ長が３２ビットの０以上の整数で、ｉ番目のＰＮＧ画像のデータＰＮＧ＿ｉｍａｇｅ（ｉ）が開始される位置を、このＧｒａｐｈｉｃｓＯｂｊｅｃｔ（）の先頭からの相対バイト数で表現する。

フィールドＰＮＧ＿ｆｉｌｅ＿ｎａｍｅ（ｉ）は、フィールドｓｔａｒｔ＿ａｄｄｒｅｓｓ＿ｏｆ＿ＰＮＧ＿ｉｍａｇｅ（ｉ）から始まるＰＮＧデータのファイル名を表す。ブロックＰＮＧＩｍａｇｅＲｅｇｉｏｎ（）内のフィールドであるフィールドＰＮＧ＿ｉｍａｇｅ（ｉ）の内容は、単体のＰＮＧファイルと同一であり、ブロックＰＮＧＩｍａｇｅＲｅｇｉｏｎ（）は、一つ以上のＰＮＧファイルを連結して作られる。例えば、上述の第７６図Ａでは、ＰＮＧデータ２０３Ａ、２０３Ｂ、・・・、２０３ｎが連結されてブロックＰＮＧＩｍａｇｅＲｅｇｉｏｎ（）が作られる。その際にファイル名が失われないよう、フィールドＰＮＧ＿ｆｉｌｅ＿ｎａｍｅ（ｉ）にファイル名を保持しておくことができる。逆に、ブロックＰＮＧＩｍａｇｅＲｅｇｉｏｎ（）を分解して個別のＰＮＧファイルを復元する際には、各フィールドＰＮＧ＿ｉｍａｇｅ（ｉ）を、ｉが対応するフィールドＰＮＧ＿ｆｉｌｅ＿ｎａｍｅ（ｉ）で示されるファイル名の、独立したファイルとすればよい。

フィールドｓｔａｒｔ＿ａｄｄｒｅｓｓ＿ｏｆ＿ｓｏｕｎｄ＿ｄａｔａ（ｉ）は、データ長が３２ビットの０以上の整数で、ｉ番目の音声データｓｏｕｎｄ＿ｄａｔａ（ｉ）が開始される位置を、このブロックＧｒａｐｈｉｃｓＯｂｊｅｃｔ（）の先頭からの相対バイト数で表現する。

ブロックＧＯＢＪＣｏｍｍａｎｄＴａｂｌｅ（）は、実行時刻が同じ表示制御命令を集めた命令グループＤｉｓｐＣｍｄｓ（ｉ）から構成される。命令グループＤｉｓｐＣｍｄｓ（ｉ）は、必ず実行時刻を指定する命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｔｉｍｅ）から開始して、表示制御命令が並べられる。換言すると、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｔｉｍｅ）から次の命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｔｉｍｅ）の前までが一つの命令グループＤｉｓｐＣｍｄｓ（ｉ）を構成している。

ブロックＰＮＧＩｍａｇｅＲｅｇｉｏｎ（）は、上述したように、ＰＮＧ方式で圧縮符号化された画像一枚分のデータであるフィールドＰＮＧ＿ｉｍａｇｅ（ｉ）の並びで構成される。

ブロックＳｏｕｎｄＤａｔａＲｅｇｉｏｎ（）は、音声データｓｏｕｎｄ＿ｄａｔａ（ｉ）として実際の音声データが格納される。

ブロックＧｒａｐｈｉｃｓＯｂｊｅｃｔＨｅａｄｅｒ（）とブロックＧＯＢＪＣｏｍｍａｎｄＴａｂｌｅ（）との間には、任意の数のｐａｄｄｉｎｇ＿ｗｏｒｄを挿入することができる。同様に、ブロックＧＯＢＪＣｏｍｍａｎｄＴａｂｌｅ（）とブロックＰＮＧＩｍａｇｅＲｅｇｉｏｎ（）の間には任意の数のｐａｄｄｉｎｇ＿ｗｏｒｄを挿入することができる。

第８４図は、上述したブロックＧｌｏｂａｌＰａｌｅｔｔｅＴａｂｌｅ（）の一例の構造を表すシンタクスを示す。フィールドｎｕｍｂｅｒ＿ｏｆ＿ｐａｌｅｔｔｅ＿ｅｎｔｒｉｅｓは、次に羅列されているパレットデータの数を表す。データ長が８ビットのインデックス番号で画像を表現する場合、このフィールドｎｕｍｂｅｒ＿ｏｆ＿ｐａｌｅｔｔｅ＿ｅｎｔｒｉｅｓの値は、最大で２５６となり、２５６色を用いることができる。この２５６色のうち一部の色だけを使用する場合、必要なパレットデータだけが存在すればよい。このフィールドｎｕｍｂｅｒ＿ｏｆ＿ｐａｌｅｔｔｅ＿ｅｎｔｒｉｅｓにより、用いるインデックスの数が示される。

フィールドｐａｌｅｔｔｅ＿ｉｎｄｅｘ＿ｎｕｍｂｅｒは、次のフィールドｒｅｄ＿ｖａｌｕｅ、フィールドｇｒｅｅｎ＿ｖａｌｕｅ、フィールドｂｌｕｅ＿ｖａｌｕｅおよびフィールドａｌｐｈａでそれぞれ定義される色や透明度に割り当てられたインデックス番

号を表す。画像データからは、このインデックス番号によって色や透明度αが参照される。

このブロックＧｌｏｂａｌＰａｌｅｔｔｅＴａｂｌｅ（）においてｆｏｒ文で示されるループの中では、同じ値を持つフィールドｐａｌｅｔｔｅ＿ｉｎｄｅｘ＿ｎｕｍｂｅｒが二度以上現れてはならない。フィールドｒｅｄ＿ｖａｌｕｅ、フィールドｇｒｅｅｎ＿ｖａｌｕｅおよびフィールドｂｌｕｅ＿ｖａｌｕｅは、それぞれデータ長が８ビットの０以上の整数であって、それぞれ赤、緑、青の色を指定する。８ビットのデータ長を有するフィールドａｌｐｈａは、透明度αを表し、値０が完全に透明、値２５５が完全に不透明な状態を表す。

各ＰＮＧ画像は、パレット情報ＰＬＴＥを格納するチャンクを持つことができるが、この実施の一形態では、そのパレット情報ＰＬＴＥを使用せず、ブロックＧｌｏｂａｌＰａｌｅｔｔｅＴａｂｌｅ（）で定義されたパレット情報を使うものとする。これは、同時に複数のＰＮＧ画像を表示する場合、各ＰＮＧ画像が異なるパレットテーブルに基づく色を使っていると、それぞれ正しい色で表示することが困難になるためである。ＧｒａｐｈｉｃｓＯｂｊｅｃｔ（）に属する、フィールドＰＮＧ＿ｉｍａｇｅ（ｉ）でそれぞれ表される複数のＰＮＧ画像は、全て共通のブロックＧｌｏｂａｌＰａｌｅｔｔｅＴａｂｌｅ（）を参照し、ブロックＧｌｏｂａｌＰａｌｅｔｔｅＴａｂｌｅ（）に示される共通のパレットテーブルを用いるものとする。

命令グループＤｉｓｐＣｍｄｓ（ｉ）について説明する。命令グループＤｉｓｐＣｍｄｓ（ｉ）は、グラフィクスオブジェクト２００による表示を制御する表示制御命令が羅列される。命令グループＤｉｓｐＣｍｄｓ（ｉ）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）は、次の命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）の前までの命令を、指定の時刻ｓｔａｒｔ＿ｔｉｍｅに実行することを指示する。時刻ｓｔａｒｔ＿ｔｉｍｅの原点は、当該グラフィクスオブジェクト２００のｐｔｓとする。また時刻ｓｔａｒｔ＿ｔｉｍｅの単位は、ｐｔｓと同じものが用いられる。

一つの命令グループＤｉｓｐＣｍｄｓ（ｉ）は、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により時刻ｓｔａｒｔ＿ｔｉｍｅから始まり、その他の複数の命令を置くことができる。命令グループＤｉｓｐＣｍｄｓ（ｉ）に置かれたこれらの命令は、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）で指定される時刻ｓｔａｒｔ＿ｔｉｍｅに同時に、実行されるものとする。命令グループＤｉｓｐＣｍｄｓ（ｉ）による命令の実行完了時刻前に、次の命令グループＤｉｓｐＣｍｄｓ（ｉ＋１）の命令ｅｘｅｃｔｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）で示される時刻ｓｔａｒｔ＿ｔｉｍｅになった場合は、命令グループＤｉｓｐＣｍｄｓ（ｉ）の実行は中断され、次の命令グループＤｉｓｐＣｍｄｓ（ｉ＋１）が実行される。

命令グループＤｉｓｐＣｍｄｓ（ｉ）に置かれる、命令ｅｘｅｃｔｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）以外の表示制御命令としては、第８５図Ａおよび第８５図Ｂに一覧が例示されるように、以下のものが考えられる。各行頭の番号は、第８５図の各番号にそれぞれ対応している。（すなわち、番号（１）を、命令ｅｘｅｃｔｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）とする）

（２）グラフィクスオブジェクトの表示を開始する命令
（３）グラフィクスオブジェクトの表示を停止する命令
（４）使用中のパレットテーブルの色や透明度を変更する命令
（５）グラフィクスオブジェクトのプレーン上での表示位置やサイズを設定する命令
（６）グラフィクスオブジェクトの中の表示範囲を設定する命令
（７）効果音の再生を行う命令
（８）画像データ（ＰＮＧデータ）に効果音を割り当てる命令

これらｅｘｅｃｔｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）の後に置かれる７種類の命令は、一例であって、命令グループＤｉｓｐＣｍｄｓ（ｉ）に置かれる命令は、この７種類に限定されるものではない。表示制御命令をさらに定義し追加することは可能である。

（２）および（３）の、グラフィクスオブジェクト２００の表示の開始および終了命令は、所謂フェードイン／フェードアウトを行うための命令であって、それぞれ命令ｆａｄｅ＿ｉｎ（ｆａｄｅ＿ｉｎ＿ｔｉｍｅ）、命令ｆａｄｅ＿ｏｕｔ（ｆａｄｅ＿ｏｕｔ＿ｔｉｍｅ）の形で用いられる。

フェードインは、命令ｆａｄｅ＿ｉｎ（ｆａｄｅ＿ｉｎ＿ｔｉｍｅ）で指示され、グラフィクスオブジェクト２００が非表示の状態から徐々に表示の状態に変化していく。アルファブレンディングの透明度αの値を、時刻ｆａｄｅ＿ｉｎ＿ｔｉｍｅに基づき徐々に増加させていくことで、フェードインが実現できる。命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）の後に命令ｆａｄｅ＿ｉｎ（ｆａｄｅ＿ｉｎ＿ｔｉｍｅ）が置かれていると、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）で指定される時刻ｓｔａｒｔ＿ｔｉｍｅの直後から当該グラフィクスオブジェクト２００が徐々に透明から不透明になっていき、引数の時刻ｆａｄｅ＿ｉｎ＿ｔｉｍｅで指定された時間が経過すると、全てのパレットインデックスの透明度αの値が共通パレットテーブルで指定した値に設定される。

命令ｆａｄｅ＿ｉｎ（ｆａｄｅ＿ｉｎ＿ｔｉｍｅ）において、時刻ｆａｄｅ＿ｉｎ＿ｔｉｍｅが０に設定されていると、当該グラフィクスオブジェクト２００がパレットテーブルに指定される色および透明度αで即座に表示される。

フェードアウトは、フェードインの逆の処理であって、命令ｆａｄｅ＿ｏｕｔ（ｆａｄｅ＿ｏｕｔ＿ｔｉｍｅ）で指示され、グラフィクスオブジェクト２００が表示の状態から徐々に非表示の状態に変化していく。アルファブレンディングの透明度αの値を、時刻ｆａｄｅ＿ｏｕｔ＿ｔｉｍｅに基づき徐々に減少させていくことで、フェードアウトが実現できる。命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）の後に命令ｆａｄｅ＿ｏｕｔ（ｆａｄｅ＿ｏｕｔ＿ｔｉｍｅ）が置かれていると、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）で指定される時刻ｓｔａｒｔ＿ｔｉｍｅの直後から当該グラフィクスオブジェクト２００が徐々に不透明から透明になっていき、引数の時刻ｆａｄｅ＿ｏｕｔ＿ｔｉｍｅで指定された時間が経過すると、全てのパレットインデックスの透明度αの値が０とされ、当該グラフィクスオブジェクト２００が完全に透明になり、見えなくなる。

命令ｆａｄｅ＿ｏｕｔ（ｆａｄｅ＿ｏｕｔ＿ｔｉｍｅ）において、時刻ｆａｄｅ＿ｏｕｔ＿ｔｉｍｅが０に設定されていると、当該グラフィクスオブジェクト２００の表示が即座に停止される。

フェードインおよびフェードアウトにおいて、透明度αの値は、時間の経過と共にできるだけ滑らかに変化させると、より自然なフェードインおよびフェードアウトの効果が得られ、好ましい。これは、この例に限られず、例えば、フェードインにおいて、透明度αの値は、時刻ｆａｄｅ＿ｉｎ＿ｔｉｍｅで示される時間が経過した後に、パレットテーブルで指定した値になっていればよく、どの程度の分解能や階調で透明度αを変化させるかについては、命令によって指定していない。実際には、透明度αの変化の分解能や階調は、実装に依存する。

ここでは可読性を上げるために、命令を「ｆａｄｅ＿ｉｎ（）」、「ｆａｄｅ＿ｏｕｔ（）」のようにテキストで表記して説明したが、実際には、これらの命令ｆａｄｅ＿ｉｎ（）、ｆａｄｅ＿ｏｕｔ（）は、その引数と共に所定のバイナリ値に変換されてＤｉｓｐＣｍｄｓ（ｉ）に記述される。これは、以降に説明される他の命令についても同様である。

（４）の、使用中のパレットテーブルの色や透明度αを変更する命令は、パレット情報を変更する。この命令は、命令ｃｈａｎｇｅ＿ｐａｌｅｔｔｅ（ｉｎｄｅｘ，ｎｅｗＲ，ｎｅｗＧ，ｎｅｗＢ，ｎｅｗＡｌｐｈａ）の形で用いられる。字幕プレーン１１やグラフィクスプレーン１２に同時に表示されるＰＮＧ画像は、第８３図に示したシンタクスで定義される、第２４図のような共通のパレットテーブルを参照している。通常は、ＧｌｏｂａｌＰａｌｅｔｔｅＴａｂｌｅ（）で定義されたパレット情報がそのまま共通パレットテーブルとして使用される。この命令ｃｈａｎｇｅ＿ｐａｌｅｔｔｅ（ｉｎｄｅｘ，ｎｅｗ

Ｒ，ｎｅｗＧ，ｎｅｗＢ，ｎｅｗＡｌｐｈａ）を用いることで、この共通のパレット情報を変更することができる。

命令ｃｈａｎｇｅ＿ｐａｌｅｔｔｅ（ｉｎｄｅｘ，ｎｅｗＲ，ｎｅｗＧ，ｎｅｗＢ，ｎｅｗＡｌｐｈａ）に引数として記述される値ｉｎｄｅｘ、ｎｅｗＲ、ｎｅｗＧ、ｎｅｗＢおよびｎｅｗＡｌｐｈａにより、パレット２２において、パレット番号ｉｎｄｅｘで示されるカラーインデックス値の三原色の値Ｒ、ＧおよびＢが値ｎｅｗＲ、ｎｅｗＧおよびｎｅｗＢにそれぞれ変更され、透明度αの値が値ｎｅｗＡｌｐｈａに変更される。

（５）の、グラフィクスオブジェクトのプレーン上での表示位置やサイズを設定する命令は、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）の形で用いられ、プレーン上の座標（ｘ１，ｙ１）および（ｘ２，ｙ２）で定められる矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）に、当該グラフィクスオブジェクト２００を配置する。また、（６）の、グラフィクスオブジェクトの中の表示範囲を設定する命令は、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）の形で用いられ、グラフィクスオブジェクト２００によるＰＮＧ画像上の座標（ａ１，ｂ１）および（ａ２，ｂ２）で定められる矩形領域（ａ１，ｂ１）（ａ２，ｂ２）を、実際にプレーン上に表示する。

第８６図Ａおよび第８６図Ｂを用いて、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）および命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）について、より具体的に説明する。第８６図Ａおよび第８６図Ｂにおける座標は、第８７図に一例が示されるように、表示画面の左上隅を原点とし、水平右方向をｘ、垂直下方向をｙとして、座標（ｘ，ｙ）と表す。

命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）により、第８６図Ａに示されるように、グラフィクスオブジェクト２００によるＰＮＧ画像２５０内の、実際に表示される矩形領域（ａ１，ｂ１）（ａ２，ｂ２）が設定される。この第８６図Ａの例では、設定される矩形領域（ａ１，ｂ１）（ａ２，ｂ２）がＰＮＧ画像２５０に対して小さいものとされている。命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）により、この矩形領域（ａ１，ｂ１）（ａ２，ｂ２）のプレーン上における実際の表示位置が矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）に設定される（第８６図Ｂ参照）。画面上の矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）に対して、ＰＮＧ画像２５０のうち矩形領域（ａ１，ｂ１）（ａ２，ｂ２）の部分が表示される。

矩形領域（ａ１，ｂ１）（ａ２，ｂ２）が実際に表示される矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）よりも大きい場合は、矩形領域（ａ１，ｂ１）（ａ２，ｂ２）のうち矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）の範囲のＰＮＧ画像だけが表示される。一方、矩形領域（ａ１，ｂ１）（ａ２，ｂ２）の大きさが実際に表示される矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）よりも小さい場合は、矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）内の矩形領域（ａ１，ｂ１）（ａ２，ｂ２）の外側は、透明色として扱うとよい。

