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

特許第4906197号
(P4906197)

(45)発行日　平成24年3月28日(2012.3.28)　(24)登録日　平成24年1月20日(2012.1.20)

(51)Int.Cl.			F I		
H04N	5/85	(2006.01)	H04N	5/85	A
H04N	5/91	(2006.01)	H04N	5/91	N
H04N	5/92	(2006.01)	H04N	5/92	H
H04N	7/32	(2006.01)	H04N	7/137	Z

請求項の数 11　(全 37 頁)

(21)出願番号　特願2001-147594(P2001-147594)
(22)出願日　平成13年5月17日(2001.5.17)
(65)公開番号　特開2002-57986(P2002-57986A)
(43)公開日　平成14年2月22日(2002.2.22)
審査請求日　平成20年1月9日(2008.1.9)
審判番号　不服2010-18949(P2010-18949/J1)
審判請求日　平成22年8月23日(2010.8.23)
(31)優先権主張番号　特願2000-158546(P2000-158546)
(32)優先日　平成12年5月29日(2000.5.29)
(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　100082131
弁理士　稲本　義雄
(72)発明者　上田　衛
東京都品川区北品川６丁目７番３５号　ソニー株式会社内

合議体
審判長　藤内　光武
審判官　小池　正彦
審判官　▲徳▼田　賢二

最終頁に続く

(54)【発明の名称】復号装置および方法、並びに記録媒体

(57)【特許請求の範囲】
【請求項１】
符号化ストリームを入力する入力手段と、
バッファリングされている前記符号化ストリームをスライス単位で読み出して復号する複数の復号手段と、
前記複数の復号手段を並行して動作させるように制御する復号制御手段と、
前記複数の復号手段のそれぞれから復号結果として出力された複数の画像データのうち、前記複数の復号手段からの要求に対応したものを選択する選択手段と、
選択された前記画像データに対して必要に応じて動き補償処理を施す動き補償手段と、
必要に応じて動き補償処理が施された前記画像データを任意の再生速度で出力させる出力制御手段と
を含み、
前記復号制御手段は、前記復号手段がスライス単位で復号対象とする前記符号化ストリームがバッファリングされている位置を示す書き込みポインタを前記複数の復号手段のうちのいずれかに供給し、
前記復号手段は、供給された前記書込みポインタに従い、バッファリングされている前記符号化ストリームをスライス単位で読み出して復号する
復号装置。
【請求項２】
前記符号化ストリームは、ビットレートが所定数倍に高速化されたMPEG２ビデオビット

ストリームである
請求項１に記載の復号装置。
【請求項３】
前記出力制御手段は、ビットレートが前記所定数倍に高速化された前記MPEG２ビデオビットストリームの復号結果である画像データを、０倍乃至前記所定数倍の再生速度で出力させる
請求項２に記載の復号装置。
【請求項４】
前記復号手段は、復号処理が終了したことを示す終了信号を前記復号制御手段に出力し、
前記復号制御手段は、前記終了信号を出力した前記復号手段に対して、新たな前記符号化ストリームを復号させるように制御する
請求項１に記載の復号装置。
【請求項５】
前記符号化ストリームをバッファリングする第１のバッファ手段と、
前記符号化ストリームから、前記符号化ストリームに含まれる所定の情報の単位の始まりを表わすスタートコードを読み出すとともに、前記第１のバッファ手段から、前記スタートコードが保持されている位置に関する位置情報を読み出す読み出し手段と、
読み出された前記スタートコードおよび前記位置情報をバッファリングする第２のバッファ手段と、
前記第１のバッファ手段による前記符号化ストリームのバッファリング、および前記第２のバッファ手段による前記スタートコードおよび前記位置情報のバッファリングを制御するバッファリング制御手段と
をさらに含む請求項１に記載の復号装置。
【請求項６】
前記復号手段は、復号処理が終了したことを示す終了信号を前記選択手段に出力し、
前記選択手段は、
複数の前記復号手段それぞれの処理状態に対応する複数の処理状態値を記憶する記憶手段を含み、
記憶している全ての処理状態値が第１の値になった場合、前記終了信号を出力している前記復号手段に対応する前記処理状態値を、前記第１の値から第２の値に変更し、
対応する前記処理状態値が前記第２の値である復号手段により復号された前記画像データのうち、いずれかの前記画像データを選択し、
選択された前記画像データを復号した前記復号手段に対応する前記処理状態値を前記第１の値に変更する
請求項１に記載の復号装置。
【請求項７】
前記選択手段により選択された前記画像データ、または前記動き補償手段により動き補償が施された前記画像データを保持する保持手段と、
前記選択手段により選択された前記画像データ、または前記動き補償手段により動き補償が施された前記画像データの前記保持手段による保持を制御する保持制御手段と
をさらに含む請求項１に記載の復号装置。
【請求項８】
前記保持手段は、前記画像データの輝度成分と色差成分をそれぞれ分けて保持する
請求項７に記載の復号装置。
【請求項９】
前記復号手段は、スライスデコーダであり、前記符号化ストリームのピクチャ層のデコード結果であるパラメータに従い、前記第１のバッファ手段からスライス単位で読み出した前記符号化ストリームを復号する
請求項１に記載の復号装置。

【請求項１０】
復号装置による、
符号化ストリームを入力する入力ステップと、
複数の復号手段を並行して動作させるように制御する復号制御ステップと、
前記複数の復号手段により、バッファリングされている前記符号化ストリームをスライス単位で読み出して復号する復号ステップと、
前記複数の復号手段のそれぞれから復号結果として出力された複数の画像データのうち、前記複数の復号手段からの要求に対応したものを選択する選択ステップと、
選択された前記画像データに対して必要に応じて動き補償処理を施す動き補償手段と、
必要に応じて動き補償処理が施された前記画像データを任意の再生速度で出力させる出力制御ステップと
を含み、
前記復号制御ステップは、前記復号手段がスライス単位で復号対象とする前記符号化ストリームがバッファリングされている位置を示す書き込みポインタを前記複数の復号手段のうちのいずれかに供給し、
前記復号ステップは、供給された前記書込みポインタに従い、バッファリングされている前記符号化ストリームをスライス単位で読み出して復号する
復号方法。
【請求項１１】
符号化ストリームを入力する入力ステップと、
複数の復号手段を並行して動作させるように制御する復号制御ステップと、
前記複数の復号手段により、バッファリングされている前記符号化ストリームをスライス単位で読み出して復号する復号ステップと、
前記複数の復号手段のそれぞれから復号結果として出力された複数の画像データのうち、前記複数の復号手段からの要求に対応したものを選択する選択ステップと、
選択された前記画像データに対して必要に応じて動き補償処理を施す動き補償手段と、
必要に応じて動き補償処理が施された前記画像データを任意の再生速度で出力させる出力制御ステップと
を含み、
前記復号制御ステップは、前記復号手段がスライス単位で復号対象とする前記符号化ストリームがバッファリングされている位置を示す書き込みポインタを前記複数の復号手段のうちのいずれかに供給し、
前記復号ステップは、供給された前記書込みポインタに従い、バッファリングされている前記符号化ストリームをスライス単位で読み出して復号する
処理をコンピュータに実行させるプログラムが記録されている記録媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、復号装置および方法、並びに記録媒体に関し、例えば、MPEG２ビデオビットストリームをデコードする場合に用いて好適な復号装置および方法、並びに記録媒体に関する。
【０００２】
【従来の技術】
MPEG(Moving Picture Experts Group)２ビデオは、ISO/IEC(International Standards Organization/International Electrotechnical Commission)13818-2、およびITU-T(International Telecommunication Union-Telecommunication sector)勧告H.262に規定されているビデオ信号の高能率圧縮符号化方式である。
【０００３】
MPEG２ビデオでは、ビデオ画像の各画像は符号化の効率が異なる３つのピクチャタイプ（

フレーム内符号化画像（Ｉ（Intra)ピクチャ）、フレーム間順方向予測符号化画像（Ｐ(Predictive)ピクチャ）、および双方向予測符号化画像（Ｂ(Bidirectionally predictive)ピクチャ））のうちのいずれかに分類される。Ｉピクチャに分類された画像は、当該画像のフレーム内の空間的相関関係に基づいて符号化される。Ｐピクチャに分類された画像は、当該画像の前に存在するＩピクチャまたはＰピクチャからの動き補償予測によって符号化される。Ｂピクチャに分類された画像は、当該画像の前後に存在するＩピクチャまたはＰピクチャからの動き補償予測によって符号化される。したがって、符号化の効率は、Ｉピクチャ、Ｐピクチャ、Ｂピクチャの順に高くなる。

【０００４】

図１を参照して具体的に説明する。ビデオ信号の画像が$Ｉ_1$，$Ｂ_2$，$Ｂ_3$，$Ｐ_4$、・・・、$Ｐ_{13}$に分類された場合（下付の数字は表示順序を示す）、例えば、画像$Ｉ_1$はフレーム内の空間的相関関係に基づいて符号化され、画像$Ｐ_4$は、画像$Ｉ_1$からの動き補償予測によって符号化され、画像$Ｐ_7$は、画像$Ｐ_4$からの動き補償予測によって符号化される。例えばまた、画像$Ｂ_2$は、画像$Ｉ_1$および画像$Ｐ_4$からの動き補償予測によって符号化され、画像$Ｂ_5$は、画像$Ｐ_4$および画像$Ｐ_7$からの動き補償予測によって符号化される。

【０００５】

MPEG２符号化ストリームは、符号化の手法によって決まるプロファイルと、取り扱う画素数によって決まるレベルによってクラス分けされ、広範囲なアプリケーションに対応できるようになされている。例えば、MPEG２符号化ストリームのクラスのうちの１つであるMP@ML（メイン・プロファイル・メイン・レベル）は、DVB(Digital Video Broadcast)や、DVD(Digital Versatile Disk)に広く実用化されている。プロファイルおよびレベルは、図６を用いて後述するsequence_extensionに記述される。

【０００６】

また、放送局における用途に適用させたMPEG２符号化ストリームのプロファイルとして、ビデオの色差信号を従来のベースバンドと同様に4:2:2方式で取り扱うことができるようにビットレートの上限を高く設定した4:2:2P（4:2:2プロファイル）が規定されている。さらに、MPEG２符号化ストリームのレベルとして、次世代の高解像度ビデオ信号に対応するＨＬ（ハイ・レベル）が規定されている。

【０００７】

図２は、MPEG２で規定されている代表的なクラスである、4:2:2P@HL（4:2:2プロファイル・ハイ・レベル）、4:2:2P@ML（4:2:2プロファイル・メイン・レベル）、MP@HL（メイン・プロファイル・ハイ・レベル）、MP@HL-1440（メイン・プロファイル・ハイ・レベル－１４４０）、MP@ML（メイン・プロファイル・メイン・レベル）、MP@LL（メイン・プロファイル・ロー・レベル）、および、SP@ML（シンプル・プロファイル・メイン・レベル）に関し、各クラスのパラメータ（ビットレート、１ラインあたりのサンプル数、１フレームあたりのライン数、フレームの処理時間、およびサンプルの処理時間）の上限値を示している。

【０００８】

図２に示すように、4:2:2P@HLのビットレートの上限値は、３００（メガビット／秒）であり、処理する画素数の上限値は、62,668,800（画素／秒）である。一方、MP@MLのビットレートの上限値は、１５（メガビット／秒）であり、処理する画素数の上限値は、10,368,000（画素／秒）である。すなわち、4:2:2P@HLをデコードするビデオデコーダは、MP@MLをデコードするビデオデコーダに比較して、ビットレートは２０倍、処理する画素数は約６倍の処理能力が必要であることがわかる。

【０００９】

ここで、MPEG２ビデオビットストリームのレベル構造について、図３を参照して説明する。最上位層であるピクチャ層の最初には、sequence_headerが記述されている。sequence_headerは、MPEGビットストリームのシーケンスのヘッダデータを定義するものである。シーケンス最初のsequence_headerに、sequence_extensionが続かない場合、当該ビットストリームには、ISO/IEC11172-2の規定が適応される。シーケンスの最初のsequence_heade

rに、sequence_extensionが続く場合、その後に発生する全てのsequence_headerの直後には、sequence_extensionが続く。すなわち、図３に示す場合においては、全てのsequence_headerの直後に、sequence_extensionが続く。

【００１０】

sequence_extensionは、MPEGビットストリームのシーケンス層の拡張データを定義するものである。sequence_extensionは、sequence_headerの直後にのみ発生し、かつ、復号後、およびフレームリオーダリング後にフレームの損失がないようにするために、ビットストリームの最後に位置するsequence_end_codeの直前にきてはならない。また、ビットストリーム中に、sequence_extensionが発生した場合、それぞれのpicture_headerの直後にpicture_cording_extentionが続く。

【００１１】

GOP(Group Of Picture)内には、複数の画像(picture)が含まれる。GOP_headerは、MPEGビットストリームのGOP層のヘッダデータを定義するものであり、さらに、このビットストリーム中には、picture_headerとpicture_coding_extensionによって定義されたデータエレメントが記述されている。１枚の画像は、picture_headerおよびpicture_coding_extensionに続くpicture_dataとして符号化される。また、GOP_headerに続く最初の符号化フレームは、符号化されたＩフレームである（すなわち、GOP_headerの最初の画像はＩピクチャである）。ITU－T勧告H.262には、sequence_extensionおよびpicture_cording_extentionの他、各種の拡張が定義されているが、ここでは図示、および説明は省略する。

【００１２】

picture_headerは、MPEGビットストリームのピクチャ層のヘッダデータを定義するものであり、picture_coding_extensionは、MPEGビットストリームのピクチャ層の拡張データを定義するものである。

【００１３】

picture_dataは、MPEGビットストリームのスライス層およびマクロブロック層に関するデータエレメントを記述するものである。picture_dataは、図３に示されるように、複数のslice（スライス）に分割され、スライスは、複数のmacro_block（マクロブロック）に分割される。

【００１４】

マクロブロックは、１６×１６の画素データで構成されている。スライスの最初のマクロブロックおよび最後のマクロブロックは、スキップマクロブロック（情報を含まないマクロブロック）ではない。なお、フレームDCT（Discrete Cosine Transform：離散コサイン変換）符号化およびフィールドDCT符号化の使用が可能なフレーム画像においては、フレーム符号化が使用されたマクロブロックの内部構成とフィールド符号化が使用されたマクロブロックの内部構成が相違する。

【００１５】

マクロブロックは、輝度成分および色差成分の１区画を含む。マクロブロックという用語は、情報源および復号データまたは対応する符号化データ成分のいずれかを示す。マクロブロックには、4:2:0、4:2:2および4:4:4の３つの色差フォーマットがある。マクロブロックにおけるブロックの順序は、それぞれの色差フォーマットによって異なる。

【００１６】

図４（Ａ）に、色差フォーマットが4:2:0方式である場合におけるマクロブロックを示す。4:2:0方式の場合、マクロブロックは、４個の輝度（Ｙ）ブロックと、それぞれ１個の色差（Ｃｂ，Ｃｒ）ブロックで構成される。図４（Ｂ）に、色差フォーマットが4:2:2方式である場合におけるマクロブロックを示す。4:2:2方式の場合、マクロブロックは、４個の輝度（Ｙ）ブロックと、それぞれ２個の色差（Ｃｂ，Ｃｒ）ブロックで構成される。

