(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3709721号
(P3709721)
(45)発行日　平成17年10月26日(2005.10.26)　(24)登録日　平成17年8月19日(2005.8.19)

(51)Int.Cl.[7]		F I		
H04N	5/268	H04N	5/268	
H04H	1/00	H04H	1/00	A
H04J	3/00	H04J	3/00	M
H04N	7/24	H04N	7/13	Z

請求項の数 24　(全 47 頁)

(21)出願番号　特願平10-282609
(22)出願日　平成10年10月5日(1998.10.5)
(65)公開番号　特開平11-220655
(43)公開日　平成11年8月10日(1999.8.10)
審査請求日　平成14年7月8日(2002.7.8)
(31)優先権主張番号　特願平9-271275
(32)優先日　平成9年10月3日(1997.10.3)
(33)優先権主張国　日本国(JP)

(73)特許権者　000002185
ソニー株式会社
東京都品川区北品川６丁目７番３５号
(74)代理人　100082131
弁理士　稲本　義雄
(72)発明者　田原　勝己
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　安田　幹太
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　根岸　愼治
東京都品川区北品川６丁目７番３５号　ソニー株式会社内

最終頁に続く

(54)【発明の名称】符号化ストリームスプライシング装置及び符号化ストリームスプライシング方法、符号化ストリーム生成装置及び符号化ストリーム生成方法、並びに情報処理装置及び方法

(57)【特許請求の範囲】

【請求項１】

第１の符号化ストリームと第２の符号化ストリームとをスプライシングポイントにおいてスプライシングする符号化ストリームスプライシング装置において、

上記第１の符号化ストリームのシンタックスを解析することによって、上記第１の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、

上記ストリーム解析手段によって得られた上記第１の符号化ストリームの符号化パラメータに基いて、上記スプライシングポイントにおいて上記第１の符号化ストリームと上記第２の符号化ストリームとがシームレスに接続されるように、上記第２の符号化ストリームの符号化パラメータを変更し、上記第１の符号化ストリームと上記符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシング手段と

を備えたことを特徴とする符号化ストリームスプライシング装置。

【請求項２】

上記ストリーム解析手段によって抽出される符号化パラメータは、

ＶＢＶバッファの初期状態を示すデータvbv_delayであることを特徴とする請求項１記載の符号化ストリームスプライシング装置。

【請求項３】

上記スプライシング手段は、

上記第２の符号化ストリームにおけるスプライスポイント後の最初のピクチャのvbv_delayの値を、上記第１の符号化ストリームにおけるスプライスポイント後の最初のピクチ

ャのvbv_delayの値に書き換えることを特徴とする請求項２記載の符号化ストリームスプライシング装置。

【請求項４】

上記スプライシング手段は、

上記第２の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量が、上記書き換えられたvbv_delayの値に対応するビット量となるように、上記第２の符号化ストリームにスタッフィングバイトを挿入することを特徴とする請求項３記載の符号化ストリームスプライシング装置。

【請求項５】

上記スプライシング手段は、

上記スタッフィングバイトのデータ量を、上記第１の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量、及び上記第２の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量に基いて演算することを特徴とする請求項４記載の符号化ストリームスプライシング装置。

【請求項６】

上記スプライシング手段は、

上記スタッフィングバイトのデータ量を、上記第１の符号化ストリームのスプライスポイントにおけるＶＢＶバッファのデータ占有量と、上記第２の符号化ストリームのスプライスポイントにおけるＶＢＶバッファのデータ占有量とに基いて演算することを特徴とする請求項５記載の符号化ストリームスプライシング装置。

【請求項７】

上記ストリーム解析手段によって抽出される符号化パラメータは、

復号時にリピートフィールドを生成するか否かを示すフラグrepeat_first_fieldであることを特徴とする請求項１記載の符号化ストリームスプライシング装置。

【請求項８】

上記スプライシング手段は、

上記第１の符号化ストリームにおけるスプライスポイント前のピクチャのフレーム構造と、上記第２の符号化ストリームにおけるスプライスポイント後のピクチャのフレーム構造との整合性が取れるように、上記第１の符号化ストリームにおけるスプライスポイント前のピクチャのrepeat_first_fieldの値、又は、上記第２の符号化ストリームにおけるスプライスポイント後のピクチャのrepeat_first_fieldの値を変更することを特徴とする請求項７記載の符号化ストリームスプライシング装置。

【請求項９】

上記ストリーム解析手段によって抽出される符号化パラメータは、

最初のフィールドがトップフィールドであるかボトムフィールドであるかを示すフラグtop_field_firstであることを特徴とする請求項１記載の符号化ストリームスプライシング装置。

【請求項１０】

上記スプライシング手段は、

上記第１の符号化ストリームにおけるスプライスポイント前のピクチャのフレーム構造と、上記第２の符号化ストリームにおけるスプライスポイント後のピクチャのフレーム構造との整合性が取れるように、上記第１の符号化ストリームにおけるスプライスポイント前のピクチャのtop_field_firstの値、又は、上記第２の符号化ストリームにおけるスプライスポイント後のピクチャのtop_field_firstの値を変更することを特徴とする請求項９記載の符号化ストリームスプライシング装置。

【請求項１１】

第１の符号化ストリームと第２の符号化ストリームとをスプライシングポイントにおいてスプライシングする符号化ストリームスプライシング方法において、

上記第１の符号化ストリームのシンタックスを解析することによって、上記第１の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、

上記ストリーム解析ステップによって得られた上記第１の符号化ストリームの符号化パラメータに基いて、上記スプライシングポイントにおいて上記第１の符号化ストリームと上記第２の符号化ストリームとがシームレスに接続されるように、上記第２の符号化ストリームの符号化パラメータを変更し、上記第１の符号化ストリームと上記符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシングステップと
を備えたことを特徴とする符号化ストリームスプライシング方法。

【請求項１２】
上記ストリーム解析ステップによって抽出される符号化パラメータは、
ＶＢＶバッファの初期状態を示すデータvbv_delayであることを特徴とする請求項１１記載の符号化ストリームスプライシング方法。

【請求項１３】
上記スプライシングステップは、
上記第２の符号化ストリームにおけるスプライスポイント後の最初のピクチャのvbv_delayの値を、上記第１の符号化ストリームにおけるスプライスポイント後の最初のピクチャのvbv_delayの値に書き換えることを特徴とする請求項１２記載の符号化ストリームスプライシング方法。

【請求項１４】
上記スプライシングステップは、
上記第２の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量が、上記書き換えられたvbv_delayの値に対応するビット量となるように、上記第２の符号化ストリームにスタッフィングバイトを挿入することを特徴とする請求項１３記載の符号化ストリームスプライシング方法。

【請求項１５】
上記スプライシングステップは、
上記スタッフィングバイトのデータ量を、上記第１の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量、及び上記第２の符号化ストリームにおけるスプライスポイント後の最初のピクチャの発生ビット量に基いて演算することを特徴とする請求項１４記載の符号化ストリームスプライシング方法。

【請求項１６】
上記スプライシングステップは、
上記スタッフィングバイトのデータ量を、上記第１の符号化ストリームのスプライスポイントにおけるＶＢＶバッファのデータ占有量と、上記第２の符号化ストリームのスプライスポイントにおけるＶＢＶバッファのデータ占有量とに基いて演算することを特徴とする請求項１５記載の符号化ストリームスプライシング方法。

【請求項１７】
上記ストリーム解析ステップによって抽出される符号化パラメータは、
復号時にリピートフィールドを生成するか否かを示すフラグrepeat_first_fieldであることを特徴とする請求項１１記載の符号化ストリームスプライシング方法。

【請求項１８】
上記スプライシングステップは、
上記第１の符号化ストリームにおけるスプライスポイント前のピクチャのフレーム構造と、上記第２の符号化ストリームにおけるスプライスポイント後のピクチャのフレーム構造との整合性が取れるように、上記第１の符号化ストリームにおけるスプライスポイント前のピクチャのrepeat_first_fieldの値、又は、上記第２の符号化ストリームにおけるスプライスポイント後のピクチャのrepeat_first_fieldの値を変更することを特徴とする請求項１７記載の符号化ストリームスプライシング方法。

【請求項１９】
上記ストリーム解析ステップによって抽出される符号化パラメータは、
最初のフィールドがトップフィールドであるかボトムフィールドであるかを示すフラグtop_field_firstであることを特徴とする請求項１１記載の符号化ストリームスプライシ

ング方法。
【請求項２０】
上記スプライシングステップは、
上記第１の符号化ストリームにおけるスプライスポイント前のピクチャのフレーム構造と、上記第２の符号化ストリームにおけるスプライスポイント後のピクチャのフレーム構造との整合性が取れるように、上記第１の符号化ストリームにおけるスプライスポイント前のピクチャのtop_field_firstの値、又は、上記第２の符号化ストリームにおけるスプライスポイント後のピクチャのtop_field_firstの値を変更することを特徴とする請求項１９記載の符号化ストリームスプライシング方法。
【請求項２１】
第１の符号化ストリームと第２の符号化ストリームとをスプライシングポイントにおいてスプライシングすることによってスプライスド符号化ストリームを生成する符号化ストリーム生成装置において、
上記第１の符号化ストリームのシンタックスを解析することによって、上記第１の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、
上記ストリーム解析手段によって得られた上記第１の符号化ストリームの符号化パラメータに基いて、上記スプライシングポイントにおいて上記第１の符号化ストリームと上記第２の符号化ストリームとがシームレスに接続されるように、上記第２の符号化ストリームの符号化パラメータを変更し、上記第１の符号化ストリームと上記符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシング手段と
を備えたことを特徴とする符号化ストリーム生成装置。
【請求項２２】
第１の符号化ストリームと第２の符号化ストリームとをスプライシングポイントにおいてスプライシングすることによってスプライスド符号化ストリームを生成する符号化ストリーム生成方法において、
上記第１の符号化ストリームのシンタックスを解析することによって、上記第１の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、
上記ストリーム解析ステップによって得られた上記第１の符号化ストリームの符号化パラメータに基いて、上記スプライシングポイントにおいて上記第１の符号化ストリームと上記第２の符号化ストリームとがシームレスに接続されるように、上記第２の符号化ストリームの符号化パラメータを変更し、上記第１の符号化ストリームと上記符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシングステップと
を備えたことを特徴とする符号化ストリーム生成方法。
【請求項２３】
第１の符号化ストリームと第２の符号化ストリームとをスプライシングする符号化ストリームスプライシング装置において、
上記第１の符号化ストリームの符号化パラメータに基いて、上記第２の符号化ストリームの符号化パラメータを変更する符号化パラメータ変更手段と、
上記第１の符号化ストリームと、上記符号化パラメータ変更手段により符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシング手段と
を備えたことを特徴とする符号化ストリームスプライシング装置。
【請求項２４】
第１の符号化ストリームと第２の符号化ストリームとをスプライシングする符号化ストリームスプライシング方法において、
上記第１の符号化ストリームの符号化パラメータに基いて、上記第２の符号化ストリームの符号化パラメータを変更する符号化パラメータ変更ステップと、
上記第１の符号化ストリームと、上記符号化パラメータ変更ステップの処理により符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシングステップと

を備えたことを特徴とする符号化ストリームスプライシング方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

この発明は、デジタル放送システムにおいて使用される符号化ストリームスプライシング装置及び符号化ストリームスプライシング方法、符号化ストリーム生成装置及び符号化ストリーム生成方法、並びに情報処理装置及び方法であって、特には、２つの符号化ストリームをストリームレベルでスプライシングすることによってシームレスなスプライスドストリームを生成する符号化ストリームスプライシング装置及び符号化ストリームスプライシング方法、符号化ストリーム生成装置及び符号化ストリーム生成方法、並びに情報処理装置及び方法に関するものである。

【０００２】

【従来の技術】

図１は、現在のテレビジョン放送システムを説明するための図である。現在のテレビジョン放送システムにおいて、各家庭にテレビ番組を配信するための放送局は、全国規模のテレビ番組を制作する本局（キーステーション又はメインステーション）ＳＫと、地方特有のテレビ番組を作成する本局系列の複数の地方局（支局）ＳＡ、ＳＢ、及びＳＣとから構成されている。本局ＳＫは、全国共通のテレビ番組を作成し、その作成したテレビ番組を地方局に伝送するための放送局であって、地方局は、本局から局間伝送によって送られて来たオリジナルテレビ番組、及びそのオリジナルテレビ番組一部を地方特有向けに編集したテレビ番組の両方の番組を地方の家庭に配信するための放送局である。例えば、図１に示されるように、地方局ＥＡは、放送エリアＥＡ内の家庭に伝送するテレビ番組を作成する局であって、地方局ＥＢは、放送エリアＥＢ内の家庭に伝送するテレビ番組を作成する局であって、地方局ＥＣは、放送エリアＥＣ内の家庭に伝送するテレビ番組を作成する局である。尚、この各地方局において行われる編集処理とは、例えば、本局から送られてきたニュース番組に、地方独自の天気予報のプログラムを挿入したり、映画やドラマ等の番組に、地方向けのコマーシャルを挿入したりする処理のことである。

【０００３】

図２は、各地方局における編集処理を説明するための図であって、図２（Ａ）は、本局において制作されたオリジナルテレビ番組ＰＧOLDを示し、図２（Ｂ）は、地方局において制作された地方向けの差し替えテレビ番組ＰＧNEWであって、図２（Ｃ）は、地方局において編集されたテレビ番組ＰＧEDITを示している。

図２に示された編集処理の例は、本局から伝送されてきたオリジナルテレビ番組のうち、コマーシャルＣＭ１、プログラム２、及びコマーシャルＣＭ３を、地方局において地方向けに制作されたコマーシャルＣＭ１’、プログラム２’、及びコマーシャルＣＭ３’に置き換える編集処理の例である。この地方局における編集処理の結果、図２（Ｃ）に示されるように、本局において生成されたテレビ番組（プログラム１、ＣＭ２、プログラム３Ｃ、及びプログラム４）と地方局において生成されたテレビ番組（コマーシャルＣＭ１’、プログラム２’、及びコマーシャルＭ３’）とが混在する地方向けのテレビ番組が生成される。

【０００４】

近年、現在のテレビジョン放送システムはアナログのベースバンドのテレビジョン信号を各家庭に配信するアナログ放送であるため、これらのアナログ方法システムを、デジタル技術を使用した次世代の放送システムに置換えようという試みがなされている。このデジタル放送システムは、ＭＰＥＧ２（Moving Picture Experts Group Phase2）等の圧縮符号化技術を用いてビデオデータやオーディオデータを圧縮符号化し、その符号化されたストリームを地上波や衛星波を利用して各家庭や他局に伝送するシステムである。特に、このデジタル放送システムとして提案されている放送技術の中で、ヨーロッパにおいて次世代の放送方式として提案されているＤＶＢ（Digital Video Broadcasting）規格が最も有力であって、このＤＶＢ規格がデファクトスタンダートとなりつつある。

【０００５】
次に、図３を参照してＭＰＥＧ規格を用いて、ビデオデータとオーディオデータを含んだプログラムを送信側システムから受信側システムに伝送する一般的なデジタル伝送システムに関して説明する。
【０００６】
一般的なデジタル伝送システムにおいて、伝送側システム１０は、ＭＰＥＧビデオエンコーダ１１と、ＭＰＥＧオーディオエンコーダ１２と、マルチプレクサ１３とを備え、受信側システム２０は、デマルチプレクサ２１と、ＭＰＥＧビデオエンコーダ２２と、ＭＰＥＧオーディオデコーダとを備えている。
【０００７】
ＭＰＥＧビデオエンコーダ１１は、ベースバンドのソースビデオデータＶをＭＰＥＧ規格に基いて符号化し、その符号化したストリームをビデオエレメンタリストリームＥＳとして出力する。ＭＰＥＧオーディオエンコーダ１２は、ベースバンドのソースオーディオデータＡをＭＰＥＧ規格に基いて符号化し、その符号化したストリームをオーディオエレメンタリストリームＥＳとして出力する。マルチプレクサ１３は、ＭＰＥＧビデオエンコーダ１１とＭＰＥＧオーディオエンコーダ１２から、それぞれビデオエレメンタリーストリームとオーディオエレメンタリストリームを受取り、それらのストリームをトランスポートストリームパケットの形態に変換し、ビデオエレメンタリストリームを含んだトランスポートストリームパケットとオーディオエレメンタリストリームを含んだトランスポートストリームパケットを生成する。さらに、マルチプレクサ１３は、ビデオエレメンタリストリームを含んだトランスポートストリームパケットとオーディオエレメンタリストリームを含んだトランスポートストリームパケットとが混在するように、それぞれのトランスポートストリームパケットを多重化し、受信システム２０に伝送されるトランスポートストリームを生成する。
【０００８】
デマルチプレクサ２１は、伝送路を介して伝送されたトランスポートストリームを受取り、ビデオエレメンタリストリームを含んだトランスポートストリームとオーディオエレメンタリストリームを含んだトランスポートストリームパケットとに分離する。さらに、デマルチプレクサ２１は、ビデオエレメンタリストリームを含んだトランスポートストリームパケットからビデオエレメンタリーストリームを生成するとともに、オーディオエレメンタリストリームを含んだトランスポートストリームパケットからオーディオエレメンタリーストリームを生成する。ＭＰＥＧビデオデコーダ２２は、デマルチプレクサ２１からビデオエレメンタリーストリームを受取り、このビデオエレメンタリストリームをＭＰＥＧ規格に基いてデコードし、ベースバンドのビデオデータＶを生成する。ＭＰＥＧオーディオデコーダ２２は、デマルチプレクサ２１からオーディオエレメンタリーストリームを受取り、このオーディオエレメンタリストリームをＭＰＥＧ規格に基いてデコードし、ベースバンドのオーディオデータＡを生成する。
【０００９】
さて、従来のアナログ放送システムをこのようなデジタル伝送システムの技術を使用してデジタル放送システムに置き換えようとした場合、本局から地方局に向けて伝送されるテレビ番組のビデオデータは、ＭＰＥＧ２規格に基いて圧縮符号化された符号化ストリームとなる。従って、地方局において本局から伝送されたオリジナル符号化ストリームの一部を、地方局において制作された符号化ストリームに置換えるための編集処理を行なうためには、この編集処理の前に、一端、符号化ストリームをデコードして、ベースバンドのビデオデータに戻さなければいけない。なぜなら、ＭＰＥＧ規格に準じた符号化ストリームに含まれる各ピクチャの予測方向は、前後のピクチャの予測方向と相互に関連しているので、ストリーム上の任意の位置において符号化ストリームを接続することができないからである。もし強引に２つの符号化ストリームをつなげたとすると、符号化ストリームのつなぎめが不連続になってしない、正確にデコードできなくなってしまうことが発生する。
【００１０】

従って、図２において説明したような編集処理を実現するためには、本局から供給されたオリジナルの符号化ストリームと、地方向けに制作された符号化ストリームの両方を一端デコードし、それぞれをベースバンドのビデオ信号に戻すデコード処理と、２つのベースバンドのビデオデータを編集してオンエア用の編集されたビデオデータを生成する編集処理と、編集されたビデオデータを再び符号化して、符号化ビデオストリームを生成するという符号化処理とを行なわなくてはいけない。しかしながら、ＭＰＥＧ規格に基く符号化／復号化処理は１００％可逆の符号化／復号化処理ではないので、復号化処理及び符号化処理を繰り返すたびに画質が劣化してしまうという問題があった。

【００１１】

そこで、近年では、供給された符号化ストリームを復号化処理せずに、符号化ストリームの状態のまま編集することを可能にする技術が要求されるようになってきた。尚、このように符号化されたビットストリームレベルで、異なる２つの符号化ビットストリームを連結し、連結されたビットストリームを生成することを「スプライシング」と呼んでいいる。つまり、スプライシングとは、符号化ストリームの状態のままで複数のストリームを編集及び接続することを意味する。