上述した表示制御命令を、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）を複数用いて並べることにより、時間経過に従って変化する字幕や同期型グラフィクスを表示することができる。例えば、第８３図を用いて説明したグラフィクスオブジェクト２００において、ブロックＧＯＢＪＣｏｍｍａｎｄＴａｂｌｅ（）内に、複数の命令グループＤｉｓｐＣｍｄｓ（ｉ）を記述する。そして、命令グループＤｉｓｐＣｍｄｓ（ｉ）のそれぞれは、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）における時刻ｓｔａｒｔ＿ｔｉｍｅを所望に異ならせて記述し、時刻ｓｔａｒｔ＿ｔｉｍｅで示される時刻に開始する表示制御命令を記述する。

第８８図は、命令グループＤｉｓｐＣｍｄｓ（ｉ）の記述とグラフィクスオブジェクト２００の表示変化の一例を示す。第８８図において、横軸は時間の経過を表し、縦軸はグラフィクスオブジェクト２００の透明度を表す。透明度は、第８８図で上に向かうほど大きくなる。ｐｔｓで示される時刻が原点とされる。

最初の命令グループＤｉｓｐＣｍｄｓ（０）により、プレーン上の表示領域が命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（８００，８００，１３００，９００）として、グラフィクスオブジェクト２００のＰＮＧ画像における表示領域が命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿

ｂｏｘ（０，０，５００，１００）として設定し、命令ｆａｄｅ＿ｉｎ（２ｓｅｃ）により、２秒かけてフェードインする処理を、時刻〔０〕から開始することが指示される。次の命令グループＤｉｓｐＣｍｄｓ（１）は、命令ｃｈａｎｇｅ＿ｐａｌｅｔｔｅ（ｉｎｄｅｘ，ｎｅｗＲ，ｎｅｗＧ，ｎｅｗＢ，Ａｌｐｈａ）がカラーインデックス値〔１〕、〔２〕〔３〕および〔４〕についてそれぞれ記述され、時刻〔８００〕から、パレットテーブル２２においてカラーインデックス値〔１〕、〔２〕〔３〕および〔４〕で参照される色および透明度αを変更する旨が指示される。そして、次の命令グループＤｉｓｐＣｍｄｓ（２）により、表示されているグラフィクスオブジェクト２００を、時刻〔２０００〕から２秒かけてフェードアウトすることが指示される。

この第８８図に示されるように、命令グループＤｉｓｐＣｍｄｓ（０）、ＤｉｓｐＣｍｄｓ（１）およびＤｉｓｐＣｍｄｓ（２）を並べることで、時間経過に従って表示が変化する例えば字幕を実現することができる。命令グループＤｉｓｐＣｍｄｓ（０）、ＤｉｓｐＣｍｄｓ（１）およびＤｉｓｐＣｍｄｓ（２）を適当に用いることで、字幕やボタン画像のアニメーション表示が可能とされる。

第８９図Ａ、第８９図Ｂ、第８９図Ｃおよび第８９図Ｄは、字幕表示が徐々に現れるフェードインの例を示す。第８９図Ａ～第８９図Ｄと、徐々に字幕表示が現れるように表示制御される。このような単純なフェードインは、上述した第８８図の命令グループＤｉｓｐＣｍｄｓ（０）と同様の命令で実現することができる。

第９０図Ａおよび第９０図Ｂは、字幕のＰＮＧ画像２６０がプレーン内を移動する例を示す。これは、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）を複数、用いることで実現できる。例えば、最初の命令グループＤｉｓｐＣｍｄｓ（０）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により開始時刻が設定され、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）により第９０図Ａに一例が示されるようにＰＮＧ画像２６０の表示領域を設定すると共に、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）により、ＰＮＧ画像２６０のプレーン内での初期の表示領域を設定する。

次の命令グループＤｉｓｐＣｍｄｓ（１）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により命令グループＤｉｓｐＣｍｄｓ（０）から所定時間経過した時刻が開始時刻として指定されると共に、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１’，ｙ１’，ｘ２’，ｙ２’）により、プレーン内の移動先の表示領域が設定される。同様にして、次の命令グループＤｉｓｐＣｍｄｓ（２）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により命令グループＤｉｓｐＣｍｄｓ（１）から所定時間経過した時刻が開始時刻として指定されると共に、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１”，ｙ１”，ｘ２”，ｙ２”）により、プレーン内の移動先の表示領域が設定される。

このようにすることで、第９０図Ｂに一例が示されるように、字幕のＰＮＧ画像２６０がプレーン上の矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）、矩形領域（ｘ１’，ｙ１’）（ｘ２’，ｙ２’）、矩形領域（ｘ１”，ｙ１”）（ｘ２”，ｙ２”）と移動するようにできる。

第９１図Ａおよび第９１図Ｂは、字幕のＰＮＧ画像２６１内の表示領域２６２が移動し、字幕表示をスクロールさせる例を示す。これは、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）を複数、用いることで実現できる。例えば、最初の命令グループＤｉｓｐＣｍｄｓ（０）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により開始時刻が設定され、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）により、第９１図Ａに一例が示されるように、ＰＮＧ画像２６０において初期に表示される矩形領域２６２を設定すると共に、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）により、ＰＮＧ画像２６０がプレーン内で表示される矩形領域を設定する。

次の命令グループＤｉｓｐＣｍｄｓ（１）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により命令グループＤｉｓｐＣｍｄｓ（０）から所定時間経

過した時刻が開始時刻として指定されると共に、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１’，ｂ１’，ａ２’，ｂ２’）により、ＰＮＧ画像２６０内の移動先の表示領域が設定される。同様にして、次の命令グループＤｉｓｐＣｍｄｓ（２）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により命令グループＤｉｓｐＣｍｄｓ（１）から所定時間経過した時刻が開始時刻として指定されると共に、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１”，ｂ１”，ａ２”，ｂ２”）により、ＰＮＧ画像２６０内の移動先の矩形領域が設定される。

このようにすることで、第９１図Ｂに一例が示されるように、プレーン上の矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）内を、字幕のＰＮＧ画像２６１内の一部の矩形領域が矩形領域（ａ１，ｂ１）（ａ２，ｂ２）、矩形領域（ａ１’，ｂ１’）（ａ２’，ｂ２’）、矩形領域（ａ１”，ｂ１”）（ａ２”，ｂ２”）と移動するようにされ、字幕表示のスクロールを実現できる。

第９２図Ａおよび第９２図Ｂは、ＰＮＧ画像２６５の一部を表示する枠を設定し、ＰＮＧ画像２６５上でこの枠を移動させつつ、プレーン上での表示位置も移動させる例を示す。これは、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）と、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）とを複数、同時に用いることで実現できる。例えば、最初の命令グループＤｉｓｐＣｍｄｓ（０）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により開始時刻が設定され、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）と命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）とにより、枠２６６Ａが設定される（第９２図Ａ参照）。

例えば、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１，ｙ１，ｘ２，ｙ２）によりプレーン上で表示される矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）が設定されると共に、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１，ｂ１，ａ２，ｂ２）によりＰＮＧ画像２６５において表示される矩形領域（ａ１，ｂ１）（ａ２，ｂ２）が設定される。これら矩形領域（ｘ１，ｙ１）（ｘ２，ｙ２）および矩形領域（ａ１，ｂ１）（ａ２，ｂ２）により、枠２６６Ａが設定される。

そして、次の命令グループＤｉｓｐＣｍｄｓ（１）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により命令グループＤｉｓｐＣｍｄｓ（０）から所定時間経過した時刻が開始時刻として指定され、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１’，ｙ１’，ｘ２’，ｙ２’）によるプレーン内の矩形領域（ｘ１’，ｙ１’）（ｘ２’，ｙ２’）と、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１’，ｂ１’，ａ２’，ｂ２’）によるＰＮＧ画像２６５上の矩形領域（ａ１’，ｂ１’）（ａ２’，ｂ２’）により、枠２６６Ａに対する移動先の枠２６６Ｂが設定される。同様に、次の命令グループＤｉｓｐＣｍｄｓ（２）において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により命令グループＤｉｓｐＣｍｄｓ（１）から所定時間経過した時刻が開始時刻として指定され、命令ｓｅｔ＿ｄｉｓｐｌａｙ＿ｂｏｘ（ｘ１”，ｙ１”，ｘ２”，ｙ２”）によるプレーン内の矩形領域（ｘ１”，ｙ１”）（ｘ２”，ｙ２”）と、命令ｓｅｔ＿ｃｌｉｐｐｉｎｇ＿ｂｏｘ（ａ１”，ｂ１”，ａ２”，ｂ２”）によるＰＮＧ画像２６５上の矩形領域（ａ１”，ｂ１”）（ａ２”，ｂ２”）により、枠２６６Ｂに対する移動先の枠２６５Ｂが設定される。

こうすることで、第９２図Ｂに一例が示されるように、字幕のＰＮＧ画像２６５内の一部の矩形領域が移動しつつ、当該矩形領域がプレーン内を領域２６５Ａ、２６５Ｂ、２６５Ｃと移動するような表示が可能となる。

このように、この発明の実施の一形態では、グラフィクスオブジェクト２００の表示制御を、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）により各表示制御命令をグループ化した命令グループＤｉｓｐＣｍｄｓ（ｉ）を用いて行っているため、字幕プレーン１１およびグラフィクスプレーン１２における多彩な表示を、容易に実現することができる。

２－１３．効果音について

この発明の実施の一形態では、さらに、グラフィクスオブジェクト２００の表示制御に同期して音声を出力することができる。上述した、命令グループＤｉｓｐＣｍｄｓ（ｉ）に置かれる、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）以外の表示制御命令（２）～（８）のうち、（７）の効果音の再生を行う命令と、（８）の画像データに効果音を割り当てる命令により、音声出力が定義される。また、音声データには、それぞれ識別子ｓｏｕｎｄ＿ｉｄが付与され、互いが識別される。

（７）の、効果音の再生を行う命令は、命令ｐｌａｙ＿ｓｏｕｎｄ（ｓｏｕｎｄ＿ｉｄ）の形で用いられ、識別子ｓｏｕｎｄ＿ｉｄで指定された音声データの再生を指示する。この命令を命令グループＤｉｓｐＣｍｄｓ（ｉ）に含めることにより、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（ｓｔａｒｔ＿ｔｉｍｅ）で指定された時刻ｓｔａｒｔ＿ｔｉｍｅに、識別子ｓｏｕｎｄ＿ｉｄで指定された音声データが再生される。

一例として、命令ｐｌａｙ＿ｓｏｕｎｄ（ｓｏｕｎｄ＿ｉｄ）を命令ｆａｄｅ＿ｉｎ（ｆａｄｅ＿ｉｎ＿ｔｉｍｅ）および命令ｆａｄｅ＿ｏｕｔ（ｆａｄｅ＿ｉｎ＿ｔｉｍｅ）と同時に使用することで、字幕の表示および／または消去に連動して音声データを再生し、効果音を発生させることが可能となる。第８８図は、命令ｐｌａｙ＿ｓｏｕｎｄ（ｓｏｕｎｄ＿ｉｄ）を使用した例である。第８８図の例では、最初の命令グループＤｉｓｐＣｍｄｓ（０）において、開始時刻〔０〕で２秒かけて表示がフェードインすると共に、命令ｐｌａｙ＿ｓｏｕｎｄ（１）により識別子ｓｏｕｎｄ＿ｉｄが〔１〕で特定される音声データが再生される。次いで、命令グループＤｉｓｐＣｍｄｓ（１）において、時刻〔８００〕で表示色が変更されると共に、命令ｐｌａｙ＿ｓｏｕｎｄ（２）により識別子ｓｏｕｎｄ＿ｉｄが〔２〕で特定される音声データが再生される。そして、命令グループＤｉｓｐＣｍｄｓ（２）において、時刻〔２０００〕から１秒かけて表示がフェードアウトすると共に、命令ｐｌａｙ＿ｓｏｕｎｄ（１）により識別子ｓｏｕｎｄ＿ｉｄが〔１〕で特定される音声データが再生される。

命令ｐｌａｙ＿ｓｏｕｎｄ（ｓｏｕｎｄ＿ｉｄ）は、必須の命令ではない。

（８）の、ＰＮＧデータに効果音を割り当てる命令は、命令ｓｅｔ＿ｓｏｕｎｄ（ＰＮＧ＿ｉｍａｇｅ＿ｉｄ，ｓｏｕｎｄ＿ｉｄ）の形で用いられ、識別子ＰＮＧ＿ｉｍａｇｅ＿ｉｄで指定されるＰＮＧデータに対して、識別子ｓｏｕｎｄ＿ｉｄで指定される音声データを割り当てる。この命令ｓｅｔ＿ｓｏｕｎｄ（ＰＮＧ＿ｉｍａｇｅ＿ｉｄ，ｓｏｕｎｄ＿ｉｄ）により、識別子ＰＮＧ＿ｉｍａｇｅ＿ｉｄで指定されるＰＮＧデータが表示される際に、識別子ｓｏｕｎｄ＿ｉｄで指定される音声データが再生される。ＰＮＧデータの識別子ＰＮＧ＿ｉｍａｇｅ＿ｉｄは、例えば、上述した第８３図における、ブロックＰＮＧＩｍａｇｅＲｅｇｉｏｎ（）中のＰＮＧ＿ｉｍａｇｅ（ｉ）に対して指定されるループカウンタｉと同じ値を用いることができる。

この命令ｓｅｔ＿ｓｏｕｎｄ（ＰＮＧ＿ｉｍａｇｅ＿ｉｄ，ｓｏｕｎｄ＿ｉｄ）は、主に、選択状態および実行状態のボタンのＰＮＧデータに用いることが想定されている。こうすることで、ボタンが通常状態から選択状態へ遷移する際や、選択状態から実行状態に遷移する際に、それぞれの状態を表すＰＮＧデータに割り当てられた音声データを効果音として発生させることが可能となる。この例に限らず、ボタン以外のＰＮＧデータに対してこの命令ｓｅｔ＿ｓｏｕｎｄ（ＰＮＧ＿ｉｍａｇｅ＿ｉｄ，ｓｏｕｎｄ＿ｉｄ）を用いてもよい。

第９３図は、ボタン画像に音声データを割り当てたグラフィクスオブジェクト２００の一例のデータ構造を示す。ＰＮＧデータ領域２０３に、通常状態、選択状態および実行状態それぞれにおけるボタンのＰＮＧデータ２０３Ａ、２０３Ｂおよび２０３Ｃが格納される。この第９３図の例では、表示制御命令ではＰＮＧデータに対する座標指定と、音声データの割り当てのみを行い、ＰＮＧデータの表示開始時刻や、表示開始時のボタンの初期状態は、外部のスクリプトプログラムにより制御するようにしている。そのため、表示制御命令は、便宜上、実行時刻〔０〕の命令として記述する。

第９３図に例示されるグラフィクスオブジェクト２００では、表示制御命令テーブル２０２において、命令ｅｘｅｃｕｔｉｏｎ＿ｔｉｍｅ（０）に基づき時刻〔０〕で実行され

る、命令グループＤｉｓｐＣｍｄｓ（０）だけが存在している。識別子ＰＮＧ＿ｉｍａｇｅ＿ｉｄは、〔０〕から始まるので、識別子ＰＮＧ＿ｉｍａｇｅ＿ｉｄが〔０〕で通常状態時のＰＮＧデータ２０３Ａを、識別子ＰＮＧ＿ｉｍａｇｅ＿ｉｄが〔１〕で選択状態時のＰＮＧデータ２０３Ｂを、識別子ＰＮＧ＿ｉｍａｇｅ＿ｉｄが〔２〕で実行状態時のＰＮＧデータ２０３Ｃをそれぞれ表している。

命令ｓｅｔ＿ｓｏｕｎｄ（１，１０）に基づき、識別子ＰＮＧ＿ｉｍａｇｅ＿ｉｄが〔１〕である選択状態のボタンを表示するＰＮＧデータ２０３Ｂが表示されたときには、識別子ｓｏｕｎｄ＿ｉｄが〔１０〕で指定される音声データが効果音として再生される。同様に、命令ｓｅｔ＿ｓｏｕｎｄ（２，１１）に基づき、識別子ＰＮＧ＿ｉｍａｇｅ＿ｉｄが〔２〕である実行状態のボタンを表示するＰＮＧデータ２０３Ｃが表示されたときには、識別子ｓｏｕｎｄ＿ｉｄが〔１１〕で指定される音声データが効果音として再生されることになる。

第７３図Ａ、第７３図Ｂおよび第７３図Ｃでは省略されているが、プレーヤは、予め１または複数種類の音声データを、プレーヤ内部のメモリなどに保持することができる。例えば、プレーヤは、プレーヤ内部に設けられた不揮発性メモリなどに、出荷時に予め所定の音声データを記憶しておくようにできる。

これに限らず、グラフィックオブジェクト２００や動画データによるコンテンツが記録されたディスクに、効果音として用いるための音声データを記録しておき、当該ディスクの再生時にこの音声データを読み込んでおくようにしてもよい。ディスクへの音声データの格納方法としては、音声データが格納されたファイルを別に用意し、ディスク再生開始時にこのファイルを先に読み込んでおき、プレーヤ内のメモリに置いておく方法がある。