【００１７】

各マクロブロックは、いくつかの方法により、予測符号化処理が可能である。予測モードは、フィールド予測とフレーム予測の２種類に大別される。フィールド予測においては、先に復号された、１つ、もしくは複数のフィールドのデータを使用し、各フィールドにつ

いて、独自に予測を行う。フレーム予測は、先に復号された、１つ、もしくは複数のフレームを使用してフレームの予測を行う。フィールド画像内では、予測は全てフィールド予測である。一方、フレーム画像においては、フィールド予測、またはフレーム予測のいずれかにより予測が可能であり、その予測方法は、マクロブロックごとに選択される。また、マクロブロックの予測符号化処理においては、フィールド予測およびフレーム予測以外に、１６×８動き補償およびデュアルプライムの２種類の特別予測モードを使用することができる。

【００１８】

動きベクトル情報、および他の周辺情報は、各マクロブロックの予測誤差信号とともに符号化される。動きベクトルの符号化については、可変長符号を使用して符号化された最後の動きベクトルを予測ベクトルとして、予測ベクトルとの差分ベクトルを符号化する。表示可能なベクトルの最大長は、画像毎にプログラムすることができる。また、適切な動きベクトルの計算は符号器により実行される。

【００１９】

picture_dataの後には、次のsequence_headerとsequence_extensionが配置されている。このsequence_headerとsequence_extensionによって記述されたデータエレメントは、ビデオストリームのシーケンスの先頭に記述されたsequence_headerとsequence_extensionによって記述されたデータエレメントと全く同じである。このように、同じデータをストリーム中に記述することにより、ビットストリーム受信装置側において、データストリームの途中（例えばピクチャ層に対応するビットストリーム部分）から受信が開始された場合、シーケンス層のデータを受信できなくなってストリームをデコードできなくなることが抑止される。

【００２０】

最後のsequence_headerとsequence_extensionとによって定義されたデータエレメントの次、つまり、データストリームの最後には、シーケンスの終わりを示す３２ビットのsequence_end_codeが記述されている。

【００２１】

次に、図５乃至１２を用いて、それぞれのデータエレメントの詳細について説明する。

【００２２】

図５に、sequence_headerのデータ構成を示す。sequence_headerに含められるデータエレメントは、sequence_header_code, horizontal_size_value, vertical_size_value、aspect_ratio_information, frame_rate_code, bit_rate_value、marker_bit, vbv_buffer_size_value, constrained_parameter_flag, load_intra_quantiser_matrix, intra_quantiser_matrix, load_non_intra_quantiser_matrix、およびnon_intra_quantiser_matrix等から構成される。

【００２３】

sequence_header_codeは、シーケンス層のスタート同期コードを表すデータである。horizontal_size_valueは、画像の水平方向の画素数の下位１２ビットからなるデータである。vertical_size_valueは、画像の縦のライン数の下位１２ビットからなるデータである。aspect_ratio_informationは、画素のアスペクト比（縦横比）または表示画面アスペクト比を表すデータである。frame_rate_codeは、画像の表示周期を表すデータである。bit_rate_valueは、発生ビット量に対する制限のためのビットレートの下位１８ビットのデータである。

【００２４】

marker_bitは、スタートコードエミュレーションを防止するために挿入されるビットデータである。vbv_buffer_size_valueは、発生符号量制御用の仮想バッファVBV(Video Buffering Verifier)の大きさを決める値の下位１０ビットデータである。constrained_parameter_flagは、各パラメータが制限以内であることを示すデータである。load_non_intra_quantiser_matrixは、非イントラマクロブロック用量子化マトリックス・データの存在を示すデータである。load_intra_quantiser_matrixは、イントラマクロブロック用量子化

マトリックス・データの存在を示すデータである。intra_quantiser_matrixは、イントラマクロブロック用量子化マトリックスの値を示すデータである。non_intra_quantiser_matrixは、非イントラマクロブロック用量子化マトリックスの値を表すデータである。

【００２５】

図６に、sequence_extensionのデータ構成を示す。sequence_extensionは、extension_start_code, extension_start_code_identifier, profile_and_level_indication, progressive_sequence, chroma_format, horizontal_size_extension, vertical_size_extension, bit_rate_extension, marker_bit, vbv_buffer_size_extension, low_delay, frame_rate_extension_n、およびframe_rate_extension_d等のデータエレメントから構成されている。

【００２６】

extension_start_codeは、エクステンションデータ（拡張データ）のスタート同期コードを表すデータである。extension_start_code_identifierは、拡張データの種類を示すデータである。profile_and_level_indicationは、ビデオデータのプロファイルとレベルを指定するためのデータである。progressive_sequenceは、ビデオデータが順次走査（プログレッシブ画像）であることを示すデータである。chroma_formatは、ビデオデータの色差フォーマットを指定するためのデータである。horizontal_size_extensionは、シーケンスヘッダのhorizntal_size_valueに加える上位２ビットのデータである。vertical_size_extensionは、シーケンスヘッダのvertical_size_valueに加える上位２ビットのデータである。

【００２７】

bit_rate_extensionは、シーケンスヘッダのbit_rate_valueに加える上位１２ビットのデータである。marker_bitは、スタートコードエミュレーションを防止するために挿入されるビットデータである。vbv_buffer_size_extensionは、シーケンスヘッダのvbv_buffer_size_valueに加える上位８ビットのデータである。low_delayは、Ｂピクチャを含まないことを示すデータである。frame_rate_extension_nは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを得るためのデータである。frame_rate_extension_dは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを得るためのデータである。

【００２８】

図７に、GOP_headerのデータ構成を示す。GOP_headerを表わすデータエレメントは、group_start_code, time_code, closed_gop、およびbroken_linkから構成される。

【００２９】

group_start_codeは、GOP層の開始同期コードを示すデータである。time_codeは、GOPの先頭ピクチャの時間を示すタイムコードである。closed_gopは、GOP内の画像が他のGOPから独立再生可能なことを示すフラグデータである。broken_linkは、編集などのためにGOP内の先頭のＢピクチャが正確に再生できないことを示すフラグデータである。

【００３０】

図８に、picture_headerのデータ構成を示す。picture_headerに関するデータエレメントは、picture_start_code, temporal_reference, picture_coding_type、vbv_delay, full_pel_forward_vector, forward_f_code, full_pel_backward_vector、および backward_f_code等から構成される。

【００３１】

picture_start_codeは、ピクチャ層の開始同期コードを表すデータである。temporal_referenceは、ピクチャの表示順を示す番号でGOPの先頭でリセットされるデータである。picture_coding_typeは、ピクチャタイプを示すデータである。vbv_delayは、ランダムアクセス時の仮想バッファの初期状態を示すデータである。full_pel_forward_vector, forward_f_code, full_pel_backward_vector、およびbackward_f_codeは、MPEG２では使用されない固定データである。

【００３２】

図９に、picture_coding_extensionのデータ構成を示す。picture_coding_extensionは、extension_start_code, extension_start_code_identifier, f_code[0][0], f_code[0][1], f_code[1][0], f_code[1][1], intra_dc_precision, picture_structure, top_field_first, frame_pred_frame_dct, concealment_motion_vectors, q_scale_type, intra_vlc_format, alternate_scan, repeat_firt_field, chroma_420_type, progressive_frame, composite_display_flag, v_axis, field_sequence, sub_carrier, burst_amplitude、およびsub_carrier_phase等から構成される。

【００３３】

extension_start_codeは、ピクチャ層のエクステンションデータ（拡張データ）のスタートを示す同期コードを表すデータである。extension_start_code_identifierは、拡張データの種類を示すデータである。f_code[0][0]は、フォワード方向の水平動きベクトル探索範囲を表すデータである。f_code[0][1]は、フォワード方向の垂直動きベクトル探索範囲を表すデータである。f_code[1][0]は、バックワード方向の水平動きベクトル探索範囲を表すデータである。f_code[1][1]は、バックワード方向の垂直動きベクトル探索範囲を表すデータである。

【００３４】

intra_dc_precisionは、ＤＣ係数の精度を表すデータである。ブロック内の各画素の輝度および色差信号を表した行列ｆにDCTを施すと、８×８のDCT係数行列Ｆが得られる。DCT係数行列Ｆの左上隅の係数をＤＣ係数と呼ぶ。ＤＣ係数はブロック内の平均輝度、平均色差を表わす信号である。picture_structureは、フレームストラクチャであるか、フィールドストラクチャであるかを示すデータであり、フィールドストラクチャであることを示している場合はさらに、上位フィールドであるか、下位フィールドであるかを示すデータを含んでいる。top_field_firstは、フレームストラクチャである場合において、最初のフィールドが上位であるか、下位であるかを示すデータである。frame_predictive_frame_dctは、フレームストラクチャである場合において、フレーム・モードDCTの予測がフレーム・モードだけであることを示すデータである。concealment_motion_vectorsは、イントラマクロブロックに伝送エラーを隠蔽するための動きベクトルがついていることを示すデータである。

【００３５】

q_scale_typeは、線形量子化スケールを利用するか、非線形量子化スケールを利用するかを示すデータである。intra_vlc_formatは、イントラマクロブロックに、別の２次元VLC(Variable Length Cording)を使うか否かを示すデータである。alternate_scanは、ジグザグスキャンを使うか、オルタネート・スキャンを使うかの選択を表すデータである。repeat_firt_fieldは、２：３プルダウンの際に使われるデータである。chroma_420_typeには、色差フォーマットが4:2:0方式である場合、次のprogressive_frameと同じ値が記述され、色差フォーマットが4:2:0方式ではない場合には０が記述される。progressive_frameは、このピクチャが順次走査であるか否かを示すデータである。composite_display_flagは、ソース信号がコンポジット信号であったか否かを示すデータである。v_axis, field_sequence, sub_carrier, burst_amplitude、およびsub_carrier_phaseは、ソース信号がコンポジット信号であった場合に使われるデータである。

【００３６】

図１０に、picture_dataのデータ構成を示す。picture_data()関数によって定義されるデータエレメントは、slice()関数によって定義されるデータエレメントである。このslice()関数によって定義されるデータエレメントは、ビットストリーム中に少なくとも１個記述されている。

【００３７】

slice()関数は、図１１に示されるように、slice_start_code, quantiser_scale_code, intra_slice_flag, intra_slice, reserved_bits, extra_bit_slice, およびextra_information_slice等のデータエレメントと、macroblock()関数によって定義される。

【００３８】

slice_start_codeは、slice()関数によって定義されるデータエレメントのスタートを示すスタートコードである。quantiser_scale_codeは、このスライス層に存在するマクロブロックに対して設定された量子化ステップサイズを示すデータであるが、マクロブロック毎に、quantiser_scale_codeが設定されている場合には、各マクロブロックに対して設定されたmacroblock_quantiser_scale_codeのデータが優先して使用される。

【００３９】

intra_slice_flagは、ビットストリーム中にintra_sliceおよびreserved_bitsが存在するか否かを示すフラグである。intra_sliceは、スライス層中にノンイントラマクロブロックが存在するか否かを示すデータである。スライス層におけるマクロブロックのいずれかがノンイントラマクロブロックである場合には、intra_sliceは「０」となり、スライス層におけるマクロブロックの全てがイントラマクロブロックである場合には、intra_sliceは「１」となる。reserved_bitsは、７ビットの予備のデータ領域である。extra_bit_sliceは、追加の情報が存在するか否かを示すフラグであって、次にextra_information_sliceが存在する場合には「１」に設定され、追加の情報が存在しない場合には「０」に設定される。

【００４０】

これらのデータエレメントの次には、macroblock()関数によって定義されたデータエレメントが記述されている。macroblock()関数は、図１２に示すように、macroblock_escape，macroblock_address_increment，quantiser_scale_code、およびmarker_bit等のデータエレメントと、macroblock_modes()関数、motion_vectors()関数、およびcoded_block_pattern()関数によって定義されたデータエレメントを記述するための関数である。

【００４１】

macroblock_escapeは、参照マクロブロックと前のマクロブロックとの水平方向の差が３４以上であるか否かを示す固定ビット列である。参照マクロブロックと前のマクロブロックとの水平方向の差が３４以上である場合、macroblock_address_incrementの値に３３が加えられる。macroblock_address_incrementは、参照マクロブロックと前のマクロブロックとの水平方向の差を示すデータである。もし、macroblock_address_incrementの前にmacroblock_escapeが１つ存在するのであれば、このmacroblock_address_incrementの値に３３を加えた値が、実際の参照マクロブロックと前のマクロブロックとの水平方向の差分を示すデータとなる。

【００４２】

quantiser_scale_codeは、各マクロブロックに設定された量子化ステップサイズを示すデータであり、macroblock_quantが「１」のときだけ存在する。各スライス層には、スライス層の量子化ステップサイズを示すslice_quantiser_scale_codeが設定されているが、参照マクロブロックに対してscale_codeが設定されている場合には、この量子化ステップサイズを選択する。

【００４３】

macroblock_address_incrementの次には、macroblock_modes()関数によって定義されるデータエレメントが記述されている。macroblock_modes()関数は、図１３に示すように、macroblock_type，frame_motion_type，field_motion_type，dct_type等のデータエレメントを記述するための関数である。macroblock_typeは、マクログブロックの符号化タイプを示すデータである。

【００４４】

macroblock_motion_forwardまたはmacroblock_motion_backwardが「１」であり、ピクチャ構造がフレームであり、さらにframe_pred_frame_dctが「０」である場合、macroblock_typeを表わすデータエレメントの次にframe_motion_typeを表わすデータエレメントが記述される。なお、このframe_pred_frame_dctは、frame_motion_typeがビットストリーム中に存在するか否かを示すフラグである。

【００４５】

frame_motion_typeは、フレームのマクロブロックの予測タイプを示す２ビットのコード

である。予測ベクトルが２個であって、フィールドベースの予測タイプである場合、frame_motion_typeには「００」が記述される。予測ベクトルが１個であって、フィールドベースの予測タイプでる場合、frame_motion_typeには「０１」が記述される。予測ベクトルが１個であって、フレームベースの予測タイプである場合、frame_motion_typeには「１０」が記述される。予測ベクトルが１個であって、デュアルプライムの予測タイプである場合、frame_motion_typeには「１１」が記述される。

【００４６】

field_motion_typeは、フィールドのマクロブロックの動き予測を示す２ビットのコードである。予測ベクトルが１個であって、フィールドベースの予測タイプである場合、field_motion_typeには「０１」が記述される。予測ベクトルが２個であって、１８×８マクロブロックベースの予測タイプである場合、field_motion_typeには「１０」が記述される。予測ベクトルが１個であって、デュアルプライムの予測タイプである場合、field_motion_typeには「１１」が記述される。

【００４７】

ピクチャ構造がフレームであり、frame_pred_frame_dctが、そのビットストリーム中にframe_motion_typeが存在することを示し、frame_pred_frame_dctが、そのビットストリーム中にdct_typeが存在することを示している場合、macroblock_typeを表わすデータエレメントの次にはdct_typeを表わすデータエレメントが記述される。なお、dct_typeは、DCTがフレームDCTモードであるか、フィールドDCTモードであるかを示すデータである。