【００１２】

しかしながら、このスプライシング処理を実現するためには以下のような２つの問題点がある。

【００１３】

まず、第１の問題点について説明する。

【００１４】

上述したＭＰＥＧビデオエンコーダ１１及びＭＰＥＧビデオデコーダ２２において使用されているＭＰＥＧ規格では、符号化方式として双方向予測符号化方式が採用されている。この双方向予測符号化方式では、フレーム内符号化、フレーム間順方向予測符号化および双方向予測符号化の３つのタイプの符号化が行われ、各符号化タイプによる画像は、それぞれＩピクチャ（intra coded picture ）、Ｐピクチャ（predictive coded picture）およびＢピクチャ（bidirectionally predictive coded picture）と呼ばれる。また、Ｉ，Ｐ，Ｂの各ピクチャを適切に組み合わせて、ランダムアクセスの単位となるＧＯＰ（Group of Picture）が構成される。一般的には、各ピクチャの発生符号量は、Ｉピクチャが最も多く、次にＰピクチャが多く、Ｂピクチャが最も少ない。

【００１５】

ＭＰＥＧ規格のようにピクチャ毎にビット発生量が異なる符号化方法では、得られる符号化ビットストリーム（以下、単にストリームとも言う。）をビデオデコーダにおいて正確に復号化して画像を得るためには、ビデオデコーダ２２における入力バッファ内のデータ占有量を、ビデオエンコーダ１１で把握していなければならない。そこで、ＭＰＥＧ規格では、ビデオデコーダ２２における入力バッファに対応するバッファとして‘ＶＢＶ（Video Buffering Verifier）バッファ’という仮想バッファを想定し、ビデオエンコーダ１１では、ＶＢＶバッファを破綻、つまりアンダフローやオーバフローさせないように、符号化処理を行なうように定義されている。例えば、このＶＢＶバッファの容量は、伝送される信号の規格に応じて決められており、メインプロファイル・メインレベル（ＭＰ@ＭＬ）のスタンダードビデオ信号の場合であれば、１．７５Ｍビットの容量を有している。ビデオエンコーダ１１は、このＶＢＶバッファをオーバフローやアンダフローさせないように、各ピクチャのビット発生量をコントロールする。

【００１６】

次に、図４を参照して、ＶＢＶバッファについて説明する。

【００１７】

図４（Ａ）は、本局において制作されたプログラム１とコマーシャルＣＭ１とを含んだオリジナルテレビ番組を、ビデオエンコーダで符号化したオリジナルストリームＳＴOLDと、そのオリジナルストリームＳＴOLDに対応するＶＢＶバッファのデータ占有量の軌跡を示す図である。図４（Ｂ）は、地方向けに制作されたコマーシャルであって、オリジナル

テレビ番組のコマーシャルＣＭ１の部分に差し替えられるコマーシャルＣＭ１’を、地方局のビデオエンコーダで符号化した差し替えストリームＳＴNEWと、その差し替えストリームＳＴNEWのＶＢＶバッファのデータ占有量の軌跡を示す図である。尚、以下の説明において、本局から支局に伝送されたオリジナルテレビ番組を符号化したストリームの一部が支局において作成された新たなストリームによって置き換えられるので、このオリジナルテビ番組を符号化したオリジナルストリームを、古いストリームであることを示す‘ＳＴOLD’と表現し、オリジナルストリームＳＴOLDの一部に新たに差し替えられる差し替えストリームを‘ＳＴNEW’と表現する。図４（Ｃ）は、スプライスポイントＳＰにおいてオリジナルストリームＳＴOLDに対して差し替えストリームＳＴNEWをスプライシングすることによって得られたスプライスドストリームＳＴSPL（ spliced stream ）と、そのスプライスドストリームＳＴSPLのＶＢＶバッファのデータ占有量の軌跡を示す図である。

【００１８】

尚、図４において、ＶＢＶバッファのデータ占有量の軌跡において、右上がり部分（傾き部分）は伝送ビットレートを表し、垂直に落ちている部分は各ピクチャの再生のためにビデオデコーダがデコーダバッファから引き出すビット量を表している。ビデオデコーダがこのデコーダバッファからビットを引き出すタイミングは、デコーデイングタイムスタンプ（ＤＴＳ）と呼ばれる情報によって指定される。なお、図４においてＩ，Ｐ，Ｂは、それぞれＩピクチャ，Ｐピクチャ，Ｂピクチャを表している。

【００１９】

オリジナル符号化ストリームＳＴOLDは本局のビデオエンコーダ符号化されたストリームであって、差し替えストリームＳＴNEWは地方局のビデオエンコーダで符号化されたストリームであるので、オリジナル符号化ストリームＳＴOLDと差し替えストリームＳＴNEWとは、それぞれのビデオエンコーダで全く関係無く符号化されたストリームである。よって、地方局のビデオエンコーダは、本局のビデオエンコーダのＶＢＶバッファのデータ占有量の軌跡を全く知らずに、独自に差し替えストリームＳＴNEWを生成するための符号化処理を行っているので、スプライスポイントＳＰにおけるオリジナルストリームＳＴOLDのＶＢＶバッファのデータ占有量ＶＢＶOLDと、スプライスポイントＳＰにおける差し替えストリームＳＴNEWのＶＢＶバッファのデータ占有量ＶＢＶNEWとは異なってしまう。

【００２０】

つまり、スプライスドストリームＳＴSPLのスプライスポイントＳＰの前後において、ＶＢＶバッファのデータ占有量の軌跡が不連続にならないようにするためには、スプライスドストリームＳＴSPLにおける差し替えストリームＳＴNEWのＶＢＶバッファのデータ占有量の初期レベルは、ＶＢＶバッファのデータ占有量ＶＢＶOLDにしなければいけない。その結果、図４に示したように、オリジナルストリームＳＴOLDのＶＢＶバッファのデータ占有量ＶＢＶOLDよりも、差し替えストリームＳＴNEWのＶＢＶバッファのデータ占有量ＶＢＶNEWの値が小さい場合には、スプライスドストリームＳＴSPLにおける差し替えストリームＳＴNEWの部分で、ＶＢＶバッファがオーバーフローしてしまう。また、逆に、オリジナルストリームＳＴOLDのＶＢＶバッファのデータ占有量ＶＢＶOLDよりも、差し替えストリームＳＴNEWのＶＢＶバッファのデータ占有量ＶＢＶNEWの値が大きい場合には、スプライスストリームＳＴSPLにおける差し替えストリームＳＴNEWの部分で、ＶＢＶバッファがアンダーフローしてしまう。

【００２１】

次に、第２の問題について説明する。

【００２２】

ＭＰＥＧ規格に基いて符号化されたストリームのヘッダには、符号化情報を示すさまざまなデータエレメントやフラグが記述されており、これらのデータエレメントやフラグを使用して符号化ストリームを復号化するようになされている。

【００２３】

図２に示されたオリジナルテレビ番組の本編を構成するプログラム１、プログラム２、プ

ログラム３及びプログラム４は、ビデオカメラ等によって撮影された２９．９７Ｈｚ（約３０Ｈｚ）のフレームレートを有するＮＴＳＣ方式のテレビジョン信号とは限らず、２４Ｈｚ（毎秒２４コマ）のフレームレートを有する映画素材からテレビジョン信号に変換された信号であることもある。一般的に、このように２４Ｈｚの映画素材を２９．９７Ｈｚのテレビジョン信号に変換する処理を、オリジナル素材における２個のフィールドを、所定のシーケンスで３個のフィールドに変換する処理を含むことから、‘２：３プルダウン処理’と呼んでいる。

【００２４】

図５は、この２：３プルダウン処理を説明するための図である。図５において、Ｔ１からＴ８は２４Ｈｚのフレーム周波数を有する映画素材のトップフィールドを示し、Ｂ１からＢ９は２４Ｈｚのフレーム周波数を有する映画素材のボトムフィールドを示している。さらに、図５において示された楕円及び三角形は、トップフィールドとボトムフィールドから構成されるフレームの構造を示している。

【００２５】

具体的には、この２：３プルダウン処理において、２４Ｈｚのフレーム周波数を有する映画素材（８個のトップフィールドＴ１～Ｔ８、及び８個のボトムフィールドＢ１～Ｂ８）に、ボトムフィールドＢ２を繰り返すことによって生成されたリピートフィールドＢ２’、トップフィールドＴ４を繰り返すことによって生成されたリピートフィールドＴ４’、ボトムフィールドＢ６を繰り返すことによって生成されたリピートフィールドＢ６’、及びトップフィールドＴ８を繰り返すことによって生成されたリピートフィールドＴ８’の４つのリピートフィールドを挿入する処理が行われる。その結果、この２：３プルダウン処理によって、２４Ｈｚのフレーム周波数を有する映画素材から２９．９７Ｈｚのフレーム周波数を有するテレビジョン信号が生成される。

【００２６】

ＭＰＥＧエンコーダにおいて、２：３プルダウン処理されたテレビジョン信号は、ビデオエンコーダにおいてそのまま符号化処理されるのではなく、２：３プルダウン処理されたテレビジョン信号からリピートフィールドを除去してから、符号化処が行われる。図５に示した例では、２：３プルダウンされたテレビジョン信号から、リピートフィールドＢ２’、Ｔ４’、Ｂ６’及びＴ８’が除去される。このように符号化処理前にリピートフィールドを除去する理由は、このリピートフィールドは、２：３プルダウン処理時に挿入された冗長なフィールドであって、圧縮符号化効率を向上させるために削除したとしても何ら画質劣化が発生しないからである。

【００２７】

また、ＭＰＥＧ規格においては、符号化ストリームをデコードする際に、フレームを構成する２つのフィールドのいずれかのフィールドを繰り返すことによって、リピートフィールドを生成するか否かを示す‘ repeat_first_field ’というフラグを記述することを定義付けている。具体的には、ＭＰＥＧデコーダは、符号化ストリームをデコードする際に、符号化ストリーム中のフラグ「repeat_first_field 」が「１」の場合にはリピートフィールドを生成し、符号化ストリーム中のフラグ「repeat_first_field 」が「０」の場合にはリピートフィールドを生成しないという処理を行なう。

【００２８】

図５に示した例の場合には、トップフィールドＴ１とボトムフィールドＢ１とから構成されるフレームを符号化したストリームの「 repeat_first_field 」は「０」であって、トップフィールドＴ２とボトムフィールドＢ２とから構成されるフレームを符号化したストリームの「 repeat_first_field 」は「１」であって、トップフィールドＴ３とボトムフィールドＢ３とから構成されるフレームを符号化したストリームの「 repeat_first_field 」は「０」であって、トップフィールドＴ４とボトムフィールドＢ４とから構成されるフレームを符号化したストリームの「 repeat_first_field 」は「１」であるので、トップフィールドＴ２とボトムフィールドＢ２とから構成されるフレームの符号化ストリームをデコードする時には、リピートフィールドＢ２’を生成し、トップフィールドＴ４とボ

トムフィールドＢ４とから構成されるフレームの符号化ストリームをデコードする時には、リピートフィールドＢ４’を生成する処理を行なう。

【００２９】

さらに、ＭＰＥＧ規格においては、フレームを構成する２つのフィールドのうち、最初のフィールドがトップフィールドかボトムフィールドかを示す‘ top_field_first ’というフラグを、符号化ストリーム中に記述することを定義している。具体的には、「top_field_first」が「１」である場合には、トップフィールドがボトムフィールドよりも時間的に早いフレーム構造であることを示し、「top_field_first」が「０」の場合には、ボトムフィールドがトップフィールドよりも時間的に早いフレーム構造であることを示す。

【００３０】

図５に示した例では、トップフィールドＴ１とボトムフィールドＢ１から構成されるフレームの符号化ストリームの「top_field_first」は「０」であって、トップフィールドＴ２とボトムフィールドＢ２から構成されるフレームの符号化ストリームの「top_field_first」は「１」であって、トップフィールドＴ３とボトムフィールドＢ３から構成されるフレームの符号化ストリームの「top_field_first」は「０」であって、トップフィールドＴ４とボトムフィールドＢ４から構成されるフレームの符号化ストリームの「top_field_first」は「１」である。

【００３１】

次に、図６を参照して、符号化ストリームをスプライシング処理したときの「top_field_first」及び「 repeat_first_field 」等のＭＰＥＧ規格において定義されているフラグに関して発生する問題点について説明する。

【００３２】

図６（Ａ）は、本局において制作されたオリジナルテレビ番組を符号化したオリジナルストリームＳＴOLDのフレーム構造を示している図であって、図５（Ｂ）は、地方局において制作された地方向けのコマーシャルＣ１’を符号化した差し替えストリームＳＴNEWのフレーム構造を示している図であって、図５（Ｃ）は、スプライシング処理されたスプライスドストリームＳＴSPLのフレーム構造を示している図である。

【００３３】

オリジナルストリームＳＴOLDにおけるプログラム１及びプログラム２は、共に２：３プルダウン処理された符号化ストリームであって、本編のコマーシャルＣＭ１の各フレームは、「 top_field_first 」が「０」となっているフレーム構造の符号化ストリームである。また、図６（Ｂ）に示された、地方コマーシャルＣＭ１’は、オリジナルテレビ番組のコマーシャルＣＭ１の部分に差し替えられる符号化ストリームであって、「top_field_first」が「１」となっているフレーム構造の符号化ストリームである。図６（Ｃ）に示されたスプライスドストリームＳＴSPLは、プログラム１で示されるオリジナルストリームＳＴOLDの後に、差し替えストリームＳＴNEWをスプライスし、さらに、差し替えストリームＳＴNEWの後に、プログラム２で示されるオリジナルストリームＳＴOLDをスプライスすることによって生成されたストリームである。つまり、スプライスドストリームＳＴSPLは、オリジナルストリームＳＴOLDの本編コマーシャルＣＭ１の替わりに地方コマーシャルＣＭ１’を挿入したストリームである。

【００３４】

図６に示した本局において制作されたコマーシャルＣＭ１の各フレームは、top_field_firstが「０」のフレーム構造の符号化ストリームであって、地方局において制作されたコマーシャルＣＭ１’は、top_field_firstが「１」のフレーム構造の符号化ストリームであることを示している。

【００３５】

図６（Ａ）及び図６（Ｂ）に示したように、コマーシャルＣＭ１のフレーム構造と、コマーシャルＣＭ１に対して差し替えられる差し替えコマーシャルＣＭ１’のフレーム構造とが異なる場合に、オリジナルストリームＳＰOLDのスプライスポイントＳＰ１で、プログラム１のストリームの後にコマーシャルＣＭ１’のストリームをスプライスすると、スプ

ライスドストリームＳＴSPLにおいてフィールドのギャップが生じてしまう。フィールドのギャップとは、図６（Ｃ）に示されたように、スプライスポイントＳＰ１におけるボトムフィールドＢ６が、スプライスドストリームＳＴSPLから脱落し、トップフィールドとボトムフィールドの繰り返しパターンが不連続になっていることを意味している。

【００３６】

このようにフィールドのギャップが生じてフィールドパターンが不連続になっている符号化ストリームは、ＭＰＥＧ規格違反の符号化ストリームであって、通常のＭＰＥＧデコーダでは正常に復号化できない。

【００３７】

また、図６（Ａ）及び図６（Ｂ）に示したように、オリジナルストリームＳＰOLDのスプライスポイントＳＰ２で、コマーシャルＣＭ１’のストリームの後にプログラム２のストリームをスプライスすると、スプライスドストリームＳＴSPLにおいてフィールドの重複が生じてします。このフィールドの重複とは、図６（Ｃ）に示されたように、スプライスポイントＳＰ２におけるボトムフィールドｂ１２及びボトムフィールドＢ１２が同じ表示時間に存在することになってしまうことを意味している。

【００３８】

このようにフィールドの重複が生じてフィールドパターンが不連続になっている符号化ストリームは、ＭＰＥＧ規格違反の符号化ストリームであって、通常のＭＰＥＧデコーダでは正常に復号化できない。

【００３９】

つまり、単純にスプラシング処理を行なうと、フィールドパターン／フレームパターンが不連続になってしまい、ＭＰＥＧ規格に準じたスプライスドストリームを生成することができなかった。

【００４０】

【発明が解決しようとする課題】

本発明の目的は、スプライシングされたストリームのＶＢＶバッファのデータ占有量の軌跡が連続であって、且つ、ＶＢＶバッファば破綻しないシームレスなスプライシング処理を実現するための符号化ストリームスプライシング装置を提供することにある。さらには、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならないシームレスなスプライシング処理を実現するための符号化ストリームスプライシング装置を提供することにある。

【００４１】

【課題を解決するための手段】

請求項１に記載の符号化ストリームスプライシング装置は、第１の符号化ストリームのシンタックスを解析することによって、第１の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、ストリーム解析手段によって得られた第１の符号化ストリームの符号化パラメータに基いて、スプライシングポイントにおいて第１の符号化ストリームと第２の符号化ストリームとがシームレスに接続されるように、第２の符号化ストリームの符号化パラメータを変更し、第１の符号化ストリームと符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシング手段とを備えたことを特徴とする。

【００４２】

請求項１１に記載の符号化ストリームスプライシング方法は、第１の符号化ストリームのシンタックスを解析することによって、第１の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、ストリーム解析ステップによって得られた第１の符号化ストリームの符号化パラメータに基いて、スプライシングポイントにおいて第１の符号化ストリームと第２の符号化ストリームとがシームレスに接続されるように、第２の符号化ストリームの符号化パラメータを変更し、第１の符号化ストリームと符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシングステップとを備えたことを特徴とする。

【００４３】
請求項２１に記載の符号化ストリーム生成装置は、第１の符号化ストリームのシンタックスを解析することによって、第１の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、ストリーム解析手段によって得られた第１の符号化ストリームの符号化パラメータに基いて、スプライシングポイントにおいて第１の符号化ストリームと第２の符号化ストリームとがシームレスに接続されるように、第２の符号化ストリームの符号化パラメータを変更し、第１の符号化ストリームと符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシング手段とを備えたことを特徴とする。

【００４４】
請求項２２に記載の符号化ストリーム生成方法は、第１の符号化ストリームのシンタックスを解析することによって、第１の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、ストリーム解析ステップによって得られた第１の符号化ストリームの符号化パラメータに基いて、スプライシングポイントにおいて第１の符号化ストリームと第２の符号化ストリームとがシームレスに接続されるように、第２の符号化ストリームの符号化パラメータを変更し、第１の符号化ストリームと符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシングステップとを備えたことを特徴とする。

請求項２３に記載の符号化ストリームスプライシング装置は、第１の符号化ストリームの符号化パラメータに基いて、第２の符号化ストリームの符号化パラメータを変更する符号化パラメータ変更手段と、第１の符号化ストリームと、符号化パラメータ変更手段により符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシング手段とを備えたことを特徴とする。

請求項２４に記載の符号化ストリームスプライシング方法は、第１の符号化ストリームの符号化パラメータに基いて、第２の符号化ストリームの符号化パラメータを変更する符号化パラメータ変更ステップと、第１の符号化ストリームと、符号化パラメータ変更ステップの処理により符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシングステップとを備えたことを特徴とする。

【００４７】
請求項１の符号化ストリームスプライシング装置及び請求項２１の符号化ストリーム生成装置においては、第１の符号化ストリームのシンタックスを解析することによって、第１の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、ストリーム解析手段によって得られた第１の符号化ストリームの符号化パラメータに基いて、スプライシングポイントにおいて第１の符号化ストリームと第２の符号化ストリームとがシームレスに接続されるように、第２の符号化ストリームの符号化パラメータを変更し、第１の符号化ストリームと符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシング手段とを備えることによって、スプライシングされたストリームのＶＢＶバッファのデータ占有量の軌跡が連続であって、且つ、ＶＢＶバッファが破綻しないシームレスなスプライシング処理を実現することができる。請求項２３に記載の符号化ストリームスプライシング装置においては、第１の符号化ストリームの符号化パラメータに基いて、第２の符号化ストリームの符号化パラメータが変更され、第１の符号化ストリームと、符号化パラメータが変更された第２の符号化ストリームとがスプライシングされる。また、本発明によれば、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならない整合性の取れたシームレスなストリームを生成することのできるスプライシング処理を実現できる。

