

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第3630107号

(P3630107)

(45) 発行日 平成17年3月16日(2005.3.16)

(24) 登録日 平成16年12月24日(2004.12.24)

(51) Int.Cl.<sup>7</sup>

G 1 0 H 7/02

G 1 0 H 1/00

F I

G 1 0 H 7/00 5 2 1 K

G 1 0 H 1/00 1 0 2 Z

請求項の数 7 (全 40 頁)

(21) 出願番号	特願2001-86166 (P2001-86166)	(73) 特許権者	000004075
(22) 出願日	平成13年3月23日(2001.3.23)		ヤマハ株式会社
(65) 公開番号	特開2002-287757 (P2002-287757A)		静岡県浜松市中沢町 1 0 番 1 号
(43) 公開日	平成14年10月4日(2002.10.4)	(74) 代理人	100104798
審査請求日	平成13年10月16日(2001.10.16)		弁理士 山下 智典
		(72) 発明者	田邑 元一
			静岡県浜松市中沢町 1 0 番 1 号 ヤマハ株式会社内
		審査官	板橋 通孝

最終頁に続く

(54) 【発明の名称】 音データ転送方法、音データ転送装置およびプログラム

(57) 【特許請求の範囲】

【請求項 1】

楽音波形の音データを記憶する低速記憶装置と、該音データをキャッシングする高速記憶装置とを用いて、前記低速記憶装置に記憶された音データのうちの一部を前記高速記憶装置に転送する音データ転送方法であって、

前記低速記憶装置に記憶されている音データは発音開始部に対応する第 1 音データと発音開始部以外の部分に対応する第 2 音データとを含み、

何れかの第 1 音データの指定を受信する第 1 音データ指定受信過程と、

該第 1 音データの指定に基づいて、後に指定される第 2 音データの候補を予測する第 2 音データ予測過程と、

該予測した第 2 音データを前記低速記憶装置から前記高速記憶装置に転送する過程とを有することを特徴とする音データ転送方法。

【請求項 2】

前記低速記憶装置に記憶されている音データは、前記第 1 および第 2 音データ以降に用いられる第 3 音データをさらに含み、

何れかの第 2 音データの指定を受信する第 2 音データ指定受信過程と、

該第 2 音データの指定に基づいて、後に指定される第 3 音データの候補を予測する第 3 音データ予測過程と、

該予測した第 3 音データを前記低速記憶装置から前記高速記憶装置に転送する過程とをさらに有することを特徴とする請求項 1 記載の音データ転送方法。

10

20

## 【請求項 3】

前記何れかの音データの指定を受信すると、前記高速記憶装置に記憶され、かつ、予測が外れた音データの候補に対して、解放が可能である旨を示す識別情報を付与する過程をさらに有することを特徴とする請求項 1 乃至 2 の何れかに記載の音データ転送方法。

## 【請求項 4】

前記高速記憶装置に転送されるべき音データに係る音色指定を受信する音色指定受信過程と、

該受信した音色指定に基づいて、前記第 1 音データの候補を予測する音データ予測過程と、

該予測した第 1 音データの候補を前記低速記憶装置から前記高速記憶装置に転送する過程と

をさらに有することを特徴とする請求項 1 乃至 3 の何れかに記載の音データ転送方法。

10

## 【請求項 5】

前記音色指定受信過程の前に、各音色毎に基本的音データを予め前記高速記憶装置に転送する過程をさらに有し、

前記第 1、第 2 または第 3 音データ指定受信過程において指定された音データが前記高速記憶装置に転送されていなかった場合には、該指定された音データに代えて、前記基本的音データを用いる

ことを特徴とする請求項 4 に記載の音データ転送方法。

## 【請求項 6】

請求項 1 ないし 5 の何れかに記載の方法を実行することを特徴とする音データ転送装置。

20

## 【請求項 7】

請求項 1 ないし 5 の何れかに記載の方法を実行することを特徴とするプログラム。

## 【発明の詳細な説明】

## 【0001】

## 【発明の属する技術分野】

この発明は、ハードディスク等、比較的低速な記憶媒体からの波形データの読み出し等に基づき、楽音あるいは音声若しくはその他任意の音の波形を生成する音データ転送方法、音データ転送装置およびプログラムに関し、特に、演奏者により行われた自然楽器固有の各種奏法若しくはアーティキュレーションによる音色変化を忠実に表現した波形を生成するものに関する。この発明は、電子楽器は勿論のこと、自動演奏装置、コンピュータ、電子ゲーム装置その他のマルチメディア機器等、楽音あるいは音声若しくはその他任意の音を発生する機能を有するあらゆる分野の機器若しくは装置または方法において広範囲に応用できるものである。なお、この明細書において、楽音波形という場合、音楽的な音の波形に限るものではなく、音声あるいはその他任意の音の波形を含んでいてもよい意味合いで用いるものとする。

30

## 【0002】

## 【従来の技術】

波形メモリにおいて、PCM（パルス符号変調）あるいはDPCM（差分PCM）又はADPCM（適応差分PCM）等の任意の符号化方式で符号化した波形データ（つまり波形サンプルデータ）を記憶しておき、これを所望の音楽ピッチに対応して読み出すことにより、楽音波形を形成するようにした、いわゆる「波形メモリ読み出し」技術は既に公知であり、また、様々なタイプの「波形メモリ読み出し方式」技術が知られている。従来知られた「波形メモリ読み出し方式」技術のほとんどは、発音開始から終了までの1つの音の波形を発生するためのものである。一例として、発音開始から終了までの1音の全波形の波形データを記憶する方式がある。また、別の例として、変化の複雑なアタック部などについてはその全波形の波形データを記憶し、変化のあまりないサステイン部などについては所定のループ波形を記憶する方式がある。なお、本明細書において、「ループ波形」とは繰り返し読み出し（ループ読み出し）される波形という意味で用いるものとする。

40

## 【0003】

50

波形データを記憶するための手段としては、ROM、RAM、ハードディスク、CD-ROM等が知られている。ハードディスク、CD-ROM等は単位記憶容量あたりの単価が安価であり大容量のデータ記憶に適している。しかし、ハードディスク等はアクセス時間が遅く、かつ一定していないため、楽音信号を出力するタイミングで必要な波形データを即座に読み出すことは不可能である。このため、以下のように様々な技術が提案されている。

まず、特開平6-308964号公報においては、ハードディスク等に記憶した複数の波形データの先頭部分を予めRAMに転送しておく技術が開示されている。すなわち、発音指示が供給されると、ハードディスク等の当該波形データの後続部分の読み出しが開始されるとともに予めRAMに記憶された先頭部分が再生される。そして、先頭部分の再生が終了した後に、後続部分の波形データが再生されるのである。

#### 【0004】

また、特開昭63-181188号公報においては、順次発音すべき波形データがシーケンスデータとして予め定義されている場合に、各波形データを順次読み出しながら再生する技術が開示されている。この技術においては、各波形データの読出しを開始する時刻が発音タイミングよりも早くなるように、シーケンスタイミングに対応して予め定められる。