グラフィクスオブジェクト２００と同様に音声データを格納したＰＥＳパケットを作り、それをＴＳパケットに分割してクリップＡＶストリームに多重化する方法も考えられる。

上述の第７６図Ａ、第７６図Ｂおよび第７６図Ｃのグラフィクスオブジェクト２００中のグラフィクスオブジェクトヘッダ２０１内（図示は省略する）、或いは、上述したようにＰＮＧ画像データ領域２０３の直後に音声データ領域２０４を設け、グラフィクスオブジェクト２００に含まれるＰＮＧ画像に対応する音声データを格納する方法も考えられる。

何れの方法であっても、音声データを予めディスクからプレーヤのメモリに読み込んで蓄積しておくことができるので、ＰＮＧ画像で作られたボタンが選択状態や実行状態になると同時に効果音を発生させることが可能となる。音声データには、それぞれユニークな識別子ｓｏｕｎｄ＿ｉｄが割り当てられており、音声データを一意に特定することができるようにされている。

この、音声データのディスクへの格納方法について、より詳細に説明する。グラフィクスオブジェクト２００に音声データを格納する方法について、第７６図Ａ、第７６図Ｂおよび第７６図Ｃを用いて説明する。第７６図Ａおよび第７６図Ｃは、音声データをグラフィクスオブジェクト２００に付加してクリップＡＶストリームに多重化する例が示されている。

第７６図Ａは、グラフィクスオブジェクト２００中のＰＮＧデータ領域２０３の後ろに音声データ領域２０４を設けた例である。音声データ領域２０４には、複数の音声データ２０４Ａ、２０４Ｂ、・・・、２０４ｎを格納することができる。これら音声データ２０４Ａ、２０４Ｂ、・・・、２０４ｎを、同一のグラフィクスオブジェクト２００に格納されるＰＮＧデータ２０３Ａ、２０３Ｂ、・・・、２０３ｎとそれぞれ関連性のあるものとすると、ＰＮＧ画像および音声データを対応付けての扱いが容易になる。

音声データ２０４Ａ、２０４Ｂ、・・・、２０４ｎは、例えばＡＩＦＦ（Ａｕｄｉｏ Ｉｎｔｅｒｃｈａｎｇｅ Ｆｉｌｅ Ｆｏｒｍａｔ）形式やＷＡＶＥファイルといった、圧縮符号化されていないデータでもよいし、ＭＰ３（Ｍｏｖｉｎｇ Ｐｉｃｔｕｒｅｓ Ｅｘｐｅｒｔｓ Ｇｒｏｕｐ １ Ａｕｄｉｏ Ｌａｙｅｒ ３）やＡＡＣ（Ａｄｖａｎｃｅｄ Ａｕｄｉｏ Ｃｏｄｉｎｇ）、ＡＴＲＡＣ（Ａｄａｐｔｉｖｅ Ｔｒａｎｓｆｏ

ｒｍ　Ａｃｏｕｓｔｉｃ　Ｃｏｄｉｎｇ）といった、圧縮符号化されたデータでもよい。圧縮符号化された音声データを格納する場合には、プレーヤ側が音声データの圧縮符号化方式に対応したオーディオデコーダを持っている必要がある。

第７６図Ｃは、ボタンを構成するグラフィクスオブジェクト２００に対して、ボタンのそれぞれの状態に対応した音声データを格納した例である。この例では、例えばボタンの選択状態時に再生する音声データ２０４Ａ－２と、実行状態時に再生する音声データ２０４Ｂ－２とが、音声データ領域２０３に格納される。一方、ＰＮＧデータ領域２０３には、通常状態、選択状態および実行状態それぞれのボタン表示に用いるＰＮＧデータ２０３Ａ－２、２０３Ｂ－２および２０３Ｃ－２が格納される。

この場合、例えば、ボタン画像が格納されるＰＮＧデータ領域２０３に続く音声データ領域２０４には、ボタンが選択状態時に遷移したときに再生する音声データ２０４Ａ－２と、実行状態時に遷移したときに再生する音声データ２０４Ｂ－２とが格納されるように、予め決めておく。そして、ボタン画像のＰＮＧデータが表示されると同時に、当該ボタンの状態に対応した音声データが再生されるようにする。このプレーヤで再生される効果音は、主にボタンクリック音として使われることを想定しているため、このようなルールでも、十分に、主要な目的を達成することができる。

音声データをクリップＡＶストリームに多重化しない方法について説明する。例えば、第９４図に一例が示されるように、ディレクトリＢＤＡＶの下に音声データを格納するディレクトリＳＯＵＮＤを設ける。このディレクトリＳＯＵＮＤには、音声データであるＰＣＭ波形データを格納する、例えばＡＩＦＦ形式の音声データファイル「ｓｏｕｎｄ１．ａｉｆｆ」を置く。ディレクトリＳＯＵＮＤに格納された音声データファイルは、例えば当該ディスクのプレーヤへの最初のローディング時にまとめて読み込まれ、プレーヤの内部メモリに格納される。

各音声データには、音声データそれぞれを識別可能な識別子ｓｏｕｎｄ＿ｉｄが予め割り当てられ、プログラムやスクリプトなどから識別子ｓｏｕｎｄ＿ｉｄにより、所望の音声データがメモリから呼び出されるモデルが構成される。

この場合、例えば、第９５図に一例が示されるように、グラフィクスオブジェクト２００に対してサウンドｉｄ領域２０５を設け、このサウンドｉｄ領域２０５に対してサウンドｉｄデータ２０５Ａ、２０５Ｂを格納する。この第９５図の例では、ＰＮＧデータ領域２０３にボタンの通常状態時、選択状態時および実行状態時にそれぞれ対応したＰＮＧデータ２０３Ａ、２０３Ｂおよび２０３Ｃが格納され、サウンドｉｄデータ２０５Ａおよび２０５Ｂは、ＰＮＧデータ２０３Ｂおよび２０３Ｃにそれぞれ対応する識別子ｓｏｕｎｄ＿ｉｄである。ＰＮＧデータ２０３Ｂの表示の際に、プレーヤ内部のメモリに予め格納された、サウンドｉｄデータ２０５Ａが示す識別子ｓｏｕｎｄ＿ｉｄが対応する音声データを再生するように指示される。

例えば、第９３図を用いて既に説明したように、表示制御命令テーブル２０２の記述によって、ＰＮＧデータと音声データとを関連付けるようにしてもよい。

この第９５図に一例が示される構成のデータは、ディスクから読み出すだけでなく、インターネットなどのネットワークから取得し、入力端１０１に入力するようにもできる。

識別子ｓｏｕｎｄ＿ｉｄにより音声データを指定するため、第８５図を用いて既に説明した表示制御コマンドなどにより、グラフィクス表示とは関係の無い任意の時刻に、音声データによる効果音などを発生させることができる。

この方法では、識別子ｓｏｕｎｄ＿ｉｄで音声データをメモリから呼び出すため、効果音の種類は、第１に、識別子ｓｏｕｎｄ＿ｉｄの数で制限される。第２に、使える効果音の種類が、上述したように、プレーヤの内部メモリの容量により制限される。

第９６図を用いて、より具体的に説明する。ディスク４００のローディング時に、プレーヤによりディスクに対してイニシャルアクセスがなされ、ディレクトリＢＤＡＶの下に配置されるディレクトリＳＯＵＮＤから、音声データがまとめて読み込まれる。読み込まれた音声データ（ＰＣＭデータ）は、プレーヤの内部メモリ４１０に格納される。このとき、音声データのそれぞれに対して固有の識別子ｓｏｕｎｄ＿ｉｄが割り当てられる。デ

ィスク４００に記録されている音声データに対して予め識別子ｓｏｕｎｄ＿ｉｄを付加しておいてもよい。

この例では、１６個の音声データがディスク４００から読み込まれ、識別子ｓｏｕｎｄ＿ｉｄ＝１～１６がそれぞれの音声データに割り当てられる。音声データそれぞれのデータサイズが取得される。この第９６図の例では、識別子ｓｏｕｎｄ＿ｉｄ＝１～１６の音声データがそれぞれｄ１バイト、ｄ２バイト、・・・、ｄ１６バイトのデータサイズを有しているものとする。

例えば、ボタン４２０Ａ、４２０Ｂおよび４２０Ｃが表示されるメニュー画面４２０における、ボタン４２０Ｃに対する操作に基づき、ボタン４２０Ｃに対して指定された識別子ｓｏｕｎｄ＿ｉｄにより対応する音声データがメモリ４１０から読み出される。この第９６図の例では、ボタン４２０Ｃの実行状態に対して識別子ｓｏｕｎｄ＿ｉｄ＝１で示される音声データが指定されている。メモリ４１０から読み出されたこの音声データは、所定に処理されてバッファ４５０Ｂに一旦溜め込まれる。そして、オーディオミキサ２３１に供給され、例えば本編の動画像データに付随する音声データと混合されて、音声として出力される。

バッファ４５０Ａには、例えば本編の動画像データに付随する音声データが一旦、溜め込まれる。例えば、バッファ４５０Ａおよび４５０Ｂの間で、溜め込まれた音声データを読み出すタイミングを調整することで、バッファ４５０Ｂに溜め込まれたボタン４２０Ｃの操作に対応した効果音がバッファ４５０Ａに溜め込まれた音声データの適当なタイミングで出力される。この例では、識別子ｓｏｕｎｄ＿ｉｄ＝０で、音声データの再生無しが指定される。

このようなモデルにおいて、ディスク４００から読み込み可能な音声データの合計容量がメモリ４１０の容量に基づき制限される。それぞれの音声データの容量がバッファ４５０Ｂの容量に基づき制限される。メモリ４１０の容量を容量Ｍ（バイト）、バッファ４５０Ｂの容量を容量Ｄｍａｘ（バイト）とした場合、次の２つの条件を満たす必要がある。

（１）メモリ４１０に格納された音声データのそれぞれの容量ｄｉがバッファ４５０Ｂの容量Ｄｍａｘより小さい。

（２）メモリ４１０に格納された音声データの総容量（ｄ１＋ｄ２＋・・・＋ｄｎ）がメモリ４１０の容量Ｍより小さい。

換言すれば、この（１）および（２）の条件をプレーヤ側とディスク制作側とに対してルール化し、規格として定めておくことで、効果音などの音声データの再生に関して再生互換性を確保することができる。

音声データをクリップＡＶストリームに多重化しない場合に、音声データをディスクのプレーヤへの最初のローディング時にまとめて読み込むように説明したが、これはこの例に限定されない。一例として、音声データを複数回に分けて読み出すようにできる。例えば、シナリオの進行に応じてある区切となる箇所で、次の区切りまでに用いるだけの音声データをまとめて読み出し、メモリに記憶させる。このとき、それまでメモリに記憶されていた音声データを消去する。こうすることで、１つのシナリオの再生において、メモリの容量以上の音声データを扱うことが可能とされる。

音声データは、ディスクの所定領域にまとめて記録しておいてもよいし、ディスクの複数領域に分けて記録しておいてもよい。ディスクの複数領域に分けて記録する場合には、例えば、シナリオの進行に応じて、シナリオの区切りに対応したディスク上の位置に、次の区切りまでに用いられる音声データをまとめて記録することが考えられる。ネットワークで結ばれたサーバーから音声データをダウンロードすることも、ファイルが置かれている場所を例えばＵＲＬ（Ｕｎｉｆｏｒｍ　Ｒｅｓｏｕｒｃｅ　Ｌｏｃａｔｏｒ）で指定するなどすれば、ディスクからの読み出しと同様の仕組みで実現可能である。

第７６図Ａおよび第７６図Ｂを用いて説明した、音声データをクリップＡＶストリームに多重化する方法では、音声データの種類に制約が無く、１画像毎に異なる音声データを指定できる利点がある。音声データは、必要に応じてクリップＡＶストリームにより供給されるため、その都度異なる音声データを用いることができる。音声データをクリップＡ

Ｖストリームに多重化する方法では、音声データが画像データと同時にクリップＡＶストリームから読み込まれるため、読み込みモデルが簡単になる、音声データのファイル数の制限やファイルサイズの制限は、ディスク容量以外の制約を受けない、などの利点がある。

音声データをクリップＡＶストリームに多重化する方法では、異なるグラフィクスオブジェクトで同一の音声データを扱うような場合、それぞれのグラフィックオブジェクトが同一の音声データをそれぞれ持たなければいけないので、データが冗長になってしまう。グラフィクスオブジェクトから音声データを抽出する必要があるため、クリップＡＶストリームのデマルチプレクス処理の後に、音声データを分離する処理が必要となる。

第７３図Ａ、第７３図Ｂおよび第７３図Ｃを用いてグラフィクスオブジェクト２００に連動可能な音声データ処理を説明する。

上述の、クリップＡＶストリームに多重化されない音声データは、例えば入力チャンネル（１）のデータとして入力端１０１に入力され、スイッチ回路１０２および１０３を介してコンテンツバッファ１０５に供給される。一方、音声データが格納されたグラフィクスオブジェクト２００が多重化されたクリップＡＶストリームは、入力端２０２に入力される。そして、ＰＩＤフィルタ１１０でグラフィクスオブジェクト２００が振り分けられバッファＴＢｎ１１１Ａに一旦溜め込まれ、スイッチ回路１０３を介してコンテンツバッファ１０５に供給される。

音声データの格納されていないグラフィクスオブジェクト２００がクリップＡＶストリームに多重化されて入力端２０２から入力される。ＰＩＤフィルタ１１０でグラフィクスオブジェクト２００を構成するトランスポートパケットが振り分けられ、バッファＴＢｎ１１１ＢまたはバッファＴＢｎ１１１Ｃに一旦溜め込まれる。バッファＴＢｎ１１１Ｂに溜め込まれたトランスポートパケットは、バッファＢｎ１１２Ｂに供給され、ＰＩＤヘッダに基づきグラフィクスオブジェクト２００がまとめられる。グラフィクスオブジェクト２００は、スイッチ回路１０７を介してグラフィクスデコーダＡ１１６に供給される。バッファＴＢｎ１１１Ｃに溜め込まれたトランスポートパケットも、同様にしてバッファＢｎ１１２Ｃを用いてグラフィクスオブジェクト２００がまとめられ、スイッチ回路１０８を介してグラフィクスデコーダＢ１１７に供給される。

グラフィクスデコーダＡ１１６およびＢ１１７では、供給されたトランスポートパケットのヘッダ情報を除去すると共に、当該トランスポートパケットに格納された画像データをデコードして字幕またはグラフィクスを表示するための画像データ、例えばビットマップデータとする。

コンテンツバッファ１０５に供給された、音声データが格納されたグラフィクスオブジェクト２００に格納される画像データは、コンテンツバッファ１０５から取り出され、スイッチ回路１０７および１０８を介してグラフィクスデコーダＡ１１６およびＢ１１７にもそれぞれ供給される。

この第７３図Ａ、第７３図Ｂおよび第７３図Ｃの例では、グラフィクスデコーダＡ１１６は、字幕プレーンに展開する画像データをデコードし、グラフィクスデコーダＢ１１７は、グラフィクスプレーンに展開する画像データをデコードする。グラフィクスデコーダＡ１１６およびＢ１１７は、これに限らず、さらに他のデータ形式の画像データをデコードするものであってもよいし、複数の形式の画像データに対応するようにもできる。

グラフィクスデコーダＡ１１６の出力は、スイッチ回路１３０の入力端１３０Ｂおよびスイッチ回路１３１の入力端１３１Ｃに供給され、スイッチ回路１３０および１３１を介してそれぞれ字幕プレーン１３２およびグラフィクスプレーン１３３に供給される。

マルチメディアエンジン１０６は、サウンドプレーヤ１０６Ｄを有する。バッファ１０９は、サウンドバッファ１０９Ｅを有する。サウンドプレーヤ１０６Ｄは、サウンドバッファ１０９Ｅを用いて、コンテンツバッファ１０５から読み出した音声データを復号化して、例えばリニアＰＣＭオーディオデータにして出力する。サウンドプレーヤ１０６Ｄから出力された音声データは、プレゼンテーションプロセッサ１５７に供給され、オーディオデコーダ１１８から出力された音声データと混合されて、出力端１５８に導出される。

例えばボタン画像をクリックした際に発生されるクリック音など、効果音となる音声データは、サウンドプレーヤ１０６Ｄによって再生される。音声データは、サウンドバッファ１０９Ｅに蓄積され、サウンドプレーヤ１０６Ｄにより再生される。

マルチメディアエンジン１０６では、例えばシナリオ記述言語としてＨＴＭＬとＥＣＭＡスクリプトとを組み合わせて用いている場合コードバッファ１０４に蓄えられたＥＣＭＡスクリプトを読み出して解析し、コードバッファ１０４からの他のＥＣＭＡスクリプトやＨＴＭＬ文書の読み出し、コンテンツバッファ１０５からの画像データや音声データの読み出しなどを行う。コンテンツバッファ１０５に格納された音声データは、コンテンツバッファ１０５に格納された他のデータと同様、コンテンツバッファ１０５に保持しておくことができる。

マルチメディアエンジン１０６により、ユーザからの、リモートコントロールコマンダやポインティングデバイスなどによる入力が受け取られ、その入力に基づく処理が行われる。マルチメディアエンジン１０６では、このユーザ入力に基づく処理結果や各スクリプトに応じてコントロール信号を生成する。このコントロール信号は、グラフィクスデコーダＡ１１６およびＢ１１７、オーディオデコーダ１１８、ＭＰＥＧビデオデコーダ１２０およびシステムデコーダ１２１にも供給される。

グラフィクスレンダラ１０６Ｃで処理された画像データは、スイッチ回路１３０および１３１をそれぞれ介して字幕プレーン１３２およびグラフィクスプレーン１３３に供給される。字幕プレーン１３２およびグラフィクスプレーン１３３は、例えばフレームメモリからなり、第２０図を用いて説明した字幕プレーン１１およびグラフィクスプレーン１２にそれぞれ対応する。