【００４８】

MPEG２ビデオビットストリーム中の各データエレメントは、start codeと称される、特殊なビットパターンで開始される。これらのスタートコードは、別の状況では、ビデオストリーム中に現れない特定のビットパターンである。各スタートコードは、スタートコードプレフィクスと、それに続くスタートコード値から構成される。スタートコードプレフィクスは、ビット列“0000 0000 0000 0000 0000 0001”である。スタートコード値は、スタートコードのタイプを識別する８ビットのデータである。

【００４９】

図１４に、MPEG２のスタートコード値(Start code value)を示す。多くのスタートコードには、１個のスタートコード値が設定されている。しかしながら、slice_start_codeには、複数のスタートコード値（０１乃至ＡＦ）が設定されており、このスタートコード値は、スライスに対する垂直位置を表わす。これらのスタートコードは、全てバイト単位であるため、スタートコードプレフィクスの最初のビットがバイトの最初のビットになるように、スタートコードプレフィクスの前に、複数のビット“０”が挿入され、スタートコードがバイト単位になるように調整される。

【００５０】

次に、MP@MLのMPEG２ビデオビットストリームに対応した従来のMPEGビデオデコーダについて、図１５を参照して説明する。図１５は、当該MPEGビデオデコーダの構成の一例を示している。

【００５１】

当該MPEGビデオデコーダは、ストリーム入力回路１１、バッファ制御回路１２、クロック発生回路１３、スタートコード検出回路１４、デコーダ１５、動き補償回路１６、および表示出力回路１７から構成されるＩＣ(Integrated Circuit)１と、ストリームバッファ２１およびビデオバッファ２２で構成され、例えば、DRAM(Dynamic Random Access Memory)からなるバッファ２により構成される。

【００５２】

ＩＣ１のストリーム入力回路１１は、高能率符号化された符号化ストリーム（MP@MLのMPEG２ビデオビットストリーム）の入力を受け付けて、バッファ制御回路１２に供給する。バッファ制御回路１２は、クロック発生回路１３から供給される基本クロックに従って、入力された符号化ストリームをバッファ２のストリームバッファ２１に入力する。ストリームバッファ２１は、少なくとも、MP@MLのデコードに要求されるVBVバッファサイズであ

る1,835,008ビットの容量を有する。ストリームバッファ２１に保存されている符号化ストリームは、バッファ制御回路１２の制御に従って、先に書き込まれたデータから順に読み出され、スタートコード検出回路１４に供給される。スタートコード検出回路１４は、入力されたストリームから、図１４を用いて説明したスタートコードを検出し、検出したスタートコードおよび入力されたストリームをデコーダ１５に出力する。

【００５３】

デコーダ１５は、入力されたストリームをMPEGシンタックスに基づいて、デコードする。デコーダ１５は、入力されたスタートコードに従って、まず、ピクチャ層のヘッダパラメータをデコードし、それを基に、スライス層をマクロブロックに分離してマクロブロックをデコードし、その結果得られる予測ベクトルおよび画素を、動き補償回路１６に出力する。

【００５４】

圧縮符号化方式としてのMPEGでは、隣接した画像間の時間的冗長性を利用して、近接した画像間で動き補償した差分を得ることにより、符号化効率を改善している。当該MPEGビデオデコーダでは、動き補償を用いた画素に対しては、現在デコードしている画素にその動きベクトルが示す参照画像の画素データを加算することにより動き補償を行い、符号化前の画像データに復号する。

【００５５】

デコーダ１５から出力されるマクロブロックが動き補償を使用していない場合、動き補償回路１６は、その画素データを、バッファ制御回路１２を介してバッファ２のビデオバッファ２２に書き込み、表示出力に備えるとともに、この画素データが、他の画像の参照データとされる場合に備える。

【００５６】

デコーダ１５から出力されるマクロブロックが動き補償を使用している場合、動き補償回路１６は、デコーダ１５から出力される予測ベクトルに従って、バッファ制御回路１２を介して、バッファ２のビデオバッファ２２から参照画素データを読み出す。そして、読み出した参照画素データを、デコーダ１５から供給された画素データに加算し、動き補償を行う。動き補償回路１６は、動き補償を行った画素データを、バッファ制御回路１２を介してバッファ２のビデオバッファ２２に書き込み、表示出力に備えるとともに、この画素データが、他の画素の参照データとされる場合に備える。

【００５７】

表示出力回路１７は、デコードした画像データを出力するための同期タイミング信号を発生し、このタイミングを基に、バッファ制御回路１２を介して、ビデオバッファ２２から画素データを読み出し、復号ビデオ信号として出力する。

【００５８】

【発明が解決しようとする課題】

以上説明したように、MPEG２ストリームは階層構造を有している。図３を用いて説明したピクチャ層のsequence_header乃至picture_coding_extensionのデータは、図２を用いて説明したプロファイルおよびレベルから成るクラスが異なる場合においても、そのデータ量は、あまり変更されない。一方、スライス層以下のデータ量は、符号化する画素数に依存する。

【００５９】

図２に示したように、ＨＬにおいて１枚のピクチャで処理しなければならないマクロブロックの数は、ＭＬのそれに対して約６倍になる。さらに、図４に示したように、4:2:2Pにおいて１個のマクロブロックで処理するブロックの数は、ＭＰのそれに対して４／３倍になる。

【００６０】

したがって、図１５に示したMP@MLに対応する従来のMPEGビデオデコーダを用いて4:2:2P@HLの符号化ストリームを復号しようとした場合、VBVバッファサイズおよび画素数の増加に伴って、ストリームバッファ２１のバッファサイズが不足する。また、ビットレートの

増加に伴い、入力ストリームのストリームバッファ２１へのアクセスが増加し、画素数の増加に伴って、動き補償回路１６のビデオバッファ２２へのアクセスが増加するため、バッファ制御回路１２の制御が間に合わなくなる。さらに、ビットレートの増加、マクロブロックおよびブロック数の増加に伴って、デコーダ１５の処理が間に合わなくなる。

【００６１】

一般に、信号処理を高速で実行させようとした場合、回路規模が大幅に増加し、部品点数の増加および消費電力の増加を招いてしまう。したがって、MP@MLを復号する従来技術を用いて、4:2:2P@HLを復号する装置を実現しようとした場合、今日の半導体技術の進展によって信号処理回路、メモリ（バッファ）回路ともに、その動作速度は著しく向上し、その回路規模は縮小化されてはいるものの、実現可能な回路規模で実時間動作が可能な4:2:2P@HLに対応したビデオデコーダを実現することは困難である課題があった。

【００６２】

ところで、MP@MLの符号化ストリームを、図１５に示したMP@MLに対応する従来のMPEGビデオデコーダを用い、１倍速以上で高速再生させることを考えた場合、入力するMP@MLの符号化ストリームから画像を間引いて当該MPEGビデオデコーダに入力する方法が考えられる。

【００６３】

しかしながら、MPEG方式においては、図１を用いて説明したように、近傍の画像を参照する予測符号化方式を採用しているので、任意の画像を間引いた場合、復号することができない画像が生じてしまう。例えば、図１６に示すように、画像P_{10}を間引いた場合、画像P_{10}を参照して復号する画像B_8，B_9，B_{11}，B_{12}、およびP_{13}を復号することができなくなってしまう。

【００６４】

そこで、図１７に示すように、他の画像に参照されるＩピクチャおよびＰピクチャは間引かず、他の画像に参照されることがないＢピクチャだけを間引いたMP@MLの符号化ストリームを入力する方法が考えられる。しかしながら、Ｂピクチャだけを間引く方法は、Ｂピクチャだけを検出して間引く機能が必要になる上、Ｂピクチャだけしか間引かれないので、符号化ストリームの再生速度がＩピクチャ、Ｐピクチャ、およびＢピクチャの配置に依存して特定の速度となってしまう課題があった。

【００６５】

また、復号したＩピクチャ、およびＰピクチャを適宜繰り返して表示することにより、見かけ上の再生速度は任意の速度に設定できるものの、そのような表示出力は画像の動きがギクシャクしたものとなってしまう課題があった。

【００６６】

さらに、従来のMPEGビデオデコーダでは、複数の符号化ストリームを同時に入力したり、または、DVBで供給される多チャンネル分の複数の符号化ストリーム（エレメンタリストリーム）が多重化されているチャンネル多重ストリーム（トランスポートストリーム）を入力したりして、それらを同時にデコードし、得られる多チャンネル分の複数の復号ビデオ信号を同時に、あるいは、複数の復号ビデオ信号のうちの１つを選択して出力するようなことができない課題があった。

【００６７】

本発明はこのような状況に鑑みてなされたものであり、実現可能な回路規模であって、4:2:2P@HLの符号化ストリームを実時間再生でき、且つ、MP@MLの符号化ストリームを高速再生できるビデオデコーダを実現することを目的とする。

【００６８】

また、多チャンネル分の複数の符号化ストリームを平行してデコードすることができるビデオデコーダを実現することを目的とする。

【００６９】

【課題を解決するための手段】

本発明の復号装置は、符号化ストリームを入力する入力手段と、バッファリングされて

いる符号化ストリームをスライス単位で読み出して復号する複数の復号手段と、複数の復号手段を並行して動作させるように制御する復号制御手段と、複数の復号手段のそれぞれから復号結果として出力された複数の画像データのうち、複数の復号手段からの要求に対応したものを選択する選択手段と、選択された画像データに対して必要に応じて動き補償処理を施す動き補償手段と、必要に応じて動き補償処理が施された画像データを任意の再生速度で出力させる出力制御手段とを含み、復号制御手段は、復号手段がスライス単位で復号対象とする符号化ストリームがバッファリングされている位置を示す書き込みポインタを複数の復号手段のうちのいずれかに供給し、復号手段は、供給された書込みポインタに従い、バッファリングされている符号化ストリームをスライス単位で読み出して復号する。

【００７０】

前記高速化された符号化ストリームは、ビットレートが所定数倍に高速化されたMPEG２ビデオビットストリームであるようにすることができる。

【００７１】

前記出力制御手段は、ビットレートが所定数倍に高速化されたMPEG２ビデオビットストリームの復号結果である画像データを、０倍乃至所定数倍の再生速度で出力させるようにすることができる。

【００７２】

前記復号手段は、復号処理の終了を示す信号を復号制御手段に出力するようにすることができ、前記復号制御手段は、復号処理の終了を示す信号を出力した復号手段に、新たな符号化ストリームを復号させるように制御するようにすることができる。

【００７３】

本発明の第１の復号装置は、符号化ストリームをバッファリングする第１のバッファ手段と、符号化ストリームから、符号化ストリームに含まれる所定の情報の単位の始まりを表わすスタートコードを読み出すとともに、第１のバッファ手段から、スタートコードが保持されている位置に関する位置情報を読み出す読み出し手段と、読み出されたスタートコードおよび位置情報をバッファリングする第２のバッファ手段と、第１のバッファ手段による符号化ストリームのバッファリング、および第２のバッファ手段によるスタートコードおよび位置情報のバッファリングを制御するバッファリング制御手段とをさらに含むことができる。

【００７４】

前記復号手段は、復号処理が終了したことを示す終了信号を前記選択手段に出力し、前記選択手段は、複数の復号手段それぞれの処理状態に対応する複数の処理状態値を記憶する記憶手段を含むことができ、記憶している全ての処理状態値が第１の値になった場合、終了信号を出力している復号手段に対応する処理状態値を、第１の値から第２の値に変更し、対応する処理状態値が第２の値である復号手段により復号された画像データのうち、いずれかの画像データを選択し、選択された画像データを復号した復号手段に対応する処理状態値を第１の値に変更するようにすることができる。

【００７５】

本発明の第１の復号装置は、選択手段により選択された画像データ、または動き補償手段により動き補償が施された画像データを保持する保持手段と、選択手段により選択された画像データ、または動き補償手段により動き補償が施された画像データの保持手段による保持を制御する保持制御手段とをさらに含むことができる。

【００７６】

前記保持手段は、画像データの輝度成分と色差成分をそれぞれ分けて保持するようにさうることができる。

【００７７】

前記復号手段は、スライスデコーダとすることができる。

【００７８】

本発明の復号方法は、符号化ストリームを入力する入力ステップと、複数の復号手段を

並行して動作させるように制御する復号制御ステップと、複数の復号手段により、バッファリングされている符号化ストリームをスライス単位で読み出して復号する復号ステップと、複数の復号手段のそれぞれから復号結果として出力された複数の画像データのうち、複数の復号手段からの要求に対応したものを選択する選択ステップと、選択された画像データに対して必要に応じて動き補償処理を施す動き補償手段と、必要に応じて動き補償処理が施された画像データを任意の再生速度で出力させる出力制御ステップとを含み、復号制御ステップは、復号手段がスライス単位で復号対象とする符号化ストリームがバッファリングされている位置を示す書き込みポインタを複数の復号手段のうちのいずれかに供給し、復号ステップは、供給された書込みポインタに従い、バッファリングされている符号化ストリームをスライス単位で読み出して復号する。

【００７９】

本発明の記録媒体は、符号化ストリームを入力する入力ステップと、複数の復号手段を並行して動作させるように制御する復号制御ステップと、複数の復号手段により、バッファリングされている符号化ストリームをスライス単位で読み出して復号する復号ステップと、複数の復号手段のそれぞれから復号結果として出力された複数の画像データのうち、複数の復号手段からの要求に対応したものを選択する選択ステップと、選択された画像データに対して必要に応じて動き補償処理を施す動き補償手段と、必要に応じて動き補償処理が施された画像データを任意の再生速度で出力させる出力制御ステップとを含み、復号制御ステップは、復号手段がスライス単位で復号対象とする符号化ストリームがバッファリングされている位置を示す書き込みポインタを複数の復号手段のうちのいずれかに供給し、復号ステップは、供給された書込みポインタに従い、バッファリングされている符号化ストリームをスライス単位で読み出して復号する処理をコンピュータに実行させるプログラムが記録されている。

【００８０】

本発明の復号装置および方法、並びに記録媒体のプログラムにおいては、符号化ストリームが入力され、複数の復号手段が並行して動作されるように制御されて、バッファリングされている符号化ストリームがスライス単位で読み出されて復号される。さらに、複数の復号手段のそれぞれから復号結果として出力された複数の画像データのうち、複数の復号手段からの要求に対応したものが選択され、選択された画像データに対して必要に応じて動き補償処理が施され、必要に応じて動き補償処理が施された画像データが任意の再生速度で出力される。なお、復号制御の処理では、復号手段がスライス単位で復号対象とする符号化ストリームがバッファリングされている位置を示す書き込みポインタが複数の復号手段のうちのいずれかに供給され、復号の処理は、供給された書込みポインタに従い、バッファリングされている符号化ストリームがスライス単位で読み出されて復号される。

【００８２】

前記符号化ストリームは、MPEG２ビデオビットストリームとすることができる。

【００８３】

前記復号手段は、復号処理が終了したことを示す終了信号を前記復号制御手段に出力し、前記復号制御手段は、前記終了信号を出力した前記復号手段に対して、他の前記符号化ストリームを復号させるように制御するようにすることができる。

【００８４】

本発明の第２の復号装置は、符号化ストリームをバッファリングする第１のバッファ手段と、符号化ストリームから、符号化ストリームに含まれる所定の情報の単位の始まりを表わすスタートコードを読み出すとともに、第１のバッファ手段から、スタートコードが保持されている位置に関する位置情報を読み出す読み出し手段と、読み出されたスタートコードおよび位置情報をバッファリングする第２のバッファ手段と、第１のバッファ手段による符号化ストリームのバッファリング、および第２のバッファ手段によるスタートコードおよび位置情報のバッファリングを制御するバッファリング制御手段とをさらに含むことができる。