#### 【0005】

##### 【発明が解決しようとする課題】

しかし、特開平6-308964号公報に係る技術によれば、予め全ての波形データの先頭部分を予めRAMに転送しておく必要があるため、RAMの使用効率が低くなるという問題があった。また、特開昭63-181188号公報に係る技術は、ノンリアルタイムで楽音信号を生成することを前提としており、シーケンスデータがリアルタイムで供給される場合において適用することは不可能である。

本発明は上述した事情に鑑みてなされたものであり、RAMの使用効率が高く、しかもハードディスク等を用いながらもリアルタイムに楽音信号を生成することができる音データ転送方法、音データ転送装置およびプログラムを提供することを目的としている。

#### 【0006】

##### 【課題を解決するための手段】

上記課題を解決するため本発明にあっては、下記構成を具備することを特徴とする。なお、括弧内は例示である。

請求項1記載の構成にあっては、楽音波形の音データ（ベクトルデータ）を記憶する低速記憶装置（ハードディスク109）と、該音データをキャッシングする高速記憶装置（キャッシュメモリ44）とを用いて、前記低速記憶装置に記憶された音データのうちの一部を前記高速記憶装置に転送する音データ転送方法であって、前記低速記憶装置に記憶されている音データは発音開始部（アタック部）に対応する第1音データと発音開始部以外の部分に対応する第2音データ（ボディ部のベクトルデータ）とを含み、何れかの第1音データの指定を受信する第1音データ指定受信過程と、該第1音データの指定に基づいて、後に指定される第2音データの候補を予測する第2音データ予測過程（ステップS30）と、該予測した第2音データを前記低速記憶装置から前記高速記憶装置に転送する過程（ステップS43）とを有することを特徴とする。

さらに、請求項2記載の構成にあっては、請求項1記載の音データ転送方法において、前記低速記憶装置に記憶されている音データは、前記第1および第2音データ以降に用いられる第3音データ（リリース部またはジョイント部）をさらに含み、何れかの第2音データの指定を受信する第2音データ指定受信過程と、該第2音データの指定に基づいて、後に指定される第3音データの候補を予測する第3音データ予測過程（ステップS33）と、該予測した第3音データを前記低速記憶装置から前記高速記憶装置に転送する過程（ステップS43）とをさらに有することを特徴とする。

さらに、請求項3記載の構成にあっては、請求項1乃至2の何れかに記載の音データ転送方法において、前記何れかの音データの指定を受信すると、前記高速記憶装置に記憶され

10

20

30

40

50

、かつ、予測が外れた音データの候補に対して、解放が可能である旨を示す識別情報を付与する（メンバdwStatusを‘USED’に設定する）過程をさらに有することを特徴とする。さらに、請求項4記載の構成にあっては、請求項1乃至3の何れかに記載の音データ転送方法において、前記高速記憶装置に転送されるべき音データに係る音色指定（プログラムチェンジ）を受信する音色指定受信過程と、該受信した音色指定に基づいて、前記第1音データの候補を予測する音データ予測過程（ステップS30）と、該予測した第1音データの候補を前記低速記憶装置から前記高速記憶装置に転送する過程（ステップS43）とをさらに有することを特徴とする。

さらに、請求項5記載の構成にあっては、請求項4に記載の音データ転送方法において、前記音色指定受信過程の前に、各音色毎に基本的音データ（代替ページ）を予め前記高速記憶装置に転送する過程をさらに有し、前記第1，第2または第3音データ指定受信過程において指定された音データが前記高速記憶装置に転送されていなかった場合には、該指定された音データに代えて、前記基本的音データを用いることを特徴とする。

また、請求項6記載の構成にあっては、請求項1ないし5の何れかに記載の方法を実行することを特徴とする。

また、請求項7記載の構成にあっては、請求項1ないし5の何れかに記載の方法を実行することを特徴とする。

#### 【0007】

#### 【発明の実施の形態】

##### 1．実施形態のハードウェア構成

以下、この発明の一実施形態を添付図面に従って詳細に説明する。

図1は、この発明に係る波形生成装置のハードウェア構成例を示すブロック図である。ここに示されたハードウェア構成例はコンピュータを用いて構成されており、そこにおいて、波形生成処理は、コンピュータがこの発明に係る波形生成処理を実現する所定のプログラム（ソフトウェア）を実行することにより実施される。勿論、この波形生成処理はコンピュータソフトウェアの形態に限らず、DSP（ディジタル・シグナル・プロセッサ）によって処理されるマイクロプログラムの形態でも実施可能であり、また、この種のプログラムの形態に限らず、ディスクリート回路又は集積回路若しくは大規模集積回路等を含んで構成された専用ハードウェア装置の形態で実施してもよい。また、この波形生成装置は、電子楽器あるいはカラオケ装置又は電子ゲーム装置又はその他のマルチメディア機器又はパーソナルコンピュータ等、任意の製品応用形態をとってよい。

#### 【0008】

図1に示されたハードウェア構成例においては、コンピュータのメイン制御部としてのCPU101に対して、バスラインBL（データあるいはアドレスバス等）を介してリードオンリメモリ（ROM）102、ランダムアクセスメモリ（RAM）103、パネルスイッチ104、パネル表示器105、ドライブ106、波形取込部107、波形出力部108、ハードディスク109、通信インタフェース111がそれぞれ接続されている。CPU101は、後述する「波形データベース作成」や「制作したデータベースに基づく楽音合成（ソフトウェア音源）」等の処理を、所定のプログラムに基づいて実行する。これらのプログラムは、通信インタフェース111を介したネットワークあるいはドライブ106に装着されたCDやMO等の外部記憶メディア106A等から供給されてハードディスク109に記憶される。そして、実行時にハードディスク109からRAM103にロードされる。あるいは、ROM102にプログラムが記録されていてもよい。ROM102は、CPU101により実行あるいは参照される各種プログラムや各種データ等を格納するものである。RAM103は、演奏に関する各種情報やCPU101がプログラムを実行する際に発生する各種データを一時的に記憶するワーキングメモリとして、あるいは現在実行中のプログラムやそれに関連するデータを記憶するメモリとして使用される。RAM103の所定のアドレス領域がそれぞれの機能に割り当てられ、レジスタやフラグ、テーブル、メモリなどとして利用される。パネルスイッチ104は、楽音をサンプリングする指示やサンプリングされた波形データ等のエディットや各種情報の入力等を行うための

各種の操作子を含んで構成される。例えば、数値データ入力用のテンキーや文字データ入力用のキーボード、あるいはパネルスイッチ等である。この他にも音高、音色、効果等を選択・設定・制御するための各種操作子を含んでいてよい。パネル表示器 105 は、パネルスイッチ 104 により入力された各種情報やサンプリングされた波形データ等を表示する、例えば液晶表示パネル (LCD) や CRT 等のディスプレイである。