グラフィクスレンダラ１０６Ｃから字幕プレーン１３２およびグラフィクスプレーン１３３に供給される画像データは、例えばランレングス圧縮形式、ＰＮＧ形式またはＪＰＥＧ形式の画像データをグラフィクスレンダラ１０６Ｃでデコードした後のビットマップデータであるものとする。

マルチメディアエンジン１０６は、プレゼンテーションプロセッサ１５５に対して、字幕プレーン１３２、グラフィクスプレーン１３３および動画像プレーン１３４を切り換える制御信号を供給し、プレゼンテーションプロセッサ１４１に対して、オーディオストリーム出力を制御するような制御信号を供給する。

字幕プレーン１３２上の画像データは、第２２図におけるパレット２２に対応するパレット１５０に供給され、２５６色からなるパレットに対してインデックスによる参照がなされ、ＲＧＢデータが出力されると共に、透明度データα１が出力される。ＲＧＢデータは、第２２図におけるＲＧＢ／ＹＣｂＣｒ変換回路２９に対応するＲＧＢ／ＹＣｂＣｒ変換回路１５１に供給され、カラーシステムがＲＧＢ（４：４：４）からＹＣｂＣｒ（４：４：４）に変換される。ＲＧＢ／ＹＣｂＣｒ変換回路１５１から出力されたＹＣｂＣｒデータは、プレゼンテーションプロセッサ１５５に供給される。

グラフィクスプレーン１３３上の画像データは、第２２図におけるパレット２６に対応するパレット１５２に供給され、２５６色からなるパレットに対してインデックスによる参照がなされ、ＲＧＢデータが出力されると共に、透明度データα２が出力される。ＲＧＢデータは、第２２図におけるＲＧＢ／ＹＣｂＣｒ変換回路２７に対応するＲＧＢ／ＹＣｂＣｒ変換回路１５３に供給され、カラーシステムがＲＧＢ（４：４：４）からＹＣｂＣｒ（４：４：４）に変換される。ＲＧＢ／ＹＣｂＣｒ変換回路１５３から出力されたＹＣｂＣｒデータは、プレゼンテーションプロセッサ１５５に供給される。

プレゼンテーションプロセッサ１５５には、動画像プレーン１３４上の動画像データがアップ／ダウンコンバータ１５４を介して供給される。

プレゼンテーションプロセッサ１５５は、字幕プレーン１１（字幕プレーン１３２）による透明度α１と、グラフィクスプレーン１２（グラフィクスプレーン１３３）による透明度α２とを用いたアルファブレンディング処理を行う。この処理により、動画プレーン１０、字幕プレーン１１およびグラフィクスプレーン１２の画像データが合成される。プレゼンテーションプロセッサ１５５は、画像データに対してリアルタイムでエフェクト処

理を行うこともできる。このように、プレーン間の合成処理やエフェクト処理がなされた画像データが出力端１５６に導出される。

２－１４．その他

以上説明したように、この発明によれば、記録済み大容量ディスクにおいて、動画を表示する動画プレーンと、字幕を表示する字幕プレーンと、メニュー画面などインタラクティブな機能を有する画面を表示するグラフィクスプレーンとがそれぞれ独立して設けられ、これらのプレーンが合成されて表示されるようにしているため、動画プレーンに動画を表示させ、その動画を背景としてグラフィクスプレーンにメニュー画面などを表示させることができるという効果がある。

この発明によれば、グラフィクスプレーンに表示するための画像データを格納するためのバッファが設けられているため、グラフィクスプレーンの表示を同一の画像データを繰り返し用いて表示することができる。これにより、メニュー画面などをより高い自由度で構成することができるという効果がある。

この発明によれば、メニュー画面などに表示するボタンを３状態に分類し、それぞれに対して画像データを用意し、ユーザの入力などに応じて切り替えることにより、従来のＤＶＤビデオでは実現できなかった多彩な見栄えのメニューを実現できるという効果がある。

この発明では、グラフィクスプレーンに表示されるグラフィクスの表示制御を、表示制御命令を用いて記述しているため、インタラクティブな機能をグラフィクスプレーンに表示される画面を用いて実現することができるという効果がある。字幕やボタンのスクロール表示、移動表示といった簡単なアニメーションや、ユーザ入力に応じて画像の内容が変化する表現力のあるボタンを実現することができるという効果がある。

この発明によれば、記録済み大容量ディスクにおいて、動画を表示する動画プレーンと、字幕を表示する字幕プレーンと、メニュー画面などインタラクティブな機能を有する画面を表示するグラフィクスプレーンとがそれぞれ独立して設けられ、これらのプレーンが合成されて表示されるようにしている。そして、字幕プレーンとグラフィクスプレーンに表示するオブジェクトの形式として共通のグラフィクスオブジェクトを定義し、デコーダモデル、表示制御命令と動作モデルを定義した。これにより、動画に同期して字幕やボタンを表示することが可能となる効果がある。

この発明では、グラフィクスオブジェクトのデコーダモデルを定義し、実装のためにはオブジェクトバッファからプレーンバッファへの転送レートを制限する方法を示した。そして、同一のオブジェクトであっても、オブジェクトのプレーン上の位置、さらにオブジェクトの変形・移動などによって変化するプレーンバッファへの書き換えデータ量について、ウィンドウと称する矩形の更新領域を定義した。これにより、最小更新時間間隔を見積もることが可能となり、実装の実現性と再生互換性を高めることができるという効果がある。

この発明では、グラフィクスオブジェクトに音声データを含め、グラフィクスオブジェクトに含まれるボタン画像の表示と同時に音声データを再生するデコーダモデルを定義した。そのため、グラフィクスオブジェクトの表示に同期した音声データの再生を容易に行うことができるという効果がある。

この発明では、グラフィクスオブジェクトの表示制御命令に対して、音声データの再生を指示する命令を定義した。また、グラフィクスオブジェクトの表示制御命令に対して、グラフィクスオブジェクトに含まれる画像データに対して音声データを割り当てることができるようにしたため、任意の時刻での効果音などの音声データの再生や、効果音を伴う字幕やボタンの実現などが可能となる効果がある。

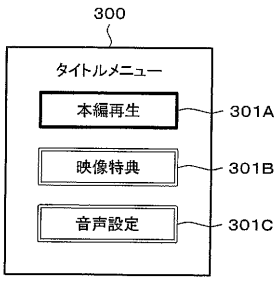
【符号の説明】

１０　動画プレーン
１１　字幕プレーン
１２　グラフィクスプレーン
２２　パレット

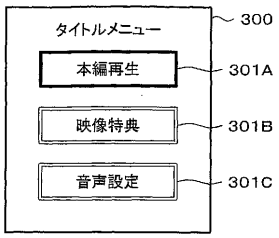
３０　ＢＤ仮想プレーヤ
３１　プレーヤコマンド
３２　共通パラメータ
４０　プレイバックコントロールプログラム
４１　メソッド
６０　メニュー画面
７０，７０’　シナリオ
７３Ａ～７３Ｍ　プレイリスト
１００　プレーヤデコーダ
１０４　コードバッファ
１０５　コンテンツバッファ
１０６　マルチメディアエンジン
１０９　バッファ
１１０　ＰＩＤフィルタ
１１６　グラフィックスデコーダＡ
１１７　グラフィックスデコーダＢ
１１８　オーディオデコーダ
１２０　ＭＰＥＧビデオデコーダ
１３２　字幕プレーン
１３３　グラフィクスプレーン
１３４　動画像プレーン
２２６　ＰＮＧデコーダバッファ
２２７　ＰＮＧデコーダ
２２８　オブジェクトバッファ
２２９　プレーンバッファ
２３１　オーディオミキサ

【図１】

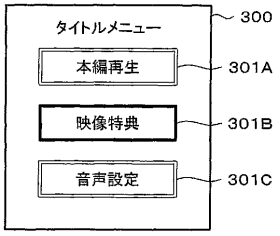
第１図



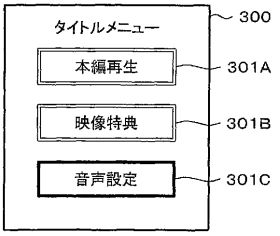
第２図Ａ



第２図Ｂ

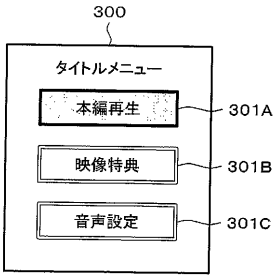


第２図Ｃ

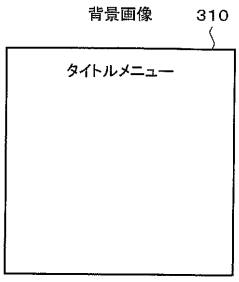


【図３】

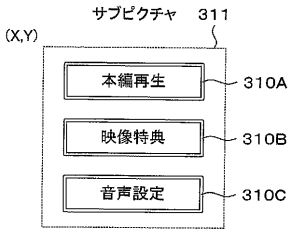
第３図



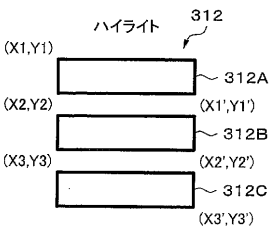
第４図Ａ



第４図Ｂ



第４図Ｃ



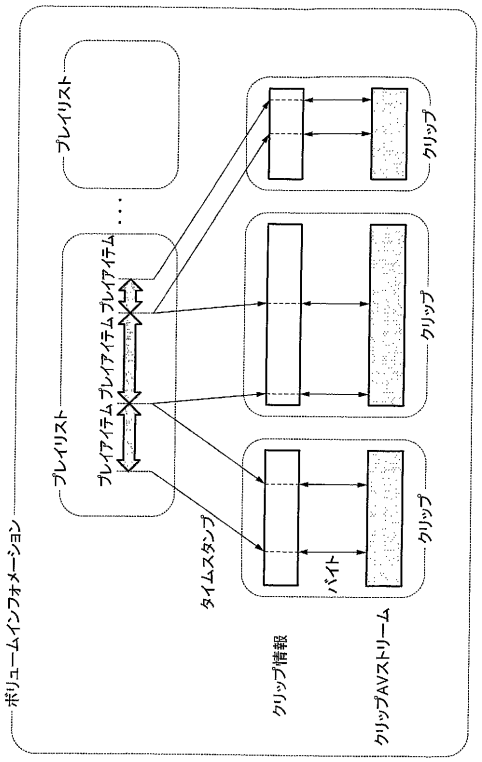
【図５】

第５図

サブピクチャ	1ピクセルあたり2ビットの情報で表現された一枚の画像データ 色(A0,B0,C0,D0)	
	表示を開始する座標(X,Y)	
	ハイライト	選択状態の色(A1,B1,C1,D1) 実行状態の色(A2,B2,C2,D2)
		色を変化させる領域の座標の組(＝ボタンの数＝3) (X1,Y1,X1',Y1') (X2,Y2,X2',Y2') (X3,Y3,X3',Y3')

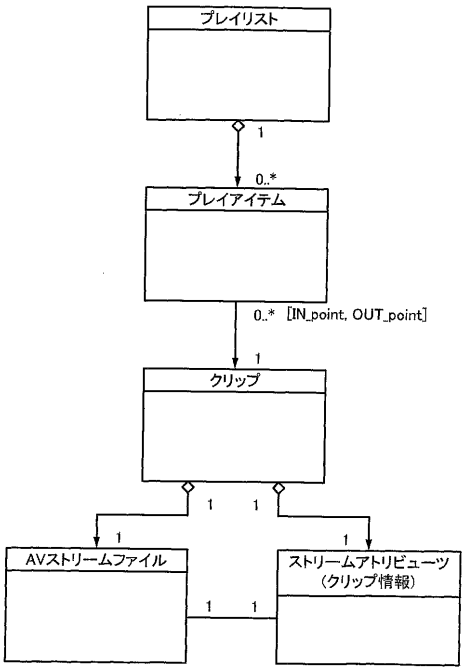
【図６】

第６図



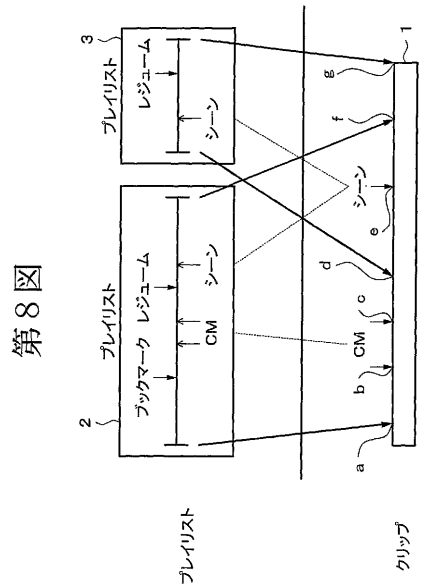
【図７】

第７図



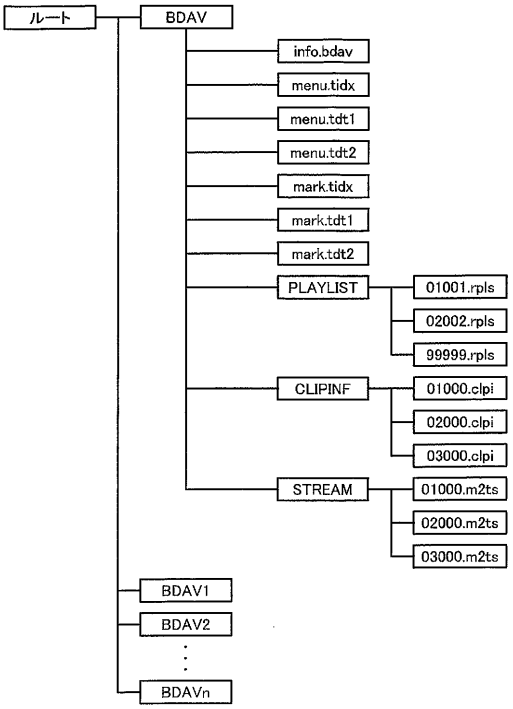
【図８】

第８図



【図９】

第９図



【図１０】

第１０図

シンタクス	データ長(ビット)	ニーモニック
info.bdav{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
TableOfPlayLists_start_address	32	unimsbf
MakersPrivateData_start_address	32	unimsbf
reserved_for_future_use	192	bslbf
UIAppInfoBDAV()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
TableOfPlayLists()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
MakersPrivateData()		
for(i=0;i<N3;i++){		
padding_word	16	bslbf
}		
}		

【図１１】

第１１図

シンタクス	データ長(ビット)	ニーモニック
UIAppInfoBDAV(){		
length	32	unimsbf
reserved_for_future_use	16	bslbf
BDAV_character_set	8	bslbf
reserved_for_word_align	6	bslbf
BDAV_protect_flag	1	bslbf
resume_valid_flag	1	bslbf
PIN	8*4	bslbf
resume_PlayList_file_name	8*10	bslbf
ref_to_menu_thumbnail_index	16	unimsbf
BDAV_name_length	8	unimsbf
BDAV_name	8*255	bslbf
}		

【図１２】

第１２図

シンタクス	データ長(ビット)	ニーモニック
TableOfPlayLists(){		
length	32	unimsbf
number_of_PlayLists	16	unimsbf
for(i=0;i<*number_of_PlayLists*;i++){		
PlayList_file_name	8*10	bslbf
}		
}		

【図１３】

第１３図

シンタクス	データ長(ビット)	ニーモニック
xxxxx.rpls/yyyyy.vpls{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
PlayList_start_address	32	unimsbf
PlayListMark_start_address	32	unimsbf
MakersPrivateData_start_address	32	unimsbf
reserved_for_future_use	160	bslbf
UIAppInfoPlayList()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
PlayList()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
PlayListMark()		
for(i=0;i<N3;i++){		
padding_word	16	bslbf
}		
MakersPrivateData()		
for(i=0;i<N4;i++){		
padding_word	16	bslbf
}		
}		

【図１４】

第１４図

シンタクス	データ長(ビット)	ニーモニック
UIAppInfoPlayList(){		
length	32	unimsbf
reserved_for_future_use	16	bslbf
PlayList_character_set	8	unimsbf
reserved_for_word_align	4	bslbf
playback_protect_flag	1	bslbf
write_protect_flag	1	bslbf
is_played_flag	1	bslbf
is_edited_flag	1	bslbf
time_zone	8	bslbf
reserved_for_word_align	8	bslbf
record_time_and_date	4*14	bslbf
PlayList_duration	4*6	bslbf
maker_ID	16	unimsbf
maker_model_code	16	unimsbf
channel_number	16	unimsbf
reserved_for_word_align	8	bslbf
channel_name_length	8	unimsbf
channel_name	8*20	bslbf
PlayList_name_length	8	unimsbf
PlayList_name	8*255	bslbf
PlayList_detail_length	16	unimsbf
PlayList_detail	8*1200	bslbf
}		

【図１５】

第１５図

シンタクス	データ長(ビット)	ニーモニック
PlayList(){		
length	32	uimsbf
reserved_for_word_align	12	bslbf
PL_CPI_type	4	bslbf
number_of_PlayItems	16	uimsbf
if(〈Virtual-PlayList〉&&PL_CPI_type==1){		
number_of_SubPlayItems	16	uimsbf
}else{		
reserved_for_word_align	16	bslbf
}		
for(PlayItem_id=0;PlayItem_id〈number_of_PlayItems;PlayItem_id++){		
PlayItem()		
}		
if(〈Virtual-PlayList〉&&CPI_type==1){		
for(i=0;i〈number_of_SubPlayItems;i++){		
SubPlayItem()		
}		
}		
}		