【００８５】

前記復号手段は、復号処理が終了したことを示す終了信号を前記選択手段に出力し、前記選択手段は、複数の復号手段それぞれの処理状態に対応する複数の処理状態値を記憶する記憶手段を含むことができ、記憶している全ての処理状態値が第１の値になった場合、終了信号を出力している復号手段に対応する処理状態値を、第１の値から第２の値に変更し、対応する処理状態値が第２の値である復号手段により復号された画像データのうち、いずれかの画像データを選択し、選択された画像データを復号した復号手段に対応する処理状態値を第１の値に変更するようにすることができる。

【００８６】

本発明の第２の復号装置は、選択手段により選択された画像データ、または動き補償手段により動き補償が施された画像データを保持する保持手段と、選択手段により選択された画像データ、または動き補償手段により動き補償が施された画像データの保持手段による保持を制御する保持制御手段とをさらに含むことができる。

【００８７】

前記保持手段は、画像データの輝度成分と色差成分をそれぞれ分けて保持するようにすることができる。

【００８８】

本発明の第２の復号装置は、複数の符号化ストリームが多重化されている多重ストリームの入力を受け付ける受付手段と、多重ストリームを複数の符号化ストリームに分離して、入力手段に供給する供給手段とをさらに含むことができる。

【００８９】

前記復号手段は、スライスデコーダとすることができる。

【０１０１】

【発明の実施の形態】

本発明を適用したMPEGビデオデコーダの第１の構成例について、図１８を参照して説明する。

【０１０２】

このMPEGビデオデコーダ４０は、ストリーム入力回路４１、スタートコード検出回路４２、ストリームバッファ制御回路４３、クロック発生回路４４、ピクチャデコーダ４５、スライスデコーダ制御回路４６、スライスデコーダ４７乃至４９、動き補償回路５０、輝度バッファ制御回路５１、色差バッファ制御回路５２、および表示出力回路５３から構成されるＩＣ３１、ストリームバッファ６１およびスタートコードバッファ６２で構成され、例えば、DRAMからなるバッファ３２、例えば、DRAMからなる輝度バッファ７１、例えば、DRAMからなる色差バッファ７２、CPU(Central Processing Unit)などよりなるコントローラ３４、並びに、ドライブ３５で構成される。

【０１０３】

ストリーム入力回路４１は、高能率符号化された符号化ストリーム（MPEG２ビデオビットストリーム）の入力を受け、スタートコード検出回路４２に供給する。スタートコード検出回路４２は、入力された符号化ストリームをストリームバッファ制御回路４３に供給するとともに、図１４を用いて説明したスタートコードを検出して、それを基に、そのスタートコードの種類と、ストリームバッファ６１にそのスタートコードが書き込まれる位置を示す書き込みポインタとを含む、スタートコード情報を生成し、ストリームバッファ制御回路４３に供給する。

【０１０４】

クロック発生回路４４は、図１５を用いて説明したクロック発生回路１３の２倍の周期の基本クロックを発生し、ストリームバッファ制御回路４３に供給する。ストリームバッファ制御回路４３は、クロック発生回路４４から供給される基本クロックに従って、入力された符号化ストリームを、バッファ３２のストリームバッファ６１に書き込み、入力されたスタートコード情報を、バッファ３２のスタートコードバッファ６２に書き込む。ストリームバッファ６１は、少なくとも4:2:2P@HLのデコードに要求されるVBVバッファサイズである47,185,920ビットの容量を有している。

【０１０５】
ピクチャデコーダ４５は、ストリームバッファ制御回路４３を介して、スタートコードバッファ６２からスタートコード情報を読み出す。例えば、デコード開始時は、図３を用いて説明したsequence_headerからデコードが開始されるので、ピクチャデコーダ４５は、図１４を用いて説明したスタートコードであるsequence_header_codeに対応する書き込みポインタをスタートコードバッファ６２から読み出し、その書き込みポインタを基に、ストリームバッファ６１からsequence_headerを読み出してデコードする。続いて、ピクチャデコーダ４５は、sequence_headerの読み出しと同様に、sequence_extension, GOP_header, picture_coding_extension等をストリームバッファ６１から読み出してデコードする。

【０１０６】
ピクチャデコーダ４５が、スタートコードバッファ６２から、最初のslice_start_codeを読み出した時点で、そのピクチャのデコードに必要な全てのパラメータが揃ったことになる。ピクチャデコーダ４５は、デコードしたピクチャ層のパラメータを、スライスデコーダ制御回路４６に出力する。

【０１０７】
スライスデコーダ制御回路４６は、ピクチャ層のパラメータの入力を受け、符号化ストリームのクラス（4:2:2@ML、MP@ML等）を判別する。スライスデコーダ制御回路４６はまた、ピクチャ層のパラメータの入力を受け、ストリームバッファ制御回路４３を介して、スタートコードバッファ６２から、対応するスライスのスタートコード情報を読み出す。さらに、スライスデコーダ制御回路４６は、スライスデコーダ４７乃至４９のいずれかにデコードさせるスライスが、符号化ストリームに含まれる何番目のスライスであるかを示すレジスタを有し、そのレジスタを参照しながら、ピクチャ層のパラメータと、スタートコード情報に含まれるスライスの書き込みポインタをスライスデコーダ４７乃至４９のいずれかに供給する。スライスデコーダ制御回路４６が、スライスデコーダ４７乃至４９のうち、デコードを実行させるスライスデコーダを選択する処理については、図１９および図２０を用いて後述する。

【０１０８】
スライスデコーダ４７は、マクロブロック検出回路８１、ベクトル復号回路８２、逆量子化回路８３、および逆DCT回路８４で構成され、スライスデコーダ制御回路４６から入力されたスライスの書き込みポインタを基に、対応するスライスを、ストリームバッファ制御回路４３を介してストリームバッファ６１から読み出す。そして、スライスデコーダ制御回路４６から入力されたピクチャ層のパラメータに従って、読み出したスライスをデコードして、動き補償回路５０に出力する。

【０１０９】
マクロブロック検出回路８１は、スライス層のマクロブロックを分離し、各マクロブロックのパラメータをデコードし、可変長符号化された各マクロブロックの予測モードおよび予測ベクトルをベクトル復号回路８２に供給し、可変長符号化された係数データを逆量子化回路８３に供給する。ベクトル復号回路８２は、可変長符号化された、各マクロブロックの予測モードおよび予測ベクトルをデコードして、予測ベクトルを復元する。逆量子化回路８３は、可変長符号化された係数データをデコードして逆DCT回路８４に供給する。逆DCT回路８４は、デコードされた係数データに逆DCTを施し、符号化前の画素データに復元する。

【０１１０】
スライスデコーダ４７は、動き補償回路５０に、デコードしたマクロブロックに対する動き補償の実行を要求し（すなわち、図中、REQで示される信号を１にする）、動き補償回路５０から動き補償の実行要求に対する受付を示す信号（図中ACKで示される信号）を受けて、デコードされた予測ベクトルおよびデコードされた画素を動き補償回路５０に供給する。スライスデコーダ４７は、ACK信号の入力を受けて、デコードされた予測ベクトルおよびデコードされた画素を動き補償回路５０に供給した後に、REQ信号を１から０に変

更する。そして、次に入力されたマクロブロックのデコードが終了した時点で、REQ信号を、再び０から１に変更する。

【０１１１】

また、スライスデコーダ４８のマクロブロック検出回路８５乃至逆DCT回路８８およびスライスデコーダ４９のマクロブロック検出回路８９乃至逆DCT回路９２においても、スライスデコーダ４７のマクロブロック検出回路８１乃至逆DCT回路８４と同様の処理が行われるので、その説明は省略する。

【０１１２】

動き補償回路５０は、スライスデコーダ４７乃至４９から入力されたデータの動き補償が終了したか否かを示すReg_REQ_A，Reg_REQ_BおよびReg_REQ_Cの３つのレジスタを有し、これらのレジスタの値を参照しながら、適宜、スライスデコーダ４７乃至４９のうちの１つを選択して、動き補償実行要求を受け付け（すなわち、REQ信号に対して、ACK信号を出力して、予測ベクトルと画素の入力を受ける）、動き補償処理を実行する。このとき、動き補償回路５０は、スライスデコーダ４７乃至４９のうち、所定のタイミングにおいてREQ信号が１であるスライスデコーダ４７乃至４９に対する動き補償が、それぞれ１回ずつ終了した後に、次の動き補償要求を受け付ける。例えば、スライスデコーダ４７が連続して動き補償要求を出しても、スライスデコーダ４８およびスライスデコーダ４９の動き補償が終了するまで、スライスデコーダ４７の２つ目の動き補償要求は受け付けられない。動き補償回路５０が、スライスデコーダ４７乃至４９のいずれのデコーダの出力に対して動き補償を実行するかを選択する処理については、図２１および図２２を用いて後述する。

【０１１３】

スライスデコーダ４７乃至４９のいずれかから入力されるマクロブロックが動き補償を使用していない場合、動き補償回路５０は、その画素データが輝度データであれば、輝度バッファ制御回路５１を介して輝度バッファ７１に書き込み、その画素データが色差データであれば、色差バッファ制御回路５２を介して色差バッファ７２に書き込み、表示出力に備えるとともに、この画素データが、他の画像の参照データとされる場合に備える。

【０１１４】

また、スライスデコーダ４７乃至４９のいずれかから出力されるマクロブロックが動き補償を使用している場合、動き補償回路５０は、スライスデコーダ４７乃至４９のうち対応するデコーダから入力される予測ベクトルに従って、その画素データが輝度データであれば、輝度バッファ制御回路５１を介して、輝度バッファ７１から参照画素を読み込み、その画素データが色差データであれば、色差バッファ制御回路５２を介して、色差バッファ７２から参照画素データを読み込む。そして、動き補償回路５０は、読み込んだ参照画素データを、スライスデコーダ４７乃至４９のいずれかから供給された画素データに加算し、動き補償を行う。

【０１１５】

動き補償回路５０は、動き補償を行った画素データを、その画素データが輝度データであれば、輝度バッファ制御回路５１を介して、輝度バッファ７１に書き込み、その画素データが色差データであれば、色差バッファ制御回路５２を介して、色差バッファ７２に書き込み、表示出力に備えるとともに、この画素データが、他の画素の参照データとされる場合に備える。

【０１１６】

表示出力回路５３は、デコードされた画像データを出力するための同期タイミング信号を発生し、このタイミングに従って、輝度バッファ制御回路５１を介して、輝度バッファ７１から輝度データを読み出し、色差バッファ制御回路５２を介して、色差バッファ７２から色差データを読み出して、復号ビデオ信号として出力する。

【０１１７】

ドライブ３５は、コントローラ３４に接続されており、必要に応じて装着される磁気ディスク１０１、光ディスク１０２、光磁気ディスク１０３、および半導体メモリ１０４など

とデータの授受を行う。また、コントローラ３４は、以上説明したＩＣ３１、およびドライブ３５の動作を制御するものである。コントローラ３４は、例えば、ドライブ３５に装着されている磁気ディスク１０１、光ディスク１０２、光磁気ディスク１０３、および半導体メモリ１０４などに記録されているプログラムに従って、ＩＣ３１に処理を実行させることができる。

【０１１８】

次に、図１９のフローチャートを参照して、スライスデコーダ制御回路４６の処理について説明する。

【０１１９】

ステップＳ１において、スライスデコーダ制御回路４６は、画像の垂直方向のマクロブロック数を設定した後、処理するスライスが画像内の何番目のスライスであるかを表わすレジスタの値Ｎを１に初期化する。ステップＳ２において、スライスデコーダ制御回路４６は、スライスデコーダ４７が処理中であるか否かを判断する。

【０１２０】

ステップＳ２において、スライスデコーダ４７が処理中ではないと判断された場合、ステップＳ３に進む。ステップＳ３において、スライスデコーダ制御回路４６は、ピクチャ層のパラメータと、スタートコード情報に含まれるスライスＮの書き込みポインタをスライスデコーダ４７に供給し、スライスデコーダ４７にスライスＮをデコードさせ、処理はステップＳ８に進む。

【０１２１】

ステップＳ２において、スライスデコーダ４７が処理中であると判断された場合、ステップＳ４に進む。ステップＳ４において、スライスデコーダ制御回路４６は、スライスデコーダ４８が処理中であるか否かを判断する。ステップＳ４において、スライスデコーダ４８が処理中ではないと判断された場合、ステップＳ５に進む。ステップＳ５において、スライスデコーダ制御回路４６は、ピクチャ層のパラメータと、スタートコード情報に含まれるスライスＮの書き込みポインタをスライスデコーダ４８に供給し、スライスデコーダ４８にスライスＮをデコードさせ、処理はステップＳ８に進む。

【０１２２】

ステップＳ４において、スライスデコーダ４８が処理中であると判断された場合、ステップＳ６に進む。ステップＳ６において、スライスデコーダ制御回路４６は、スライスデコーダ４９が処理中であるか否かを判断する。ステップＳ６において、スライスデコーダ４９が処理中であると判断された場合、ステップＳ２に戻り、それ以降の処理が繰り返される。

【０１２３】

ステップＳ６において、スライスデコーダ４９が処理中ではないと判断された場合、ステップＳ７に進む。ステップＳ７において、スライスデコーダ制御回路４６は、ピクチャ層のパラメータと、スタートコード情報に含まれるスライスＮの書き込みポインタをスライスデコーダ４９に供給し、スライスデコーダ４９にスライスＮをデコードさせ、処理はステップＳ８に進む。

【０１２４】

ステップＳ８において、スライスデコーダ制御回路４６は、処理するスライスが符号化ストリームの何番目のスライスであるかを示すレジスタの値Ｎを１だけインクリメントする。ステップＳ９において、スライスデコーダ制御回路４６は、全スライスのデコードが終了したか否かを判断する。ステップＳ９において、全スライスのデコードが終了されていないと判断された場合、ステップＳ２に戻り、それ以降の処理が繰り返される。ステップＳ９において、全スライスのデコードが終了されたと判断された場合、スライスデコーダ制御回路４６の当該処理は終了される。

【０１２５】

図１９を用いて説明したスライスデコーダ制御回路４６の処理の具体例について、図２０を参照して説明する。上述したように、ピクチャデコーダ４５でピクチャ層のデータがデ

コードされ、そのパラメータがスライスデコーダ制御回路４６に供給される。ここで、図１９を用いて説明したステップＳ１において、スライスデコーダ制御回路４６は、画像の垂直方向のマクロブロック数（レジスタの値Ｎが取り得る最大値）を設定した後、レジスタの値Ｎを１に初期化する。ステップＳ２において、スライスデコーダ４７は処理中ではないと判断されるので、ステップＳ３において、スライスデコーダ制御回路４６は、ピクチャ層のパラメータと、スタートコード情報に含まれるスライス１の書き込みポインタをスライスデコーダ４７に供給し、スライスデコーダ４７にスライスＮ（Ｎ＝１）をデコードさせ、ステップＳ８において、レジスタの値Ｎを１だけインクリメントする。そして、ステップＳ９において、全スライスのデコードが終了していないと判断されるため、処理はステップＳ２に戻る。