#### 【0009】

波形取込部 107 は A/D 変換器を内蔵し、外部波形入力 (例えば、マイクロフォンなどからの入力) されたアナログ楽音信号をデジタルデータに変換 (サンプリング) して RAM 103 あるいはハードディスク 109 に該デジタル波形データをオリジナル波形データ (生成すべき波形データの素材となる波形データ) として取り込むものである。CPU 101 によって実行する「波形データベース作成」処理では、上記取り込んだオリジナル波形データを基にして本発明に従う「波形データベース」の作成を行う。また、CPU 101 によって実行する「データベースに基づく楽音合成」処理では、上記「波形データベース」を使用して演奏情報に応じた任意の楽音信号の波形データを生成する。勿論、複数の楽音信号の同時発生が可能である。生成された楽音信号の波形データはバスライン BL を介して波形出力部 108 に与えられ、適宜バッファ記憶される。波形出力部 108 ではバッファ記憶された波形データを所定の出力サンプリング周波数にしたがって出力し、これを D/A 変換してサウンドシステム 108A に送出する。こうして、波形出力部 108 から出力された楽音信号は、サウンドシステム 108A を介して発音される。ハードディスク 109 は、波形データや奏法に応じた波形を合成するためのデータ (後述する奏法テーブル、コードブック等のデータ)、各種音色パラメータ等からなる音色データなどのような演奏に関する複数種類のデータを記憶したり、前記 CPU 101 が実行する各種プログラム等の制御に関するデータを記憶したりするものである。

#### 【0010】

ドライブ 106 は、波形データや奏法に応じた波形を合成するためのデータ (後述する奏法テーブル、コードブック等の各種データ)、多種多様な音色パラメータ等からなる音色データなどのような演奏に関する複数種類のデータを記憶したり、前記 CPU 101 が実行する各種プログラム等の制御に関するデータを記憶したりするための着脱可能なディスク (外部記憶メディア 106A) をドライブするものである。なお、前記ドライブ 106 によりドライブされる外部記憶メディア 106A はフロッピーディスク (FD) の他に、コンパクトディスク (CD-ROM・CD-RAM)、光磁気ディスク (MO)、あるいは DVD (Digital Versatile Disk の略) 等の着脱自在な様々な形態の記憶メディアであってよい。制御プログラムを記憶した外部記憶メディア 106A をドライブ 106 にセットし、その内容 (制御プログラム) をハードディスク 109 に落とさずに、RAM 103 に直接ロードしてもよい。なお、外部記憶メディア 106A を用いて、あるいはネットワークを介して制御プログラムを提供するやり方は、制御プログラムの追加やバージョンアップ等を容易に行うことができるので好都合である。

#### 【0011】

通信インタフェース 111 は、例えば LAN やインターネット、電話回線等の通信ネットワーク (図示せず) に接続されており、該通信ネットワークを介して、サーバコンピュータ等 (図示せず) と接続され、当該サーバコンピュータ等から制御プログラムや各種データあるいは演奏情報などを波形生成装置側に取り込むためのものである。すなわち、ROM 102 やハードディスク 109 に制御プログラムや各種データが記憶されていない場合に、サーバコンピュータから制御プログラムや各種データをダウンロードするために用いられる。クライアントとなる波形生成装置は、通信インタフェース 111 を介してサーバコンピュータへと制御プログラムや各種データのダウンロードを要求するコマンドを送信する。サーバコンピュータは、このコマンドを受け、要求された制御プログラムやデータなどを通信インタフェース 111 を介してハードディスク 109 に蓄積することにより、ダウンロードが完了する。更に、MIDI インタフェースを含み、MIDI の演奏情報を受け取るようにしてもよいのは勿論である。また、音楽演奏用キーボードや演奏操作機器

10

20

30

40

50

をバスライン B L に接続し、リアルタイム演奏によって演奏情報を供給するようにしてもよいのは言うまでもない。勿論、所望の音楽曲の演奏情報を記憶した外部記憶メディア 106 A を使用して、演奏情報を供給するようにしてもよい。

#### 【0012】

### 2. 波形データベース作成処理

#### 2.1. 処理の概要

図2は、上述した波形生成装置において実行される「波形データベース作成処理」の一実施形態を示すフローチャートである。当該処理は、いろいろな奏法（若しくはアーティキュレーション）に対応するために、いろいろな奏法（若しくはアーティキュレーション）で演奏された演奏音の波形を素材としてベクトルデータを作成するための処理である。

ステップ S1 では、後述する奏法テーブル及びコードブックを記憶するためのデータベースを準備する。このデータベースとなる媒体としては、例えばハードディスク 109 を使用する。そして、様々な自然楽器の様々な演奏態様による波形データを収集する（ステップ S2）。すなわち、様々な自然楽器の様々な実際の演奏音を外部波形入力（例えば、マイクロフォン等）から波形取込部 107 を介して取り込み、それらの演奏音の波形データ（オリジナル波形データ）をハードディスク 109 の所定のエリアに記憶する。この際に取り込む演奏音の波形データは演奏全体の波形データであってもよいし、あるフレーズ、あるいは1音、あるいはアタック部やリリース部といった特徴のある演奏の一部の波形データだけでもよい。

#### 【0013】

次に、こうして得られた自然楽器固有の様々な演奏態様による演奏音の波形データを特徴的な部分毎に切り分けて、チューニング及びファイル名付けする（ステップ S3）。すなわち、取り込んだオリジナル波形データを波形形状の変化を代表する一部の波形（例えば、アタック部波形、ボディ部波形、リリース部波形、ジョイント部波形等）毎に分離して（1 切り分け）、分離した1周期乃至複数周期の波形データがそれぞれいかなるピッチであるかを判定し（2 チューニング）、さらにそれぞれ分離した波形データに対してファイル名を付与する（3 ファイル名付け）。ただし、アタック部分やリリース部分といった演奏の一部の波形データを取り込んでいる場合には、このような波形の分離（1 切り分け）を省略できる。

次に、周波数分析による成分分離を行う（ステップ S4）。すなわち、ステップ S3 で分離生成された一部の波形データを F F T（高速フーリエ変換）分析して複数成分に分離し（この実施形態では、調和成分と調和外成分に分離する）、さらに各成分（調和成分、調和外成分等）から波形、ピッチ、振幅の各要素毎の特徴抽出、つまり特徴分離を行う（ただし、調和成分と調和外成分に分離する場合、調和外成分はピッチを持たないものであることから調和外成分についてのピッチ分離は行わなくてよい）。例えば「波形」（T i m b r e）要素は、ピッチと振幅をノーマライズした波形形状のみ特徴を抽出したものである。「ピッチ」（P i t c h）要素は、基準ピッチに対するピッチ変動特性を抽出したものである。「振幅」（A m p l i t u d e）要素は、振幅エンベロープ特性を抽出したものである。

#### 【0014】

ステップ S5 では、ベクトルデータの作成が行われる。すなわち、分離された各成分（調和成分、調和外成分等）の波形（T i m b r e）やピッチ（P i t c h）や振幅（A m p l i t u d e）の各要素毎に複数のサンプル値を分散的に又は必要に応じて連続的に抽出し、当該サンプル値列に対して各々異なったベクトル I D（識別情報）を付与して、サンプル値の時刻位置のデータとともにコードブックに記憶する（以下、このようなサンプルデータをベクトルデータと呼ぶ）。この実施形態では、調和成分の波形（T i m b r e）要素のベクトルデータ、ピッチ（P i t c h）要素のベクトルデータ、振幅（A m p l i t u d e）要素のベクトルデータと、調和外成分の波形（T i m b r e）要素のベクトルデータ、振幅（A m p l i t u d e）要素のベクトルデータとがそれぞれ作成される。このように、これらの各成分要素毎のベクトルデータは、時間軸の進行に伴い変化するデ