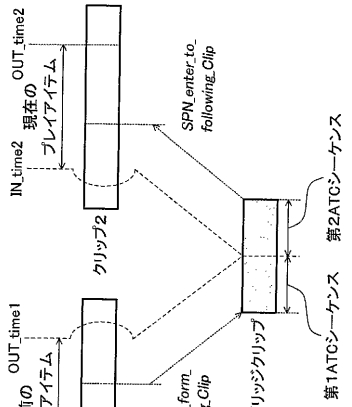
【図１６】

第１６図

シンタクス	データ長(ビット)	ニーモニック
PlayItem(){		
length	16	uimsbf
Clip_Infomation_file_name	8*5	bslbf
Clip_codec_identifier	8*4	bslbf
reserved_for_future_use	6	bslbf
connection_condition	2	bslbf
if(CPI_type==1){		
ref_to_STC_id	8	uimsbf
}else{		
reserved_for_word_align	8	bslbf
}		
IN_time	32	uimsbf
OUT_time	32	uimsbf
if(〈Vietual-PlayList〉&&connection_condition==5){		
BridgeSequenceInfo()		
}		
}		

【図１７】

第１７図



【図１８】

第１８図

シンタクス	データ長(ビット)	ニーモニック
PlayListMark(){		
length	32	uimsbf
number_of_PlayList_marks	16	uimsbf
for(i=0;i〈number_of_PlayList_marks;i++){		
mark_invalid_flag	1	uimsbf
mark_type	7	uimsbf
mark_name_length	8	uimsbf
maker_ID	16	uimsbf
ref_to_PlayItem_id	16	uimsbf
mark_time_stamp	32	uimsbf
entry_ES_PID	16	uimsbf
if(mark_type==0x01||mark_type==0x02){		
ref_to_menu_thumbnail_index	16	uimsbf
}else{		
ref_to_menu_thumbnail_index	16	uimsbf
}		
duration	32	uimsbf
makers_infomation	32	bslbf
mark_name	8*24	bslbf
}		
}		

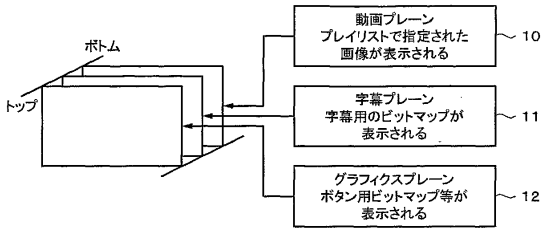
【図１９】

第１９図

シンタクス	データ長(ビット)	ニーモニック
zzzzz.clpi{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
SequenceInfo_start_address	32	unimsbf
ProgramInfo_start_address	32	unimsbf
CPI_start_address	32	unimsbf
ClipMark_start_address	32	unimsbf
MakersPrivateData_start_address	32	unimsbf
reserved_for_future_use	96	bslbf
ClipInfo()		
for(i=0;<N1;i++){		
padding_word	16	bslbf
}		
SequenceInfo()		
for(i=0;<N2;i++){		
padding_word	16	bslbf
}		
ProgramInfo()		
for(i=0;<N3;i++){		
padding_word	16	bslbf
}		
CPI()		
for(i=0;<N4;i++){		
padding_word	16	bslbf
}		
ClipMark()		
for(i=0;<N5;i++){		
padding_word	16	bslbf
}		
MakersPrivateData()		
for(i=0;<N6;i++){		
padding_word	16	bslbf
}		
}		

【図２０】

第２０図



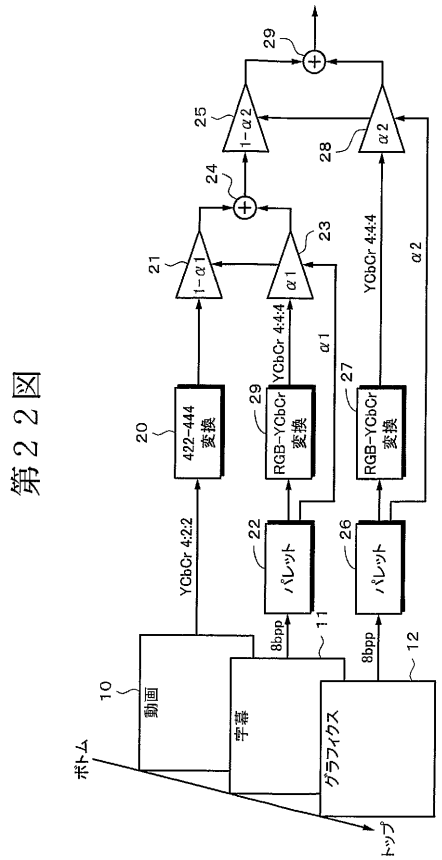
【図２１】

第２１図

項目	規定内容
動画プレーン	1920x1080x16ビット, YCbCr(4:2:2),各8ビット
字幕プレーン	1920x1080x8ビット, 8ビットカラーマップアドレス(パレット) +256段階のアルファ ブレンディング
グラフィックスプレーン	1920x1080x8ビット, 8ビットカラーマップアドレス(パレット)+ 256段階のアルファ ブレンディング

【図２２】

第２２図



【図２３】

第２３図

入力	入力アドレス 8ビット
出力	出力データ 8ビットx4、(R, G, B, α)出力

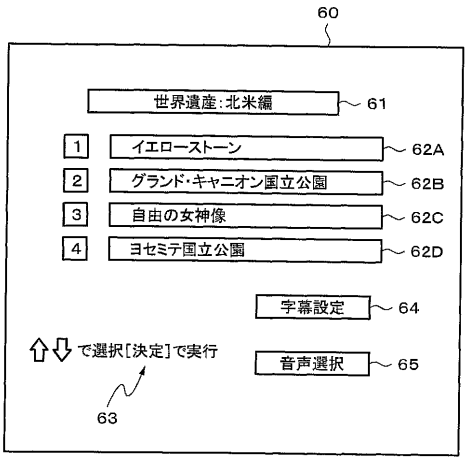
【図２４】

第２４図

	三原色の値			不透明度
カラーインデックス値	R	G	B	α
0x00	0	0	0	0
0x01	10	100	30	0.5
⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮
0xFF	200	255	100	0.8

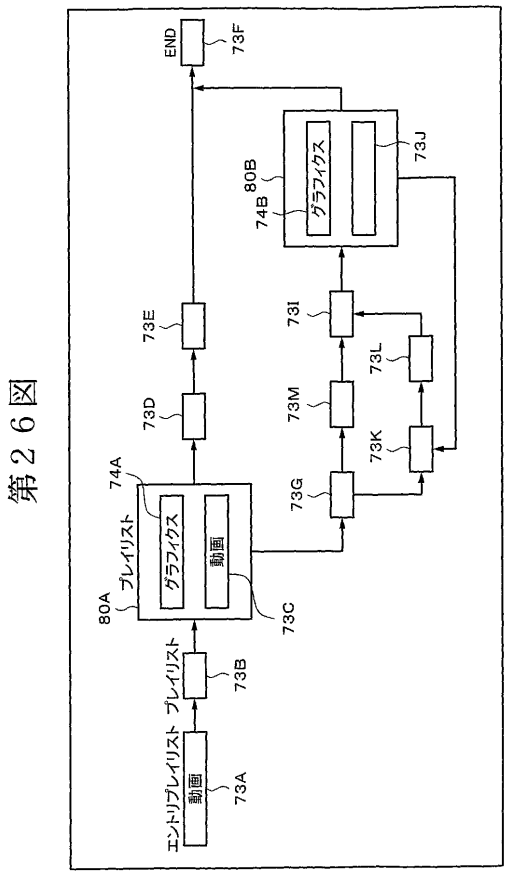
【図２５】

第２５図



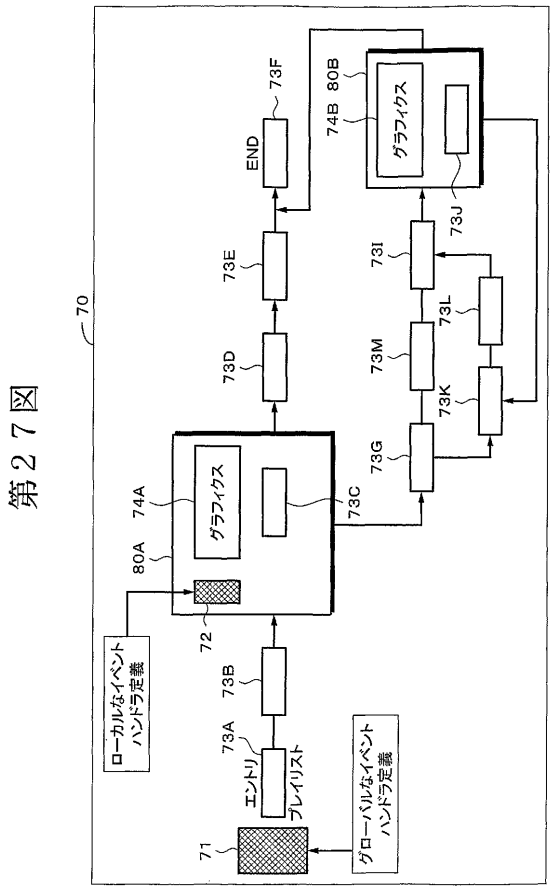
【図２６】

第２６図



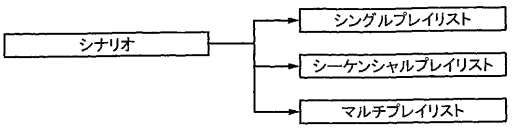
【図２７】

第２７図

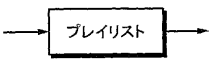


【図２８】

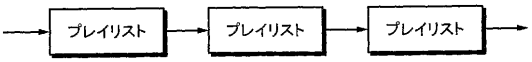
第２８図



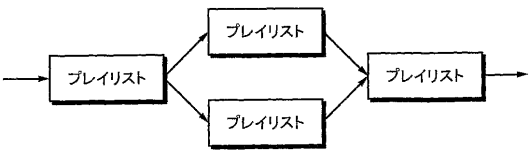
第２９図Ａ



第２９図Ｂ

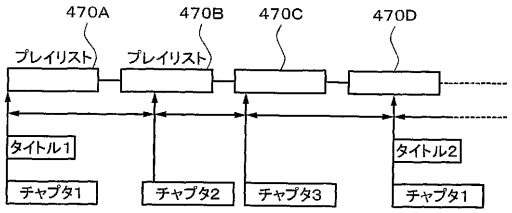


第２９図Ｃ



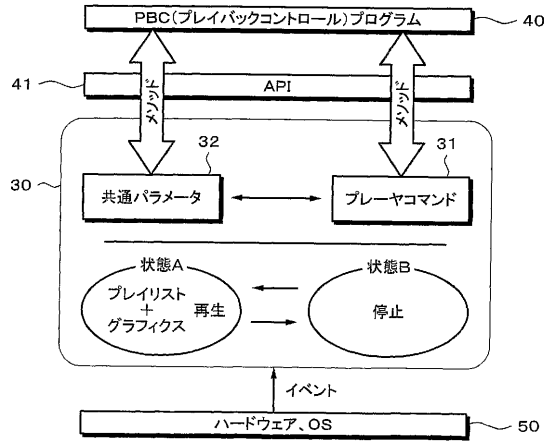
【図３０】

第３０図



【図３１】

第３１図



第３２図

第３２図A
第３２図B
第３２図C

第３２図A

イベント名	説明
TimeFired	カウントダウンタイマーが0になった カウントアップタイマーが指定の値になった
PlayStopped	再生が停止した
PlayStilled	再生が一時停止した
StillReleased	一時停止が解除された
PlayPaused	ユーザにより一時停止された
PauseReleased	ポーズが解除された

第３２図B

イベント名	説明
PlayStarted	再生が開始された
PlayRepeated	繰り返し再生時の先頭を検出した
SPDisplayStatusChanged	SPストリームの表示・非表示状態が変わった
SelectedAudioChanged	再生するオーディオストリームが変わった
VideoStopped	再生するビデオストリームが変わった
ScenarioStarted	シナリオの先頭を検出
ScenarioEnded	シナリオの終わりを検出
PlayListStarted	プレイリストの先頭を検出
PlayListEnded	プレイリストの終わりを検出
PlayItemStarted	プレイアイテムの先頭を検出
PlayItemEnded	プレイアイテムの終わりを検出

第３２図C

イベント名	説明
MarkEncountered	再生中にマークを検出 グラフィックス画面を表示する際に利用 共通パラメータ列に、種類と番号が格納される
ButtonPressed	画面上のボタンが押された 共通パラメータ列に、押されたボタンのIDが格納される
ValidPeriodStarted	有効期間が開始した リンクの選択有効期間を設定する際に利用
ValidPeriodEnded	有効期間が終了した リンクの強制実行の時に利用
KeyPressed	リモコンのキーが押された どのキーが押されたかは、イベントハンドラ内のスイッチなどで識別する

第３３図Ａ

第３３図

第３３図Ａ
第３３図Ｂ
第３３図Ｃ
第３３図Ｄ
第３３図Ｅ
第３３図Ｆ
第３３図Ｇ
第３３図Ｈ

メソッド	備考
再生開始位置指定	
LinkPlayList(playListNumber)	playListNumberで指定されたPlayListの再生を開始する。
LinkPlayItem(playListNumber,playItemNumber)	指定のPlayListの指定のPlayItemから再生を開始する。 playItemNumberはPlayItem_idであり、0から始まる。 PlayListの先頭から再生するのであれば、playItemNumberは0になる。

第３３図Ｂ

メソッド	備考
再生開始位置指定	
Link(position)(object) position=("prev"|"next"|"top"|"Parent"|"tail") object=(PlayList | PlayItem | Chapter)	シナリオ内移動。 現在再生中の箇所から前後のPlayList,PlayItem,Chapterに移動する。
Exit	シナリオの再生を停止する。標準レジスタの値は保持されない。
RSM	前回再生を停止した箇所から再生を再開する。 保存されているresume情報を呼び出し、レジスタにセットして再生を開始する。

第３３図Ｃ

Player状態の取得	
getMenuDescriptionLanguage()	メニュー表示の際の言語を取得
getScenarioNumber()	再生中のシナリオ番号を得る。
getPlayListNumber()	再生中のプレイリスト番号を得る。
getChapterNumber()	再生中のチャプター番号を得る。
getPlayerSupport()	プレーヤのバージョン、有する機能を取得。

第３３図Ｄ

Videoストリームに関するもの	
getVideoStreamAvailability()	指定のビデオストリームが含まれているかを得る。
setVideoStreamNumber()	デコードするビデオストリームを指定する。
getVideoStreamNumber()	選択中のビデオストリームの番号を得る。
getVideoStreamAttribute()	ビデオストリームの属性を得る。 （符号化方式、解像度、アスペクト日、4:3時のディスプレイモード、ClosedCaption）
setAngleNumber()	アングル番号を指定する。
getAngleNumber()	選択中のアングル番号を得る。
getMaxVideoStreams()	選択可能なビデオストリームの数を得る。 getVideoStreamAvailability()があれば十分か？

JP 4442564 B2 2010.3.31

第３３図Ｅ

Audioストリームに関するもの	
getAudioStreamAvailability()	指定のオーディオストリームが含まれているかを得る。
getAudioStreamLanguage()	指定のオーディオストリームの言語についての情報を得る。
getAudioStreamStatus()	再生するオーディオストリームを指定する。
setAudioStreamStatus()	再生しているオーディオストリームの番号を得る。
getAudioStreamAttribute()	オーディオストリームの属性を得る。(符号化方式、ch数、Q、fs)

第３３図Ｆ

サブピクチャストリーム	
getSPStreamAvailability()	指定のSPストリームが含まれているかを得る。
getSPStreamLanguage()	指定のSPストリームの言語を得る。
getSPDisplayStatus()	SPの表示状態(表示または非表示)を得る。
setSPDisplayStatus()	SPの表示状態(表示または非表示)を設定する。
getSpStreamAttribute()	SPの属性を得る。解像度別、4:3 またはワイド用など。

第３３図Ｇ

レジスタリード/ライト	
clearReg()	全レジスタの初期化する。
setReg()	レジスタへの値のセット
getReg()	レジスタから値の読み出し

第３３図Ｈ

タイマ	
sleep()	指定されたミリ秒間、処理を停止する。
setTimeout()	指定されたミリ秒後に関数や処理を実行する。
setInterval()	指定されたミリ秒ごとに処理を実行する。
clearTimer()	指定された登録タイマIDの処理を中止する。
pauseTimer()	指定した登録タイマIDのタイマを一時停止する。
resumeTimer()	指定した登録タイマIDのタイマを一時停止から再開する。
その他	
playSoundEffect(sound_id)	選択された効果音を再生。ボタンコマンドで使用できる。

JP 4442564 B2 2010.3.31

第３４図

第３４図Ａ
第３４図Ｂ
第３４図Ｃ
第３４図Ｄ
第３４図Ｅ
第３４図Ｆ
第３４図Ｇ
第３４図Ｈ
第３４図Ｉ
第３４図Ｊ

第３４図Ａ

オブジェクト	メソッド	備考
bdp		
プレーヤー動作		
	playScenario(scenarioNumber,[scenarioTime])	scenarioNumberで指定されるシナリオを再生する。 ScenarioNumberは、実際にはシナリオ構成を記述した ファイル等を指すURIになる。 scenarioTimeはオプションであるが、 シナリオ内の経過時刻を指定する。
	playPlayList(playListNumber)	playListNumberで指定されたPlayListを再生する。
	playChapterMark(playListNumber,chapterNumber)	playListNumberで決まるPlayList中の chapterNumberで指定されるあるチャプターから再生する。