【００４８】
また、請求項１１の符号化ストリームスプライシング方法及び請求項２２の符号化ストリーム生成方法においては、第１の符号化ストリームのシンタックスを解析することによって、第１の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、ストリーム解析ステップによって得られた第１の符号化ストリームの符号化パラメータ

に基いて、スプライシングポイントにおいて第１の符号化ストリームと第２の符号化ストリームとがシームレスに接続されるように、第２の符号化ストリームの符号化パラメータを変更し、第１の符号化ストリームと符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシングステップとを行なうことによって、スプライシングされたストリームのＶＢＶバッファのデータ占有量の軌跡が連続であって、且つ、ＶＢＶバッファが破綻しないシームレスなスプライシング処理を実現することができる。請求項２４に記載の符号化ストリームスプライシング方法においては、第１の符号化ストリームの符号化パラメータに基いて、第２の符号化ストリームの符号化パラメータが変更され、第１の符号化ストリームと、符号化パラメータが変更された第２の符号化ストリームとがスプライシングされる。また、本発明によれば、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならない整合性の取れたシームレスなストリームを生成することのできるスプライシング処理を実現できる。

【００５０】

【発明の実施の形態】

図７は、本発明の実施の形態に係る符号化ストリームスプライシング装置を含むデジタル放送システムを構成を示す図である。

【００５１】

図７に示されているように、一般的には、デジタル放送システムは主局（Key station）３０と、この主局系列の地方局４０とから構成されている。

【００５２】

主局３０は、系列の地方局に対して共通のテレビプログラムを制作及び伝送するための放送局であって、放送システムコントローラ３１と、素材サーバ３２と、ＣＭサーバ３３と、マトリックススイッチャブロック３４と、ＭＰＥＧエンコーダブロック３５と、マルチプレクサ３６と、変調回路３７とから構成されている。

【００５３】

放送システムコントローラ３１は、素材サーバ３２、ＣＭサーバ３３、マトリックススイッチャブロック３４、ＭＰＥＧエンコーダブロック３５、マルチプレクサ３６、及び変調回路３７等の放送局に設けられている全ての装置及び回路を統合的に管理及びコントロールするシステムである。この放送システムコントローラ３１は、番組供給会社から供給された番組素材及びプロモーション素材や自局で作成した番組素材やＣＭ素材などのあらゆる素材の放映時間を管理するための番組編成表が登録されており、放送番組編成システム１は、この番組編成表に従って、上述した各装置及び回路を制御する。この番組編成表は、例えば、１時間単位又は１日単位の放送番組スケジュールが記録されているイベント情報ファイル、及び１５秒単位の放送番組のタイムスケジュールが記録されている運行情報ファイル等から構成されている。

【００５４】

素材サーバ３２は、テレビ番組の本編として放送される映画番組、スポーツ番組、娯楽番組、ニュース番組等のビデオデータ及びオーディオデータを格納すると共に、放送システムコントローラ３１によって指定されたプログラムを、番組編成表のタイムスケジュールに従ったタイミングで出力するためのサーバである。また、この映画番組とは、先に説明したように、２４Ｈｚのフレーム周波数を有するフィルム素材から２：３プルダウン処理することによって３０Ｈｚのフレーム周波数を有するテレビジョン信号に変換されたビデオデータである。この素材サーバ３２から本編のプログラムとして出力されたビデオデータ及びオーディオデータはマトリックススイッチャブロック３４に供給される。例えば、図２に示した例では、プログラム１、プログラム２、プログラム３及びプログラム４等がこの素材サーバ３２に記録されている。尚、この素材サーバ３２に記憶されているビデオデータ及びオーディオデータは、圧縮符号化されていないベースバンドのビデオデータ及びオーディオデータである。

【００５５】

ＣＭサーバ３３は、素材サーバ３２から再生された本編のプログラム間に挿入されるコマ

ーシャルを格納すると共に、放送システムコントローラ３１によって指定されたコマーシャルを、番組編成表のタイムスケジュールに従ったタイミングで出力するためのサーバである。このＣＭサーバ３３からコマーシャルとして出力されたビデオデータ及びオーディオデータはマトリックススイッチャ３４に供給される。例えば、図２に示した例では、コマーシャルＣＭ１、コマーシャルＣＭ２、及びコマーシャルＣＭ３等が、このＣＭサーバ３３に記録されている。尚、このＣＭサーバ３３に記憶されているビデオデータ及びオーディオデータは、圧縮符号化されていないベースバンドのビデオデータ及びオーディオデータである。

【００５６】

マトリックススイッチャブロック３４は、スポーツ中継やニュース番組等のライブプログラム、素材サーバ３２から出力された本編プログラム、及びＣＭサーバ３３から出力されたコマーシャルプログラムをそれぞれルーティングするマトリックス回路を有している。さらに、マトリックススイッチャブロック３４は、放送システムコントローラによって決定された番組編成表のタイムスケジュールに従ったタイミングで、素材サーバ３２から供給された本編プログラムとＣＭサーバ３３から供給されたコマーシャルプログラムを接続してスイッチングするスイッチング回路を有している。このスイッチング回路によって、本編プログラムとＣＭプログラムをスイッチングすることによって、例えば図２に示した伝送プログラムＰＧOLDを生成することができる。

【００５７】

ＭＰＥＧエンコーダブロック３５は、マトリックススイッチャブロックから出力されたベースバンドのビデオデータ及びオーディオデータをＭＰＥＧ２規格に基いて符号化するためのブロックであって、複数のビデオエンコーダ及びオーディオエンコーダを有している。

【００５８】

マルチプレクサ３６は、ＭＰＥＧエンコーダブロック３５から出力された９チャンネルのトランスポートストリームを多重化して、１つの多重化トランスポートストリームを生成する。従って、出力される多重化トランスポートストリーム中には、１から９チャンネルまでの符号化ビデオエレメンタリーストリームを含んだトランスポートストリームパケットと１チャンネルから９チャンネルまでの符号化オーディオエレメンタリーストリームを含んだトランスポートストリームパケットとが混在しているストリームである。

【００５９】

変調回路３７は、トランスポートストリームをＱＰＳＫ変調して、その変調されたデータを伝送路を介して、地方局４０及び家庭６１に伝送する。

【００６０】

次に、地方局４０の全体構成に関して、図７を参照して説明する。

【００６１】

地方局４０は、本局から送られてきた共通のテレビプログラムを地方向けに編集し、地方向けに編集されたテレビ番組みを各家庭に放映するめの放送局であって、放送システムコートローラ４１と、復調回路４２と、デマルチプレクサ４３と、ストリーム変換回路４４と、素材サーバ４６と、ＣＭサーバ４７と、エンコーダブロック４８と、ストリームサーバ４９と、ストリームスプライサ５０と、ストリーム変換回路５１と、マルチプレクサ５２と、変調回路５３とを備えている。

【００６２】

放送システムコントローラ４１は、本局３０の放送システムコントローラ３１と同じように、復調回路４２、デマルチプレクサ４３、ストリーム変換回路４４、素材サーバ４６、ＣＭサーバ４７、エンコーダブロック４８、ストリームサーバ４９、ストリームスプライサ５０、ストリーム変換回路５１、マルチプレクサ５２、変調回路５３等の地方局に設けられている全ての装置及び回路を統合的に管理及びコントロールするシステムである。この放送システムコントローラ４１は、本局３０における放送システムコントローラ３１と同様に、本局３０から供給された伝送プログラムに対して、自局で作成した番組やＣＭな

どを挿入した編集テレビ番組の放映時間を管理するための番組編成表が登録されており、この番組編成表に従って、上述した各装置及び回路を制御する。

【００６３】

復調回路４２は、伝送路を介して本局３０から伝送された伝送ピログラムをＱＰＳＫ復調することによって、トランスポートストリームを生成する。

【００６４】

デマルチプレクサ４３は、復調回路４２から出力されたトランスポートスリームをデマルチプレックスして９チャンネルのトランスポートストリームを生成し、それぞれのチャンネルのトランスポートストリームをストリーム変換回路４４に出力する。つまり、このデマルチプレクサ４３は、本局３０のマルチプレクサ３６とは逆の処理を行なう。

【００６５】

ストリーム変換回路４４は、デマルチプレクサ４３から供給されたトランスポートストリームをエレメンタリーストリームの形態に変換するための回路である。

【００６６】

素材サーバ４６は、地方向けのテレビ番組として放送される娯楽番組、ニュース番組等のビデオデータ及びオーディオデータを格納しているサーバである。ＣＭサーバ４７は、本局３０から供給された本編プログラム間に挿入される地方向けコマーシャルのビデオデータ及びオーディオデータを格納するためのサーバである。この素材サーバ４６及びＣＭサーバ４７に記憶されているビデオデータ及びオーディオデータは、圧縮符号化されていないベースバンドのビデオデータ及びオーディオデータである。

【００６７】

エンコーダブロック４８は、素材サーバ４６及びＣＭサーバ４７から供給された複数チャンネルのビデオデータ及び複数チャンエルのオーディオデータを符号化するためのブロックであって、複数チャンネルに対応した複数のビデオエンコーダ及び複数のオーディオエンコーダを備えている。このエンコーダブロック４８と本局３０のＭＰＥＧエンコーダブロック３５との違いは、本局３０のＭＰＥＧエンコーダブロック３５はトランスポートストリームを出力するのに対して、この地方局４０のエンコーダブロック４８は、エレメンタリーストリーム出力する点で異なっているが、このエンコーダブロック４８の実質的な機能及び処理は本局３０のＭＰＥＧエンコーダブロック３５と全く同じである。エンコーダブロック４８から出力された複数チャンネルのエレメンタリーストリームのうち３チャンネル分のエレメンタリーストリームが、ストリームサーバ４９に供給され、残りのチャンネルのエレメンタリーストリームは、ストリームスプライサ５０に供給される。

【００６８】

ストリームサーバ４９は、エンコーダブロックから供給された３チャンネル分のエレメンタリーストリームを受取り、ストリームの状態でランダムアクセス可能な記録媒体にストリームの状態で記録すると共に、放送システムコントローラ４１からの制御にしたがって、そのエレメンタリーストリームをランダムアセクス可能な記録媒体から再生する。

【００６９】

ストリームスプライサ５０は、エンコーダブロック４８及びストリームサーバ４９から供給された複数のエレメンタリースをルーティングして所定の出力ラインに出力すると共に、本局３０から供給されたエレメンタリーストリームと、地方局４０において生成されたエレメンタリーストリームとをストリームレベルでスプライシングするためのブロックである。このストリームスプライサ５０における処理は、詳しくは後述する。

【００７０】

ストリーム変換回路５１は、ストリームスプライサ５０からスプライスドストリームとして出力されたエレメンタリーストリームを受けとり、このエレメンタリーストリームをトランスポートストリームに変換する回路である。

【００７１】

マルチプレクサ５２は、本局３０のマルチプレクサ３６と同じように、ストリーム変換回路から出力された９チャンネルのトランスポートストリームを多重化して、１つの多重化

トランスポートストリームを生成する。

【００７２】

変調回路５３は、トランスポートストリームをＱＰＳＫ変調して、その変調されたデータを伝送路を介して、各家庭６２に配信する。

【００７３】

図８は、本局３０のＭＰＥＧエンコーダブロック３５及び地方局４０のエンコーダブロック４８の構成を詳細に説明するためのブロック図である。本局３０のＭＰＥＧエンコーダブロック３５と地方局４０のエンコーダブロック４８とは実質的に同じ構成であるので、本局３０のＭＰＥＧエンコーダブロック３５を例にあげて、その構成及び機能を説明する。

【００７４】

ＭＰＥＧエンコーダブロック３５は、このＭＰＥＧエンコーダブロック３５の全ての回路を集中的に制御するためのエンコーダコントローラ３５０と、供給された複数チャンネルのビデオデータをエンコードするための複数のＭＰＥＧビデオエンコーダ３５１－１Ｖ～３５１－９Ｖと、ビデオデータにそれぞれ対応する複数のオーディオデータをＭＰＥＧ２規格に基いて符号化するためのＭＰＥＧオーディオエンコーダ３５１－１Ａ～３５１－９Ａとを有している。

【００７５】

さらに、ＭＰＥＧエンコーダブロック３５は、各ビデオエンコーダ３５１－１Ｖ～３５１－９Ｖからそれぞれ出力された符号化エレメンタリーストリーム（ＥＳ）をトランスポートストリームに変換するストリーム変換回路３５２－１Ｖ～３５２－９Ｖと、各オーディオエンコーダ３５１－１Ａ～３５１－９Ａからそれぞれ出力された符号化エレメンタリーストリーム（ＥＳ）をトランスポートストリームに変換するストリーム変換回路３５２－１Ａ～３５２－９Ａと、第１チャンネル（１ｃｈ）のビデオエレメンタリーストリームを含んだトランスポートストリームと、第１チャンネル（１ｃｈ）のオーディオエレメンタリーストリームを含んだトランスポートストリームとをトランスポートストリームパケット単位で多重化するマルチプレクサ３５３－１と、第２チャンネル（２ｃｈ）のビデオエレメンタリーストリームを含んだトランスポートストリームと、第２チャンネル（２ｃｈ）のオーディオエレメンタリーストリームを含んだトランスポートストリームとをトランスポートストリームパケット単位で多重化するマルチプレクサ３５３－２と、．．．．．．、第９チャンネル（９ｃｈ）のビデオエレメンタリーストリームを含んだトランスポートストリームと、第９チャンネル（９ｃｈ）のオーディオエレメンタリーストリームを含んだトランスポートストリームとをトランスポートストリームパケット単位で多重化するマルチプレクサ３５３－９とを備えている。

【００７６】

図８に示したＭＰＥＧエンコーダブロック３５は、９チャンネルの伝送プログラムをエンコードする構成となっているが、９チャンネルに限らず何チャンネルであっても良いことは言うまでもない。

【００７７】

図８に示したＭＰＥＧエンコーダブロック３５は、各チャンネルの伝送プログラムの伝送レートを、符号化するビデオデータの絵柄に応じてダイナミックに変化させる統計多重という制御を行なう。この統計多重という手法は、あるチャンネルの伝送プログラムのピクチャの絵柄が比較的簡単であって、このピクチャを符号化するためにそれほど多くのビットを必要としない場合であって、一方、その他のプログラムのピクチャの絵柄が比較的難しく、このピクチャを符号化するために多くのビットを必要とするような場合には、あるチェンネルのピクチャを符号化するためのビットを、その他のチャンネルのピクチャを符号化するビットに割当てることによって、伝送路の伝送レートを効率良くしようすることができる方法である。以下に、このようにダイナミックに各ビデオエンコーダの符号化レートを変化させる方法について簡単に説明する。

【００７８】

各ビデオエンコーダ３５１－１Ｖ～３５１－９Ｖは、まず、符号化処理の前に行われる動き補償の結果得られた動き補償残差やイントラＡＣ等の統計量から、符号化対象となっているピクチャを符号化するためにどの程度のビット量が必要かを示すディフィカルティデータ（ Difficulty Data ）Ｄ１～Ｄ９を生成する。このディフィカルティデータとは、符号化難易度を示す情報であって、ディフィカルティが大きいということは符号化対象となっているピクチャの絵柄が複雑であることを表わし、ディフィカルティが小さいということは符号化対象となっているピクチャの絵柄が簡単であることを表わしている。このディフィカルティデータは、ビデオエンコーダにおける符号化処理時に使用される、イントラＡＣや動き補償残差（ＭＥ残差）等の統計量に基いて、概算することができる。

【００７９】

エンコーダコントローラ３５０は、各ビデオエンコーダ３５１－１Ｖ～３５１－９Ｖからそれぞれ出力されたディフィカルティデータＤ１～Ｄ９を受取り、それらのディフィカルティデータＤ１～Ｄ９に基いて、各ビデオエンコーダ３５１－１Ｖ～３５１－９Ｖに対するターゲットビットレートＲ１～Ｒ９をそれぞれ演算する。具体的には、以下の式（１）に示すように、エンコーダコントローラ３５０は、伝送路のトータル伝送レート Total_Rate を、ディフィカルティデータＤ１～Ｄ９を用いて比例配分することによって、ターゲットビットレートＲ１～Ｒ９を求めることができる。

Ｒｉ ＝ （ Ｄｉ／ Σ Ｄｋ ） × Total_Rate ・・・ （１）

【００８０】

式（１）において、「 Ｒｉ 」は「ｉ」チャンネルの伝送プログラムのターゲットビットレート、「 Ｄｉ 」は「ｉ」チャンネルの伝送プログラムのピクチャを符号化するためのディフィカルティデータ、「 Σ 」はｋ＝１～９チャンネルのディフィカルティデータの総和を意味する。

【００８１】

エンコーダコントローラ３５０は、式（１）に基いて演算したターゲットビットレートＲ１～Ｒ９を、それぞれ対応するビデオエンコーダ３５１－１Ｖ～３５１－９Ｖに供給する。このターゲットビットレートＲ１～Ｒ９を演算する単位は、各ピクチャ毎であっても良いし、またＧＯＰ単位であっても良い。

【００８２】

ビデオエンコーダ３５１－１Ｖ～３５１－９Ｖは、エンコーダコントローラ３５０から供給されたターゲットビットレートＲ１～Ｒ９をそれぞれ受取り、このターゲットビットレートＲ１～Ｒ９に対応するように符号化処理を行なう。このように符号化するピクチャの符号化難易度を示すディフィカルティデータに基いて、各ビデオエンコーダから出力される符号化ストリームのビットレートをダイナミックに変更することによって、符号化すべきピクチャの符号化難易度に対して最適な量のビットを割り当てることができ、さらに、各ビデオエンコーダからそれぞれ出力されるビットレートの総量が、伝送路のトータル伝送レート Total_Rate をオーバーフローすることがない。

【００８３】

ストリーム変換回路３５２－１Ｖ～３５２－９Ｖ及び、ストリーム変換回路３５２－１Ａ～３５２－９Ａは、共に、エレメンタリーストリームをトランスポートストリームに変換するための回路である。

【００８４】

図９を参照して、ビデオエンコーダ３５１－１Ｖにおいて、供給されたソースビデオデータを符号化してビデオエレメンタリーストリームを生成し、そのビデオエレメンタリーストリームを、ストリーム変換回路３５２－１Ｖにおいてトランスポートストリームに変換する例を挙げて、ビデオエレメンタリーストリームからトランスポートストリームを生成する過程について説明する。

【００８５】

図９（Ａ）は、ビデオエンコーダ３５１－１Ｖに供給されるソースビデオデータを示し、図９（Ｂ）は、ビデオエンコーダ３５１－１Ｖから出力されたビデオエレメンタリースト

リーム（ＥＳ）を示し、図９（Ｃ）は、パケッライズドエレメンタリーストリーム（ＰＥＳ）、図９（Ｄ）は、トランスポートストリーム（ＴＳ）を示している。

【００８６】

図９（Ｂ）に示したストリームＶ１、Ｖ２、Ｖ３及びＶ４のように、ＭＰＥＧ２規格において符号化されたエレメンタリーストリームのデータ量は、ビデオフレームのピクチャタイプ（Ｉピクチャ、Ｐピクチャ又はＢピクチャ）及び動き補償の有無に応じて異なってくる。図９（Ｃ）に示したパケッタイズドエレメンタリーストリーム（ＰＥＳ）は、その複数のエレメンタリーストリームをパケット化し、その先頭にＰＥＳヘッダを付加することによって生成される。例えば、このＰＥＳヘッダには、ＰＥＳパケットの開始を示す２４〔bit〕のパケット開始コードと、ＰＥＳパケットの実データ部分に収容されるストリームデータの種別（例えばビデオや音声等の種別）を示す８〔bit〕のストリームＩＤと、以降に続くデータの長さを示す１６〔bit〕のパケット長と、値「１０」を示すコードデータと、各種フラグ情報が格納されるフラグ制御部と、コンディショナル・コーデイング部のデータの長さを示す８〔bit〕のＰＥＳヘッダ長と、ＰＴＳ（Presentation Time Stamp）と呼ばれる再生出力の時間情報やＤＴＳ（Decoding Time Stamp）と呼ばれる復号時の時刻管理情報、或いはデータ量調整のためのスタッフィングバイト等が格納される可変長のコンディショナル・コーデイング部とによつて構成される。