10

20

30

40

50

ータである。次に、奏法モジュールのデータ（詳しい内容については後述する）を作成して奏法モジュールを奏法テーブルに記憶する。こうして作成された奏法モジュール及びベクトルデータは、データベースにおける奏法テーブル及びコードブックへ書き込まれ（ステップS6）、データベースへのデータ蓄積がはかれる。上述したように、ベクトルデータは取り込んだオリジナル波形データそのままでなく取り込んだオリジナル波形の形状を代表する波形を各要素毎に分離したデータであって、このベクトルデータは各々が最終的には奏法モジュールの単位となるデータである。このように、コードブックには抽出した波形形状の変化を代表する一部の波形データを圧縮した形で記憶する。一方、奏法テーブルには、奏法モジュールのデータ（つまり、圧縮された形で記憶されたベクトルデータを元の波形形状の波形データに戻すために必要な各種のデータや、コードブックに記憶されたベクトルデータを指定するためのIDデータ）などが記憶される（詳しくは後述する）。

10

#### 【0015】

上述した特徴分離（ステップS4参照）の際に、振幅、ピッチ、波形要素の他に時間を要素として特徴抽出を行う（以下、抽出された時間要素のベクトルデータのことを「タイムベクトルデータ」と呼ぶ）。この時間要素については、分離生成された一部の波形データの時間区間におけるオリジナル波形データの時間長をそのまま用いる。従って、当該時間区間のオリジナルの時間長（可変値である）を比「1」で示すこととすれば、当該「波形データベース作成処理」時においてこの時間長をあえて分析・測定する必要はない。その場合、時間要素についてのデータ（すなわち、「タイムベクトルデータ」）はどの時間区間でも同じ値“1”であるから、これをコードブックにあえて記憶しておかなくてもよい。勿論、これに限らず、この実際の時間長を分析・測定し、これを「タイムベクトルデータ」としてコードブックに記憶するようにする変形例も実施可能である。

20

#### 【0016】

そして、データベース作成が充分に行われたか否かを判定する（ステップS7）。すなわち、外部波形入力から得られた様々な自然楽器の様々な演奏態様による演奏音のオリジナル波形データの収集を充分に行って、様々な奏法モジュールのデータ及びベクトルデータを充分に得たか否かを判定する。この判定は自動判定に限らず、ユーザによるスイッチ入力操作に基づく処理続行可否指示に従って行うようにしてもよい。オリジナル波形データの収集とそれに基づくベクトルデータの作成が充分に行われたと判定されたならば（ステップS7のYES）、当該処理を終了する。引き続き、オリジナル波形データの収集とそれに基づくベクトルデータの作成を行う場合（ステップS7のNO）、ステップS2の処理へ戻り、上述した各処理（ステップS2～ステップS7）を繰り返し実行する。上記「ベクトルデータの作成が充分に行われたか否か」の判定（ステップS7）は、「作成したベクトルデータを実際に使用して楽音を生成してみる」ことにより行われてもよい。すなわち、ステップS7で一旦「ベクトルデータを充分作成した」（ステップS7のYES）と判断して図2に示すフローを抜けた後に、「その作成したベクトルデータを使用して楽音を再生してみたら満足できなかったので、再びステップS2以降の処理を行ってベクトルデータを追加する」というような処理を行ってもよい。つまり、「ベクトルデータを作成してデータベースに追加する」という処理は、必要に応じて随時行われる。

30

40

なお、上述の「波形データベース作成処理」において、奏法モジュールを任意に追加・削除したり、あるいは奏法モジュールのデータ等の編集を行うことができるようにしてもよい。

#### 【0017】

### 2.2. 奏法モジュールのデータ構成

ここで、奏法モジュールのデータについて具体的に説明する。

奏法モジュールはハードディスク109上にデータベース化されて構成される奏法テーブルに記憶され、1つの奏法モジュールは「奏法ID」と「奏法パラメータ」の組み合わせによって指定することができるようになっている。「奏法ID」は、その中に楽器情報及びモジュールパーツ名を含む。例えば、「奏法ID」は次のように定義される。例えば、

50

1つの「奏法ID」が32ビット(第0～31ビット)列で表現されているとすると、そのうちの6ビットを使用して楽器情報を表現する。例えば、当該6ビット列が「000000」であればAltoSax(アルト・サクス)を示し、「001000」であればViolin(バイオリン)を示す楽器情報である。この楽器情報は前記6ビット列のうち上位3ビット列を楽器種類の大分類に使用し、下位3ビット列を楽器種類の小分類のために使用するなどしてよい。また、32ビット列の別の6ビットを使用してモジュールパーツ名を表現する。例えば、当該6ビット列が「000000」であればNormalAttack、「000001」であればBendAttack、「000010」であればGraceNoteAttack、「001000」であればNormalShortBody、「001001」であればVibBody、「001010」であればNormalLongBody、「010000」であればNormalRelease、「011000」であればNormalJoint、「011001」であればGraceNoteJointを示すモジュールパーツ名である。勿論、上述した構成に限られないことは言うまでもない。

#### 【0018】

上述したように、個々の奏法モジュールは、上記「奏法ID」と「奏法パラメータ」との組み合わせで特定される。すなわち、「奏法ID」に応じて所定の奏法モジュールが特定され、その内容が「奏法パラメータ」に応じて可変設定される。この「奏法パラメータ」は該奏法モジュールに対応する波形データを特徴付ける、若しくは制御するパラメータであり、各奏法モジュール毎に所定の種類の「奏法パラメータ」が存在している。例えば、AltoSax [NormalAttack]モジュールの場合にはAttack直後の絶対音高やAttack直後の音量などの種類の奏法パラメータが与えられてよいし、AltoSax [BendUpAttack]モジュールの場合にはBendUpAttack終了時の絶対音高、BendUpAttack時のBend深さの初期値、BendUpAttack開始(ノートオンタイミング)～終了までの時間、Attack直後の音量、あるいはBendUpAttack中のデフォルトのカーブの時間的な伸縮などの種類の奏法パラメータが与えられてよい。また、AltoSax [NormalShortBody]モジュールの場合には当該モジュールの絶対音高、NormalShortBodyの終了時刻-開始時刻、NormalShortBody開始時のダイナミクス、NormalShortBody終了時のダイナミクスなどの種類の奏法パラメータが与えられてよい。なお、奏法モジュールには、「奏法パラメータ」の採りうる全ての値に対応するデータ(後述する要素データ)を必ずしも有しない。「奏法パラメータ」の飛び飛びの一部の値だけに応じたデータを記憶している場合もある。すなわち、例えばAltoSax [NormalAttack]モジュールの場合、Attack直後の絶対音高やAttack直後の音量の全ての値ではなく、一部のデータだけに対応したデータを記憶していてもよい。