第３４図Ｂ

	playPlayItem(playListNumber,playItemNumber)	playListNumberとplayItemiNumberで決まる PlayItemから再生する。 playItemNumberはPlayItem_idであり、0から始まる。 PlayListの先頭から再生するのであれば、 playItemNumberは0になる。
	play(position)(object) position = (“prev”|“next”|“top”|“goUp”|“tail”) object = (PlayList|PlayItem|Chapter)	シナリオ内移動。現在再生中の箇所から前後の PlayList,PlayItem等に移動する。
	stop()	シナリオの再生を停止する。 標準レジスタの値は保持されない。
	resume()	前回再生を停止した箇所から再生を再開する。
	playSoundEffect()	選択された効果音を再生

第３４図Ｃ

プレーヤー状態		
	getMenuDescriptionLanguage()	メニュー表示の際の言語を取得
	getScenarioNumber()	再生中のシナリオ番号を得る。
	getPlayListNumber()	再生中のプレイリスト番号を得る。
	getChapterNumber()	再生中のチャプター番号を得る。
	getPlayerSupport()	プレイヤのバージョン、有する機能を取得

第３４図Ｄ

ビデオストリーム		
	setVideoStreamNumber()	デコードするビデオストリームを指定する。
	getVideoStreamNumber()	選択中のビデオストリーム番号を得る。
	getVideoStreamStatus()	ビデオストリームの状態を得る。
	getVideoStreamAttr()	ビデオストリームの属性を得る。 （符号化方式、解像度、アスペクト比、 4:3時のディスプレイモード、クローズドキャプション）
	setAngleNumber()	アングル番号を指定する。
	getAngleNumber()	選択中のアングル番号を得る。
	getMaxVideoStreams()	ビデオストリームの最大数を得る。

JP 4442564 B2 2010.3.31

第３４図Ｇ

レジスタリード/ライト		
	clearReg()	プレイヤが内蔵しているメモリ領域（レジスタ）に対する操作。 全レジスタを初期化する。
	setReg()	レジスタへの値のセット
	getReg()	レジスタから値の読み出し

第３４図Ｈ

タイマ		
	sleep()	指定されたミリ秒間、処理を停止する。
	setTimeout()	指定されたミリ秒後に関数や処理を実行する。
	setInterval()	指定されたミリ秒ごとに処理を実行する。
	clearTimer()	指定された登録タイマIDの処理を中止する。
	pauseTimer()	指定した登録タイマIDのタイマを一時停止する。
	resumeTimer()	指定した登録タイマIDのタイマを一時停止から再開する。

第３４図Ｅ

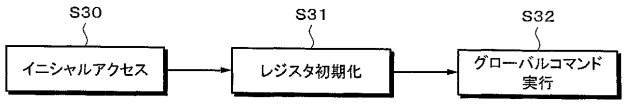
オーディオストリーム		
	getAudioStreamAvailability()	指定のオーディオストリームが含まれているかを得る。
	getAudioStreamLanguage()	指定のオーディオストリームの言語についての情報を得る。
	getAudioStreamStatus()	オーディオストリームの状態（再生または非再生）をセットする。
	setAudioStreamStatus()	オーディオストリームの状態（再生または非再生）をセットする。
	getAudioStreamAttribute()	オーディオストリームの属性を得る。（符号化方式、ch数、Q、fs）

第３４図Ｆ

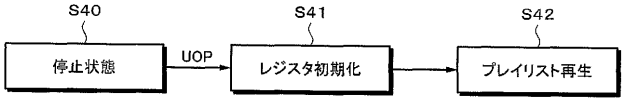
サブピクチャストリーム		
	getSPStreamAvailability()	指定のSPストリームが含まれているかを得る。
	getSPStreamLanguage()	指定のSPストリームの言語を得る。
	getSPDisplayStatus()	SPの表示状態を得る。
	setSPDisplayStatus()	SPの表示状態を設定する。SPの表示のON/OFF
	getSpStreamAttribute()	SPの属性を得る。4:3またはワイド用など。

JP 4442564 B2 2010.3.31

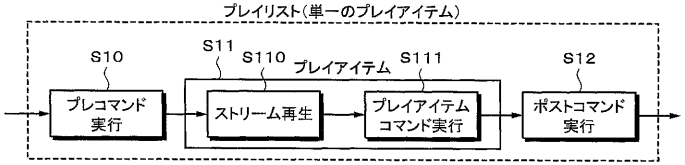
第３５図Ａ



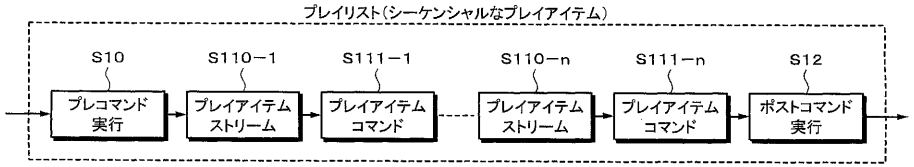
第３５図Ｂ



第３６図Ａ



第３６図Ｂ



第３４図Ｉ

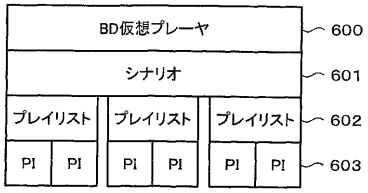
キー入力		
	getPressedKey()	入力されたキーの種類を取得
グラフィックス関連		
	loadGraphics(htmlfile,ID)	htmlfileで指定されるファイルを読み込み、 グラフィクスプレーンに展開する。 ただし、表示はしない。この画像は、IDで参照される。
	showGraphics(ID)	IDで指定されるグラフィクス画像を表示する。 すでにloadGraphics()でプレーン上に展開されていることが必要。
	hideGraphics(ID)	IDで指定されるグラフィクス画像を消す。

第３４図Ｊ

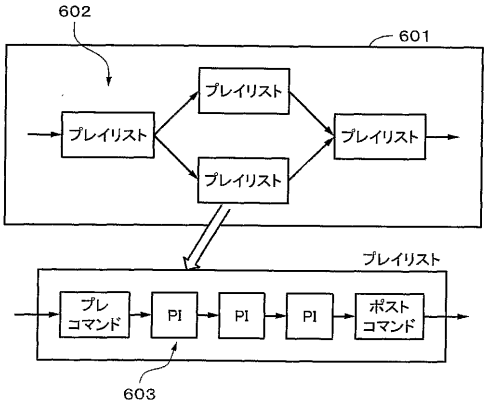
その他		
	random(input Number num)	1からnumまでの乱数を発生する。 ECMAスクリプトのMathオブジェクトを使用せず、独自に定義する。
	catchEvent(eventname,eventhandler)	eventnameで指定されたイベントが発生した際、 eventhandlerで指定した関数を実行することを登録する。

JP 4442564 B2 2010.3.31

第３７図Ａ

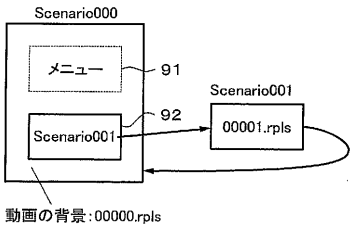


第３７図Ｂ



【図３８】

第３８図



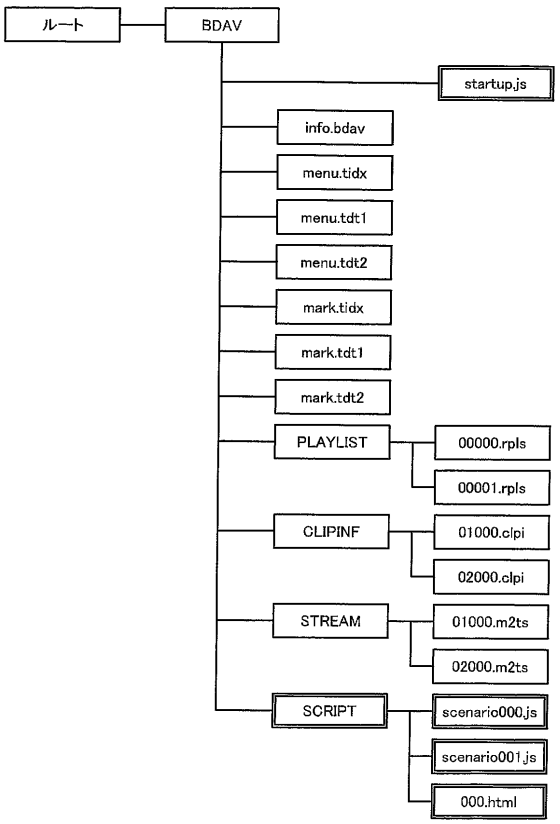
【図３９】

第３９図

ファイル名	内容
startup.js	ディスク挿入時に最初に実行されるスクリプトファイル
scenario000.js	シナリオ一覧画面(シナリオ000)の構成情報を記述したスクリプトファイル
000.html	シナリオ一覧画面のレイアウトを記述したhtmlファイル
00000.rpls	シナリオ一覧画面表示中に背景として再生されるプレイリストファイル
scenario001.js	シナリオ001の構成情報を記述したスクリプトファイル
00001.rpls	シナリオ001で再生されるプレイリストの情報が記述されたプレイリストファイル

【図４０】

第４０図



【図４１】

第４１図

```
==========
startup.js
==========
function makeArray(n){
        this.length=n;
        for(i=0;i<n;i++){
                this[i]=null;
        }
}
//シナリオの数と名前の定義
var scenario=new makeArray(2);
scenario[0]="scenario000";
scenario[1]="scenario001";


//最初のシナリオ実行
bdp.playScenario("scenario000");
```

【図４２】

第４２図

```
==========
scenario000.js
==========
function UOPControl(){
var keyID=getPressedKey();
    switch(keyID){
        case menu:
            bdp.playScenario("scenario000.js");  //最初に戻る。
            break;
        default;
            break;
    }
}
function playListEnded(){
    bdp.playScenario("scenario000.js");  //最後まで再生し終わったら、また繰り返す。
    return();
}
function MarkEncountered(){   //グラフィクスを読み込むタイミングを示したマーク
    bdp.loadGraphics("000.html","id_1");
    return();
}
function ValidPeriodStarted(f){   //グラフィクスを表示するタイミングを示すマーク
    if(f =="id_1"){
        bdp.showGraphics("id_1");
    }
    return();
}
function ValidPeriodEnded(f){   //グラフィクス表示を停止するタイミングを示すマーク
    if(f =="id_1"){
        bdp.hideGraphics("id_1");
    }
    return();
}
bdp.catchEvent("onKeyPressed","UOPControl()");
bdp.catchEvent("onPlayListEnded","PlayListEnded()");
bdp.catchEvent("onMarkEncountered","MarkEncountered()");
bdp.catchEvent("onValidPeriodStarted","ValidPeriodStarted()");
bdp.catchEvent("onValidPeriodEnded","ValidPeriodEnded()");

bdp.playPlayList("00000.rpls");
```

【図４３】

第４３図

```
==========
000.html
==========
<html>
<head>
<style type="text/css">
<![CDATA[
img#menu{position:absolute;top:200px;left:800px;width:200px;height:50px}
img#scenario001{position:absolute;top:700px;left:700px;width:400px;height:100px}
]]>
</style>
<script type="text/javascript">
function onMoverhandler(f){
        switch(f){
                case scenario001:
                        f.src="201.png";
                        break;
                default;
                        break;
        }
}
function onMouthandler(f){
        switch(f){
                case scenario001:
                        f.src="200.png";
                        break;
                default;
                        break;
        }
}
function onMclickhandler(f){
        switch(f){
                case scenario001:
                        f.src="202.png";
                        bdp.playScenario("scenario001.js");
                        break;
                default;
                        break;
        }
}
</script>
</head>
<body>
        <img src="100.png"  id="menu"/>
        <img src="200.png"  id="scenario001"
                onMouseover="onMoverhandler(this)"
                onMouseout="onMouthandler(this)"
                onclick="onMclickhandler(this)"/>
</body>
</html>
```

【図４４】

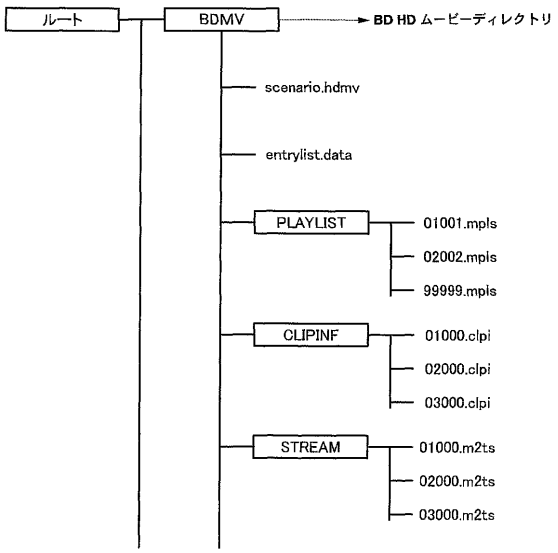
第４４図

```
==========
scenario001.js
==========
function UOPControl(){
        var keyID=getPressedKey();
        switch(keyID){
                case menu:   //シナリオ001再生中にメニューキーが押された
                        bdp.playScenario("scenario000.js");
                        break;
                default;
                        break;
        }
}
function playListEnded(){//再生が終了したので、シナリオ000(メニュー画面)に戻る。
        bdp.playScenario("scenario000.js");
        return;
}
bdp.catchEvent("onPlayListEnded","PlayListEnded()");
bdp.catchEvent("onKeyPressed","UOPControl()");


bdp.playPlayList("00001.rpls");
==========
```

【図４５】

第４５図



【図４６】

第４６図

シンタクス	データ長(ビット)	ニーモニック
scenario.hdmv{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
scenario_start_address	32	
reserved_for_future_use	224	bslbf
Autoplay()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
Scenario()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
}		

【図４７】

第４７図

シンタクス	データ長(ビット)	ニーモニック
Autoplay(){		
length	32	uimsbf
reserved	16	
number_of_commands	16	
for(i=0;i<number_of_commands;i++){		
command(i)	32	uimsbf
}		
}		

【図４８】

第４８図

シンタクス	データ長(ビット)	ニーモニック
Scenario(){		
length	32	
flags	32	
number_of_PlayLists	16	
for(i=0;i<Number_of_PlayLists;i++){		
Pre_Command_start_id	32	
Post_Command_start_id	32	
number_of_Pre_Commands	32	
number_of_Post_Commands	32	
reserved	32	
number_of_PlayItems	32	
for(PlayItem_id=0;PlayItem_id<number_of_PlayItems;PlayItem_id++){		
PI_Command_start_id	32	
number_of_PI_Commands	32	
}		
reserved		
// Command table for each PlayList		
number_of_PL_Commands	16	
for(i=0;i<number_of_PL_Commands;i++){		
PL_Command()	32	
}		
}		
}		

【図４９】

第４９図

シンタクス	データ長(ビット)	ニーモニック
entrylist.data{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
ScenarioEntry_start_address	32	uimsbf
reserved_for_future_use	224	bslbf
AppInfo()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
ScenarioEntry()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
}		

【図５０】

第５０図

シンタクス	データ長(ビット)	ニーモニック
AppInfo(){		
length	32	uimsbf
reserved_for_future_use	16	bslbf
HDMV_name_character_set	8	bslbf
reserved_for_word_align	7	bslbf
PIN_valid_flag	1	bslbf
PIN	8*4	bslbf
// UOP_mask_table() // For directory	64	
HDMV_name_length	8	uimsbf
HDMV_name	8*255	bslbf
}		

【図５１】

第５１図

シンタクス	データ長(ビット)	ニーモニック
ScenarioEntry(){		
length	32	unimsbf
name_character_set	8	bslbf
// Entry PL for the Top Menu		
Top Menu PL(){		
flags	32	bslbf
TopMenu_ref_to_PlayList_file_name	8*10	bslbf
TopMenu_ref_to_PlayItem_id	16	unimsbf
TopMenu_name_length	8	unimsbf
TopMenu_name	8*255	bslbf
}		
// Title Entries		
number_of_Titles	16	unimsbf
for(unit title_number=0;title_number<Number_of_Titles;title_number++){		
flags	32	bslbf
Title_ref_to_PlayList_file_name	8*10	bslbf
Title_ref_to_PlayItem_id	16	unimsbf
Title_name_length	8	unimsbf
Title_name	8*255	bslbf
}		
// Stream Setup Menu for each PL		
number_of_PlayLists	16	unimsbf
for(i=0;i<Number of PlayLists;i++){		
SSMenu_flags	32	bslbf
SSMenu_ref_to_PlayList_file_name	8*10	bslbf
SSMenu_ref_to_PlayItem_id	16	unimsbf
}		
}		

【図５２】

第５２図

シンタクス	データ長(ビット)	ニーモニック
xxxxx.mpls{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
PlayList_start_address	32	unimsbf
PlayListMark_start_address	32	unimsbf
reserved_for_future_use	192	bslbf
PLControlInfo()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
PlayList()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
PlayListMark()		
for(i=0;i<N3;i++){		
padding_word	16	bslbf
}		
}		

【図５３】

第５３図

シンタクス	データ長(ビット)	ニーモニック
PLControlInfo(){		
length	32	unimsbf
reserved_for_future_use	8	bslbf
PlayList_character_set	8	unimsbf
reserved_for_future_use	8	
PL_playback_type	8	
if(PL_playback_type==0x2 ||		
PL_playback_type==0x3){		
playback_count	16	
}else{		
reserved_for_word_align	16	
}		
PL_UOP_mask_table() // For PlayList	64	
reserved_for_word_align	8	
PL_random_access_mode	8	
reserved_for_word_align	8	bslbf
PlayList_duration	4*6	bslbf
PlayList_name_length	8	unimsbf
PlayList_name	8*255	bslbf
PlayList_detail_length	16	unimsbf
PlayList_detail	8*1200	bslbf
}		