【０１２６】

ステップＳ２において、スライスデコーダ４７は処理中であると判断される。そして、ステップＳ４において、スライスデコーダ４８は処理中でないと判断されるので、ステップＳ５において、スライスデコーダ制御回路４６は、ピクチャ層のパラメータと、スライス２の書き込みポインタを、スライスデコーダ４８に供給し、スライスデコーダ４８にスライスＮ（Ｎ＝２）をデコードさせ、ステップＳ８において、Ｎを１だけインクリメントする。そして、ステップＳ９において、全スライスのデコードが終了していないと判断されるため、処理はステップＳ２に戻る。

【０１２７】

ステップＳ２において、スライスデコーダ４７は処理中であると判断され、ステップＳ４において、スライスデコーダ４８は処理中であると判断される。そして、ステップＳ６において、スライスデコーダ４９は処理中ではないと判断されるので、ステップＳ７において、スライスデコーダ制御回路４６は、ピクチャ層のパラメータと、スライス３の書き込みポインタを、スライスデコーダ４９に供給し、スライスデコーダ４９にスライスＮ（Ｎ＝３）をデコードさせ、ステップＳ８において、Ｎを１だけインクリメントする。そして、ステップＳ９において、全スライスのデコードが終了していないと判断されるため、処理はステップＳ２に戻る。

【０１２８】

スライスデコーダ４７乃至４９は、入力されたスライスのデコード処理を実施した後、デコード処理の完了を示す信号をスライスデコーダ制御回路４６に出力する。すなわち、スライスデコーダ４７乃至４９のいずれかからスライスのデコードの完了を示す信号が入力されるまで、スライスデコーダ４７乃至４９は全て処理中であるので、ステップＳ２、ステップＳ４、およびステップＳ６の処理が繰り返される。そして、図２０の図中Ａで示されるタイミングで、スライスデコーダ４８がデコード処理の完了を示す信号を、スライスデコーダ制御回路４６に出力した場合、ステップＳ４において、スライスデコーダ４８が処理中ではないと判断されるので、ステップＳ５において、スライスデコーダ制御回路４６は、スライス４の書き込みポインタを、スライスデコーダ４８に供給し、スライスデコーダ４８に、スライスＮ（Ｎ＝４）をデコードさせ、ステップＳ８において、Ｎを１だけインクリメントする。そして、ステップＳ９において、全スライスのデコードが終了していないと判断されるため、処理はステップＳ２に戻る。

【０１２９】

そして、次にスライスデコーダ４７乃至４９のいずれかからデコード処理の完了を示す信号の入力を受けるまで、スライスデコーダ制御回路４６は、ステップＳ２、ステップＳ４、およびステップＳ６の処理を繰り返す。図２０においては、スライスデコーダ制御回路４６は、図中Ｂで示されるタイミングで、スライスデコーダ４９からスライス３のデコードの終了を示す信号の入力を受けるので、ステップＳ６において、スライスデコーダ４９は処理中ではないと判断される。ステップＳ７において、スライスデコーダ制御回路４６は、スライス５の書き込みポインタをスライスデコーダ４９に供給し、スライスデコーダ４９に、スライスＮ（Ｎ＝５）をデコードさせ、ステップＳ８において、Ｎを１だけインクリメントする。そして、ステップＳ９において、全スライスのデコードが終了していな

いと判断されるため、処理はステップＳ２に戻る。以下、最後のスライスのデコードが終了されるまで、同様の処理が繰り返される。

【０１３０】

このように、スライスデコーダ制御回路４６は、スライスデコーダ４７乃至４９の処理状況に対応してスライスのデコード処理を割り当てるので、スライスデコーダ４７乃至４９に効率よくデコード処理を実行させることが可能となる。

【０１３１】

次に、動き補償回路５０のスライスデコーダ調停処理について、図２１のフローチャートを参照して説明する。

【０１３２】

ステップＳ２１において、動き補償回路５０は、内部のレジスタReg_REQ_A，Reg_REQ_B、およびReg_REQ_Cを初期化する。すなわち、Reg_REQ_A＝０，Reg_REQ_B＝０、およびReg_REQ_C＝０とする。

【０１３３】

ステップＳ２２において、動き補償回路５０は、レジスタの値が全て０であるか否かを判断する。ステップＳ２２において、レジスタの値が全て０ではない（すなわち、１つでも１がある）と判断された場合、ステップＳ２４に進む。

【０１３４】

ステップＳ２２において、レジスタの値が全て０であると判断された場合、ステップＳ２３に進む。ステップＳ２３において、動き補償回路５０は、スライスデコーダ４７乃至４９から入力されるREQ信号を基に、レジスタの値を更新する。すなわち、スライスデコーダ４７からREQ信号が出力されている場合、Reg_REQ_A＝１とし、スライスデコーダ４８からREQ信号が出力されている場合、Reg_REQ_B＝１とし、スライスデコーダ４９からREQ信号が出力されている場合、Reg_REQ_C＝１とする。そして、処理はステップＳ２４に進む。

【０１３５】

ステップＳ２４において、動き補償回路５０は、Reg_REQ_A＝１であるか否かを判断する。ステップＳ２４において、Reg_REQ_A＝１であると判断された場合、ステップＳ２５に進む。ステップＳ２５において、動き補償回路５０は、スライスデコーダ４７にACK信号を送信し、Reg_REQ_A＝０とする。スライスデコーダ４７は、動き補償回路５０に、ベクトル復号回路８２で復号された予測ベクトルと、逆DCT回路８４で逆DCTされた画素を出力する。そして、処理はステップＳ３０に進む。

【０１３６】

ステップＳ２４において、Reg_REQ_A＝１ではないと判断された場合、ステップＳ２６に進む。ステップＳ２６において、動き補償回路５０は、Reg_REQ_B＝１であるか否かを判断する。ステップＳ２６において、Reg_REQ_B＝１であると判断された場合、ステップＳ２７に進む。ステップＳ２７において、動き補償回路５０は、スライスデコーダ４８にACK信号を送信し、Reg_REQ_B＝０とする。スライスデコーダ４８は、動き補償回路５０に、ベクトル復号回路８６で復号された予測ベクトルと、逆DCT回路８８で逆DCTされた画素を出力する。そして、処理はステップＳ３０に進む。

【０１３７】

ステップＳ２６において、Reg_REQ_B＝１ではないと判断された場合、ステップＳ２８に進む。ステップＳ２８において、動き補償回路５０は、Reg_REQ_C＝１であるか否かを判断する。ステップＳ２８において、Reg_REQ_C＝１ではないと判断された場合、ステップＳ２２に戻り、それ以降の処理が繰り返される。

【０１３８】

ステップＳ２８において、Reg_REQ_C＝１であると判断された場合、ステップＳ２９に進む。ステップＳ２９において、動き補償回路５０は、スライスデコーダ４９にACK信号を送信し、Reg_REQ_C＝０とする。スライスデコーダ４９は、動き補償回路５０に、ベクトル復号回路９０で復号された予測ベクトルと、逆DCT回路９２で逆DCTされた画素を出力す

る。そして、処理はステップＳ３０に進む。

【０１３９】

ステップＳ３０において、動き補償回路５０は、スライスデコーダ４７乃至４９のいずれかから入力されたマクロブロックは、動き補償を使用しているか否かを判断する。

【０１４０】

ステップＳ３０において、マクロブロックが動き補償を使用していると判断された場合、ステップＳ３１に進む。ステップＳ３１において、動き補償回路５０は、入力されたマクロブロックに動き補償処理を行う。すなわち、動き補償回路５０は、スライスデコーダ４７乃至４９のうち対応するデコーダから出力される予測ベクトルに従って、その画素データが輝度データであれば、輝度バッファ制御回路５１を介して、輝度バッファ７１から参照画素を読み出し、その画素データが色差データであれば、色差バッファ制御回路５２を介して、色差バッファ７２から参照画素データを読み出す。そして、動き補償回路５０は、読み出した参照画素データを、スライスデコーダ４７乃至４９のいずれかから供給された画素データに加算し、動き補償を行う。

【０１４１】

動き補償回路５０は、動き補償を行った画素データを、その画素データが輝度データであれば、輝度バッファ制御回路５１を介して、輝度バッファ７１に書き込み、その画素データが色差データであれば、色差バッファ制御回路５２を介して、色差バッファ７２に書き込み、表示出力に備えるとともに、この画素データが、他の画素の参照データとされる場合に備える。そして、ステップＳ２２に戻り、それ以降の処理が繰り返される。

【０１４２】

ステップＳ３０において、マクロブロックが動き補償を使用していないと判断された場合、ステップＳ３２に進む。ステップＳ３２において、動き補償回路５０は、その画素データが輝度データであれば、輝度バッファ制御回路５１を介して輝度バッファ７１に書き込み、その画素データが色差データであれば、色差バッファ制御回路５２を介して色差バッファ７２に書き込み、表示出力に備えるとともに、この画素データが、他の画像の参照データとされる場合に備える。そして、ステップＳ２２に戻り、それ以降の処理が繰り返される。

【０１４３】

図２１を用いて説明した動き補償回路５０によるデコーダの調停処理の具体例について、図２２を参照して説明する。

【０１４４】

図２２に示すタイミングＣにおいて、図２１のステップＳ２２の処理により、動き補償回路５０のレジスタが全て０であると判断された場合、スライスデコーダ４７乃至４９は、全て、REQ信号を出力しているため、ステップＳ２３の処理により、それぞれのレジスタの値は、Reg_REQ_A＝１，Reg_REQ_B＝１、またはReg_REQ_C＝１に更新される。そして、ステップＳ２４の処理により、Reg_REQ_A＝１であると判断されるため、ステップＳ２５において、動き補償回路５０は、スライスデコーダ４７にACK信号を出力して、Reg_REQ_A＝０とし、スライスデコーダ４７から予測ベクトルと画素の入力を受け、動き補償１を行う。

【０１４５】

動き補償１が終了した後、すなわち、図２２のＤで示されるタイミングにおいて、処理は、再びステップＳ２２に戻る。図中Ｄで示されるタイミングにおいては、スライスデコーダ４７から、REQ信号が出力されている。しかし、レジスタの値は、Reg_REQ_A＝０，Reg_REQ_B＝１，Reg_REQ_C＝１であり、ステップＳ２２において、レジスタの値は、全て０ではないと判断されるため、処理は、ステップＳ２４に進み、レジスタの値は更新されない。

【０１４６】

ステップＳ２４において、Reg_REQ_A＝０であると判断され、ステップＳ２６において、Reg_REQ_B＝１であると判断されるので、動き補償回路５０は、ステップＳ２７において、

スライスデコーダ４８にACK信号を出力して、Reg_REQ_B＝０とし、スライスデコーダ４８から予測ベクトルと画素の入力を受け、動き補償２を行う。

【０１４７】

動き補償２が終了した後、すなわち、図２２のＥで示されるタイミングにおいて、処理は再びステップＳ２２に戻る。図中Ｅで示されるタイミングにおいても、スライスデコーダ４７から、REQ信号が出力されている。しかし、レジスタの値は、Reg_REQ_A＝０，Reg_REQ_B＝０，Reg_REQ_C＝１であるので、ステップＳ２２において、レジスタの値は全て０ではないと判断されるので、図中Ｄで示されるタイミングのときと同様、レジスタの値は更新されない。

【０１４８】

そして、ステップＳ２４において、Reg_REQ_A＝０であると判断され、ステップＳ２６において、Reg_REQ_B＝０であると判断され、ステップＳ２８において、Reg_REQ_C＝１であると判断されるので、動き補償回路５０は、ステップＳ２９において、スライスデコーダ４９にACK信号を出力して、Reg_REQ_C＝０とし、スライスデコーダ４９から予測ベクトルと画素の入力を受け、動き補償３を行う。

【０１４９】

動き補償３が終了した後、すなわち、図２２のＦで示されるタイミングにおいて、処理は再びステップＳ２２に戻る。Ｆで示されるタイミングにおいては、レジスタの値は、Reg_REQ_A＝０，Reg_REQ_B＝０，Reg_REQ_C＝０であるので、ステップＳ２３において、レジスタの値が更新され、Reg_REQ_A＝１，Reg_REQ_B＝１，Reg_REQ_C＝０となる。

【０１５０】

そして、ステップＳ２４において、Reg_REQ_A＝１であると判断され、同様の処理により、動き補償４が実行される。

【０１５１】

このような処理を繰り返すことにより、動き補償回路５０は、スライスデコーダ４７乃至４９を調停しながら、動き補償を行う。

【０１５２】

以上説明したように、本発明を適用したMPEGビデオデコーダ４０においては、スタートコードバッファ６２を設けたことにより、ピクチャデコーダ４５乃至スライスデコーダ４９を、お互いの動作の終了を待つことなしに、ストリームバッファ６１にアクセスさせることができる。また、スライスデコーダ４７乃至４９は、スライスデコーダ制御回路４６の処理により、同時に動作させることができる。さらに、動き補償回路５０は、適宜、スライスデコーダ４７乃至４９のうちの１つを選択し、それぞれ分離された輝度バッファ７１および色差バッファ７２にアクセスし、動き補償を行うことができる。このように、MPEGビデオデコーダ４０は、デコード処理性能およびバッファへのアクセス性能が向上されているので、4:2:2P@HLのMPEG２ビデオビットストリームを実時間でデコードすることが可能となる。

【０１５３】

ところで、MP@MLのMPEG２ビデオビットストリームを実時間でデコードするために要する処理能力は、4:2:2P@HLのMPEG２ビデオビットストリームを実時間でデコードするために要する処理能力の１／６である。よって換言すれば、本発明を適用したMPEGビデオデコーダ４０は、MP@MLのMPEG２ビデオビットストリームを最大６倍速で再生することが可能である。

【０１５４】

具体的には、例えば、図２３に示すシステムにMPEGビデオデコーダ４０を用いることにより、MP@MLのMPEG２ビデオビットストリームの高速再生が実現される。

【０１５５】

当該システムのハードディスクドライブ(HDD)１１１には、MP@MLのMPEG２ビデオビットストリームが記録されている。再生装置１１２は、MPEGビデオデコーダ４０のコントローラ３４からの制御に基づき、ハードディスクドライブ１１１に記録されているMP@MLのMPEG

２ビデオビットストリームを、通常（１倍速）よりも高速（例えば６倍速）で読み出し、得られた高速再生ストリームをMPEGビデオデコーダ４０に供給する。

【０１５６】

例えば、図２４（Ａ）に示すようなMP@MLのMPEG２ビデオビットストリームは、MPEGビデオデコーダ４０により、一切の処理が省略されることなく完全にデコードされ、図２４（Ｂ）に示すように輝度バッファ７１および色差バッファ７２に書き込まれる。輝度バッファ７１および色差バッファ７２に書き込まれた画像データを、表示出力回路５３が画像のタイプに拘わりなく６枚当たり１枚の割合で読み出し、復号ビデオ出力として後段に出力することにより、６倍速の高速再生が実現される。

【０１５７】

なお、表示出力回路５３が、輝度バッファ７１および色差バッファ７２に書き込まれた画像データを、例えば、画像のタイプに拘わりなく３枚当たり１枚の割合で読み出し、復号ビデオ出力として後段に出力すれば、３倍速の高速再生が実現される。