【００８７】

トランスポートストリーム（ＴＳ）は、４バイトのＴＳヘッダと１８４バイトの実データが記録されるペイロード部とから成るトランスポートストリームパケットのデータ列である。このトランスポートストリームパケット（ＴＳパケット）を生成するためには、まず、ＰＥＳパケットのデータストリームを１８４バイト毎に分解し、その１８４バイトの実データをＴＳパケットのペイロード部に挿入し、その１８４バイトのペイロードのデータに４バイトのＴＳヘッダを付加することによってトランスポートストリームパケットが生成される。

【００８８】

次に、図１０から図１７を参照してエレメンタリーストリームのシンタックス及び構造について説明すると共に、図１８から図１９を参照してトランスポートストリームのシンタックス及び構造について詳しく説明する。

【００８９】

図１０は、ＭＰＥＧのビデオエレメンタリーストリームのシンタックスを表わした図である。ビデオエンコーダブロック３５内の各ビデオエンコーダ３５１－１Ｖ～３５１９Ｖは、この図１０に示されたシンタックスに従った符号化エレメンタリーストリームを生成する。以下に説明するシンタックスにおいて、関数や条件文は細活字で表わされ、データエレメントは、太活字で表されている。データ項目は、その名称、ビット長及びそのタイプ・伝送順序を示すニーモニック（Mnemonic）で記述されている。

【００９０】

まず、この図１０に示されているシンタックスにおいて使用されている関数について説明する。実際には、この図１０に示されているシンタックスは、ビデオデコーダ側において伝送された符号化ストリームから所定の意味のあるデータを抽出するために使用されるシンタックスである。ビデオエンコーダ側において使用されるシンタックスは、図１０に示されたシンタックスからif分やwhile文等の条件文を省略したシンタックスである。

【００９１】

video_sequesce()において最初に記述されているnext_start_code()関数は、ビットストリーム中に記述されているスタートコードを探すための関数である。この図６に示されたシンタックスに従って生成された符号化ストリームには、まず最初に、sequence_header()関数とsequence_extension()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。このsequence_header()関数は、ＭＰＥＧビットストリームのシーケンスレイヤのヘッダデータを定義するための関数であって、sequence_extension()関数は、ＭＰＥＧビットストリームのシーケンスレイヤの拡張データを定義するための関数である。

【００９２】
sequence_extension()関数の次に配置されている do{ }while構文は、while文によって定義されている条件が真である間、do文の{ }内の関数に基いて記述されたデータエレメントが符号化データストリーム中に記述されていることを示す構文である。このwhile文に使用されているnextbits()関数は、ビットストリーム中に記述されているビット又はビット列と、参照されるデータエレメントとを比較するための関数である。この図６に示されたシンタックスの例では、nextbits()関数は、ビットストリーム中のビット列とビデオシーケンスの終わりを示すsequence_end_codeとを比較し、ビットストリーム中のビット列とsequence_end_codeとが一致しないときに、このwhile文の条件が真となる。従って、sequence_extension()関数の次に配置されている do{ }while構文は、ビットストリーム中に、ビデオシーケンスの終わりを示すsequence_end_codeが現れない間、do文中の関数によって定義されたデータエレメントが符号化ビットストリーム中に記述されていることを示している。

【００９３】
符号化ビットストリームにおいて、sequence_extension()関数によって定義された各データエレメントの次には、extension_and_user_data(0)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。このextension_and_user_data(0)関数は、ＭＰＥＧビットストリームのシーケンスレイヤにおける拡張データとユーザデータを定義するための関数である。

【００９４】
このextension_and_user_data(0)関数の次に配置されている do{ }while構文は、while文によって定義されている条件が真である間、do文の{ }内の関数に基いて記述されたデータエレメントが、ビットストリームに記述されていることを示す関数である。このwhile文において使用されているnextbits()関数は、ビットストリーム中に現れるビット又はビット列と、picture_start_code又はgroup_start_codeとの一致を判断するための関数であって、ビットストリーム中に現れるビット又はビット列と、picture_start_code又はgroup_start_codeとが一致する場合には、while文によって定義された条件が真となる。よって、このdo{ }while構文は、符号化ビットストリーム中において、picture_start_code又はgroup_start_codeが現れた場合には、そのスタートコードの次に、do文中の関数によって定義されたデータエレメントのコードが記述されていることを示している。

【００９５】
このdo文の最初に記述されているif文は、符号化ビットストリーム中にgroup_start_codeが現れた場合、という条件を示しいる。このif文による条件は真である場合には、符号化ビットストリーム中には、このgroup_start_codeの次にgroup_of_picture_header()関数及びextension_and_user_data(1)関数によって定義されているデータエレメントが順に記述されている。

【００９６】
このgroup_of_picture_header()関数は、ＭＰＥＧ符号化ビットストリームのＧＯＰレイヤのヘッダデータを定義するための関数であって、extension_and_user_data(1)関数は、ＭＰＥＧ符号化ビットストリームのＧＯＰレイヤの拡張データ及びユーザデータを定義するための関数である。

【００９７】
さらに、この符号化ビットストリームにおいて、group_of_picture_header()関数及びextension_and_user_data(1)関数によって定義されているデータエレメントの次には、picture_header()関数とpicture_coding_extension()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。もちろん、先に説明したif文の条件が真とならない場合には、 group_of_picture_header()関数及びextension_and_user_data(1)関数によって定義されているデータエレメントは記述されていないので、 extension_and_user_data(0)関数によって定義されているデータエレメントの次に、picture_header()関数、picture_coding_extension()関数及びextension_and_user_data(2)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。

【００９８】

このpicture_header()関数は、ＭＰＥＧ符号化ビットストリームのピクチャレイヤのヘッダデータを定義するための関数であって、picture_coding_extension()関数は、ＭＰＥＧ符号化ビットストリームのピクチャレイヤの第１の拡張データを定義するための関数である。extension_and_user_data(2)関数は、ＭＰＥＧ符号化ビットストリームのピクチャレイヤの拡張データ及びユーザデータを定義するための関数である。このextension_and_user_data(2)関数によって定義されるユーザデータは、ピクチャレイヤに記述されているデータであって、各ピクチャ毎に記述することのできるデータであるので、本発明においては、このextension_and_user_data(2)関数によって定義されるユーザデータとして、タイムコード情報を記述するようにしている。

【００９９】

符号化ビットストリームにおいて、ピクチャレイヤのユーザデータの次には、picture_data()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。このpicture_data()関数は、スライスレイヤ及びマクロブロックレイヤに関するデータエレメントを記述するための関数である。

【０１００】

このpicture_data()関数の次に記述されているwhile文は、このwhile文によって定義されている条件が真である間、次のif文の条件判断を行うための関数である。このwhile文において使用されているnextbits()関数は、符号化ビットストリーム中に、picture_start_code又はgroup_start_codeが記述されているか否かを判断するための関数であって、ビットストリーム中にpicture_start_code又はgroup_start_codeが記述されている場合には、このwhile文によって定義された条件が真となる。

【０１０１】

次のif文は、符号化ビットストリーム中にsequence_end_code が記述されているか否かを判断するための条件文であって、sequence_end_code が記述されていないのであれば、sequence_header()関数とsequence_extension()関数と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ことを示している。sequence_end_codeは符号化ビデオストリームのシーケンスの終わりを示すコードであるので、符号化ストリームが終了しない限り、符号化ストリーム中にはsequence_header()関数とsequence_extension()関数と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。

【０１０２】

このsequence_header()関数とsequence_extension()関数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()関数とsequence_extension()関数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理由は、ビットストリーム受信装置側でデータストリームの途中（例えばピクチャレイヤ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に、シーケンスレイヤのデータを受信できなくなり、ストリームをデコード出来なくなることを防止するためである。

【０１０３】

この最後のsequence_header()関数とsequence_extension()関数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。

【０１０４】

以下に、sequence_header()関数、sequence_extension()関数、extension_and_user_data(0)関数、group_of_picture_header()関数及びextension_and_user_data(1)関数について詳細に説明する。

【０１０５】

図１１は、sequence_header()関数のシンタックスを説明するための図である。このsequence_header()関数によって定義されたデータエレメントは、sequence_header_code、sequ

ence_header_present_flag、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_quantizer_matrix、intra_quantizer_matrix、load_non_intra_quantizer_matrix、及びnon_intra_quantizer_matrix等である。

【０１０６】

sequence_header_codeは、シーケンスレイヤのスタート同期コードを表すデータである。sequence_header_present_flagは、sequence_header内のデータが有効か無効かを示すデータである。 horizontal_size_valueは、画像の水平方向の画素数の下位12ビットから成るデータである。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(400bsp単位で切り上げる)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_valueは、発生符号量制御用の仮想バッファ（ビデオバッファベリファイヤー）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_intra_quantizer_matrixは、イントラMB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_quantizer_matrixは、イントラＭＢ用量子化マトリックスの値を示すデータである。load_non_intra_quantizer_matrixは、非イントラＭＢ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。non_intra_quantizer_matrixは、非イントラＭＢ用量子化マトリックスの値を表すデータである。

【０１０７】

図１２はsequence_extension()関数のシンタックスを説明するための図である。このsequence_extension()関数によって定義されたデータエレメントとは、extension_start_code、extension_start_code_identifier、sequence_extension_present_flag、profile_and_level_indication、progressive_sequence、chroma_format、horizontal_size_extension、vertical_size_extension、bit_rate_extension、vbv_buffer_size_extension、low_delay、frame_rate_extension_n 、及び frame_rate_extension_d等のデータエレメントである。

【０１０８】

extension_start_codeは、エクステンション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、どの拡張データが送られるかを示すデータである。sequence_extension_present_flagは、シーケンスエクステンション内のデータが有効であるか無効であるかを示すスデータである。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加える上位２ビットのデータである。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。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。

【０１０９】

図１３は、extension_and_user_data(i)関数のシンタックスを説明するための図である。このextension_and_user_data(i)関数は、「i」が2以外のときは、extension_data()関数

によって定義されるデータエレメントは記述せずに、user_data()関数によって定義されるデータエレメントのみを記述する。よって、extension_and_user_data(0)関数は、user_data()関数によって定義されるデータエレメントのみを記述する。

【０１１０】

図１４は、group_of_picture_header()関数のシンタックスを説明するための図である。このgroup_of_picture_header()関数によって定義されたデータエレメントは、group_start_code、group_of_picture_header_present_flag、time_code、closed_gop、及びbroken_linkから構成される。

【０１１１】

group_start_codeは、ＧＯＰレイヤの開始同期コードを示すデータである。 group_of_picture_header_present_flagは、 group_of_picture_header内のデータエレメントが有効であるか無効であるかを示すデータである。 time_codeは、ＧＯＰの先頭ピクチャのシーケンスの先頭から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。

【０１１２】

extension_and_user_data(1)関数は、 extension_and_user_data(0)関数と同じように、user_data()関数によって定義されるデータエレメントのみを記述するための関数である。

【０１１３】

次に、図１５から図１７を参照して、符号化ストリームのピクチャレイヤに関するデータエレメントを記述するためのpicture_headr()関数、picture_coding_extension()関数、extensions_and_user_data(2)及びpicture_data()について説明する。

【０１１４】

図１５はpicture_headr()関数のシンタックスを説明するための図である。このpicture_headr()関数によって定義された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、backward_f_code、extra_bit_picture、及びextra_information_pictureである。

【０１１５】

具体的には、picture_start_codeは、ピクチャレイヤ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。

【０１１６】

vbv_delayは、ＶＢＶバッファの初期状態を示すデータであって、各ピクチャ毎に設定されている。送信側システムから受信側システムに伝送された符号化エレメンタリーストリームのピクチャは、受信側システムに設けられたＶＢＶバッファにバッファリングされ、ＤＴＳ（Decoding Time Stamp）によって指定された時間に、このＶＢＶバッファから引き出され（読み出され）、デコーダに供給される。vbv_delayによって定義される時間は、復号化対象のピクチャがＶＢＶバッファにバッファリングされ始めてから、符号化対象のピクチャがＶＢＶバッファから引き出される時間、つまりＤＴＳによって指定された時間までを意味する。本発明の符号化ストリームスプライシング装置においては、このピクチャヘッダに格納されたvbv_delayを使用することによって、ＶＢＶバッファのデータ占有量な不連続にならないシームレスなスプライシングを実現するようにしている。詳しくは後述する。

【０１１７】

full_pel_forward_vectorは、順方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。forward_f_codeは、順方向動きベクトル探索範囲を表すデータである。full_pel_backward_vectorは、逆方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。backward_f_codeは、逆方向動きベクトル探索範囲を表すデータである

。 extra_bit_pictureは、後続する追加情報の存在を示すフラグである。このextra_bit_pictureが「１」の場合には、次にextra_information_pictureが存在し、extra_bit_pictureが「０」の場合には、これに続くデータが無いことを示している。extra_information_pictureは、規格において予約された情報である。

【０１１８】

図１６は、picture_coding_extension()関数のシンタックスを説明するための図である。このpicture_coding_extension()関数によって定義されたデータエレメントとは、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_predictive_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、及びsub_carrier_phaseから構成される。

【０１１９】

extension_start_codeは、ピクチャレイヤのエクステンションデータのスタートを示す開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。 f_code[0][0]は、フォア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、フォア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。intra_dc_precisionは、DC係数の精度を表すデータである。picture_structureは、フレームストラクチャかフィールドストラクチャかを示すデータである。フィールドストラクチャの場合は、上位フィールドか下位フィールドかもあわせて示すデータである。

【０１２０】

top_field_firstは、フレームストラクチャの場合、最初のフィールドがトップフィールドであるのかボトムフィールドであるのかを示すフラグである。frame_predictive_frame_dctは、フレーム・ストラクチャの場合、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。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を使うかどう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。

【０１２１】

repeat_first_fieldは、復号化時にリピートフィールドを生成するか否かを示すフラグであって、復号化時の処理において、repeat_first_field が「１」の場合にはリピートフィールドを生成し、repeat_first_fieldが「０」の場合にはリピートフィールドを生成しないという処理を行なう。chroma_420_typeは、信号フォーマットが４：２：０の場合、次のprogressive_frame と同じ値、そうでない場合は０を表すデータである。progressive_frameは、このピクチャが、順次走査できているかどう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どうかを示すデータである。v_axisは、ソース信号が、ＰＡＬの場合に使われるデータである。field_sequenceは、ソース信号が、ＰＡＬの場合に使われるデータである。sub_carrierは、ソース信号が、ＰＡＬの場合に使われるデータである。burst_amplitudeは、ソース信号が、ＰＡＬの場合に使われるデータである。sub_carrier_phaseは、ソース信号が、ＰＡＬの場合に使われるデータである。

【０１２２】

extension_and_user_data(2)関数は、図１３に示したように、符号化ビットストリーム中にエクステンションスタートコードextension_start_code が存在する場合には、extensi

on_data()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。但し、ビットストリーム中にエクステンションスタートコードが存在しない場合には extension_data()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は記述されていない。このextension_data()関数によって定義されているデータエレメントの次には、ビットストリーム中にユーザデータスタートコードuser_data_start_codeが存在する場合には、user_data()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。

【０１２３】

図１７は、picture_data()関数のシンタックスを説明するための図である。この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。但し、ビットストリーム中に、slice()関数のスタートコードを示すslice_start_codeが存在しない場合には、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記述されていない。

【０１２４】

slice()関数は、スライスレイヤに関するデータエレメントを記述するための関数であって、具体的には、slice_start_code、slice_quantiser_scale_code、intra_slice_flag、intra_slice、reserved_bits、extra_bit_slice、extra_information_slice、及びextra_bit_slice 等のデータエレメントと、macroblock()関数によって定義されるデータエレメントを記述するための関数である。

【０１２５】

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。slice_quantiser_scale_codeは、このスライスレイヤ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。しかし、各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。intra_slice_flagは、ビットストリーム中にintra_slice及びreserved_bitsが存在するか否かを示すフラグである。intra_sliceは、スライスレイヤ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。スライスレイヤ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、スライスレイヤにおけるマクロブロックの全てがノン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ビットのデータであって「０」の値を取る。extra_bit_sliceは、符号化ストリームとして追加の情報が存在すること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」に設定される。追加の情報が存在しない場合には「０」に設定される。

【０１２６】

macroblock()関数は、マクロブロックレイヤに関するデータエレメントを記述するための関数であって、具体的には、macroblock_escape、macroblock_address_increment、及びmacroblock_quantiser_scale_code等のデータエレメントと、macroblock_modes()関数、及び macroblock_vecters(s)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。

【０１２７】

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以上の場合には、macroblock_address_incrementの値に３３をプラスする。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、この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をプラスした値が、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。macroblock_quantiser_scale_codeは、各マクロブロック毎に設定された量子化ステップサイズである。各スライスレイヤには、スライスレイヤ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macroblock_quantiser_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。

【０１２８】

次に、図１８及び図１９を参照して、トランスポートストリームパケットの構造及びトランスポートストリームパケットのシンタックスについて詳しく説明する。

【０１２９】

トランスポートストリームパケットは、４バイトのヘッダと、各種のデータ及びデータエレメントを格納するための１８４バイトのペーロード部とから構成されている。

【０１３０】

トランスポートストリームパケットのヘッダ部は、sync_byte、transport_error_indicator、payload_unit_start_indicator、transport_priority、PID、transport_scrambling_control、adaptation_field_control、continuity_counter、及びadaptation_field等の各種フィールドから構成されている。

【０１３１】

sync_byteとは、ビットストリーム中から同期パターンを検出するための固定の８ビットのフィールドである。値は‘01000111’（0x47）の固定値で定義され、このストリーム中のこのビットパターンを検出することによって、同期を検出することができる。

【０１３２】

transport_error_indicatorは、1ビットのフラグである、「１」に設定されると、少なくとも1ビットの訂正できないビットエラーがトランスポートストリームパケットに存在することを示す。

【０１３３】

payload_unit_start_indicatorは、1ビットのフラグである。ビデオ／オーディオデータ等のエレメンタリーデータまたはプログラム仕様情報（ＰＳＩ）を伝送するトランスポートストリームパケットに対して規範的な意味を有するデータである。トランスポートストリームパケットのペイロードがエレメンタリーデータを含む場合、payload_unit_start_indicatorは、次の意味を有する。payload_unit_start_indicatorが「１」の場合には、このトランスポートストリームパケットのペイロードの最初に、エレメンタリーデータが挿入されていることを示し、payload_unit_start_indicatorが「０」の場合には、このトランスポートストリームパケットのペイロードの最初に、エレメンタリーデータが挿入されていないことを示す。もし、payload_unit_start_indicatorが「１」にセットされると、ただ一つのＰＥＳパケットが任意のトランスポートストリームパケットで開始することを示す。一方、トランスポートストリームパケットのペイロードがＰＳＩデータを含む場合、payload_unit_start_indicatorは、次の意味を有する。もし、トランスポートパケットがＰＳＩセクションの第１バイトを伝送する場合、payload_unit_start_indicatorは「１」となる。もし、トランスポートストリームパケットがＰＳＩセクションの第１バイトを伝送していない場合、payload_unit_start_indicatorは「０」となる。尚、トランスポートストリームパケットがヌルパケットの場合にも、payload_unit_start_indicatorは「０」となる。