【図５４】

第５４図

PL_playback_type	意味
0x0	リザーブ領域
0x1	プレイアイテムをシーケンシャル再生する(通常再生)
0x2	プレイアイテムをランダム再生する
0x3	プレイアイテムをシャッフル再生する

【図５５】

第５５図

PL_random_access_mode	意味
0x0	飛び込み再生・変速再生許可
0x1	飛び込み再生・変速再生禁止

【図５６】

第５６図

シンタクス	データ長(ビット)	ニーモニック
PlayList(){		
length	32	unimsbf
number_of_PlayItems	16	unimsbf
number_of_SubPlayItems	16	unimsbf
for(PlayItem_id=0;PlayItem_id<number_of_PlayItems;PlayItem_id++){		
PlayItem()		
}		
for(SubPlayItem_id=0;SubPlayItem_id<number_of_SubPlayItems;SubPlayItem_id++){		
SubPlayItem()		
}		
}		

【図５７】

第５７図

シンタクス	データ長(ビット)	ニーモニック
PlayItem(){		
length	16	uimsbf
reserved_for_word_align	8	bslbf
Clip_Information_file_name	8*5	bslbf
Clip_codec_identifier	8*4	bslbf
reserved_for_future_use	7	bslbf
is_multi_angle	1	bslbf
reserved_for_future_use	4	bslbf
connection_condition	4	uimsbf
ref_to_STC_id	8	uimsbf
IN_time	32	uimsbf
OUT_time	32	uimsbf
PI_UOP_mask_table()	64	bslbf
PID_filter()		
reserved_for_word_align	8	bslbf
PI_random_access_mode	8	uimsbf
reserved_for_word_align	8	bslbf
still_mode	8	uimsbf
if(still_mode==0x1){		
still_time	16	uimsbf
}else{		
reserved_for_word_align	16	bslbf
}		
// Angle		
if(is_multi_angle){		
number_of_angles	8	uimsbf
is_seamless_angle_change	8	uimsbf
for(angle_id=1;angle_id<number_of_angles;angle_id++){		
Clip_Information_file_name	8*5	bslbf
ref_to_STC_id	8	uimsbf
IN_time	32	uimsbf
OUT_time	32	uimsbf
}		
}		
}		

【図５８】

第５８図

PI_random_access_mode	意味
0x0	飛び込み再生・変速再生許可
0x1	飛び込み再生・変速再生禁止

【図５９】

第５９図

still_mode	意味
0x0	Stillなし。
0x1	有限時間のStill。時間は次のstill_timeで指定。
0x2	無限時間のStill。ユーザが解除するまでStillを続ける。
0x3-0xf	予約

【図６０】

第６０図

is_seamless_angle_change	意味
0x0	ノンシームレス切り替えのアングル
0x1	シームレス切り替え可能なアングル

【図６１】

第６１図

シンタクス	データ長(ビット)	ニーモニック
SubPlayItem(){		
length	16	unimsbf
Clip_Information_file_name	8*5	bslbf
Clip_codec_identifier	8*4	bslbf
reserved_for_future_use	7	bslbf
is_repeat_flag	1	bslbf
SubPlayItem_type	8	bslbf
ref_to_STC_id	8	unimsbf
SubPlayItem_IN_time	32	unimsbf
SubPlayItem_OUT_time	32	unimsbf
if(is_repeat_flag==0){		
sync_PlayItem_id	16	unimsbf
sync_start_PTS_of_PlayItem	32	unimsbf
}else{		
reserved_for_word_align	16	
reserved_for_word_align	32	
}		
}		

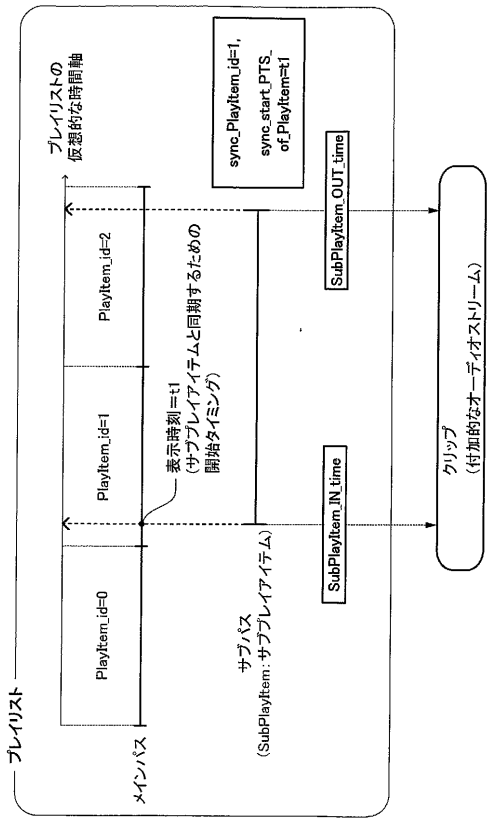
【図６２】

第６２図

is_repeat_flag	意味
0	メインパスと同期再生する。
1	メインパスと同期再生しない。再生を繰り返す。

【図６３】

第６３図



【図６４】

第６４図

シンタクス	データ長(ビット)	ニーモニック
zzzzz.clpi{		
type_indicator	8*4	bslbf
version_number	8*4	bslbf
SequenceInfo_start_address	32	uimsbf
ProgramInfo_start_address	32	uimsbf
CPI_start_address	32	uimsbf
ClipMark_start_address	32	uimsbf
reserved_for_future_use	128	bslbf
ClipInfo()		
for(i=0;i<N1;i++){		
padding_word	16	bslbf
}		
SequenceInfo()		
for(i=0;i<N2;i++){		
padding_word	16	bslbf
}		
ProgramInfo()		
for(i=0;i<N3;i++){		
padding_word	16	bslbf
}		
CPI()		
for(i=0;i<N4;i++){		
padding_word	16	bslbf
}		
ClipMark()		
for(i=0;i<N5;i++){		
padding_word	16	bslbf
}		
}		

【図６５】

第６５図

シンタクス	データ長(ビット)	ニーモニック
ClipInfo(){		
length	32	unimsbf
reserved	8	bslbf
application_type	8	unimsbf
Clip_stream_type	8	unimsbf
reserved	40	unimsbf
TS_recording_rate	32	unimsbf
num_of_source_packets	32	unimsbf
BD_system_use	1024	bslbf
TS_type_info_block()		
}		

【図６６】

第６６図

application_type	意味
0	対応するm2tsファイルが、 HDMVトランスポートストリームのルールに従っていない。
1	対応するm2tsファイルが、 HDMVトランスポートストリームのルールに従っている。 (通常のHDMVストリーム)
2	対応するm2tsファイルが、 オーディオ再生に同期する静止画用の HDMVトランスポートストリームのルールに従っている。 (タイムベーススライドショー)
3	対応するm2tsファイルが、 オーディオとは非同期に再生される静止画用の HDMVトランスポートストリームのルールに従っている。 (ブラウザブルスライドショー)

【図６７】

第６７図

シンタクス	データ長(ビット)	ニーモニック
SequenceInfo(){		
length	32	unimsbf
reserved_for_word_align	8	bslbf
num_of_ATC_sequences	8	unimsbf
for(atc_id=0;atc_id<num_of_ATC_sequences;atc_id++){		
SPN_ATC_start[atc_id]	32	unimsbf
num_of_STC_sequences[atc_id]	8	unimsbf
offset_STC_id[atc_id]	8	unimsbf
for(stc_id=offset_STC_id[atc_id];stc_id<(num_of_STC_sequences[atc_id]+offset_STC_id[atc_id]);stc_id++){		
PCR_PID[atc_id][stc_id]	16	unimsbf
SPN_STC_start[atc_id][stc_id]	32	unimsbf
presentation_start_time[atc_id][stc_id]	32	unimsbf
presentation_end_time[atc_id][stc_id]	32	unimsbf
}		
}		
}		

【図６８】

第６８図

シンタクス	データ長(ビット)	ニーモニック
ProgramInfo(){		
length	32	unimsbf
reserved_for_word_align	8	bslbf
num_of_program_sequences	8	unimsbf
for(i=0;i<num_of_program_sequences;i++){		
SPN_program_sequence_start[i]	32	unimsbf
program_map_PID[i]	16	bslbf
num_of_streams_in_ps[i]	8	unimsbf
num_of_groups[i]	8	unimsbf
for(stream_index=0;stream_index<num_of_streams_in_ps[i];stream_index++){		
stream_PID[i][stream_index]	16	unimsbf
StreamCodingInfo(i,stream_index)		
}		
}		
}		

【図６９】

第６９図

シンタクス	データ長(ビット)	ニーモニック
StreamCodingInfo(i,stream_index){		
length	8	bslbf
stream_coding_type	8	unimsbf
if(stream_coding_type==0x02){		
video_format	4	unimsbf
frame_rate	4	unimsbf
aspect_ratio	4	unimsbf
reserved_for_word_align	2	bslbf
cc_flag	1	unimsbf
reserved_for_word_align	1	bslbf
}else if(stream_coding_type==0x80//stream_coding_type==0x81//stream_coding_type==0x82){		
audio_presentation_type	4	unimsbf
sampling_frequency	4	unimsbf
language_code	16	bslbf
reserved_for_word_align	8	bslbf
}else if(stream_coding_type==0x90){		
language_code	16	bslbf
T.B.D		
}else if(stream_coding_type==0xA0){		
language_code	16	bslbf
T.B.D		
}		
}		

【図７０】

第７０図

シンタクス	データ長(ビット)	ニーモニック
CPI(){		
length	32	unimsbf
reserved_for_word_align	12	bslbf
CPI_type	4	unimsbf
EP_map_for_BDMV()		
}		

【図７１】

第７１図

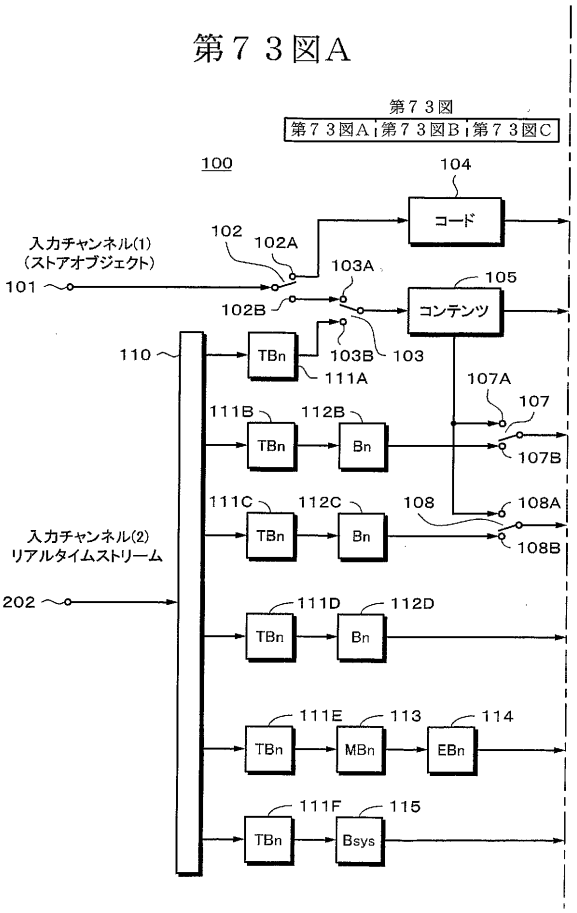
CPI_type	意味
0	将来の使用のために予約
1	EP_mapタイプ
2	TU_mapタイプ
3-7	将来の使用のために予約
8	BDMV用のEP_mapタイプ
9-15	将来の使用のために予約

【図７２】

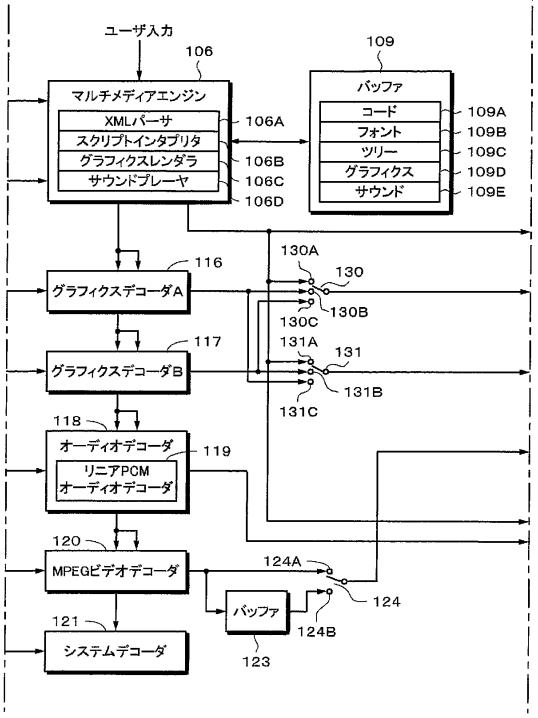
第７２図

シンタクス	データ長(ビット)	ニーモニック
EP_map_for_one_stream_PID(EP_stream_type,Nc,Nf){		
EP_fine_table_start_address	32	unimsbf
for(i=0;i<Nc;i++){		
ref_to_EP_fine_id[i]	18	unimsbf
PTS_EP_coarse[i]	14	unimsbf
SPN_EP_coarse[i]	32	unimsbf
}		
for(i=0;i<X;i++){		
padding_word	16	bslbf
}		
for(EP_fine_id=0;EP_fine_id<Nf;EP_fine_id++){		
is_angle_change_point[EP_fine_id]	1	bslbf
I_end_position_offset[EP_fine_id]	3	bslbf
PTS_EP_fine[EP_fine_id]	11	unimsbf
SPN_EP_fine[EP_fine_id]	17	unimsbf
}		
}		

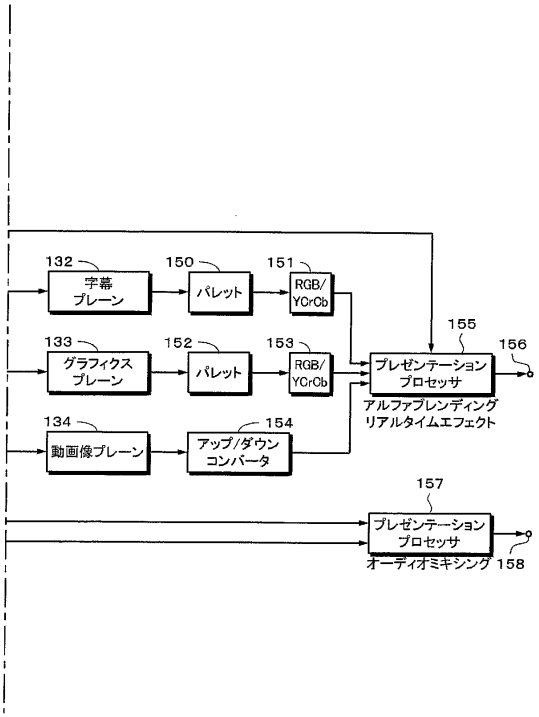
第７３図A



第７３図B

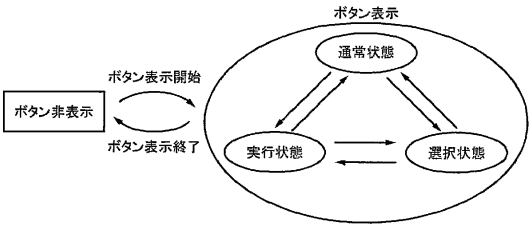


第７３図C



【図７４】

第７４図

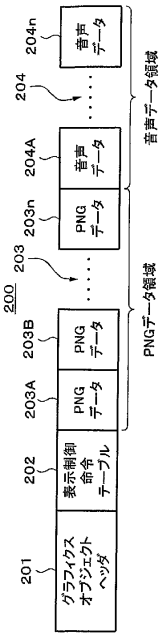


【図７５】

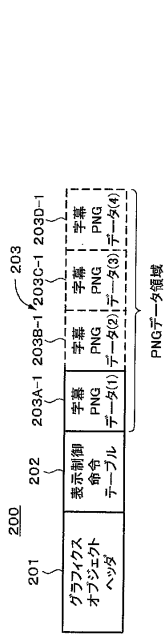
第７５図

オブジェクトの種類	動画プレーンに提示される映像との関係	データ構造	表示先のプレーン
字幕	同期型	グラフィクスオブジェクト（GOBJ）	字幕プレーン
同期型グラフィクス			グラフィクスプレーン
非同期型グラフィクス	非同期型	任意の形式	