【０１５８】

すなわち、表示出力回路５３が、輝度バッファ７１および色差バッファ７２に書き込まれた画像データを画像のタイプに拘わりなく、Ｘ（＝０乃至６）枚当たり１枚の割合で読み出し、復号ビデオ出力として後段に出力すれば、Ｘ倍速の高速再生が実現される。ここで、Ｘ＝０の場合、スチル再生となる。

【０１５９】

以上説明したように、MPEGビデオデコーダ４０においては、表示出力回路５３が、画像のタイプに拘わりなく一定の割合でデータを読み出すことによって、見た目にもギクシャクしていない自然な動きの画像を出力することが可能となる。

【０１６０】

このようにMP@MLのMPEG２ビデオビットストリームを高速再生できるMPEGビデオデコーダ４０を映像編集等に用いれば、例えば、ビデオ信号の映像素材の内容を容易に理解することができ、さらに、編集点等を快適に検索することができるので、作業効率を向上させることが可能となる。

【０１６１】

ところでまた、図１８に示したMPEGビデオデコーダ４０に、さらに表示出力回路５３と同様の回路を２個追加して表示出力を３系統とすれば、入力された６倍速のMP@MLのMPEG２ビデオビットストリームを３倍速で再生して、得られる復号ビデオ出力を当該３系統で順次出力するようにすることが可能である。

【０１６２】

また、本発明は、２：１インタレースであって３０フレーム／秒のMP@MLのMPEG２ビデオビットストリームを２倍速で記録媒体から読み出し、ノンインタレースであって６０フレーム／秒の画像として出力するような場合にも適用することが可能である。

【０１６３】

なお、図２３に示したシステムにおいては、ハードディスクドライブ１１１にMP@MLのMPEG２ビデオビットストリームを記録するようにしたが、記録されているMP@MLのMPEG２ビデオビットストリームを６倍速で読み出せるものであれば、ハードディスクドライブ１１１の代わりに他の記録再生デバイスを用いてもよい。

【０１６４】

次に、本発明を適用したMPEGビデオデコーダの第２の構成例について、図２５を参照して説明する。このMPEGビデオデコーダ１３０は、入力される多チャンネル分の符号化ストリーム（いまの場合、チャンネルＣＨ１の符号化ストリーム入力、およびチャンネルＣＨ２の符号化ストリーム入力）を平行してデコードし、得られるチャンネルＣＨ１ビデオ出力、およびチャンネルＣＨ２ビデオ出力を後段に供給するものである。また、MPEGビデオデコーダ１３０は、図１８のMPEGビデオデコーダ４０と同様に、図１５に示した従来のMPEGビデオデコーダの６倍のデコード処理能力を有する。

【０１６５】

MPEGビデオデコーダ１３０とMPEGビデオデコーダ４０との構成上の違いについて説明する。MPEGビデオデコーダ１３０においては、MPEGビデオデコーダ４０のストリーム入力回路４１、スタートコード検出回路４２、および表示出力回路５３が、それぞれ、チャンネルＣＨ１用のストリーム入力回路４１－１、スタートコード検出回路４２－１、および表示出力回路５３－１、並びにチャンネルＣＨ２用のストリーム入力回路４１－２、スタートコード検出回路４２－２、および表示出力回路５３－２に置換されている。さらに、MPEGビデオデコーダ１３０においては、バッファ３２のストリームバッファ６１およびスタートコードバッファ６２、輝度バッファ７１、および色差バッファ７２に、それぞれチャンネルＣＨ１用の領域とチャンネルＣＨ２用の領域が設けられている。

【０１６６】

ストリーム入力回路４１－１は、高能率符号化されたチャンネルＣＨ１の符号化ストリーム（MPEG２ビデオビットストリーム）の入力を受け、スタートコード検出回路４２－１に供給する。スタートコード検出回路４２－１は、入力されたチャンネルＣＨ１の符号化ストリームをストリームバッファ制御回路４３に供給するとともに、図１４を用いて説明したスタートコードを検出して、それを基に、そのスタートコードの種類と、ストリームバッファ６１のチャンネルＣＨ１用の領域にそのスタートコードが書き込まれる位置を示す書き込みポインタとを含む、スタートコード情報を生成し、ストリームバッファ制御回路４３に供給する。

【０１６７】

同様に、ストリーム入力回路４１－２は、高能率符号化されたチャンネルＣＨ２の符号化ストリーム（MPEG２ビデオビットストリーム）の入力を受け、スタートコード検出回路４２－２に供給する。スタートコード検出回路４２－２は、入力されたチャンネルＣＨ２の符号化ストリームをストリームバッファ制御回路４３に供給するとともに、図１４を用いて説明したスタートコードを検出して、それを基に、そのスタートコードの種類と、ストリームバッファ６１のチャンネルＣＨ２用の領域にそのスタートコードが書き込まれる位置を示す書き込みポインタとを含む、スタートコード情報を生成し、ストリームバッファ制御回路４３に供給する。

【０１６８】

表示出力回路５３－１は、デコードされたチャンネルＣＨ１の画像データを出力するための同期タイミング信号を発生し、このタイミングに従って、輝度バッファ制御回路５１を介して、輝度バッファ７１から輝度データを読み出し、色差バッファ制御回路５２を介して、色差バッファ７２から色差データを読み出して、チャンネルＣＨ１のビデオ出力として後段に供給する。

【０１６９】

同様に、表示出力回路５３－２は、デコードされたチャンネルＣＨ２の画像データを出力するための同期タイミング信号を発生し、このタイミングに従って、輝度バッファ制御回路５１を介して、輝度バッファ７１から輝度データを読み出し、色差バッファ制御回路５２を介して、色差バッファ７２から色差データを読み出して、チャンネルＣＨ２のビデオ出力として供給する。

【０１７０】

なお、MPEGビデオデコーダ１３０を構成する他の回路については、同一の番号が付与されたMPEGビデオデコーダ４０を構成する回路と同様であるので、それらの説明については省略する。

【０１７１】

次に、MPEGビデオデコーダ１３０がＬチャンネル分（いまの場合、Ｌ＝２）の符号化ストリームを平行してデコードするときのコントローラ３４によるパラレルでコード制御処理について、図２６のフローチャートを参照して説明する。

【０１７２】

ステップＳ５１において、コントローラ３４は、デコードするチャンネルを示す自己のレジスタＣＨを１に初期化する。ステップＳ５２において、コントローラ３４は、ピクチャ

デコーダ４５に対して、レジスタＣＨの値に対応するチャンネルのピクチャ層のデコードを指示する。ステップＳ５３において、コントローラ３４は、ピクチャデコーダ４５がレジスタＣＨの値に対応するチャンネル（いまの場合、チャンネルＣＨ１）のピクチャ層のデコードを完了したか否かを判定し、デコードを完了したと判定するまで待機する。

【０１７３】

この待機の間、コントローラ３４からの指示に対応して、ピクチャデコーダ４５は、ストリームバッファ制御回路４３を介してバッファ３２のスタートコードバッファ６２のチャンネルＣＨ１の領域からスタートコード情報を読み出し、デコードの開始点を示すsequence_headerの書き込みポインタを検出する。また、ピクチャデコーダ４５は、検出したチャンネルＣＨ１のsequence_headerの書き込みポインタに基づき、ストリームバッファ制御回路４３を介してバッファ３２のストリームバッファ６１のチャンネルＣＨ１の領域からsequence_headerを読み出してデコードする。以降、同様にして、ストリームバッファ６１のチャンネルＣＨ１の領域からsequence_extension，GOP_header，picture_header、およびpicture_coding_extensionを読み出してデコードする。さらに、ピクチャデコーダ４５は、スタートコードバッファ６２のチャンネルＣＨ１の領域からスタートコード情報を読み出し、最初のスライスの書き込みポインタを検出する。

【０１７４】

この段階、すなわち、チャンネルＣＨ１のピクチャのデコードに必要な全てのピクチャ層パラメータがそろった段階において、ピクチャデコーダ４５は、ピクチャ層のデコードが完了したことをコントローラ３４に通知する。

【０１７５】

この通知に基づき、ステップＳ５３において、コントローラ３４は、ピクチャ層のデコードが完了したと判定する。処理は、ステップＳ５４に進む。ステップＳ５４において、コントローラ３４は、ピクチャデコーダ４５からデコードされたチャンネルＣＨ１のピクチャ層パラメータを取得して保持する。ステップＳ５５において、コントローラ３４は、スライスデコーダ４６が準備完了であるか否か（以前に指示した処理を完了しているか否かなど）を判定し、スライスデコーダ４６が準備完了であると判定するまで待機する。スライスデコーダ４６が準備完了であると判定された場合、処理はステップＳ５６に進む。

【０１７６】

ステップＳ５６において、コントローラ３４は、ピクチャデコーダ４５から取得して保持しているチャンネルＣＨ１のピクチャ層パラメータをスライスデコーダ制御回路４６に供給する。ステップＳ５７において、コントローラ３４は、スライスデコーダ制御回路４６および動き補償回路５０に対して、チャンネルＣＨ１のスライス層のデコードを指示する。

【０１７７】

この指示に対応して、スライスデコーダ制御回路４６は、チャンネルＣＨ１の各スライスのデコードを、スライスデコーダ４７乃至４９に振り分ける。なお、その詳細は、図１９および図２０を参照して上述した処理と同様であるので説明を省略する。また、この指示に対応して、動き補償回路５０は、スライスデコーダ４７乃至４９からの要求を調停する。なお、その詳細は、図２１および図２２を参照して上述した処理と同様であるので説明を省略する。

【０１７８】

以上のような制御によって、チャンネルＣＨ１の１ピクチャ分のデコードが終了したとき、スライスデコーダ制御回路４６は、１ピクチャ分のデコードが終了したことをコントローラ３４に通知する。

【０１７９】

この通知に対応して、コントローラ３４は、ステップＳ５８において、レジスタＣＨの値を１だけインクリメントする。いまの場合、レジスタＣＨの値が１から２にインクリメントされる。ステップＳ５９において、コントローラ３４は、レジスタＣＨの値がＬよりも大きいか否かを判定する。レジスタＣＨの値がＬよりも大きいと判定されない場合、処理

はステップＳ５２に戻り、以降の処理が繰り返される。いまの場合、Ｌ＝２であってレジスタＣＨの値は２であるから、処理はステップＳ５２に戻り、チャンネルＣＨ２が処理対象となって、ステップＳ５２乃至ステップＳ５７の処理が実行される。

【０１８０】

そして、再びステップＳ５８において、レジスタＣＨの値が２から３にインクリメントされた場合、ステップＳ５９において、レジスタＣＨの値がＬよりも大きいと判定されて、処理はステップＳ６０に進む。ステップＳ６０において、コントローラ３４は、レジスタＣＨを１に初期化する。処理は、５２に戻り、以降の処理が繰り返される。

【０１８１】

以上説明したようにして、Ｌチャンネル分（いまの場合、２チャンネル分）の符号化ストリームが平行してデコードされる。

【０１８２】

なお、ステップＳ５４乃至ステップＳ５６において、コントローラ３４がピクチャデコーダ４５からピクチャ層パラメータを取得して保持し、保持したピクチャ層パラメータをスライスデコーダ制御回路４６に供給する処理に関し、当該処理をコントローラ３４の代わりに実行する回路をMPEGビデオデコーダ１３０に設けるようにしてもよい。

【０１８３】

MPEGビデオデコーダ１３０が１ピクチャ分のデコードに要する時間は、図１５に示した従来のMPEGビデオデコーダが実時間で１ピクチャ分のデコードに要する時間（すなわち、１倍速による処理時間）の１／６で済むことになるので、最大で６チャンネル分の符号化データを平行してデコードすることができる。

【０１８４】

図２７は、MPEGビデオデコーダ１３０を適用したMPEGビデオサーバ編集システムの構成例を示している。このMPEGビデオサーバ編集システムのハードディスクドライブ１１１には、多チャンネル分の複数のMPEG２ビデオビットストリームが記録されている。ただし、ハードディスクドライブ１１１の代わりに、光ディスクドライブ、光磁気ディスクドライブ、磁気テープドライブ、半導体メモリドライブなどの他の記録再生デバイスを用いてもよい。

【０１８５】

再生装置１１２は、コントローラ３４からの制御に基づき、ハードディスクドライブ１１１に記録されている複数のMPEG２ビデオビットストリームのうち、例えば、２チャンネル分のMPEG２ビデオビットストリーム（図２７におけるチャンネルＣＨ１の符号化ストリーム、およびチャンネルＣＨ２の符号化ストリーム）を読み出してMPEGビデオデコーダ１３０に供給する。MPEGビデオデコーダ１３０は、入力されたチャンネルＣＨ１およびＣＨ２の符号化ストリームを、上述したように平行してデコードし、得られたチャンネルＣＨ１およびＣＨ２のビデオ出力を後段に供給する。なお、デコードして得られたチャンネルＣＨ１およびＣＨ２のビデオ出力を適宜切り替えて後段に供給するようにしてもよい。

【０１８６】

次に、図２８は、本発明を適用したMPEGビデオデコーダの第３の構成例を示している。このMPEGビデオデコーダ１５０は、多重ストリームを入力として、多重ストリームに含まれる多チャンネル分の符号化ストリーム（いまの場合、チャンネルＣＨ１およびＣＨ２の符号化ストリーム（（MPEG２ビデオビットストリーム））を平行してデコードし、得られるチャンネルＣＨ１ビデオ出力、およびチャンネルＣＨ２ビデオ出力を後段に供給するものである。

【０１８７】

MPEGビデオデコーダ１５０は、図２３に示したMPEGビデオデコーダ１３０のストリーム入力回路４１－１，４１－２を、チャンネル分離回路１５１で置換したものである。チャンネル分離回路１５１は、入力される多重ストリーム（すなわち、MPEGビデオストリームが多重化されているトランストートストリーム）をチャンネルＣＨ１の符号化ストリームとチャンネルＣＨ２の符号化ストリームに分離して、チャンネルＣＨ１の符号化ストリーム

をスタートコード検出回路４２－１に供給し、チャンネルＣＨ２の符号化ストリームをスタートコード検出回路４２－２に供給する。

【０１８８】

なお、MPEGビデオデコーダ１５０を構成する他の回路については、同一の番号が付与されたMPEGビデオデコーダ１３０を構成する回路と同様であるので、それらの説明については省略する。また、MPEGビデオデコーダ１５０のスタートコード検出回路４２－１，４２－２以降の動作についても、MPEGビデオデコーダ１３０の動作と同様であるので、その説明は省略する。

【０１８９】

MPEGビデオデコーダ１５０の適用例について、図２９を参照して説明する。MPEGビデオデコーダ１５０は、図２５のMPEGビデオデコーダ１３０と同様に、図１５に示した従来のMPEGビデオデコーダの６倍のデコード処理能力を有する。したがって、MPEGビデオデコーダ１５０は、入力される多重ストリームに多重化されている多チャンネル分の符号化ストリームのうち、最大で６チャンネル分の符号化ストリームを平行してデコードすることができる。

【０１９０】

よって、例えば図２９に示すように、多重ストリームに含まれるチャンネルＣＨ１およびＣＨ２の符号化ストリームを選択して分離し、平行してデコードし、得られたチャンネルＣＨ１およびチャンネルＣＨ２ビデオ出力をビデオスイッチャ１６１に供給するようにすれば、チャンネルＣＨ１ビデオ出力の映像が徐々にチャンネルＣＨ２ビデオ出力に切り替えられるような、いわゆるワイプと呼ばれる映像合成に適用することが可能となる。