【０１３４】

transport_priorityは、トランスポートパケットの優先度を示す１ビットの識別子である。このtransport_priorityが「１」に設定されると、このトランスポートパケットは、同一のパケット識別子ＰＩＤをもつパケットであって、このtransport_priorityが「１」でないパケットより優先度が高いことを示している。例えば、このtransport_priorityのパケット識別子を設定することによって、一つのエレメンタリーストリーム内において任意のパケットに優先度をつけることができる。

【０１３５】

transport_scrambling_controlは、トランスポートストリームパケットペイロードのスクランブリングモードを示す２ビットのデータである。スクランブリングモードとは、ペイロードに格納されたデータがスクランブされているか否か及びそのスクラングルの種類を

示すためのモードである。トランスポートストリームパケットヘッダ、およびアダプテーションフィールドは、スクランブルキーＫｓによってスクランブルされてはならないように規格化されている。よって、このtransport_scrambling_controlによって、トランスポートストリームパケットペイロードに格納されたデータがスクランブルされているか否かを判断することができる。

【０１３６】

adaptation_field_controlは、このトランスポートストリームのパケットヘッダにアダプテーションフィールド及び／又はペイロードがくることを示す２ビットのデータである。具体的には、パケットヘッダにペイロードデータのみが配置される場合には、このadaptation_field_controlは「０１」となり、パケットヘッダにアダプテーションフィールドのみが配置される場合には、このadaptation_field_controlは「１０」となり、パケットヘッダにアダプテーションフィールドとペイロードとが配置される場合には、このadaptation_field_controlは「１１」となる。

【０１３７】

continuity_counterは、連続して伝送された同じＰＩＤをもつパケットが、伝送途中で一部欠落又は捨てられか否かを示すためのデータである。具体的には、continuity_counterは、同一のＰＩＤを有する各トランスポートストリームパケットごとに増加する4ビットのフィールドである。但し、このcontimlty_comterがカウントされるときは、パケットヘッダにアダムテーションフィールドが配置されている場合である。

【０１３８】

adaptation_field（ ）は、個別ストリームに関する付加情報やスタッフィングバイト等をオプションとして挿入するためのフィールドである。このアダプテーションフィールドによって、個別ストリームの動的な状態変化に関するあらゆる情報をデータと一緒に伝送することができる。

【０１３９】

図１９は、adaptation_field（ ）のシンタックスを説明するための図である。このadaptation_field（ ）は、adaptation_field_length、discontinuity_indicator、randam_access_indicator、elemntary_stream_priority_indicator、OPCR_flag、splicing_point_flag、transport_private_data_flag、adaptation_field_extension_flag、program_clock_reference（PCR）、original_program_clock_reference（OPCR）、splice_countdown、、transport_private_data_length、private_data、adaptation_field_extension_length、ltw_flag(leagal_time_window_flag)、piecewise_rate_flag、及びseamless_splice_flag等の各種フィールドから構成されている。

【０１４０】

adaptation_field_lengthは、このadaptation_field_lengthの次に続くアダプテーションフィールドのバイト数を示すデータである。adaptation_field_controlが「１１」の場合には、adaptation_field_lengthは０から１８２ビットであって、adaptation_field_controlが「１０」の場合には、adaptation_field_lengthは１８３ビットとなる。尚、トランスポートストリームのペイロードを満たすだけのエレメンタリーストリームが無い場合には、ビットを満たすためのスタッフィング処理が必要となる。

【０１４１】

discontinuity_indicatorは、同じＰＩＤを有する複数のパケットの途中において、プログラムクロックリファレンス（ＰＣＲ）がリセットされ、プログラムクロックリファレンスが不連続になっているか否かを示すデータである。もし、プログラムクロックリファレンスが不連続の場合には、このdiscontinuity_indicatorは「１」となり、プログラムクロックリファレンスが連続している場合には、このdiscontinuity_indicatorは「０」となる。尚、このプログラムクロックリファレンスとは、ビデオ及びオーディオのデコードするためのＭＰＥＧデコーダにおいて、デコーダ側のシステムタイムクロックの値をエンコーダ側において意図したタイミングに設定するための参照情報である。

【０１４２】

randam_access_indicatorは、ビデオのシーケンスヘッダ又はオーディオのフレームの始まりを示すデータである。つまり、このrandam_access_indicatorは、データエレメントのランダムアクセスを行うときに、ビデオ又はオーディオのアクセスポイント（フレームの始まりのこと）であることを示すためのデータである。

【０１４３】

elemntary_stream_priority_indicatorは、同一のＰＩＤを有するパケットにおいて、このトランスポートストリームパケットのペイロード中で伝送されるエレメンタリーストリームデータの優先度を示すデータである。例えば、エレメンタリーストリームがビデオデータが、そのビデオデータがイントラ符号化されている場合に、elemntary_stream_priority_indicatorが「１」にセットされる。それに対して、インター符号化されているビデオデータを含んだトランスポートストリームのelemntary_stream_priority_indicatorは、「０」にセットされる。

【０１４４】

PCR_flagは、アダプテーションフィールド内にＰＣＲ（program_clock_refrence）データが存在するか否かを示すデータでる。アダプテーションフィールド内にＰＣＲデータが存在する場合には、PCR_flagが「１」にセットされ、ＰＣＲデータが存在しない場合には、PCR_flagが「０」にセットされる。尚、このＰＣＲデータとは、受信機側のデコーダにおいて、伝送されたデータをデコードするデコード処理のタイミングを得るために使用されるデータである。

【０１４５】

OPCR_flagは、アダプテーションフィールド内にＯＰＣＲ(original_program_clock_refrence)データが存在するか否かを示すデータでる。アダプテーションフィールド内にＯＰＣＲデータが存在する場合には、OPCR_flagが「１」にセットされ、ＯＰＣＲデータが存在しない場合には、OPCR_flagが「０」にセットされる。このＯＰＣＲデータとは、スプライインシング処理等によって、複数のオリジナルトランスポートストリームから１つのトランスポートストリームを再構築したときのに使用されるデータであって、あるオリジナルトランスポートストリームのＰＣＲデータを表わすデータである。

【０１４６】

splicing_point_flagは、トランスポートレベルでの編集ポイント（スプライスポイント）を示すためのsplice_countdounがアダプテーションフィールド内に存在するか否かを示すデータである。アダプテーションフィールド内にsplice_countdounが存在する場合には、このsplicing_point_flagは「１」であって、アダプテーションフィールド内にsplice_countdounが存在する場合には、このsplicing_point_flagは「０」である。

【０１４７】

transport_private_data_flagは、アダプテーションフィールド内に、任意のユーザデータを記述するためのプライベートが存在するか否かを示すためのデータである。アダプテーションフィールド内にプライベートが存在する場合には、このtransport_private_data_flagは「１」にセットされ、アダプテーションフィールド内にプライベートが存在しない場合には、このtransport_private_data_flagは「０」にセットされる。

【０１４８】

adaptation_field_extension_flagは、アダプテーションフィールド内に、拡張フィールド存在するか否かを示すためのデータである。アダプテーションフィールド内に拡張フィールドが存在する場合には、このadaptation_field_extension_flagは「１」にセットされ、アダプテーションフィールド内に拡張フィールドが存在しない場合には、このadaptation_field_extension_flagは「０」にセットされる。

【０１４９】

program_clock_reference（PCR）は、送信側とクロックの位相に対して受信機側のクロックの位相を同期させるときに参照する基準クロックである。このＰＣＲデータには、トランスポートパケットが生成された時間が格納されている。このＰＣＲは、３３ビットのprogram_clock_reference_baseと９ビットのprogram_clocLreference_extensionとの４２ビ

ットから構成されるデータである。program_clocLreference_extensionによって０～２９９までのシステムクロックをカウントし、２９９から０にリセットされる際のキャリーによって、program_clock_reference_baseに１ビットを加算することによって、２４時間をカウントすることができる。

【０１５０】

original_program_clock_reference（OPCR）は、あるトランスポートストリームから単一プログラムのトランスポートストリームを再構成するときに使用されるデータである。単一プログラムドランスボートストリームが完全に再構成された場合、このoriginal_program_clock_referenceはprogram_clock_referenceにコピーされる。

【０１５１】

splice_countdownは、同一ＰＩＤのトランスポートストリームパケットにおいて、トランスポートストリームパケットレベルで編集可能（スプライシング処理可能）なポイントまでのパケットの数を示すデータである。従って、編集可能なスプライシングポイントのトランスポートストリームパケットでは、splice_countdownは「０」である。splice_countdownが「０」になるトランスポートパケットで、トランスポートストリームパケットペイロードの最終バイトは、符号化されたピクチャの最後のバイトとすることによって、スプライシング処理が可能となる。

【０１５２】

このスプライシング処理とは、トランスポートレベルで行われる２つの異なるエレメンタリーストリームを連結し、１つの新しいトランスポートストリームを生成する処理のことである。そして、スプライシング処理として、復号の不連続性を発生しないシームレススプライスと、復号の不連続性を引き起こすノンシームレススプライスとに分けることができる。符号の不連続性を発生しないとは、新しく後ろにつなげられたストリームのアクセスユニットの復号時間と、スプライス前の古いストリームのアクセスユニットの復号時間と間の矛盾が無いことを示し、符号の不連続性を発生するとは、新しく後ろにつなげられたストリームのアクセスユニットの復号時間に対して、スプライス前の古いストリームのアクセスユニットの復号時間の矛盾が生じることを示している。

【０１５３】

transport_private_data_lengthは、アダプテーションフィールドにおけるプライデートデータのバイト数を示すデータである。

【０１５４】

private_dataは、規格では特に規定されておらず、アダプテーションフィールドにおいて任意のユーザデータを記述することができるフィールドである。

【０１５５】

adaptation_field_extension_lengthは、アダプテーションフィールドにおけるアダプテーションフィールドエクステンションのデータ長を示すデータである。

【０１５６】

ltw_flag(leagal_time_window_flag)は、アダプテーションフィールドにおいて表示ウインドウのオフセット値を示すltw_offsetが存在するか否かを示すデータである。

【０１５７】

piecewise_rate_flagは、アダプテーションフィールドにおいてpiecewise_rateが存在するか否かを示すデータである。

【０１５８】

seamless_splice_flagは、スプライシングポイントが、通常のスプライシングポイントか、シームレススプライシングポイントであるかを示すデータである。このseamless_splice_flagが「０」の時は、スプライシングポイントが通常のスプライシングポイントであることを示し、このseamless_splice_flagが「１」の時は、スプライシングポイントがシームレススプイシングポイントであることを示している。通常のスプライシングポイントとは、スプライシングポイントがＰＥＳパケットの区切りに存在する場合であって、このスプライシングポイントの直前のスプライシングパケットがアクセスユニットで終了し、次

の同じＰＩＤを有するトランスポートパケットがＰＥＳパケットのヘッダで開始している場合である。これに対して、シームレススプライシングポイントとは、ＰＥＳパケットの途中にスプライシングポイントがある場合であって、新しく後ろにつなげられたストリームのアクセスユニットの復号時間と、スプライス前の古いストリームのアクセスユニットの復号時間との間に矛盾が無いようにするために、古いストリームの特性の一部を、新しいストリームの特性として使う場合である。

【０１５９】

次に、本局３０から伝送されたきたストリームＳＴOLDと地方局４０において生成されたストリームＳＴNEWとをスプライシングするスプライシング処理について、図２０から図２３を参照して説明する。

【０１６０】

図２０は、図７において説明した地方局４０の制御をよりわかりやすく説明するために、複数チャンネルうちのある１チャンネルのみを残して、他のチャンネルを省略した図である。本発明においては、スプライシング処理に関する実施例として３つのスプライシング処理に関する実施例を有している。以下に、第１、第２及び第３のスプライシング処理に関する実施例を順に説明する。

【０１６１】

第１のスプライシング処理に関する実施例は、伝送プログラムの符号化ストリームＳＴOLDが本局３０から伝送されてくる前に、新しく挿入されるコマーシャルＣＭ’の符号化ストリームＳＴNEWが既に生成されている場合に行われるスプライシング処理に関する実施例である。つまり、伝送プログラムにおけるコマーシャルＣＭの符号化ストリームＳＴOLD部分に、既に前もって符号化されていコマーシャルＣＭ１’のストリームを挿入する場合である。通常、コマーシャルというものは何度も繰り返して放映されるので、その度にコマーシャルのビデオデータを符号化するのでは効率的ではない。そこで、地方向けのコマーシャルＣＭ１’のビデオデータを符号化し、その符号化ストリームＴＳNEWをストリームサーバ４９に予め記憶しておく。そして、本局３０から置換えられるコマーシャルＣＭ１の符号化ストリームＳＴOLDが伝送されてきたときに、このストリームサーバ４９から地方向けのコマーシャルＣＭ１’の符号化ストリームＳＴNEWを再生することによって、同じコマーシャルを何度も符号化する処理を省くことができる。このような場合に、以下に具体的に説明する第１のスプライシング処理が行われる。

【０１６２】

まず地方局４０において、伝送プログラムのコマーシャルＣＭ１の部分に置換えられる地方向けのコマーシャルＣＭ１’を符号化し、符号化ストリームＳＴNEWをストリームサーバ４９に格納する初期処理について説明する。放送システムコントローラ４１は、伝送プログラムのコマーシャルＣＭの部分に置換えられるコマーシャルＣＭ１’のビデオデータを再生するようにＣＭサーバ４７を制御する。そして、エンコーダ４８１は、ＣＭサーバ４７から再生されたベースバンドのビデオデータを受取り、このビデオデータの各ピクチャの符号化難易度（Difficulty）Ｄｉをエンコーダコントローラ４８０に供給する。エンコーダコントローラ４８０は、図８において説明したエンコーダコントローラ３５０と同じように、エンコーダ４８１が適切な符号化ビットを発生するようにエンコーダ４８１に対してターゲットビットレートＲｉを供給する。エンコーダ４８１は、エンコーダコントローラ４８０から供給されたターゲットビットレートＲｉに基いて符号化処理を行なうことによって、最適なビットレートの符号化エレメンタリーストリームＳＴNEWを生成することができる。エンコーダ４８１から出力された符号化エレメンタリーストリームＳＴNEWは、ストリームサーバ４９に供給される。ストリームサーバ４９は、符号化エレメンタリーストリームをストリームの状態のまま、ランダムアクセス可能な記録媒体に記録する。これで、符号化ストリームＳＴNEWをストリームサーバ４９に格納する初期処理は終了する。

【０１６３】

次に、本局から伝送されてきた伝送プログラムの符号化ストリームＳＴOLDと、上述した

初期処理によってストリームサーバ４９に格納された符号化ストリームＳＴNEWとをスプライシングするスプライシング処理について説明する。

【０１６４】

本局３０から伝送されてきた符号化ストリームＳＴOLDは、ストリーム変換回路４４においてトランスポートストリームの形式からエレメンタリーストリームの形式に変換される。エレメンタリーストリームの形式に変換された符号化ストリームＳＴOLDは、ストリームスプライサ５０に供給される。

【０１６５】

ストリームスプライサ５０は、図２０に示されているように、スプライスコントローラ５００、スイッチ回路５０１、ストリーム解析回路５０２、ストリームプロセッサ５０３、及びスプライシング回路５０４を備えている。

【０１６６】

この第１のスプライシング処理に関する実施例では、スプライスコントローラ５００は、スイッチ回路５０１の入力端子を「ａ」に切換え、ストリームサーバ４９から供給されたエレメンタリーストリームＳＴNEWをストリーム解析回路５０２に供給する。

【０１６７】

ストリーム解析回路５０２は、符号化ストリームＳＴOLD及び符号化ストリームＳＴNEWのシンタックスを解析する回路である。具体的には、ストリーム解析回路５０２は、図１０及び図１５に開示された符号化ストリームのシンタックスから理解できるように、符号化ストリームＳＴOLD中に記述された３２ビットのpicture_start_codeを探すことによって、ストリーム中においてピクチャヘッダに関する情報が記述された場所を把握する。次に、ストリーム解析回路５０２は、picture_start_codeの１１ビット後から始まる３ビットのpicture_coding_typeを見つけることによって、ピクチャタイプを把握すると共に、この３ビットのpicture_coding_typeの次に記述された１６ビットのvbv_delayから、符号化ピクチャのvbv_delayを把握することができる。

【０１６８】

さらに、ストリーム解析回路５０２は、図１０及び図１５に開示された符号化ストリームのシンタックスから理解できるように、符号化ストリームＳＴOLD及び符号化ストリームＳＴNEW中に記述された３２ビットのextension_start_codeを探すことによって、ストリーム中においてピクチャコーディングエクステンションに関する情報が記述された場所を把握する。次に、ストリーム解析回路５０２は、picture_start_codeの２５ビット後から記述された１ビットのtop_field_firstと、そのtop_field_firstの６ビット後から記述されたrepeat_first_fieldとを探すことによって、符号化ピクチャのフレーム構造を把握することができる。例えば、符号化ピクチャの「top_field_first」が「１」である場合には、トップフィールドがボトムフィールドよりも時間的に早いフレーム構造であることを示し、「top_field_first」が「０」の場合には、ボトムフィールドがトップフィールドよりも時間的に早いフレーム構造であることを示す。また、符号化ストリーム中のフラグ「top_field_first」が「０」であって且つ「repeat_first_field 」が「１」の場合には、復号化時にトップフィールドからリピートフィールドが生成されるようなピクチャ構造であることを示し、符号化ストリーム中のフラグ「top_field_first」が「０」であって且つ「repeat_first_field 」が「１」の場合には、復号化時にボトムフィールドからリピートフィールドが生成されるようなピクチャ構造を有していることを示している。

【０１６９】

上述した、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldは、各ピクチャ毎に符号化ストリームから抽出され、スプライスコントローラ５００に供給される。ストリーム解析回路５０２に供給されたエレメンタリーストリームＳＴOLD及びエレメンタリーストリームＳＴNEWは、そのままエレメンタリーストリームＳＴOLD及びエレメンタリーストリームＳＴNEWとしてストリームプロセッサ５０３に供給される。

【０１７０】

さらに、ストリーム解析回路５０２は、供給されたストリームＳＴOLDとストリームＳＴN

EWのビット数をカウントするためのカウンタを備えており、このカウント値と各ピクチャの発生ビット量とに基いて、各ピクチャ毎にＶＢＶバッファのデータ残量をシュミレーションするようにしている。ストリーム解析回路５０２において演算された各ピクチャ毎のＶＢＶバッファのデータ残量も、スプライスコントローラ５００に供給される

【０１７１】

ストリームプロセッサ５０３は、ストリームＳＴOLDとストリームＳＴNEWとをスプライシングすることによって生成されるスプライスドストリームＳＴSPLがシームレスなストリームとなるように、このストリームＳＴOLD及びストリームＳＴNEWのストリーム構造、データエレメント及びフラグを変更するための回路である。このストリームプロセッサ５０３の具体的な処理を、図２１を参照して説明する。