このように、奏法モジュールを「奏法ID」と「奏法パラメータ」で指定できるようにすることで、例えばAltoSax [NormalAttack]であればアルトサクスのノーマルアタック部を示す複数データ(後述する要素データ)の中から所望の奏法パラメータに応じたデータを指定することができるし、Violin [BendAttack]であればバイオリンのベンドアタック部を示す複数データ(後述する要素データ)の中から所望の奏法パラメータに応じたデータを指定することができる。

#### 【0019】

##### 2.3. 奏法テーブルのデータ構成

奏法テーブルにおいては、個々の奏法モジュールにつき、当該奏法モジュールに対応する波形を生成するために必要なデータ、例えば各成分要素毎のベクトルデータ(波形要素、ピッチ要素(ピッチエンベロープ)、振幅要素(振幅エンベロープ)等)を指定するためのベクトルIDや代表点値列(複数サンプル列の中の補正のための代表的サンプル点を指示するデータ)あるいは各成分要素毎のベクトルデータ(波形要素、ピッチ要素(ピッチエンベロープ)、振幅要素(振幅エンベロープ))の開始時間位置や終了時間位置などの

10

20

30

40

50

情報等を記憶している。つまり、ベクトルデータという圧縮された形でデータベースに記憶されている波形から通常形状の波形を再生するために必要な各種のデータを記憶している（以下、このようなデータを「要素データ」とも呼ぶ）。奏法テーブルにおいて、1つの奏法モジュールに対応して記憶する具体的なデータの一例をAlto Sax [Normal Attack]モジュールの場合について説明すると、次の通りである。

データ1：奏法モジュールのサンプル長。

データ2：ノートオンタイミングの位置。

データ3：調和成分の振幅（Amplitude）要素のベクトルIDと代表点値列。

データ4：調和成分のピッチ（Pitch）要素のベクトルIDと代表点値列。

データ5：調和成分の波形（Timbre）要素のベクトルID。

10

データ6：調和外成分の振幅（Amplitude）要素のベクトルIDと代表点値列。

データ7：調和外成分の波形（Timbre）要素のベクトルID。

データ8：調和成分の波形（Timbre）要素の塊部の開始位置。

データ9：調和成分の波形（Timbre）要素の塊部の終了位置（調和成分の波形（Timbre）要素のループ部の開始位置）。

データ10：調和外成分の波形（Timbre）要素の塊部の開始位置。

データ11：調和外成分の波形（Timbre）要素の塊部の終了位置（調和外成分の波形（Timbre）要素のループ部の開始位置）。

データ12：調和外成分の波形（Timbre）要素のループ部の終了位置。

【0020】

20

上記データ1～12について、図3を参照して説明する。

図3は、当該奏法モジュールに対応する実波形区間を構成する各成分及び要素の一例を模式的に示す図であり、上から当該区間における調和成分の振幅（Amplitude）要素、調和成分のピッチ（Pitch）要素、調和成分の波形（Timbre）要素、調和外成分の振幅（Amplitude）要素、調和外成分の波形（Timbre）要素の一例を示す。なお、図に示している数字は上記各データの番号に対応するように付してある。

1は、当該奏法モジュールに該当する波形のサンプル長（波形区間長）である。例えば、当該奏法モジュールの基となったオリジナル波形データの全体の時間長さに対応している。2はノートオンタイミングの位置であり、当該奏法モジュールのどの時間位置にも可変に設定することが可能である。原則的には、このノートオンタイミングの位置から当該波形に従った演奏音の発音が開始されるが、ベンドアタックなどの奏法によってはノートオンタイミングよりも波形成分の立ち上がり開始時点が先行する場合がある。

30

【0021】

3は、コードブックに記憶された調和成分の振幅（Amplitude）要素のベクトルデータを指し示すためのベクトルID及び代表点値列を示す（図において、黒く塗りつぶした正方形で示す2点が代表点を示す）。4は、調和成分のピッチ（Pitch）要素のベクトルデータを指し示すためのベクトルID及び代表点値列を示す。6は、調和外成分の振幅（Amplitude）要素のベクトルデータを指し示すためのベクトルID及び代表点値列を示す。代表点値列データはベクトルIDによって指示されるベクトルデータ（複数サンプル列からなる）を変更制御するためのデータであり、代表的サンプル点のいくつかを指示（特定）するものである。特定された代表的サンプル点に関してその時間位置（横軸）とレベル軸（縦軸）を変更若しくは補正することにより、他の残りのサンプル点も連動して変更し、もってベクトルの形状を変更する。例えば、そのサンプル数より少ない数の分散的サンプルを示すデータであるが、勿論これに限らず、代表点値列データはサンプルとサンプルの間の中間位置のデータであってもよいし、あるいは所定の範囲（連続的な複数サンプル）にわたるデータであってもよい。また、サンプル値そのものでなく、差分や乗数等のデータであってもよい。この代表点を横軸及び/又は縦軸（時間軸）に移動することによって、各ベクトルデータの形状を変えることができる。つまり、エンベロープ波形の形状を変えることができる。

40

50

## 【 0 0 2 2 】

5 は、調和成分の波形 ( T i m b r e ) 要素のベクトルデータを指し示すためのベクトル ID である。7 は、調和外成分の波形 ( T i m b r e ) 要素のベクトルデータを指し示すためのベクトル ID である。8 は、調和成分の波形 ( T i m b r e ) 要素の波形の塊部の開始位置である。9 は、調和成分の波形 ( T i m b r e ) 要素の波形の塊部の終了位置 (あるいは、調和成分の波形 ( T i m b r e ) 要素の波形のループ部の開始位置) である。すなわち、8 から開始する三角形は特徴のある波形形状が連続的に記憶されているノンループ波形の部分を示し、その後続く 9 から開始する長方形は繰り返し読み出しすることのできるループ波形の部分を示す。ノンループ波形は、奏法 (若しくはアーティキュレーション) 等の特徴を有する高品質な波形である。ループ波形は、1 周期または適当な複数 10 周期分の波形からなる比較的単調な音部分の単位波形である。10 は、調和外成分の波形 ( T i m b r e ) 要素の波形の塊部の開始位置である。11 は、調和外成分の波形 ( T i m b r e ) 要素の波形の塊部の終了位置 (あるいは、調和外成分の波形 ( T i m b r e ) 要素の波形のループ部の開始位置) である。12 は、調和外成分の波形 ( T i m b r e ) 要素の波形のループ部の終了位置である。上記データ 3 ~ データ 7 は各成分要素毎にコードブックに記憶されているベクトルデータを指し示すための識別情報のデータであり、上記データ 2 及びデータ 8 ~ データ 12 はベクトルデータから元の (分離前の) 波形を組み立てるための時間情報のデータである。