【０１９１】

上述した一連の処理は、ソフトウェアにより実行することもできる。そのソフトウェアは、そのソフトウェアを構成するプログラムが、専用のハードウェアに組み込まれているコンピュータ、または、各種のプログラムをインストールすることで、各種の機能を実行することが可能な、例えば汎用のパーソナルコンピュータなどに、記録媒体からインストールされる。

【０１９２】

この記録媒体は、図１８に示すように、コンピュータとは別に、ユーザにプログラムを提供するために配布される、プログラムが記録されている磁気ディスク１０１（フレキシブルディスクを含む）、光ディスク１０２（CD-ROM（Compact Disk Read Only Memory）、DVD（Digital Versatile Disk）を含む)、光磁気ディスク１０３（ＭＤ(Mini Disk)を含む）、もしくは半導体メモリ１０４などよりなるパッケージメディアなどにより構成される。

【０１９３】

また、本明細書において、記録媒体に記録されるプログラムを記述するステップは、記載された順序に沿って時系列的に行われる処理はもちろん、必ずしも時系列的に処理されなくとも、並列的あるいは個別に実行される処理をも含むものである。

【０１９４】

また、本明細書において、システムとは、複数の装置により構成される装置全体を表すものである。

【０１９５】

【発明の効果】

以上のように、本発明の第１の復号装置および方法、並びに記録媒体のプログラムによれば、実現可能な回路規模であって、MP@MLのMPEG２ビデオビットストリームを任意の再生速度で再生できるビデオデコーダを実現することが可能となる。

【０１９７】

また、本発明の第２の復号装置および方法、並びに記録媒体のプログラムによれば、多チャンネル分の複数の符号化ストリームを平行してデコードすることができるビデオデコーダを実現することが可能となる。

【図面の簡単な説明】
【図１】　MPEG方式におけるピクチャタイプを説明するための図である。
【図２】　MPEG２で規定されたプロファイルおよびレベルにおける、各パラメータの上限値を示す図である。
【図３】　MPEG２ビットストリームの階層構造を説明するための図である。
【図４】マクロブロック層を説明するための図である。
【図５】　sequence_headerのデータ構造を説明するための図である。
【図６】　sequence_extensionのデータ構造を説明するための図である。
【図７】　GOP_headerのデータ構造を説明するための図である。
【図８】　picture_headerのデータ構造を説明するための図である。
【図９】　picture_coding_extensionのデータ構造を説明するための図である。
【図１０】　picture_dataのデータ構造を説明するための図である。
【図１１】　sliceのデータ構造を説明するための図である。
【図１２】　macroblockのデータ構造を説明するための図である。
【図１３】　macroblock_modesのデータ構造を説明するための図である。
【図１４】スタートコードを説明するための図である。
【図１５】　MP@MLの符号化ストリームをデコードする従来のMPEGビデオデコーダの構成例を示すブロック図である。
【図１６】符号化ストリームからＰピクチャを間引いた場合の問題点について説明するための図である。
【図１７】符号化ストリームからＢピクチャだけを間引いた場合の問題点について説明するための図である。
【図１８】本発明を適応したMPEGビデオデコーダ４０の構成例を示すブロック図である。
【図１９】スライスデコーダ制御回路４６の処理を説明するフローチャートである。
【図２０】スライスデコーダ制御回路４６の処理の具体例を説明するための図である。
【図２１】動き補償回路５０によるスライスデコーダの調停処理を説明するフローチャートである。
【図２２】動き補償回路５０によるスライスデコーダの調停処理の具体例を説明するための図である。
【図２３】　MPEGビデオデコーダ４０を用いてMP@MLの符号化ストリームを高速再生するシステムの構成例を示すブロック図である。
【図２４】　MPEGビデオデコーダ４０がMP@MLの符号化ストリームを高速再生する処理を説明するための図である。
【図２５】本発明を適応したMPEGビデオデコーダ１３０の構成例を示すブロック図である。
【図２６】　MPEGビデオデコーダ１３０のコントローラ３４によるＬチャンネル分のパラレルデコード制御処理を説明するフローチャートである。
【図２７】　MPEGビデオデコーダ１３０を用いたMPEGビデオサーバ編集システムの構成例を示すブロック図である。
【図２８】本発明を適応したMPEGビデオデコーダ１５０の構成例を示すブロック図である。
【図２９】　MPEGビデオデコーダ１５０の適用例を説明するための図である。
【符号の説明】
３１　ＩＣ，　３２　バッファ，　３４　コントローラ，　４２　スタートコード検出回路，　４３　ストリームバッファ制御回路，　４５　ピクチャデコーダ，　４６　スライスデコーダ制御回路，　４７乃至４９　スライスデコーダ，５０　動き補償回路，　５１　輝度バッファ制御回路，　５２　色差バッファ制御回路，　６１　ストリームバッファ，　６２　スタートコードバッファ，　７１　輝度バッファ，　７２　色差バッファ，　１０１　磁気ディスク，　１０２　光ディスク，　１０３　光磁気ディスク，　１０４　半導体メモリ，　１１１　ハードディスク，　１１２　再生装置，　１３０　MPEGビデオ

デコーダ，　１５０　MPEGビデオデコーダ，　１５１　チャンネル分離回路

【図１】

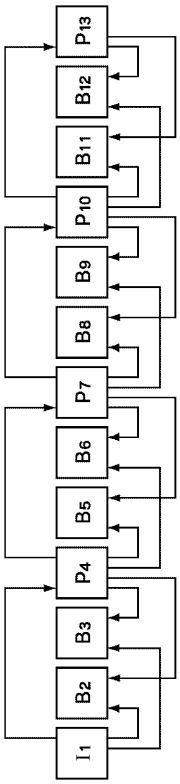


図１

【図２】

Profile and Level	上限				
	Bit rates(Mbit/s)	Samples/line	Lines/Frame	Frames/sec	Samples/sec
4:2:2P@HL	300	1920	1152	60	62,668,800
4:2:2P@ML	50	720	608	30	11,059,200
MP@HL	80	1920	1152	60	62,668,800
MP@HL-1440	60	1440	1152	60	47,001,600
MP@ML	15	720	576	30	10,368,000
MP@LL	4	352	288	30	3,041,280
SP@ML	15	720	576	30	10,368,000

図２

【図３】

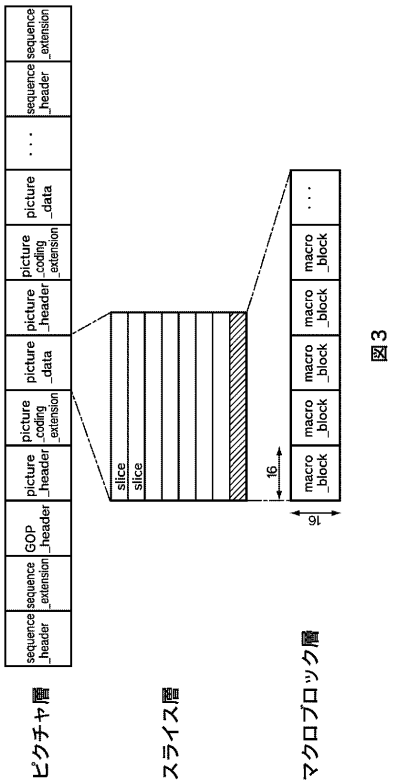
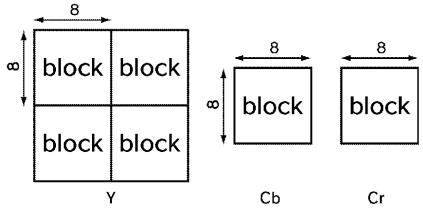


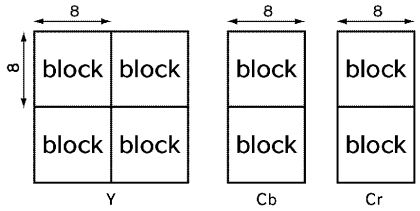
図3

【図４】



4:2:0方式のmacro block層

（A）



4:2:2方式のmacro block層

（B）

図4

【図５】

sequence_header ( ) {	ビット数	ニーモニック
sequence_header_code	32	bslbf
horizontal_size_value	12	uimsbf
vertical_size_value	12	uimsbf
aspect_ratio_information	4	uimsbf
frame_rate_code	4	uimsbf
bit_rate_value	18	uimsbf
marker_bit	1	"1"
vbv_buffer_size_value	10	uimsbf
constrained_parameters_flag	1	
load_intra_quantiser_matrix	1	
if(load_intra_quantiser_matrix)		
intra_quantiser_matrix[64]	8＊64	uimsbf
load_non_intra_quantiser_matrix	1	
if(load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix[64]	8＊64	uimsbf
next_start_code ( )		
}		

sequence_header

図5

【図６】

sequence_extension ( ) {	ビット数	ニーモニック
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
profile_and_level_indication	8	uimsbf
progressive_sequence	1	uimsbf
chroma_format	2	uimsbf
horizontal_size_extension	2	uimsbf
vertical_size_extension	2	uimsbf
bit_rate_extension	12	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_extension	8	uimsbf
low_delay	1	uimsbf
frame_rate_extension_n	2	uimsbf
frame_rate_extension_d	5	uimsbf
next_start_code ( )		
}		

sequence_extension 図6

【図７】

group_of_picture_header ( ) {	ビット数	ニーモニック
group_start_code	32	bslbf
time_code	25	bslbf
closed_gop	1	uimsbf
broken_link	1	uimsbf
next_start_code ( )		
}		

GOP_header 図7

【図８】

picture_header ( ) {	ビット数	ニーモニック
picture_start_code	32	bslbf
temporal_reference	10	uimsbf
picture_coding_type	3	uimsbf
vbv_delay	16	uimsbf
if(picture_coding_type==2 ‖ picture_coding_type==3){		
full_pel_forward_vector	1	
forward_f_code	3	uimsbf
}		
if(picture_coding_type==3)		
full_pel_forward_vector	1	
backward_f_code	3	uimsbf
}		
while(nextbits( )== “l”) {		
extra_bit_picture/∗with the value “l”∗/	1	uimsbf
extra_information_picture	8	
}		
extra_bit_picture/∗with the value “0”∗/	1	uimsbf
next_start_code ( )t		
}		

picture_header

図8

【図９】

picture_coding_extension ( ) {	ビット数	ニーモニック
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
f_code[0][0]/∗forward horizontal∗/	4	uimsbf
f_code[0][1]/∗forward vertical∗/	4	uimsbf
f_code[1][0]/∗backward horizontal∗/	4	uimsbf
f_code[1][1]/∗backward vertical∗/	4	uimsbf
intra_dc_precision	2	uimsbf
picture_structure	2	uimsbf
top_field_first	1	uimsbf
frame_pred_frame_dct	1	uimsbf
concealment_motion_vectors	1	uimsbf
q_scale_type	1	uimsbf
intra_vlc_format	1	uimsbf
alternate_scan	1	uimsbf
repeat_first_field	1	uimsbf
chroma_420_type	1	uimsbf
progressive_frame	1	uimsbf
composite_display_flag	1	uimsbf
if(composite_display_flag) {		
v_axis	1	uimsbf
field_sequence	3	uimsbf
sub_carrier	1	uimsbf
burst_amplitude	7	uimsbf
sub_carrier_phase	8	uimsbf
}		
next_start_code ( )		
}		

picture_coding_extension

図9

【図１０】

picture_data ( ) {	ビット数	ニーモニック
do {		
slice ( )		
} while(nextbits( )==slice_start_code)		
next_start_code( )		
}		

picture_data 図10

【図１１】

slice ( ) {	ビット数	ニーモニック
slice_start_code	32	bslbf
if(vertical_size>2800)		
slice_vertical_position_extention	3	uimsbf
if(<sequence_scalable_extention() is present in the bitstream>)		
if(<scalable_ mode==“data partitioning”)		
priority_breakpoint	7	uimsbf
quantiser_scale_code	5	uimsbf
if(nextbits( )== “l”) {		
intra_slice_flag	1	uimsbf
intra_slice_	1	uimsbf
reserved_bits	7	uimsbf
while(nextbits( )== “l”) {		
extra_bit_slice/∗with the value “l”∗/	1	uimsbf
extra_information_slice	8	uimsbf
}		
}		
extra_bit_slice/∗with the value “0”∗/	1	uimsbf
do {		
macroblock ( )		
} while (nextbits( )!= ‘000 0000 0000 0000 0000 0000’)		
next_start_code ( )t		
}		

slice 図11

【図１２】

macroblock( ) {	ビット数	ニーモニック
while (nextbits( )==‘0000 0001 000’)		
macroblock_escape	11	bslbf
macroblock_address_increment	1-11	vlclbf
macroblock_modes( )		
if(macroblock_quant)		
quantiser_scale_code	5	uimsbf
if(macroblock_motion_forward ‖		
(macroblock_intra && concealment_motion_vectors))		
motion_vectors(0)		
if(macroblock_motion_backward)		
motion_vectors(1)		
if(macroblock_intra && concealment_motion_vectors)		
marker_bit	1	bslbf
if(macroblock_pattern)		
coded block pattern( )		
for(1=0; i<block_count; i++) {		
block(i)		
}		
}		

macroblock

図12

【図１３】

macroblock_modes( ) {	ビット数	ニーモニック
macroblock_type	1-9	vlclbf
if((spatial_temporal_weight_code_flag==1)&&		
(spatial_temporal_weight_doce_table_index !="00")) {		
spatial_temporal_weight_code	2	uimsbf
}		
if(macroblock_motion_forward ||		
macroblock_motion_backward) {		
if(picture_structure=='frame') {		
if(frame_pred_frame_dct==0)		
frame_motion_type	2	uimsbf
} else {		
field_motion_type	2	uimsbf
}		
}		
if((picture_structure=="frame picture") &&		
(frame_pred_frame_det==0) &&		
(macroblock_intrallmacroblock_pattern)) {		
dct_type	1	uimsbf
}		
}		

macroblock_modes

図13

【図１４】

名　称	Start code value
Picture_start_code	00
Slice_start_code	01 ～ AF
Reserved	B0
Reserved	B1
User_data_start_code	B2
Sequence_header_code	B3
Sequence_error_code	B4
Extension_start_code	B5
Reserved	B6
Sequence_end_code	B7
Group_start_code	B8
System_start_code	B9 ～ FF