【０１７２】

図２１（Ａ）は、本局３０から供給されたオリジナルストリームＳＴOLDと、そのストリームＳＴOLDのＶＢＶバッファのデータ占有量の軌跡を示す図であって、図２１（Ｂ）は、ストリームサーバ４９に記憶された差し替えストリームＳＴNEWと、そのストリームＳＴNEWのＶＢＶバッファのデータ占有量の軌跡を示す図であって、図２１（Ｂ）は、スプライシングポイントＳＰ１及びＳＰ２において、ストリームＳＴOLDとストリームＳＴNEWとをスプライシングしたスプライスドストリームＳＴSPLと、そのスプライスドストリームＳＴSPLのＶＢＶバッファのデータ占有量の軌跡を示す図である。尚、図２１（Ａ）において、ＤＴＳ（デコーディングタイムスタンプ）を示し、ＳＰ１vbvは、ＶＢＶバッファのデータ占有量の軌跡上における第１のスプライシングポイントを示し、ＳＰ２vbvは、ＶＢＶバッファデータ占有量の軌跡上における第２のスプライシングポイントを示し、ＶＯ(I6)は、ピクチャＢ５がＶＢＶバッファから引き出されたときにＶＢＶバッファ上にバッファリングされているピクチャＩ６のデータ量を示し、ＧＢ(I6)は、ピクチャＩ６の発生ビット量を示し、ＶＤ(I6)は、ピクチャＩ６のvbv_delayの値を示し、ＶＯ(B7)は、ピクチャＩ６がＶＢＶバッファから引き出されたときにＶＢＶバッファ上にバッファリングされているピクチャＢ７のデータ量を示し、ＧＢ(B11)は、ピクチャＢ１１の発生ビット量を示し、ＶＤ(I12)は、ピクチャＩ１２のvbv_delayの値を示し、ＶＯ(I12)は、ピクチャＢ１１がＶＢＶバッファから引き出されたときにＶＢＶバッファ上にバッファリングされているピクチャＩ１２のデータ量を示している。また、図２１（Ｂ）において、ＧＢ(I6')は、ピクチャＩ６’の発生ビット量を示し、ＶＤ(I6')は、ピクチャＩ６’のvbv_delayの値を示し、ＶＯ(I6')は、ＶＢＶにおける第１のスプライシングポイントＳＰ１vbvにおけるピクチャＩ６’のＶＢＶバッファにバッファリングされているデータ量を示し、ＧＢ(B11')は、ピクチャＢ１１’の発生ビット量を示し、ＶＯ(I12')は、ＶＢＶにおける第２のスプライシングポイントＳＰ２vbvにおけるピクチャＢ１２’のＶＢＶバッファにバッファリングされているデータ量を示している。また、図２１（Ｃ）において、ＧＢ(I6")は、スプライスドストリームＳＴSPLがシームレスなストリームとなるようにストリーム処理されたピクチャＩ６"の発生ビット量を示し、ＶＤ(I6”)は、ピクチャＩ６”のvbv_delayの値を示し、ＧＢ(B11")は、スプライスドストリームＳＴSPLがシームレスなストリームとなるようにストリーム処理されたピクチャＢ１１”の発生ビット量を示している。

【０１７３】

オリジナルストリームＳＴOLDは本局３０において符号化されたストリームであって、差し替えストリームＳＴNEWは地方局４０において符号化されたストリームであるので、ストリームＳＴOLDとストリームＳＴNEWとは、それぞれのビデオエンコーダで全く関係無く符号化されたストリームである。つまり、ストリームＳＴOLDにおける最初のピクチャＩ６のvbv_delayの値ＶＤ(I6)と、ストリームＳＴNEWにおける最初のピクチャＩ６’のvbv_delayの値ＶＤ(I6')とは同じ値ではない。つまり、このような場合には、ＶＢＶバッファにおけるストリームスプライスポイントＳＰ１vbvのタイミングにおいて、オリジナルストリームＳＴOLDのＶＢＶハッファのデータ占有量ＶＯ(I6)と、差し替えストリームＳＴNEWのＶＢＶバッファのデータ占有量ＶＯ(I6')とは異なってしまう。

【０１７４】
つまり、本発明の背景技術において説明したように、スプライスポイントＳＰ１において、ストリームＳＴOLDとストリームＳＴNEWとを単純にスプライスしてしまうと、単純にスプライスされたストリームのＶＢＶバッファのデータ占有量が不連続になるか又はオーバーフロー／アンダーフローしてしまう。
【０１７５】
そこで、ストリームスプライサ５０では、ストリーム解析回路５０２においてストリームＳＴOLD及びストリームＳＴNEWから抽出されたデータエレメントに基いて、スプライスドストリームＳＴSPLがスプライスポイントにおいてシームレスなストリームとなるように、ストリームプロセッサ５０３において、供給されたストリームＳＴOLD及びストリームＳＴNEWのストリーム構造に関してストリーム処理を行なっている。その処理を以下に説明する。
【０１７６】
スプライスコントローラ５００は、ストリームＳＴOLDに関するデータエレメントとして、各ピクチャ毎に、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_field等の情報、各ピクチャにおける発生ビット量、各ピクチャにおけるＶＢＶバッファのデータ占有量をストリーム解析回路５０２から受け取る。図２１においては、例えば、ピクチャＩ６におけるvbv_delayの値をＶＤ(I6)と表わし、ピクチャＩ６における発生ビット量をＧＢ(I6)と表わし、ピクチャＩ６におけるＶＢＶバッファのデータ占有量をＶＯ(I6)というように表わしている。
【０１７７】
次に、スプライスポイントＳＰ１におけるスプライスコントローラ５００とストリームプロセッサ５０３のＶＢＶバッファに関する処理について説明する。
【０１７８】
まず、スプライスコントローラ５００は、スプライシングポイントＳＰ１における、オリジナルストリームＳＴOLDのピクチャＩ６のvbv_delayの値ＶＤ(I6)と、差し替えストリームＳＴNEWのピクチャＩ６’のvbv_delayの値ＶＤ(I6')とが異なると判断した場合には、差し替えストリームＳＴNEW中に記述されたピクチャＩ６’のvbv_delayの値をＶＤ(I6')からＶＤ(I6)に書き換えるようにストリームプロセッサ５０３に指示を与える。
【０１７９】
ストリームプロセッサ５０３は、スプライスコントローラ５００からの指示に従って、差し替えストリームＳＴNEWのピクチャヘッダ中に記述された１６ビットのvbv_delayの値をＶＤ(I6')からＶＤ(I6)に書き換える。
【０１８０】
ここで、単に、差し替えストリームＳＴNEWにおけるvbv_delayの値をＶＤ(I6')からＶＤ(I6)に書き換え、この書き換えたvbv_delayに従ってＶＢＶバッファからビットストリームを引き出そうとすると、ピクチャＩ６’の発生ビット量が足りないので、ＶＢＶバッファがアンダーフローしてしまう。そこで、スプライスコントローラ５００は、差し替えストリームＳＴNEWのピクチャＩ６’の発生ビット量ＧＢ(I6')が、シームレスなスプライスドストリームＳＴSPLのピクチャＩ６”の発生ビット量ＧＢ(I6")になるように、差し替えストリームＳＴNEWのピクチャＩ６’に対してスタッフィングバイトを挿入する処理を行なう。このスタッフィングバイトとは、「０」のダミービットから構成されるデータである。
【０１８１】
スタッフィングバイトを挿入する処理を行うために、スプライスコントローラ５００は、ストリームＳＴOLDにおけるピクチャＩ６及びピクチャＢ７に関する情報として受け取った発生ビット量ＧＢ(I6)、ＶＢＶバッファのデータ占有量ＶＯ(I6)、及びストリームＳＴNEWにおけるピクチャＩ６’に関する情報として受け取った発生ビット量ＧＢ(I6')、ＶＢＶバッファのデータ占有量ＶＯ(I6')を使用して、挿入すべきスタッフィングバイトのデータ量を演算する。具体的には、下記の式（２）に基いてスタッフィングバイトＳＢ１［

byte] が演算される。

$$SB1[byte] = \{ GB(I6'') - GB(I6') \} / 8$$
$$= \{ GB(I6) - GB(I6') + VO(B7) - VO(B7') \} / 8 \quad \cdots (2)$$

【０１８２】
スプライスコントローラ５００は、上式（２）に従って演算したスタフィングバイトＳＢ１を、ストリームＳＴNEWの中に挿入するようにストリームプロセッサ５０３を制御する.

【０１８３】
ストリームプロセッサ５０３は、スプライスコントローラ５００からの指令に従って、スタフィングバイトＳＢ１を、ストリームＳＴNEWの中に記述する。スタッフィングバイトをストリーム中に記述する位置としては、符号化ストリームＳＴNEWのピクチャＩ６のピクチャヘッダのスタートコードの前が最も望ましいが、他のスタートコードの前であっても問題はない。

【０１８４】
以上が、スプライスポイントＳＰ１におけるスプライスコントローラ５００とストリームプロセッサ５０３のＶＢＶバッファに関する制御である。

【０１８５】
次に、スプライスポイントＳＰ２におけるスプライスコントローラ５００とストリームプロセッサ５０３のＶＢＶバッファに関する制御について説明する。

【０１８６】
単に、スプライスポイントＳＰ２において、ストリームＳＴNEWとストリームＳＴOLDとをスプライシングしたとすると、ストリームＳＴNEWの最後のピクチャＢ１１’の発生ビット量ＧＢ(B11')が足りないので、ストリームＳＴNEWの最初のピクチャＩ１２のＶＢＶバッファのデータ占有量の軌跡と連続にならない。その結果、ＶＢＶバッファがアンダーフロー又はオーバーフローしてしまう。

【０１８７】
そこで、スプライスコントローラ５００は、ＶＢＶバッファにおけるスプライスポイントＳＰ２vbvにおいて、ＶＢＶバッファの軌跡が連続になるように、ストリームＳＴNEWの最後のピクチャＢ１１’の発生符号量ＧＢ(B11')が、図２１（Ｃ）のピクチャＢ１１”の発生符号量ＧＢ(11")となるように、ストリームＳＴNEW中にスタッフィングバイトを挿入する処理を行なう。

【０１８８】
スタッフィングバイトを挿入する処理を行うために、スプライスコントローラ５００は、ストリームＳＴOLDにおけるピクチャＩ１２に関する情報として受け取ったＶＯ(I12)、ストリームＳＴNEWの最後のピクチャ１１’の発生ビット量ＧＢ(B11')、及びストリームＳＴNEWのピクチャ１２’のＶＢＶバッファのデータ占有量ＶＯ(I12')を使用して、挿入すべきスタッフィングバイトのデータ量を演算する。尚、具体的には、下記の式（２）に基いてスタッフィングバイトＳＢ２ [byte] が演算される。

$$SB2\ [byte] = \{ GB(B11'') - GB(B11') \} / 8$$
$$= \{ VO(I12') - VO(I12) \} / 8 \quad \cdots (3)$$

【０１８９】
尚、データ占有量ＶＯ(I12')とは、最後のピクチャＢ１１’をＶＢＶバッファから引き出した後のストリームＳＴNEWに関するＶＢＶバッファのデータ占有量であると言い換えることができ、ストリームＳＴNEWのビット数をカウントすることによってＶＢＶの軌跡を

把握しているストリーム解析回路５０２によって、容易にこのデータ占有量ＶＯ(I12')検出することができる。

【０１９０】

このスプライスコントローラ５００は、上式（３）に従って演算したスタフィングバイトＳＢ２を、ストリームＳＴNEW中に挿入するようにストリームプロセッサ５０３を制御する.

【０１９１】

ストリームプロセッサ５０３は、スプライスコントローラ５００からの指令に従って、スタフィングバイトＳＢ２を、ストリームＳＴNEWのピクチャＢ１１’に関する情報として記述する。スタッフィングバイトをストリーム中に記述する位置としては、符号化ストリームＳＴNEWのピクチャＢ１１’のピクチャヘッダのスタートコードの前が最も望ましい。

【０１９２】

以上が、スプライスポイントＳＰ２におけるスプライスコントローラ５００とストリームプロセッサ５０３のＶＢＶバッファに関する制御である。

【０１９３】

図２２を参照して、スプライスポイントＳＰ１におけるスプライスコントローラ５００とストリームプロセッサ５０３のtop_field_first及びrepeat_first_field等のフラグに関する第１の処理例を説明する。

【０１９４】

図２２（Ａ）は、本局３０において制作されたプログラム１とコマーシャルＣＭ１とプログラム２から構成されるテレビ番組ＰＧOLDのフレーム構造と、そのテレビ番組ＰＧOLDを符号化したときの符号化ストリームＳＴOLDを示している図である。図２２（Ｂ）は、地方局４０において制作された差し替えコマーシャルＣＭ１’のフレーム構造と、その差し替えコマーシャルＣＭ１’ををを符号化したときの符号化ストリームＳＴNEWを示している図である。図２２（Ｃ）は、オリジナルストリームＳＴOLDと差し替えストリームＳＴNEWとをスプライシングしたときに生成されるスプライスドストリームＳＴSPLと、そのスプライスドストリームＳＴSPLをデコードしたときのフレーム構造を示している図である。

【０１９５】

スプライスコントローラ５００は、ストリーム解析回路５０２から供給されたストリームＳＴOLDにおけるコマーシャルＣＭ１の各ピクチャのtop_field_firstと、差し替えストリームＳＴNEWにおけるコマーシャルＣＭ１’のtop_field_firstとを比較する。もし、ストリームＳＴOLDにおけるtop_field_firstと、差し替えストリームＳＴNEWにおけるtop_field_firstとが一致するのであれば、フィールド構造が同じであるので、top_field_first及びrepeat_first_field等のフラグに関する処理は必要ない。しかし、図２２に示すように、オリジナルコマーシャルＣＭ１のtop_field_firstが「０」であって、差し替えコマーシャルＣＭ１’のtop_field_firstが「１」である場合には、図６において説明したようなフィールドの不連続及び重複という問題が発生する。

【０１９６】

そこで、本発明のストリームスプライサ５０は、スプライシング処理によってフィールドが欠けたり重複したりするようなＭＰＥＧストリーム違反のストリームが生成されないように、スプライシングポイント付近のピクチャのtop_field_first及びrepeat_first_fieldを書き換えるようにしている。

【０１９７】

図２２に示された例においては、スプライスコントローラ５００は、トップフィールドＴ４とボトムフィールドＢ４とからフレームが構成されるピクチャＰ３のrepeat_first_fieldを０から１に書き換えるようにストリームプロセッサ５０３を制御する。さらに、スプライスコントローラ５００は、スプライスポイントＳＰ２においてシームレスなストリームとなるように、トップフィールドｔ１０とボトムフィールドｂ１１からフレームが構成されるピクチャＰ９’のrepeat_first_fieldを０から１に書き換えるようにストリームプ

ロセッサ５０３を制御する。また、スプライスコントローラ５００は、ピクチャＰ９’のrepeat_first_fieldを書き換えたことによって、コマーシャルＣＭ１’がオリジナルコマーシャルＣＭ１に対して１フレーム時間だけシフトしたので、プログラム２において最初にディスプレイに表示されるピクチャＢ１３を、ストリームＳＴOLD中から削除するようにストリームプロセッサ５０３を制御する。

【０１９８】

ストリームプロセッサ５０３は、スプライスコントローラ５００の指示に基いて、オリジナルストリームＳＴOLDにおいてピクチャＰ３に関するpicture_coding_extensionのスタートコードを探し、その中のrepeat_first_fieldの値を０から１に書き換える。よって、このようにrepeat_first_fieldの値が書き換えられたピクチャＰ３をデコードすると、リピートフィールドＢ４’が生成されるので、スプライスポイントＳＰ１においてフィールドが連続することになる。同様に、ストリームプロセッサ５０３は、差し替えストリームＳＴNEWにおいてピクチャＰ９’に関するpicture_coding_extensionのスタートコードを探し、その中のrepeat_first_fieldの値を０から１に書き換える。このようにrepeat_first_fieldの値が書き換えられたピクチャＰ９’をデコードすると、リピートフィールドｔ１０’が生成されるので、スプライスポイントＳＰ２においてフィールドが連続することになる。また、ストリームプロセッサ５０３、オリジナルストリームＳＴOLDにおいてピクチャＢ１３に関するデータエレメントが記述された部分を、ストリームＳＴOLD中から削除又はヌルデータに置き換える。

【０１９９】

図２３は、図２２において説明したtop_field_first及びrepeat_first_field等のフラグに関する処理の他の処理例を表わしたものである。図２３を参照して、スプライスポイントＳＰ１及びＳＰ２におけるスプライスコントローラ５００とストリームプロセッサ５０３のtop_field_first及びrepeat_first_field等のフラグに関する第２の処理例を説明する。

【０２００】

図２３に示された処理例においては、スプライスコントローラ５００は、スプライスポイントＳＰ１におけるプログラム１とコマーシャルＣＭ１’とのつなぎめにおけるフィールドが連続するように、トップフィールドｔ５とボトムフィールドｂ６とから構成されるピクチャＢ７’のtop_field_firstを１から０に書き換えると共に、ピクチャＢ７’のrepeat_first_fieldを０から１に書き換えるようにストリームプロセッサ５０３を制御する。さらに、スプライスコントローラ５００は、スプライスポイントＳＰ２においてコマーシャルＣＭ１’とプログラム２のつなぎめにおけるフィールドが連続するように、トップフィールドＴ１１とボトムフィールドＢ１１から構成されるピクチャＢ１３のtop_field_firstを１から０に書き換えるようにストリームプロセッサ５０３を制御する。さらに、スプライスコントローラ５００は、トップフィールドＴ１２とボトムフィールドＢ１２から構成されるピクチャＢ１４のtop_field_firstを１から０に書き換えると共に、repeat_first_fieldを１から０に書き換えるようにストリームプロセッサ５０３を制御する。

【０２０１】

ストリームプロセッサ５０３は、スプライスコントローラ５００の制御に従って、差し替えストリームＳＴNEWにおいてピクチャＢ７’に関するpicture_coding_extensionのスタートコードを探し、そのストリーム中のtop_field_firstを１から０に書き換えると共に、repeat_first_fieldを０から１に書き換える。よって、このようにtop_field_first及びrepeat_first_fieldの値が書き換えられたピクチャＢ７’をデコードすると、ボトムフィールドｂ６の表示時間が１フレーム分だけシフトし、かつ、リピートフィールドｂ６’が生成されるので、スプライスポイントＳＰ１においてフィールドが連続することになる。同様に、ストリームプロセッサ５０３は、オリジナルストリームＳＴOLDにおいてピクチャＢ１３に関するpicture_coding_extensionのスタートコードを探し、その中のtop_field_firstを１から０に書き換える。さらに、ストリームプロセッサ５０３は、オリジナルストリームＳＴOLDにおいてピクチャＢ１４に関するtop_field_firstを１から０に書き