## 【 0 0 2 3 】

このように、奏法モジュールのデータはベクトルデータを指し示すためのデータと時間情報 20 のデータとから構成される。このような奏法テーブルに記憶されている奏法モジュールのデータを使用することにより、コードブックに記憶されている波形の素材 (ベクトルデータ) を使って、波形を自由に組み立てることができることになる。つまり、奏法モジュールは、奏法 (若しくはアーティキュレーション) に応じて生成する波形の挙動を表すデータである。なお、奏法モジュールのデータの種類や数は各奏法モジュール毎に異なっていてよい。また、上述したデータ以外にも他の情報等を具えていてよい。例えば、波形の時間軸を伸長 / 圧縮制御するためのデータなどを持っていてもよい。

## 【 0 0 2 4 】

また、上述の例では説明を理解しやすくするために、1 つの奏法モジュールが調和成分の各要素 (波形、ピッチ、振幅) 及び調和外成分の各要素 (波形、振幅) の全てを具備して 30 いる例について説明したが、これに限らず、奏法モジュールが調和成分の各要素 (波形、ピッチ、振幅) や調和外成分の各要素 (波形、振幅) の 1 つからなっているのもよいのは勿論である。例えば、奏法モジュールが調和成分の波形 ( T i m b r e ) 要素、調和成分のピッチ ( P i t c h ) 要素、調和成分の振幅 ( A m p l i t u d e ) 要素、調和外成分の波形 ( T i m b r e ) 要素、調和外成分の振幅 ( A m p l i t u d e ) 要素のいずれか 1 つの要素からなっているのもよい。こうすると、各成分毎に奏法モジュールを自由に組み合わせて使用することができることになり好ましい。

## 【 0 0 2 5 】

このように、様々な自然楽器の様々な演奏態様による演奏音の波形データを全波形データ 40 で持つのではなく、波形形状の変化に必要な一部の波形 (例えば、アタック部波形、ボディ部波形、リリース部波形、ジョイント部波形等) のみを抽出し、さらに成分、要素、代表点といった階層的な圧縮手法を用いて、データ圧縮された形で波形データをハードディスク 109 に記憶することから、波形データを記憶するために必要なハードディスク 109 の記憶容量を削減することができるようになっている。

## 【 0 0 2 6 】

## 3. 波形合成処理

図 1 に示す波形生成装置において、波形の合成はコンピュータがこの発明に係る波形合成処理を実現する所定のプログラム (ソフトウェア) を実行することにより実施される。図 4 は、前記波形合成処理を実現する所定のプログラム (「データベースに基づく楽音合成処理」) のフローチャートの一実施形態を示したものである。

10

20

30

40

50

但し、本実施形態においては、プログラムによって実現される様々な機能は相互に独立しているため、図5に示す機能ブロック図に基づいて動作を説明する。また、この種のプログラムの形態に限らず、波形合成処理を専用ハードウェア装置の形態で実施するようにしてもよい。この場合、図5は、図4と同様の波形合成処理を専用ハードウェア装置の形態で構成した場合のブロック図になる。そこで、以降は、主に、図5に従って説明し、図4については対応するステップを括弧書きして示す。

#### 【0027】

##### 3.1. 曲データ再生部101A

曲データ再生部101Aは、奏法記号付き曲データの再生処理を行う(ステップS11)。最初に、曲データ再生部101Aは奏法記号付き曲データ(演奏情報)を受信する。通常の楽譜には、そのままではMIDIデータとならないような強弱記号(クレッシェンドやデクレッシェンド等)、テンポ記号(アレグロやリタルダンド等)、スラー記号、テヌート記号、アクセント記号等の音楽記号が付されている。そこで、これらの記号を「奏法記号」としてデータ化して、この「奏法記号」を含むMIDI曲データが「奏法記号付き曲データ」である。「奏法記号」は、チャートIDとチャートパラメータとから構成する。チャートIDは楽譜に記載される音楽記号を示すIDであり、チャートパラメータはチャートIDで示される音楽記号の内容の程度を示すパラメータである。例えば、チャートIDが“ピブラート”を示す場合にはピブラートの速さや深さ等がチャートパラメータとして付与され、チャートIDが“クレッシェンド”を示す場合にはクレッシェンドのスタート時の音量、クレッシェンドのエンド時の音量、音量変化する時間長等がチャートパラメータとして付与される。

#### 【0028】

##### 3.2. 楽譜解釈部101B

楽譜解釈部(プレーヤー)101Bでは、楽譜解釈処理を行う(ステップS12)。具体的には、曲データに含まれるMIDIデータと上述した「奏法記号」(チャートIDとチャートパラメータ)を奏法指定情報(奏法IDと奏法パラメータ)に変換し、時刻情報とともに奏法合成部(アーティキュレーター)101Cに出力する。一般的に、同じ音楽記号でも演奏家により記号の解釈が異なって、演奏家毎に異なった演奏方法(すなわち、奏法若しくはアーティキュレーション)で演奏が行われることがある。あるいは、音符の並び方等によっても、演奏家毎に異なった演奏方法で演奏が行われることもある。そこで、そのような楽譜上の記号(音楽記号や音符の並び方等)を解釈する知識をエキスパートシステム化したものが楽譜解釈部101Bである。

#### 【0029】

楽譜解釈部101Bにおける楽譜上の記号を解釈する際の基準の一例としては、以下のようなものがある。例えば、ピブラートは8分音符以上でないとかけられない。スタッカートでは自然にダイナミクスが大きくなる。テヌート度で音符の減衰率が決まる。レガートは1音中で減衰しない。8分音符ピブラートのスピードは音価でほぼ決まる。音高によってダイナミクスは異なる。更には、1フレーズ内の音高の上昇又は下降によるダイナミクスの変化、減衰ダイナミクスはd b リニア、テヌートやスタッカート等に応じた音符の長さの変化、アタック部のベンドアップの記号に応じたベンドアップの幅とカーブ、といったような各種の解釈基準がある。楽譜解釈部101Bはこのような基準に従って解釈を楽譜に対して行うことにより、楽譜を演奏データに変換する。更に、楽譜解釈部101Bは、ユーザからのプレーヤー指定、すなわちユーザにより誰の演奏か(奏法か)の指定に応じて上述の楽譜解釈処理を行う。楽譜解釈部101Bは、このプレーヤー指定に応じて楽譜の解釈方法を異ならせて楽譜を解釈する。例えば、この複数プレーヤーに対応した異なる楽譜解釈方法はデータベースに蓄積されており、楽譜解釈部101Bはユーザからのプレーヤー指定に応じて選択的に楽譜解釈方法を異ならせて楽譜の解釈を行う。

#### 【0030】

なお、楽譜の解釈結果を示すデータを予め含むように曲データ(演奏情報)を構成してもよい。そのような予め楽譜を解釈した結果のデータを含む曲データを入力した場合には、

上述した処理を行う必要がないことは言うまでもない。また、楽譜解釈部 101B (ステップ S12) における楽譜の解釈処理は全自動で行うようにしてもよいし、ユーザによる人為的入力操作を適宜介在させて行うようにしてもよい。