図14

【図１５】

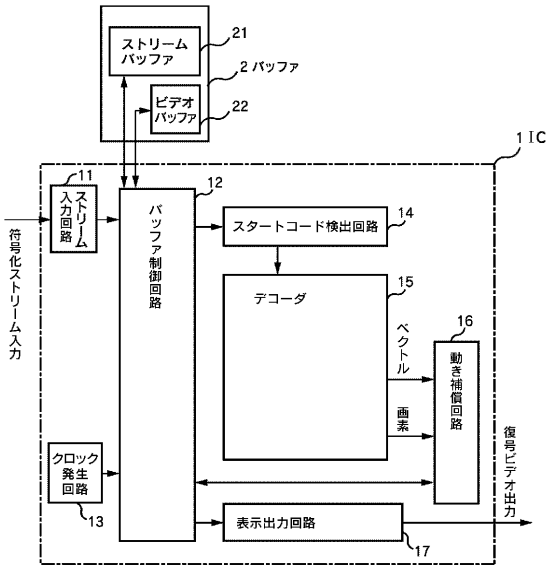


図15

【図１６】

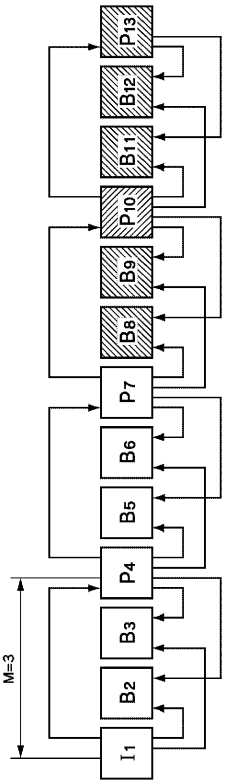


図16

【図１７】

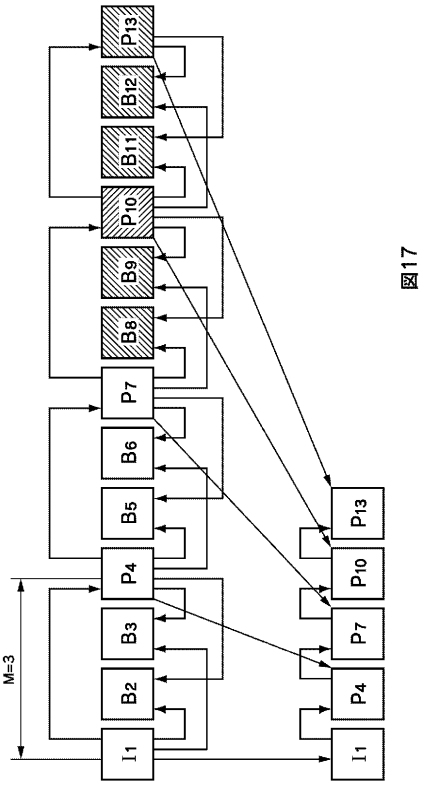


図17

【図１８】

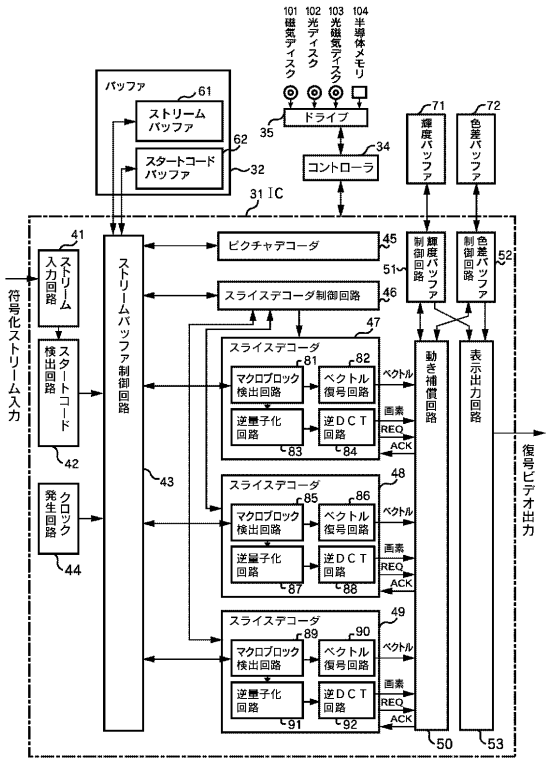


図18

【図１９】

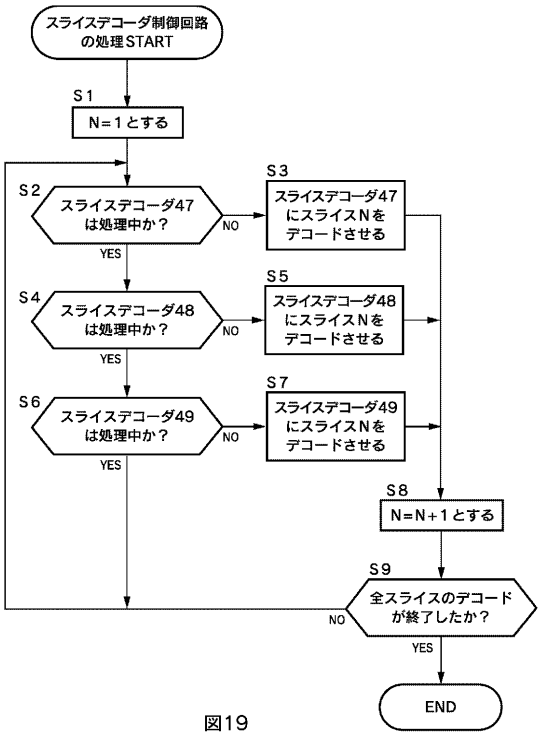


図19

【図２０】

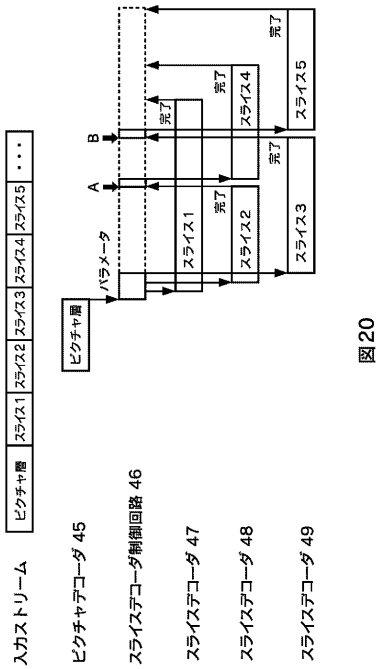


図20

【図２１】

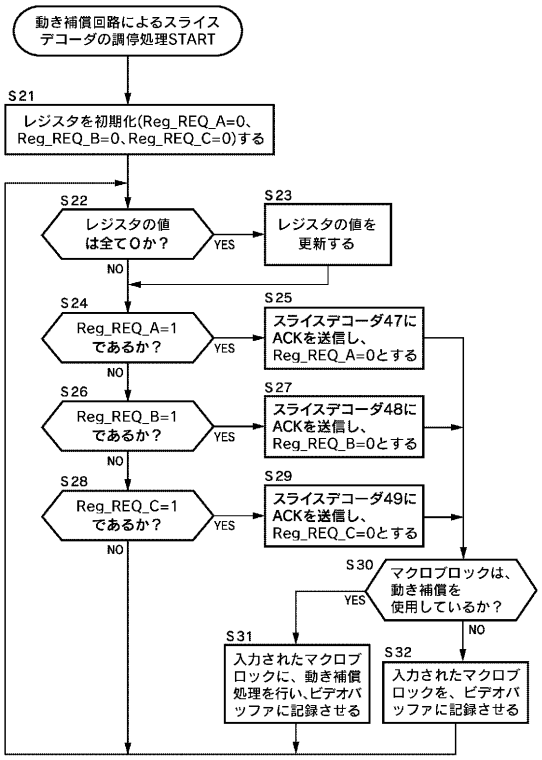


図21

【図２２】

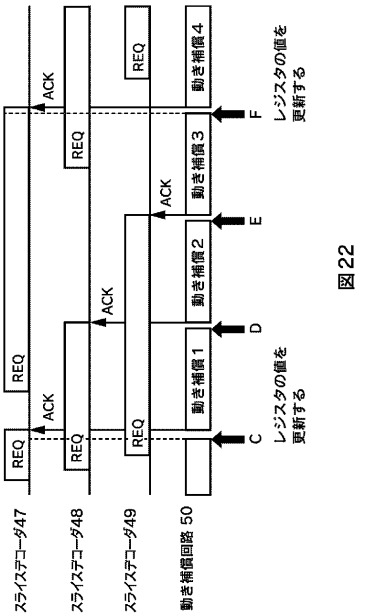


図22

【図２３】

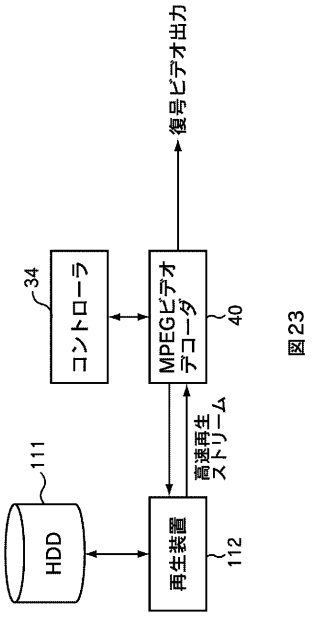


図23

【図２４】

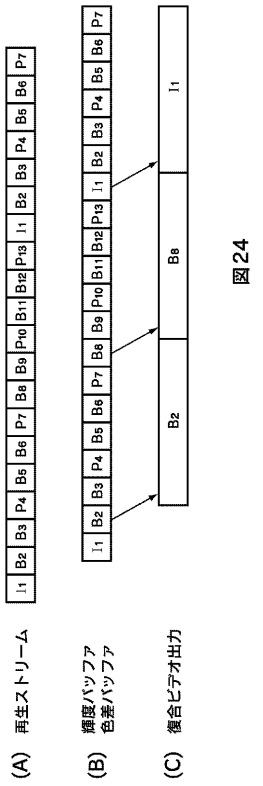


図24

【図２５】

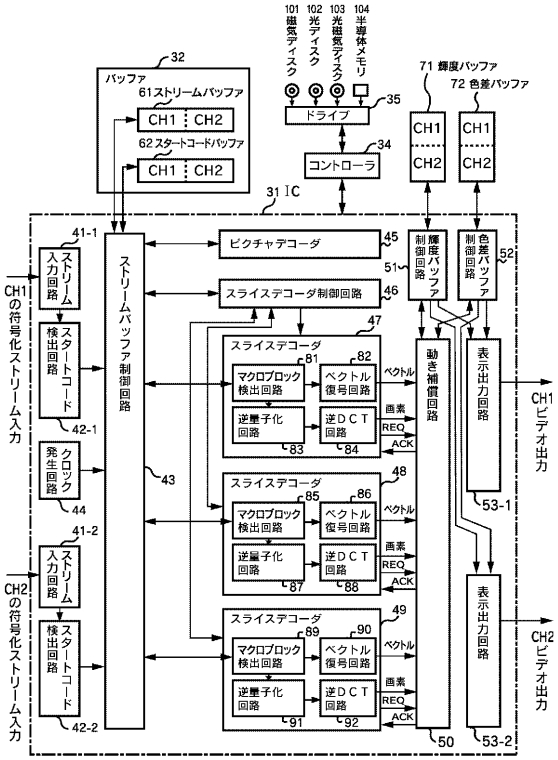


図25

【図２６】

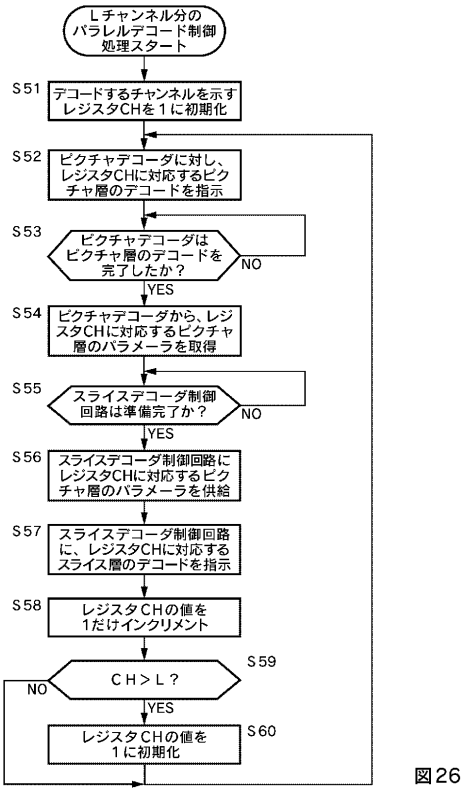


図26

【図２７】

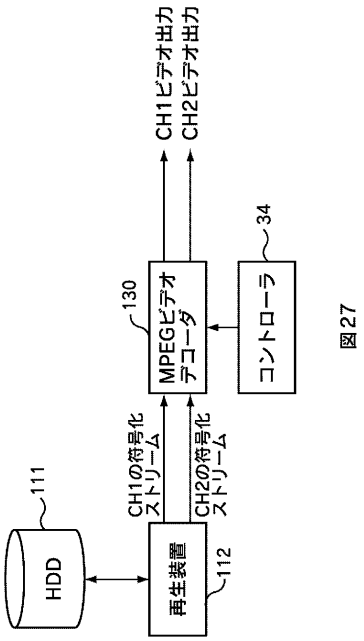


図27

【図２８】

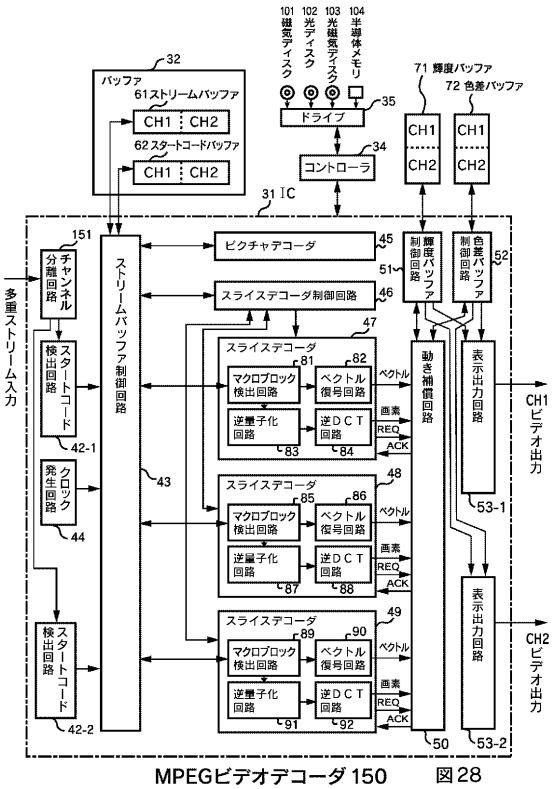


図28

【図２９】

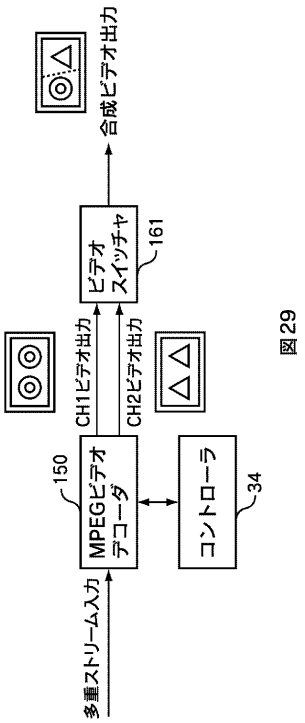

合成ビデオ出力
ビデオスイッチャ
161
CH1ビデオ出力
CH2ビデオ出力
150
MPEGビデオデコーダ
コントローラ
34
多重ストリーム入力

図29

フロントページの続き

(56)参考文献 特開平１１－３４１４８９（ＪＰ，Ａ）
特開２０００－５０２５５（ＪＰ，Ａ）
特開平１１－２５０５８２（ＪＰ，Ａ）