換えると共に、repeat_first_fieldを１から０に書き換える。
よって、このようにtop_field_first及びrepeat_first_fieldの値が書き換えられたピクチャＢ１３及びＢ１４をデコードすると、ボトムフィールドＢ１１及びＢ１２の表示時間が１フレーム分だけシフトするので、スプライスポイントＳＰ２においてフィールドが連続することになる。
【０２０２】
ここで、図２２に示された第１の処理例と、図２３に示された第２の処理例とを比較すると、図２２（Ｃ）から理解できるように、差し替えられたコマーシャルＣＭ１’の最初に表示されるピクチャＢ７’が、オリジナルコマーシャルＣＭ１の最初に表示されるピクチャＢ７よりも、１フィールドだけずれているので、差し替えられたコマーシャルＣＭ１’の表示タイミングが１フィールドだけ遅れてしまう。表示が１フィールド遅れた程度では、人間の目にはほとんどその遅れは分からない。しかしながら、放送局ではクライアント会社からのコマーシャルを放映することによって収入を得ているので、プログラム１等の本編を放映することよりもコマーシャルを遅れなく正確に放映することが要求される場合がある。このような正確な表示時間が要求される場合には、図２３に示された第２の処理例が有効である。図２３に示された第２の処理例のように、ピクチャＢ７’のtop_field_first及びrepeat_first_fieldの値を書き換えることによって、差し替えられたコマーシャルＣＭ１’の最初のピクチャＢ７’を、オリジナルコマーシャルＣＭ１の最初に表示されるピクチャに対して遅れ無く正確に表示することができる。
【０２０３】
つまり、ストリームプロセッサ５０３から出力されたストリームＳＴNEWのＶＢＶバッファのデータ占有量の軌跡は、ストリームＳＴOLDのＶＢＶバッファのデータ占有量の軌跡と整合性が取れており、且つ、フィールドパターン／フレームパターンに関する整合性が取れている。よって、スプライスコントローラ５００からの制御信号に基いてスプライシング回路５０４のスイッチング動作を制御することによって、スプライスポイントＳＰ１で、ストリームＳＴOLDの後にストリームＳＴNEWを接続し、スプライスポイントＳＰ２で、ストリームＳＴNEWの後にストリームＳＴOLDを接続することによって生成されたストリームであって、そのスプライスポイントＳＰ１及びＳＰ２において、ＶＢＶバッファのデータ占有量の軌跡が連続しており、且つフィールドパターン／フレームパターンが連続になっているスプライスドストリームＳＴSPLが生成される。
【０２０４】
次に、第２のスプライシング処理に関する実施例について説明する。第２のスプライシング処理に関する実施例は、伝送プログラムの符号化ストリームＳＴOLDが本局３０から伝送されてきたときに、新しく挿入されるコマーシャルＣＭ’を符号化して符号化ストリームＳＴNEWを生成する場合に行われるスプライシング処理に関する実施例である。つまり、本局３０から伝送されてきた伝送プログラムの符号化ストリームＳＴOLDを解析し、その解析結果に基いて、新しく挿入されるコマーシャルＣＭ１’を符号化するという方法である。
【０２０５】
まず、本局３０から伝送されてきた符号化ストリームＳＴOLDは、ストリーム変換回路４４においてトランスポートストリームの形式からエレメンタリーストリームの形式に変換される。エレメンタリーストリームの形式に変換された符号化ストリームＳＴOLDは、ストリームスプライサ５０のストリーム解析回路５０２に供給される。
【０２０６】
ストリームスプライサ５０のストリーム解析回路５０２は、符号化ストリームＳＴOLDのストリームシンタックスを解析するための回路である。この第２のスプライシング処理に関する実施例においては、このストリーム解析回路５０２は、符号化ストリームＳＴOLDのシンタックスの解析のみを行い、差し替えストリームＳＴNEWのシンタックスの解析は行なわない。
【０２０７】

具体的には、まず、ストリーム解析回路５０２は、図１０及び図１５に開示された符号化ストリームのシンタックスから理解できるように、オリジナルストリームＳＴOLD中に記述された３２ビットのpicture_start_codeを探すことによって、ストリーム中においてピクチャヘッダに関する情報が記述された場所を把握する。次に、ストリーム解析回路５０２は、picture_start_codeの１１ビット後から始まる３ビットのpicture_coding_typeを見つけることによって、ピクチャタイプを把握すると共に、この３ビットのpicture_coding_typeの次に記述された１６ビットのvbv_delayから、符号化ピクチャのvbv_delayを把握することができる。

【０２０８】

さらに、ストリーム解析回路５０２は、図１０及び図１５に開示された符号化ストリームのシンタックスから理解できるように、符号化ストリームＳＴOLD中に記述された３２ビットのextension_start_codeを探すことによって、ストリーム中においてピクチャコーディングエクステンションに関する情報が記述された場所を把握する。次に、ストリーム解析回路５０２は、picture_start_codeの２５ビット後から記述された１ビットのtop_field_firstと、そのtop_field_firstの６ビット後から記述されたrepeat_first_fieldとを探すことによって、符号化ピクチャのフレーム構造を把握することができる。

【０２０９】

ストリーム解析回路５０２は、オリジナルストリームＳＴOLD中から各ピクチャ毎に抽出された、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_field等のデータエレメントを、スプライスコントローラ５００を介して、放送システムコントローラ４１に供給する。尚、オリジナルストリームＳＴOLDの全ピクチャのデータエレメントを送る必要は無く、伝送プログラムにおけるコマーシャルＣＭ１に対応するピクチャのpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_field等のデータエレメントのみであっても良い。

【０２１０】

次に、放送システムコントローラ４１は、伝送プログラムのコマーシャルＣＭの部分に差し替えられるコマーシャルＣＭ１’のビデオデータを再生するようにＣＭサーバ４７を制御する。さらに、放送システムコントローラ４１は、オリジナルストリームＳＴOLDから抽出されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを、エンコーダブロック４８のエンコーダコントローラ４８０に供給する。

【０２１１】

エンコーダコントローラ４８０は、放送システムコントローラ４１から供給されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを使用して、差し替えコマーシャルＣＭ１’のベースバンドビデオデータをエンコードするようにエンコーダ４８１を制御する。すなわち、オリジナルコマーシャルＣＭ１の符号化ストリームＳＴOLDのpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldと、差し替えコマーシャルＣＭ１’を符号化したストリームＳＴNEWの picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldとが全く同じになるように、差し替えコマーシャルＣＭ１’を符号化する。その結果、オリジナルコマーシャルＣＭ１の符号化ストリームＳＴOLDのpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldと、全く同じ picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを有した符号化したストリームＳＴNEWが生成される。

【０２１２】

スプライスコントローラ５０１は、スイッチ回路５０１の入力端子を「ｂ」に切換え、エンコーダ４８１から出力されたエレメンタリーストリームＳＴNEWをストリーム解析回路５０２に供給する。この第２のスプライシング処理に関する実施例においては、このストリーム解析回路５０２は、符号化ストリームＳＴOLDのシンタックスの解析のみを行い、差し替えストリームＳＴNEWのシンタックスの解析は行なわないので、ストリームＳＴNEWは、ストリーム解析回路５０２において解析処理されずにそのまま出力される。

【０２１３】

ストリームプロセッサ５０３は、この第２のスプライシング処理に関する実施例では、ストリーム解析回路５０２から出力されたストリームＳＴOLD及びストリームＳＴNEWにおけるデータエレメントを変更するようなストリーム処理は必要ないので、ストリームＳＴOLDとストリームＳＴNEWとのフレーム同期を合わせる同期合わせ処理（フレームシンクロナイゼーション）のみを行なう。具体的には、このストリームプロセッサ５０３は、数フレーム分のＦＩＦＯバッファを有しており、差し替えストリームＳＴNEWがエンコーダ４８１から出力されるまで、このＦＩＦＯバッファにストリームＳＴOLDをバッファリングしておくことによって、ストリームＳＴOLDとストリームＳＴNEWのフレーム同期を合わせることができる。フレーム同期合わせ処理されたストリームＳＴOLDとストリームＳＴNEWは、スプライシング回路５０４に供給される。

【０２１４】

スプライスコントローラ５００は、スプライシングポイントＳＰ１においてストリームＳＴOLDの後にストリームＳＴNEWが接続され、スプライシングポイントＳＰ２においてストリームＳＴNEWの次にストリームＳＴOLDが接続されるように、スプライシング回路５０４のスイッチングを制御する。その結果、スプライシング回路５０４からは、スプライスドストリームＳＴSPLが出力される。

【０２１５】

単純にスプライシング回路５０４において、ストリームＳＴOLDとストリームＳＴNEWとをスイッチングしているだけであるが、スプライスドストリームＳＴSPLのＶＢＶバッファのデータ占有量の軌跡は連続となっており、且つ、スプライスポイントにおけるフレームパターンも連続している。なぜなら、オリジナルストリームＳＴOLDをストリームのシンタックスの解析結果を基に、ストリームＳＴNEWを符号化処理しているので、オリジナルストリームＳＴOLDに対して整合性の取れたストリームＳＴNEWを生成しているので、スプライスドストリームＳＴSPLのＶＢＶバッファの軌跡は、オリジナルストリームＳＴOLDのＶＢＶバッファの軌跡と全く同じであって、生成されたスプライスドストリームＳＴSPLのフレーム構造は、オリジナルストリームＳＴOLDのフレーム構造と全く同じであるからである。

【０２１６】

従って、この第２の実施例によれば、本局から伝送されてきたオリジナル符号化ストリームＳＴOLDのシンタックスを解析し、その解析結果に応じて、符号化ストリームＳＴOLDと同じストリーム構造及び符号化パラメータを有するように、差し替えコマーシャルＣＭ１’を符号化しているので、それぞれ別に生成された、符号化ストリームＳＴOLDと符号化ストリームＳＴNEWをスプライシングする際に、符号化ストリームＳＴOLDと符号化ストリームＳＴNEWとの整合性を合わせることなる簡単にスプライシングを行なうことができ、その結果、ＭＰＥＧ規格に準じた且つシームレスなスプライスドストリームＳＴSPLを生成することができる。

【０２１７】

次に、第３のスプライシング処理に関する実施例について説明する。第３のスプライシング処理に関する実施例は、オリジナルコマーシャルＣＭ１の符号化ストリームＳＴOLD及び、差し替えられるコマーシャルＣＭ１’の符号化ストリームＳＴNEWを生成する前に、予め、オリジナルコマーシャルＣＭ１の符号化ストリームＳＴOLDと差し替えコマーシャルＣＭ１’の符号化ストリームＳＴNEWを符号化するためのの符号化パラメータを決定しておき、その決っている符号化パラメータに基いてオリジナルコマーシャルＣＭ１及び差し替えコマーシャルＣＭ１’を符号化するという処理である。例えば、この符号化パラメータとは、既に説明した、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldなどや発生ビット量等によって示される情報である。

【０２１８】

まず、本局３０において、オリジナルコマーシャルＣＭ１を符号化するための符号化パラメータとして、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを決定する。本局３０の放送システムコントローラ３１は、その符号化パラメータを

、ＭＰＥＧエンコーダブロック３５のエンコーダコントローラ３５０に供給すると共に、通信回線を使用して各地方局４０の放送システムコントローラ４１にも供給する。

【０２１９】

エンコーダコントローラ３５０は、放送システムコントローラ３１から供給されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldなどの符号化パラメータを使用してオリジナルコマーシャルＣＭ１のビデオデータを符号化するようにビデオエンコーダ３５１－１Ｖを制御する。すなわち、ビデオエンコーダ３５１－１Ｖから出力された符号化ストリームＳＴOLDは、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldなどの符号化パラメータに基いているストリームである。

【０２２０】

ビデオエンコーダ３５１－１Ｖから出力された符号化ストリームＳＴOLDは、マルチプレクサ３６及び変調回路３７を介して地方局４０に供給される。

【０２２１】

一方、地方局４０は、本局３０の放送システムコントローラ３１から供給されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_field等の符号化パラメータを、エンコーダブロック４８のエンコーダコントローラ４８０に供給する。

【０２２２】

エンコーダコントローラ４８０は、放送システムコントローラ４１から供給されたpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを使用して、差し替えコマーシャルＣＭ１’のベースバンドビデオデータをエンコードするようにエンコーダ４８１を制御する。すなわち、オリジナルコマーシャルＣＭ１の符号化ストリームＳＴOLDのpicture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldと、全く同じ picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldを有した符号化したストリームＳＴNEWが生成される。

【０２２３】

スプライスコントローラ５０１は、スイッチ回路５０１の入力端子を「ｂ」に切換え、エンコーダ４８１から出力されたエレメンタリーストリームＳＴNEWをストリーム解析回路５０２に供給する。この第３のスプライシング処理に関する実施例においては、このストリーム解析回路５０２は、符号化ストリームＳＴOLD及び符号化ストリームＳＴNEWのシンタックスの解析は行なわない。

【０２２４】

ストリームプロセッサ５０３は、この第２のスプライシング処理に関する実施例では、ストリーム解析回路５０２から出力されたストリームＳＴOLD及びストリームＳＴNEWにおけるデータエレメントを変更するようなストリーム処理は必要ないので、ストリームＳＴOLDとストリームＳＴNEWとのフレーム同期を合わせる同期合わせ処理（フレームシンクロナイゼーション）のみを行なう。具体的には、このストリームプロセッサ５０３は、数フレーム分のＦＩＦＯバッファを有しており、差し替えストリームＳＴNEWがエンコーダ４８１から出力されるまで、このＦＩＦＯバッファにストリームＳＴOLDをバッファリングしておくことによって、ストリームＳＴOLDとストリームＳＴNEWのフレーム同期を合わせることができる。フレーム同期合わせ処理されたストリームＳＴOLDとストリームＳＴNEWは、スプライシング回路５０４に供給される。

【０２２５】

スプライスコントローラ５００は、スプライシングポイントＳＰ１においてストリームＳＴOLDの後にストリームＳＴNEWが接続され、スプライシングポイントＳＰ２においてストリームＳＴNEWの次にストリームＳＴOLDが接続されるように、スプライシング回路５０４のスイッチングを制御する。その結果、スプライシング回路５０４からは、スプライスドストリームＳＴSPLが出力される。

【０２２６】

単純にスプライシング回路５０４において、ストリームＳＴOLDとストリームＳＴNEWとをスイッチングしているだけであるが、スプライスドストリームＳＴSPLのＶＢＶバッファ

のデータ占有量の軌跡は連続となっており、且つ、スプライスポイントにおけるフレームパターンも連続している。なぜなら、予め本局３０の放送システムコントローラ３１において決定された、picture_coding_type、vbv_delay、top_field_first及びrepeat_first_fieldなどの符号化パラメータを使用して、オリジナルコマーシャルＣＭ１及び差し替えコマーシャルＣＭ１’が符号化されているからである。

【０２２７】

従って、この第３の実施例によれば、本局及び地方局間において、予め符号化パラメータを決定しておき、その決められた符号化パメータタに基いて本局においてオリジナルコマーシャルＣＭ１を符号化した符号化ストリームＳＴOLDを生成し、さらに、その決められた符号化パメータタに基いて本局において差し替えコマーシャルＣＭ１’を符号化した符号化ストリームＳＴNEWを生成するようにしているので、それぞれ別に生成された、符号化ストリームＳＴOLDと符号化ストリームＳＴNEWをスプライシングする際に、符号化ストリームＳＴOLDと符号化ストリームＳＴNEWとの整合性を合わせることなる簡単にスプライシングを行なうことができ、その結果、ＭＰＥＧ規格に準じた且つシームレスなスプライスドストリームＳＴSPLを生成することができる。

【０２２８】

【発明の効果】

請求項１の符号化ストリームスプライシング装置及び請求項２１の符号化ストリーム生成装置によれば、第１の符号化ストリームのシンタックスを解析することによって、第１の符号化ストリームの符号化パラメータを抽出するストリーム解析手段と、ストリーム解析手段によって得られた第１の符号化ストリームの符号化パラメータに基いて、スプライシングポイントにおいて第１の符号化ストリームと第２の符号化ストリームとがシームレスに接続されるように、第２の符号化ストリームの符号化パラメータを変更し、第１の符号化ストリームと符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシング手段とを備えることによって、スプライシングされたストリームのＶＢＶバッファのデータ占有量の軌跡が連続であって、且つ、ＶＢＶバッファば破綻しないシームレスなスプライシング処理を実現することができる。また、本発明によれば、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならない整合性の取れたシームレスなストリームを生成することのできるスプライシング処理を実現できる。

【０２２９】

また、請求項１１の符号化ストリームスプライシング方法及び請求項２２の符号化ストリーム生成方法によれば、第１の符号化ストリームのシンタックスを解析することによって、第１の符号化ストリームの符号化パラメータを抽出するストリーム解析ステップと、ストリーム解析ステップによって得られた第１の符号化ストリームの符号化パラメータに基いて、スプライシングポイントにおいて第１の符号化ストリームと第２の符号化ストリームとがシームレスに接続されるように、第２の符号化ストリームの符号化パラメータを変更し、第１の符号化ストリームと符号化パラメータが変更された第２の符号化ストリームとをスプライシングするスプライシングステップとを行なうことによって、スプライシングされたストリームのＶＢＶバッファのデータ占有量の軌跡が連続であって、且つ、ＶＢＶバッファば破綻しないシームレスなスプライシング処理を実現することができる。また、本発明によれば、スプライシングポイント前後における、符号化ストリームのストリーム構造が不連続にならない整合性の取れたシームレスなストリームを生成することのできるスプライシング処理を実現できる。

【図面の簡単な説明】

【図１】一般的なテレビジョン放送システムを説明するための図である。

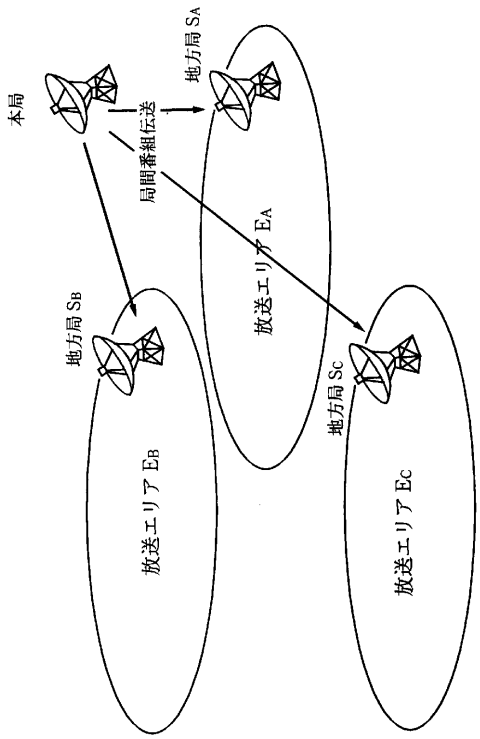
【図２】地方局におけるテレビ番組の編集処理を説明するための図である。

【図３】ＭＰＥＧ規格を用いた一般的なデジタル伝送システムを説明するための図である。

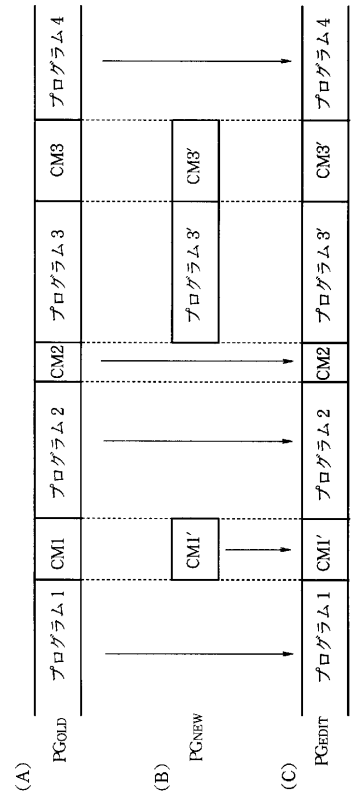
【図４】ＶＢＶバッファの制御について説明するための図である。

【図５】２：３プルダウン処理を説明するための図である。
【図６】符号化ストリームをスプライシング処理したときに発生する問題点を説明するための図である。
【図７】本発明の実施の形態に係る符号化ストリームスプライシング装置を含むデジタル放送システムを全体構成を示す図である。
【図８】本局３０のＭＰＥＧエンコーダブロック３５及び地方局４０のエンコーダブロックの構成を詳細に説明するためのブロック図である。
【図９】ストリーム変換回路においてエレメンタリーストリームからトランスポートストリームを生成するときの処理を説明するための図である。
【図１０】ＭＰＥＧのビデオエレメンタリーストリームのシーケンスのシンタックスを説明するための図である。
【図１１】シーケンスヘッダ（sequence_header）のシンタックスを説明するための図である。
【図１２】シーケンスエクステンション（sequence_extension）のシンタックスを説明するための図である。
【図１３】エクステンションアンドユーザデータ（extension_and_user_data）のシンタックスを説明するための図である。
【図１４】グループオブピクチャヘッダ（group_of_picture_header）のシンタックスを説明するための図である。
【図１５】ピクチャヘッダ（picture_headr）のシンタックスを説明するための図である。
【図１６】ピクチャコーディングエクステンション（picture_coding_extension）のシンタックスを説明するための図である。
【図１７】ピクチャヘッダ（picture_data）のシンタックスを説明するための図である。
【図１８】トランスポートストリームパケットのシンタックスを説明するための図である。
【図１９】アダプテーションフィールド（adaptation_field）のシンタックスを説明するための図である。
【図２０】地方局の簡単な構成及びストリームスプライサの構成を説明するための図である。
【図２１】ストリームスプライサのＶＢＶバッファに関する処理を説明するための図である。
【図２２】ストリームスプライサのtop_field_first及びrepeat_first_fieldに関する第１の処理例を説明するための図である。
【図２３】ストリームスプライサのtop_field_first及びrepeat_first_fieldに関する第２の処理例を説明するための図である。
【符号の説明】
３１　放送システムコントローラ，　３２　素材サーバ，　３３　ＣＭサーバ，　３４　マトリックススイッチャブロック，　３５　ＭＰＥＧエンコーダブロック，　３６　マルチプレクサ，　３７　変調回路，　４１　放送システムコントローラ，　４２　復調回路，　４３　デマルチプレクサ，　４４　ストリーム変換回路，　４６　素材サーバ，　４７　ＣＭサーバ，　４８　エンコーダブロック，　４９　ストリームサーバ，　５０　ストリームスプライサ，　５１　ストリーム変換回路，　５２　マルチプレクサ，　５３　変調回路，　６１，６２家庭