#### 【0031】

ここで、楽譜解釈部 101B において作成される演奏データの内容を図 21 を参照し説明する。

演奏データは、通常の SMF (スタンダード MIDI フォーマット) と同様に、ヘッダと複数のトラックとから構成されている。各トラックには、プログラムチェンジおよびイベントデータ (ノートオン、ノートオフ等) が含まれているが、本実施形態においては、アタック部、ボディ部、ジョイント部およびリリース部のモジュールを指定するモジュール指定部 (奏法 ID と奏法パラメータを含む奏法指定情報) が含まれている。これらモジュール指定部は、実際には未定義のシステム・エクスクルーシブ・メッセージやメタイベント、14 ビットのコントロールチェンジ等が用いられる。さらに、各トラックには、これらイベントデータまたはモジュール指定部間の時間差を指定する時間差データが含まれている。

10

#### 【0032】

ここで、各トラックに音色を指定するプログラムチェンジ、すなわち新しいデータベース (奏法テーブル + コードブック) を指定するプログラムチェンジが含まれていた場合には、楽譜解釈部 101B は該プログラムチェンジを予測データとして波形合成部 101D に供給する。これは、波形合成部 101D は、次に使用されるベクトルデータを予測してコードブックから読み出すため、予測範囲をある程度絞りこむためにプログラムチェンジを特定したものである (詳細は後述する)。なお、予測データには、他の種々のデータも含まれる。楽譜解釈部 101B においては曲データに基づいて、「その後の奏法指定を示すデータ」が予測データとして波形合成部 101D に供給される。その予測データの一つとして上記プログラムチェンジが含まれるのである。

20

#### 【0033】

##### 3.3. 奏法合成部 101C

奏法合成部 (アーティキュレーター) 101C は楽譜解釈部 (プレーヤー) 101B により変換された奏法指定 (奏法 ID + 奏法パラメータ) に基づいて奏法テーブルを参照して、奏法指定 (奏法 ID + 奏法パラメータ) に応じたパケットストリーム (あるいはベクトルストリームとも呼ぶ) 及び奏法パラメータに応じた該ストリームに関するベクトルパラメータを生成し、波形合成部 101D に供給する (ステップ S13)。パケットストリームとして波形合成部 101D に供給されるデータは、ピッチ (Pitch) 要素及び振幅 (Amplitude) 要素に関してはパケットの時刻情報、ベクトル ID、代表点値列などであり、波形 (Timbre) 要素に関してはベクトル ID、時刻情報などである (詳しくは後述する)。

30

#### 【0034】

次に、波形合成部 101D はパケットストリームに応じてコードブックからベクトルデータを取り出し、該ベクトルデータをベクトルパラメータに応じて変形し、変形したベクトルデータに基づいて波形を合成する (ステップ S14)。それから、他パートの波形生成処理を行う (ステップ S15)。ここで、他パートとは、複数の演奏パートのうち奏法合成処理を行わない、通常の楽音波形合成処理が適用される演奏パートである。例えば、これらの他のパートは通常の波形メモリ音源方式で楽音生成を行う。この「他パートの波形生成処理」は、専用のハードウェア音源 (外部の音源ユニットやコンピュータに装着可能な音源カード) に行わせてもよい。説明を簡略化するために、この実施形態では奏法 (若しくはアーティキュレーション) に応じた楽音生成を行うのは 1 パートのみの場合とする。勿論、複数パートで奏法再生してもよい。

40

#### 【0035】

図 6 は、上述した奏法合成部 101C における奏法合成処理の流れを説明するためのブロック図である。ただし、図 6 では奏法モジュールとコードブックが別々に記憶されている

50

ように図示したが、実際には両方ともハードディスク109のデータベース内に記憶されている。

奏法合成部101Cは、楽譜解釈部101Bからの奏法指定（奏法ID＋奏法パラメータ）と時刻情報のデータに基づいて、波形合成部101Dに供給する各種パケットストリームを作成する。奏法合成部101Cで各音色毎に使用している奏法モジュールは固定的ではなく、ユーザが新たに奏法モジュールを使用中の奏法モジュールに追加したり、使用している奏法モジュールの一部の奏法モジュールの使用を中止したりすることができる。また、奏法合成部101Cでは、選択された要素データと奏法パラメータの値との間のズレ分を補正するための補正情報を作成する処理や、前後の奏法モジュールの波形特性を滑らかに接続する接続部の平滑化などの処理も行う（詳しくは後述する）。

10

なお、標準的には楽譜解釈部101Bから奏法合成部101Cに対してデータが与えられるがそれに限らず、前述のとおり、楽譜解釈部101Bにより既に解釈の終わっている奏法指定付き曲データ乃至人間が楽譜の解釈をして奏法IDや奏法パラメータを付与した奏法指定付き曲データを用意して、それを再生したデータを奏法合成部101Cに供給するようにしてもよい。

#### 【0036】

図7は、奏法合成処理の一実施形態を詳細に示したフローチャートである。

奏法合成部101Cは、奏法ID及び奏法パラメータに応じて奏法テーブルから奏法モジュールの選択を行う（ステップS21）。すなわち、楽譜解釈部101Bから送信された奏法ID（楽器情報＋モジュールパーツ名）と奏法パラメータに応じて1つの奏法モジュールを選択する。この際に、楽譜解釈部101Bは楽譜を解釈する前に楽器情報の示す音色に対応してどのようなモジュールパーツが奏法テーブルに存在するかを予めデータベースをチェックして確認し、存在しているパーツの範囲で奏法IDを指定する。なお、存在しないパーツが指定された場合には、その代わりに類似の特性を有する奏法IDが選択されるようにしてもよい。次に、該指定された奏法IDと奏法パラメータに応じて複数の要素データを選択する（ステップS22）。すなわち、指定された奏法IDと奏法パラメータとにより奏法テーブルを参照することにより、奏法モジュールを特定し、該モジュールから該奏法パラメータに対応した複数の要素データを選択する。この際に、奏法モジュール中に奏法パラメータに完全一致する要素データが存在しない場合には、その値に近い奏法パラメータに対応した要素データが選択される。

20

30

#### 【0037】

次に、時刻情報に応じて要素データ中の各位置の時刻を算出する（ステップS23）。すなわち、各要素データを、時刻情報に基づいて絶対的な時間位置に配置する。具体的には、時刻情報に基づいて、各相対的な時間位置を示す要素データから対応する絶対時間を算出する。こうして、各要素データのタイミングを決定する（図3参照）。そして、奏法パラメータに応じて各要素データの値を補正する（ステップS24）。すなわち、選択された要素データと奏法パラメータの値との間のずれ分を補正する。例えば、楽譜解釈部101Bから送信されたAltosax [Normal Attack]モジュールのAttack直後の音量（奏法パラメータ）が「95」であり、奏法テーブルに存在するAltosax [Normal Attack]モジュールのAttack直後の音量が「100」である場合、奏法合成部101CはAttack直後の音量が「100」であるAltosax [Normal Attack]モジュールの要素データを選択する。しかし、このままではAttack直後の音量が「100」のままであることから、選択された要素データの代表点に対して補正を行うことによってAttack直後の音量を「95」に補正する。このように、選択された要素データの値を送信された奏法パラメータの値に近づけるようにして補正を行う。また、設定されているマイクロチューニング（楽器の調律）の値に応じた補正や楽器の音量変化特性に応じた音量の補正等も行う。これらの補正は各要素データの代表点値を変化することにより行われ、代表点値を大きく変化することもある。すなわち、補正を行うのに必要十分なデータが代表点であり、この代表点をコントロールすることによって各種の補正を行う。