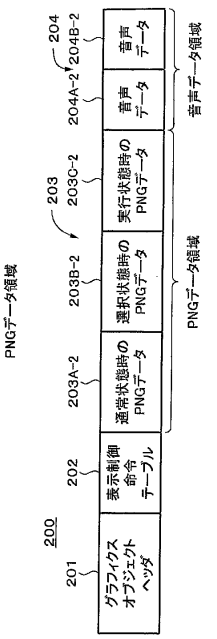
第７６図Ａ



第７６図Ｂ

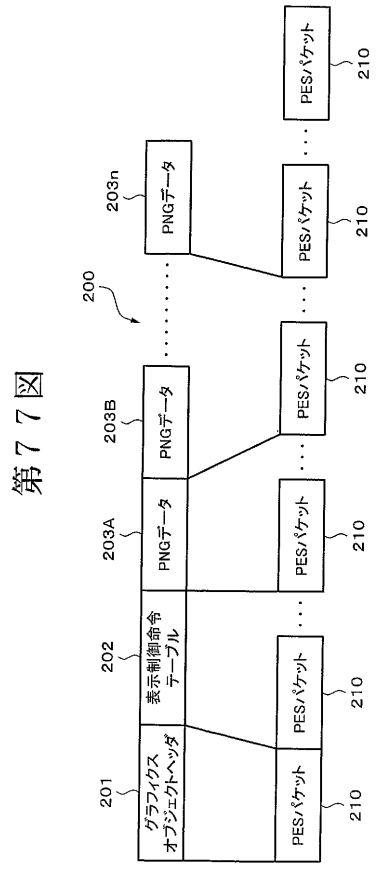


第７６図Ｃ



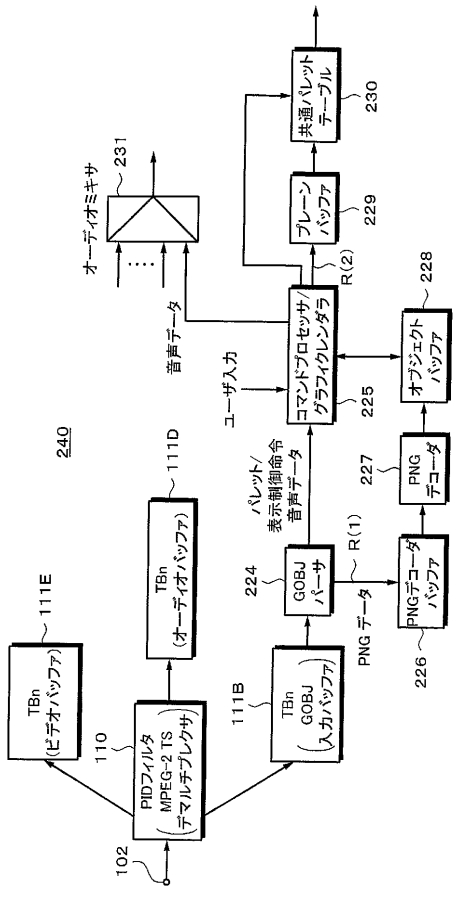
【図７７】

第７７図

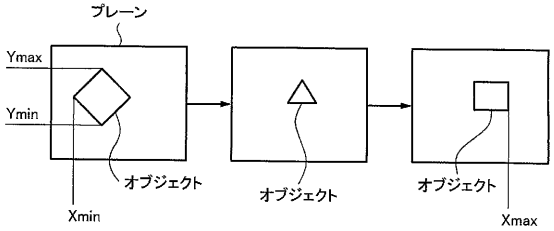


【図７８】

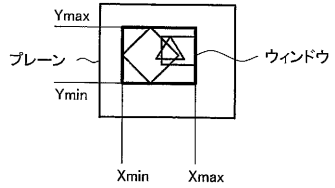
第７８図



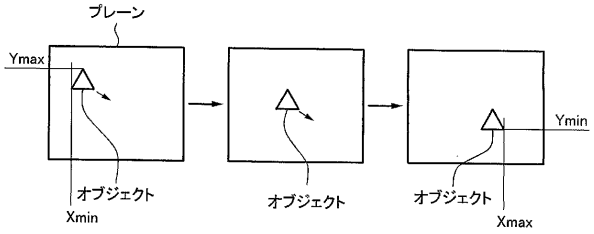
第８１図Ａ



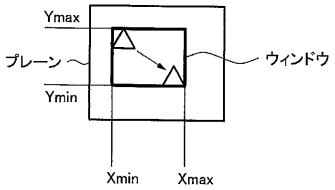
第８１図Ｂ



第８２図Ａ



第８２図Ｂ

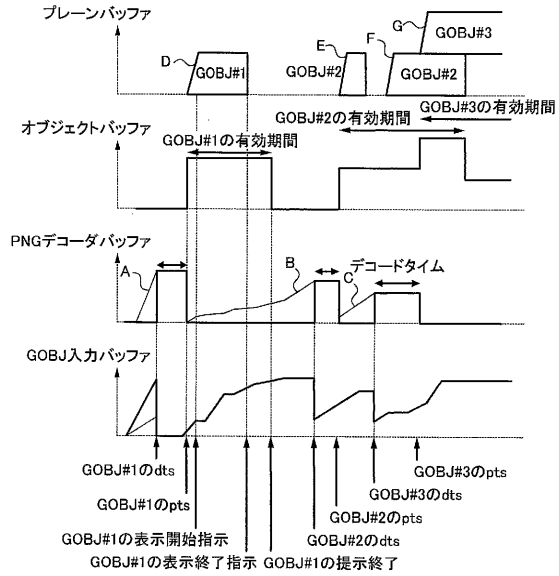


第７９図Ａ

第７９図Ｂ

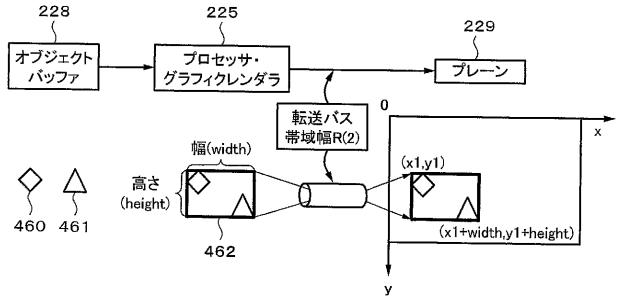
第７９図Ｃ

第７９図Ｄ



【図８０】

第８０図



JP 4442564 B2 2010.3.31

【図８３】

第８３図

シンタクス	データ長(ビット)	ニーモニック
GraphicsObject(){		
GraphicsObjectHeader(){		
length	8	unimsbf
reserved	7	
presentation_end_time_stamp	33	unimsbf
number_of_PNG_images	8	unimsbf
number_of_DispCmds	8	unimsbf
GlobalPaletteTable()		
for(i=0;i<Number_of_PNGs;i++){		
start_address_of_PNG_image(i)	32	unimsbf
PNG_file_name(i)	8*32	bslbf
}		
for(=0;i<Number_of_DispCmds;++){		
start_address_of_PNG_Cmds(i)	32	unimsbf
}		
}		
for(i=0;i<N1;i++){		
padding_word	32	bslbf
}		
GOBJCommandTable(){		
for(i=0;i<Number_of_DispCmds;i++){		
DispCmds(i)		
}		
}		
for(i=0;i<N2;i++){		
padding_word	32	bslbf
}		
PNGImageRegion (){		
for(i=0;i<Number_of_PNGs;i++){		
PNG_image(i)		
}		
}		
for(i=0;i<N3;i++){		
padding_word	32	bslbf
SoundDataRegion (){		
}		
for(i=0;i<Number_of_sound_data;i++){		
sound_data(i)		
}		
}		
}		

【図８４】

第８４図

シンタクス	データ長(ビット)	ニーモニック
GlobalPaletteTable(){		
reserved_for_future_use	8	bslbf
number_of_palette_entries	8	unimsbf
for(i=0;i<Number_of_palette_entries;i++){		
palette_index_number	8	unimsbf
red_value	8	unimsbf
green_value	8	unimsbf
bule_value	8	unimsbf
alpha	8	unimsbf
}		
}		

第８５図

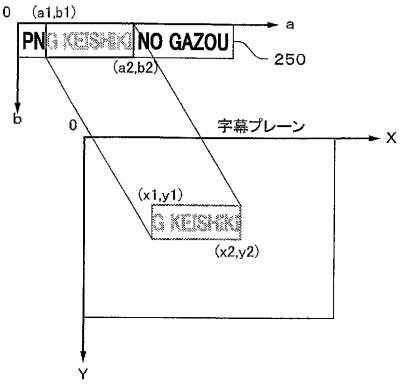
第８５図A

表示制御命令	内容
(1)execution_time(start_time)	この命令の後にあり、次のexecution_time(start_time)の前までの命令をstart_timeで指定した時刻に実行することを指示する命令。start_timeの原点は、グラフィクスオブジェクトのpts(presentation time stamp)とする。start_timeの単位はptsと同じ。
(2)fade_in(fade_in_time)	グラフィクスオブジェクトの表示を開始する。現在のパレットテーブルの不透明度の値を徐々に小さくしていき、fade_in_time後に0(透明)に設定する。fade_in_timeが0の時は、パレットテーブルの色・不透明度で即時に表示される。
(3)fade_out(fade_out_time)	グラフィクスオブジェクトの表示を停止する命令。不透明度の初期値を0から徐々に上げていき、fade_out_time経過後にパレットテーブルの値に設定する。fade_out_timeが0の時は、即時に表示停止する。

第８５図B

表示制御命令	内容
(4)change_palette (index, newR, newG, newB, newAlpha)	パレット番号indexの色を(newR, newG, newB)に、不透明度をnewAlphaに変更する。
(5)set_display_box(x1,y1,x2,y2)	プレーン上の(x1,y1,x2,y2)で定まる短形の領域にグラフィックスオブジェクトを配置する。
(6)set_clipping_box(a1,b1,a2,b2)	グラフィックスオブジェクト内の(a1,b1,a2,b2)で定まる短形の領域をプレーン上に表示する。
(7)play_sound(sound_id)	sound_idで指定された音声データを再生する。
(8)set_sound(PNG_image_id,sound_id)	PNGデータに音声データを割り当てる命令。PNG_image_idで指定されるPNGデータが表示される際に、sound_idで指定された音声データが再生されるよう指示する命令。PNG_image_idは、シンタクス中のPNG_image(i)における(ループカウンタ)と同じである。

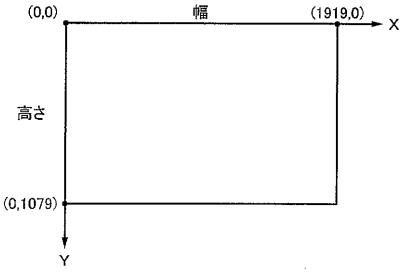
第８６図Ａ



第８６図Ｂ

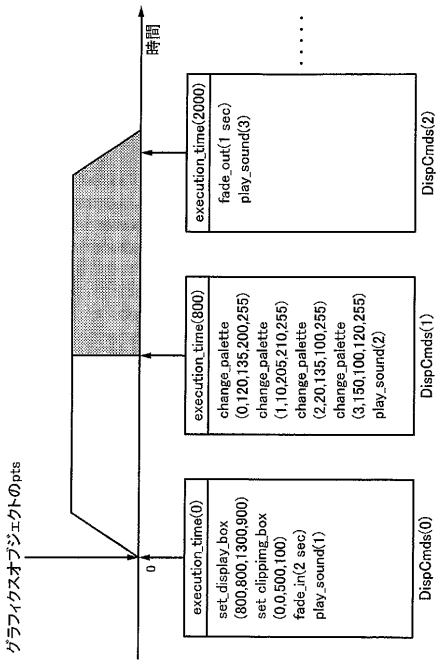
【図８７】

第８７図

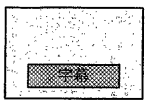


【図８８】

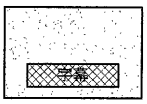
第８８図



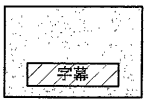
第８９図Ａ



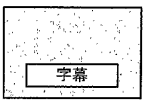
第８９図Ｂ



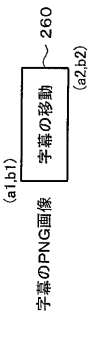
第８９図Ｃ



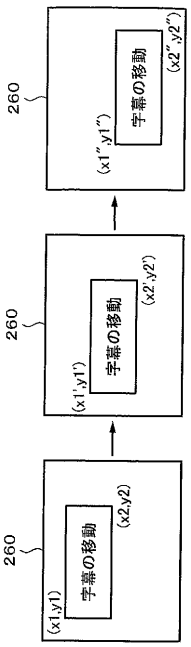
第８９図Ｄ

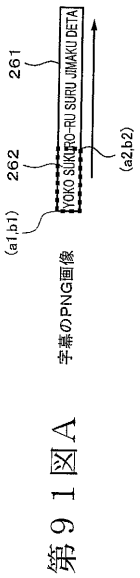


第９０図Ａ

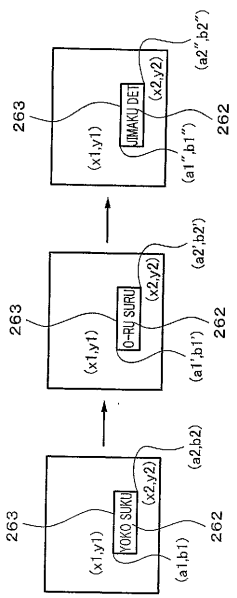


第９０図Ｂ

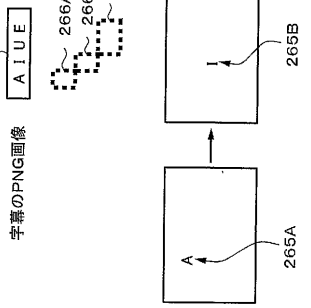




第９１図Ａ



第９１図Ｂ

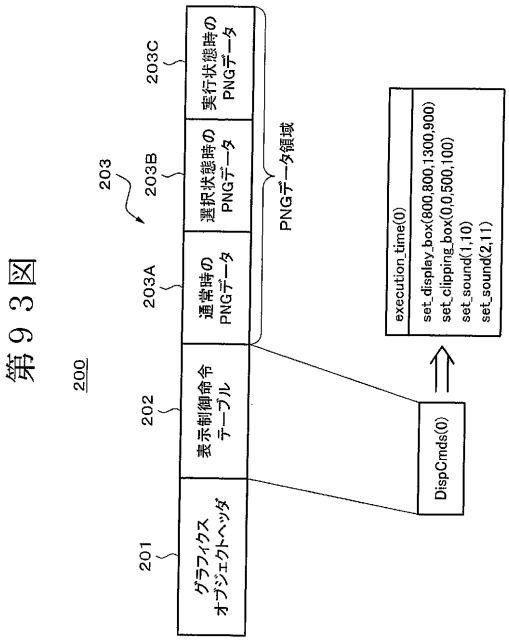


第９２図Ａ

第９２図Ｂ

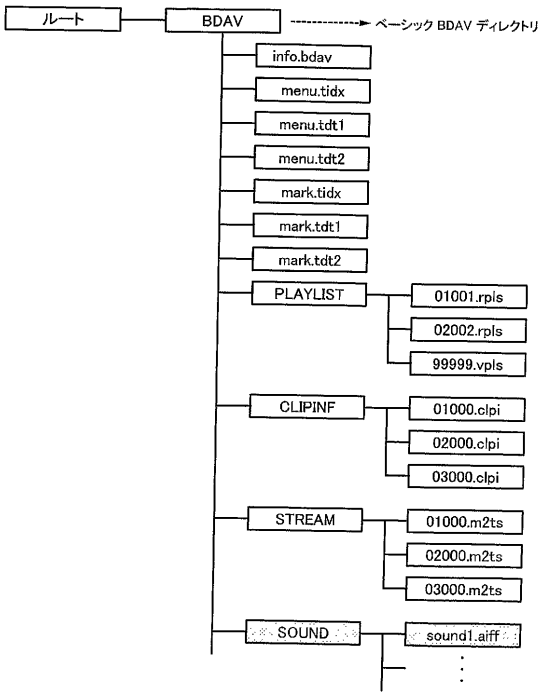
【図９３】

第９３図



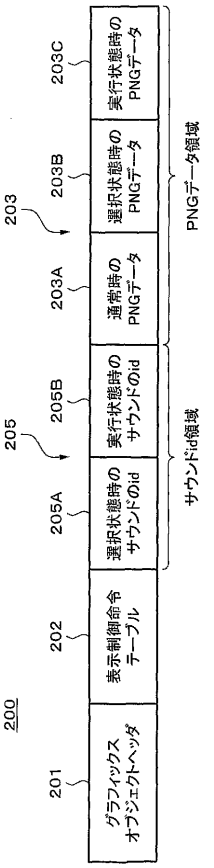
【図９４】

第９４図



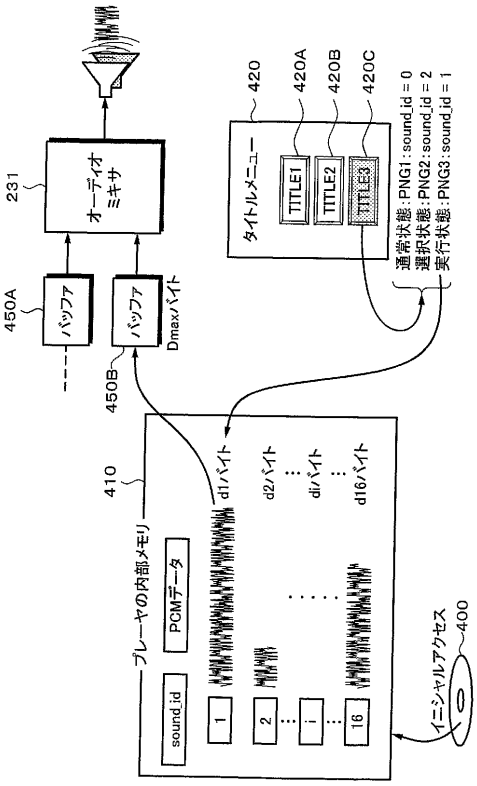
【図９５】

第９５図



【図９６】

第９６図



フロントページの続き

(51)Int.Cl. F I
H04N 5/91 (2006.01) H04N 5/91 E

(56)参考文献 特開平１１－０６９２８４（ＪＰ，Ａ）
特開２００１－３２６９１０（ＪＰ，Ａ）
特開平１０－３０８９２４（ＪＰ，Ａ）
特開平１１－１７６０９６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
H04N 5/76-5/956
G11B 20/10-20/12
G11B 27/34