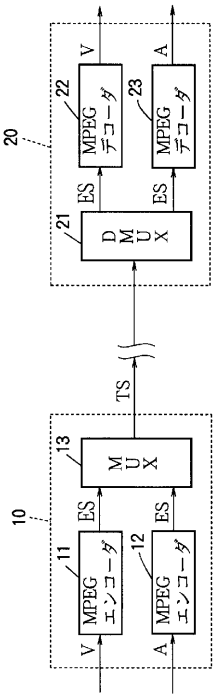
【図１】


本局
局間番組伝送
地方局 SA
地方局 SB
地方局 SC
放送エリア EA
放送エリア EB
放送エリア EC

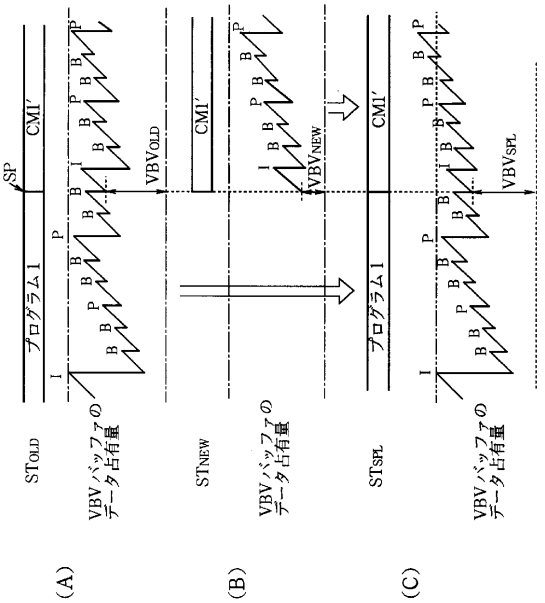
【図２】


(A)
PGOLD
プログラム1
CM1
プログラム2
CM2
プログラム3
CM3
プログラム4
(B)
PGNEW
CM1′
プログラム3′
CM3′
(C)
PGEDIT
CM1′
プログラム2
CM2
プログラム3′
CM3′
プログラム4

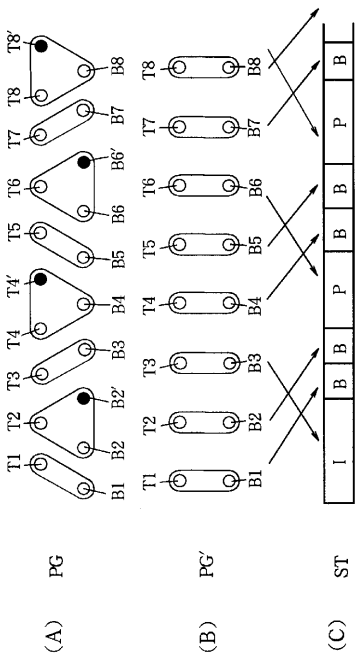
【図３】


10
20
11
MPEG エンコーダ
12
MPEG エンコーダ
13
MUX
21
DMUX
22
MPEG デコーダ
23
MPEG デコーダ
V
A
ES
TS

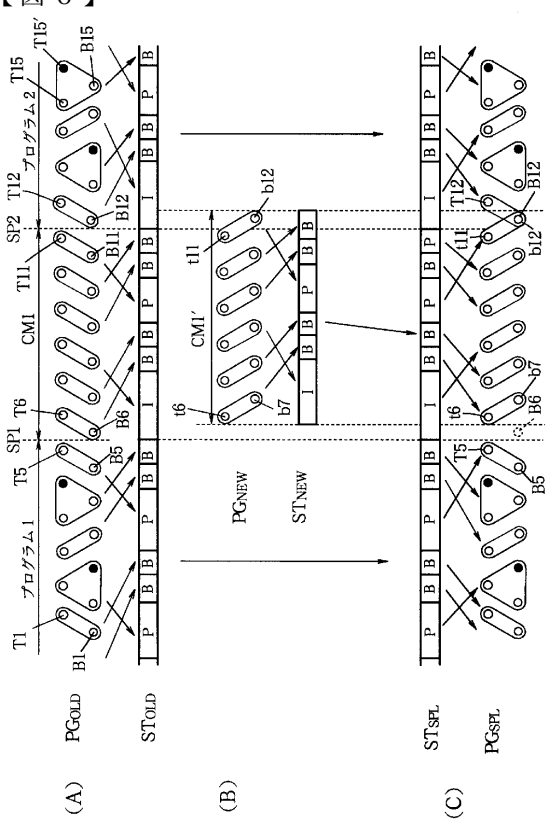
【図４】


(A)
(B)
(C)
STOLD
STNEW
STSPL
VBV バッファのデータ占有量
SP
CM1′
プログラム1
VBVOLD
VBVNEW
VBVSPL

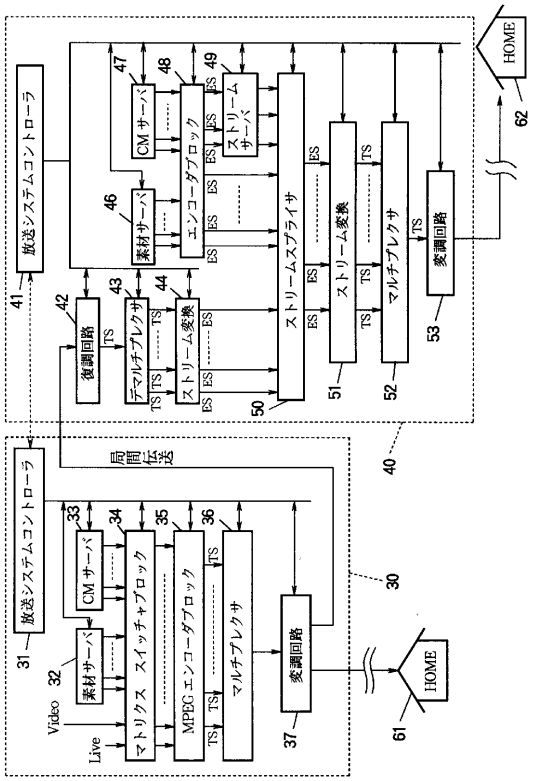
【図５】


(A) PG
(B) PG′
(C) ST

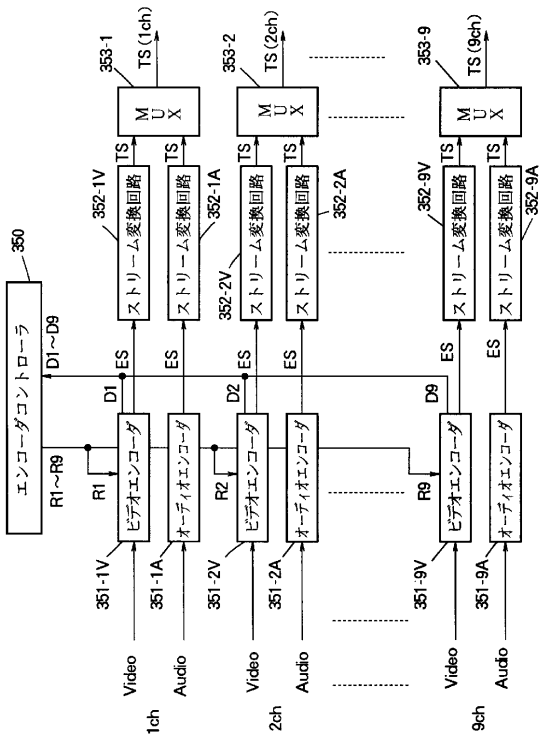
【図６】


(A) PGOLD STOLD
(B) PGNEW STNEW
(C) STSPL PGSPL

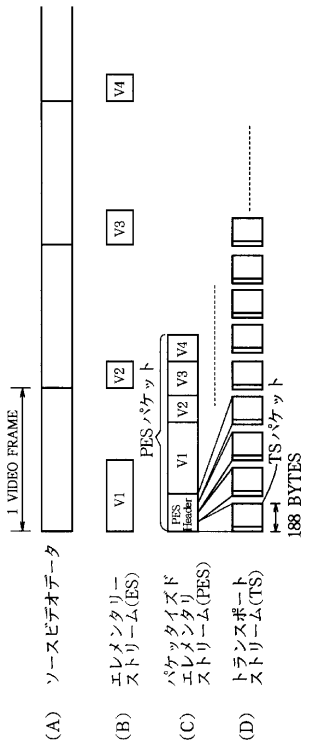
【図７】


放送システムコントローラ
局間伝送
マトリクス スイッチャブロック
MPEG エンコーダブロック
マルチプレクサ
ストリームスプライサ
ストリーム変換
デマルチプレクサ
エンコーダブロック
HOME

【図８】


エンコーダコントローラ
ビデオエンコーダ
オーディオエンコーダ
ストリーム変換回路
MUX
Video
Audio
1ch
2ch
9ch

【図９】



【図１０】

video_sequence ( ) {	ビット数	ニーモニック
next_start_code ( )		
sequence_header ( )		
sequence_extension ( )		
do {		
extension_and_user_data (0)		
do {		
if (nextbits ( )==group_start_code) {		
group_of_pictures_header ( )		
extension_and_user_data ( )		
}		
picture_header ( )		
picture_coding_extension ( )		
extension_and_user_data (2)		
picture_data ( )		
} while ((nextbits ( )==picture_start_code)‖		
(nextbits ( )==group_start_code))		
if (nextbits ( ) ! = sequence_end_code) {		
sequence_header ( )		
sequence_extension ( )		
}		
} while (nextbits ( ) ! = sequence_end_code)		
sequence_end_code	32	bslbf
)		

【図１１】

シーケンスヘッダ

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 ( )		
}		

【図１２】

シーケンス拡張

sequence_extension ( ) {	ビット	ニーモニック
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
profile_and_level_indentifier	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	bsalbf
vbv_buffer_size_extension	8	uimsbf
low_dalay	1	uimsbf
frame_rate_extension_n	2	uimsbf
frame_rate_extension_d	5	uimsbf
next_start_code ( )		
}		

【図１３】

拡張データおよびユーザデータ

extension_and_user_data (i) {	ビット数	ニーモニック
while ((((il=1) its ( )&&(nextbits ( ) extension_start_code)) ‖		
(nextbits ( )==user_data_start_code)) {		
if (nextbitS ( )==extension_start_code)		
extension_data (i)		
if (nextbits ( )=user_data_start_code)		
user_data ( )		
}		
}		

【図１４】

グループオブピクチャヘッダ

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

【図１５】

ピクチャヘッダ

picture header ( ) {	ビット数	ニーモニック
picture_start_code	32	bslbf
tenporal_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_backward_vector	1	
backward_f_code	3	uimsbf
}		
while (nextbits ( )==' 1') {		
extra_bit_picture／＊with the value " 1" ＊／	1	uimsbf
extra_information_picture	8	
}		
extra_bit_picture／＊with the value " 0" ＊／	1	uimsbf
next_start_code ( ) t		
}		

【図１６】

ピクチャコーティング拡張

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_vic_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 ( )		
}		

【図１７】

ピクチャデータ

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

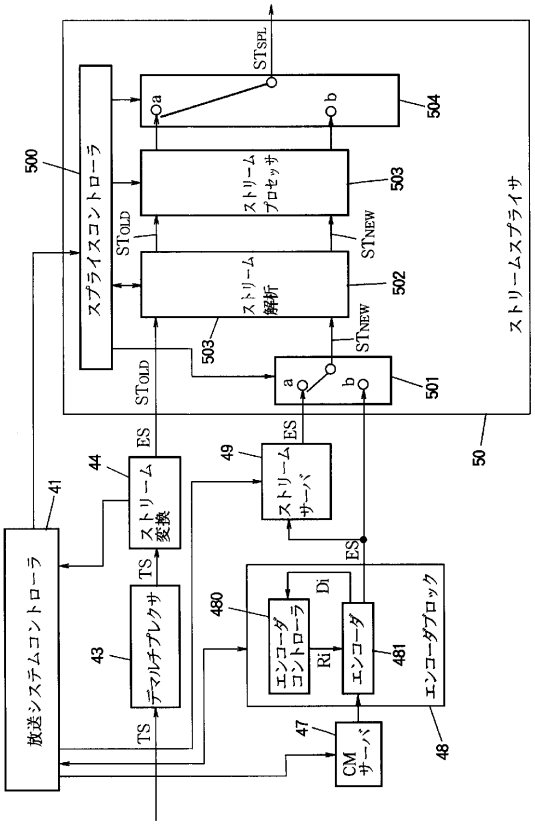
【図１８】

シンタックス	ビット数	ニーモニック
transport packet ( ) {		
sync_byte	8	bslbf
transport_error_indicator	1	bslbf
payload_unit_start_indicator	1	bslbf
transport_priority	1	bslbf
PID	13	uimsbf
transport_scrambling_control	2	bslbf
adaptaion_field_control	2	bslbf
continuity_counter	4	uimsbf
if (adaptation_field_control==' ‖ adaptation_field_control==' 11') {		
adaptation field ( )		
}		
if (adaptation_field_control==' ‖ adaptation_field_control==' 11') {		
for (i=0 ; 1<N ; i+t) {		
data_byte	8	bslbf
}		
}		
}		

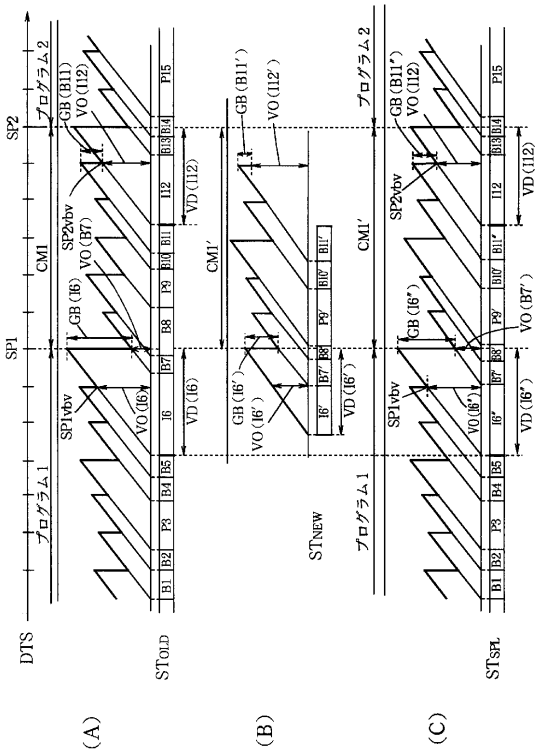
【図１９】

シンタックス	ビット数	ニーモニック
adaptation_field ( ) {		
adaptation_field_length	8	unimbf
if (adaptation_field_length>0) {		
discontinuity_indicator	1	bslbf
random access_indicator	1	bslbf
elementary_stream_priority_indicator	1	bslbf
PCR_flag	1	bslbf
OPCR_flag	1	bslbf
splicing_point_flag	1	bslbf
transport_private_data_flag	1	bslbf
adaptation_field_extension_flag	1	bslbf
if (PCR_flag==' 1') {		
program_clock_refernce_base	33	uimsbf
reserved	6	bslbf
progrqam_clock_reference_extension	9	uimsbf
}		
if (OPCR)_flag==' 1') {		
original_program_clock_reference_base	33	uimsbf
reserved	6	bslbf
original_program_clock_reference_extension	9	uimsbf
}		
if (splicing_point_flag==' 1') {		
splice_countdown	8	tcimsbf
}		
if (transport_private_data_flag==' 1') {		
transport_private_data_length	8	uimsbf
for (i=0 ; i< transport_private_data_length ; i+t) {		
provate_data_byte	8	bslbf
}		
}		
if (adaptation_field_extension_flag==' 1') {		
adaptation_field_extension_length	8	unimsbf
ltw_flag	1	bslbf
piecewise_rate_flag	1	bslbf
seamless_splice_flag	1	bslbf
reserved	5	bslbf
if (ltw_flag==' 1') {		
ltw_valid_flag	1	bslbf
ltw_offset	15	unimsbf
}		
if (plecewise_rate_flag-=' 1') {		
reserved	2	bslbf
piecewise_rate	22	uimsbf
}		
if (seamless_splice_flag==' 1') {		
splice_type	4	bslbf
DTS_next_AU [32..30]	3	bslbf
marker_bit	1	bslbf
DTS_next_AU [29..15]	15	bslbf
marker_bit	1	bslbf
DTS_next_AU [14..0]	15	bslbf
maker_bit	1	bslbf
}		
for (i=0 ; i<N ; i++) {		
reserved	8	bslbf
}		
for (i=0 ; i<N ; i++) {		
stuffing__byte	8	bslbf
}		
}		
}		

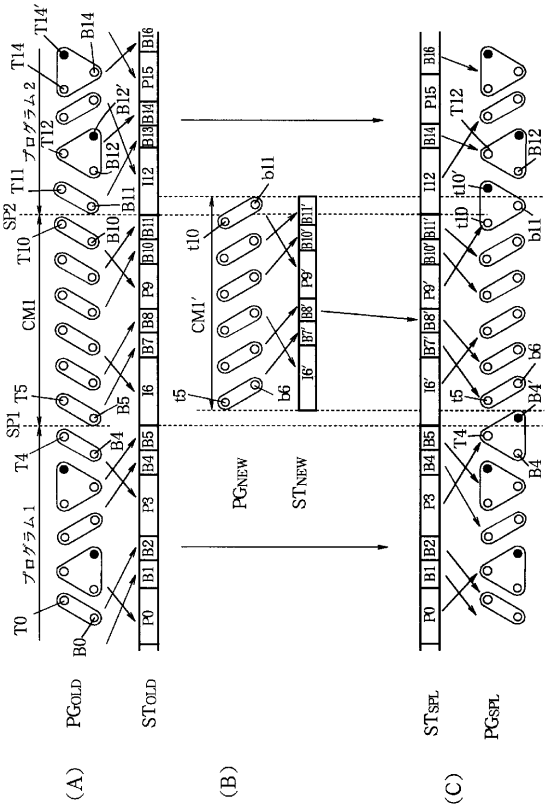
【図２０】



【図２１】



【図２２】



【図２３】

フロントページの続き

審査官　▲徳▼田　賢二

(56)参考文献　国際公開第９９／０１８７２０（WO，Ａ１）
特開平０９－２８４７６２（ＪＰ，Ａ）
特開平１１－００４４４５（ＪＰ，Ａ）
特開平０８－０８４３３３（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.[7]，ＤＢ名)
H04N 5/268 - 5/28
H04H 1/00
H04J 3/00
H04N 7/24