40

50

## 【 0 0 3 8 】

なお、上記ステップ S 2 3 では、上記奏法パラメータのような補正情報によって、上記時刻情報が示す時間位置を補正するようにしてもよい。例えば、演奏データに基づいて得られる時間位置と上記時刻情報が示す時間位置とが一致しない場合に、演奏データに基づいて得られる時間位置に近い時間位置を示す時刻情報を選択して、そこで取得した時刻位置情報を演奏データに応じて補正することで、演奏データの意図する時刻位置情報を得ることができる。また、演奏データがタッチやベロシティのような可変制御ファクタを含む場合は、その可変制御ファクタに応じて時刻位置情報を補正することで、演奏データに応じた時刻位置情報の可変制御を行うことができる。補正情報は、このような時刻位置補正を行うための情報を含む。

10

## 【 0 0 3 9 】

更に、各要素データを調整して隣り合う奏法モジュールの接続部を平滑化するためにリンク処理を行う（ステップ S 2 5）。すなわち、前後の奏法モジュールにおける接続部の代表点を互いに接近させて接続することによって、前後の奏法モジュールの波形特性が滑らかになるようにする。このような接続若しくはリンク処理は、調和成分の波形（T i m b r e）、振幅（A m p l i t u d e）、ピッチ（P i t c h）等の各要素毎に、あるいは調和外成分の波形（T i m b r e）、振幅（A m p l i t u d e）の各要素毎に、別々に行われる。

## 【 0 0 4 0 】

この際、前の奏法モジュールの「リンク開始点」から、後の奏法モジュールの「リンク終了点」までの範囲で調整を行う。すなわち、「リンク開始点」から「リンク終了点」の範囲内にある代表点を「歩みより率」に基づいて調整する。この「歩みより率」は、前の奏法モジュールと後の奏法モジュールからそれぞれどれだけ歩み寄ったところで接続するかを制御するためのパラメータであり、後述するように前後の奏法モジュールの組み合わせに従って決定される。また、前後の奏法モジュールを接続した際に、波形の接続がうまく行かない場合には、前後いずれかの奏法モジュールでその波形特性のベクトル I D を間引くことにより接続を滑らかにする。この間引きを実現するために、「奏法モジュール組み合わせテーブル」と、これから参照される「間引き実行パラメータ範囲テーブル」と、さらにこれから参照される「間引き時間テーブル」を用意する。

20

## 【 0 0 4 1 】

この他にも、以下のような楽譜解釈部 1 0 1 B におけるリンク処理により波形特性を滑らかに接続することができる。例えば、奏法モジュールとは関係なく、奏法パラメータ（ダイナミクス値、ピッチパラメータ値等）の不連続部分を滑らかに接続する。あるいは、ビブラートからリリースへと移行する場合にビブラートを早めに減少させることにより、滑らかに接続する。

30

## 【 0 0 4 2 】

ここで、上述のリンク処理について詳しく説明する。すなわち、前後の奏法モジュールの接続部を平滑化する（ステップ S 2 5 参照）ための各要素データの調整について簡単に説明する。まず、図 8 を用いて奏法モジュールが振幅（A m p l i t u d e）要素又はピッチ（P i t c h）要素と対応する場合のリンク処理について説明する。

40

## 【 0 0 4 3 】

前の奏法モジュールと後の奏法モジュールとの接続部における代表点の値の不連続により両者間の接続点に段差が生じている場合、まずダイナミクス接続点（A m p l i t u d e の場合）あるいはピッチ接続点（P i t c h の場合）の目標値を、前後どちらの奏法モジュール側の値により近づけるかという指標の「歩みより率」を決定する。本実施形態では「歩みより率」が図示のようなテーブルによって与えられるとする。例えば前の奏法モジュールのベクトル I D が「3」であり、後の奏法モジュールのベクトル I D が「7」である場合の「歩みより率」はテーブルから「30」と決定される。こうして決定された「歩みより率」により前の奏法モジュールの「リンク開始点」から「奏法モジュールの終了点」まで、徐々に目標値に向けてエンベロープ形状を変形する。また、後の奏法モジュール

50

の「リンク終了点」から「奏法モジュール開始点」まで、徐々に目標値に向けてエンベロープ形状を変形する。例えば「歩みより率」が「30」と決定された場合、前の奏法モジュールに対する目標値は「30」であり、前の奏法モジュールは後の奏法モジュール側に「30」%歩みよりを行う（本実施形態では、前の奏法モジュールにおける最後の代表点が下方に「30」%歩みよりする）。

#### 【0044】

一方、後の奏法モジュールは前の奏法モジュール側に「70」（100 - 30）%歩みよりを行う（本実施形態では、後の奏法モジュールにおける最初の代表点が上方に「70」%歩みよりする）。また、リンク開始点からリンク終了点までに存在する前後の奏法モジュールの複数代表点が上記歩みよりに伴って各々上下に歩みよりを行う。このように、歩みよりは前後する奏法モジュールの複数の代表点で行われる。なお、リンク開始点とリンク終了点は適宜定めてよいが、リンク開始点やリンク終了点を所望の代表点と同一の点に設定すると、図に示したようなリンク開始点やリンク終了点におけるエンベロープ形状の折れ曲がりがなくなるので望ましい。勿論、リンク開始点やリンク終了点を所望の代表点と同一の点に設定していない場合でも、エンベロープ形状に折れ曲がりが生じないように歩みよりを行うようにしてよいことは言うまでもない。

#### 【0045】

なお、「歩みより率」の決定は上述した例に限られるものではない。例えば、接続点の前後で指定された奏法パラメータに基づいて決定してもよい。または、奏法IDや奏法パラメータになる前の演奏データに基づいて決定してもよい。あるいは、それらのデータの組み合わせに基づいて決定してもよい。また、上述の例では「歩みより率」により歩みよりする代表点は1つであり、その他の代表点はその歩みよりに伴って適量だけ歩みよりするようにしたが、複数の代表点各々について別々に「歩みより率」を決定し、それに従って複数の代表点を各々「歩みより率」分だけ歩みよりするようにしてもよい。

#### 【0046】

次に、奏法モジュールが波形（Timbre）要素である場合のリンク処理について説明する。図9～図12は、奏法モジュールが波形（Timbre）要素である場合のリンク処理を説明するための概念図である。図9はアタック部波形とボディ部波形とを接続した場合の波形の間引きを説明するための概念図であり、図10はボディ部波形とリリース部波形とを接続した場合における波形の間引きを説明するための概念図である。図9では、ボディ部波形は5つのループ波形L1～L5からなり、各々所定の時間範囲でループ再生されるものとする。同様に、図10のボディ部波形は6つのループ波形L1'～L6'からなるものとする。

#### 【0047】

波形に関する要素データの調整（つまり、波形のリンク処理）の方法には種々あるが、その一例として、例えばアタック部あるいはジョイント部の奏法モジュールとボディ部の奏法モジュールとの接続（あるいは、ボディ部の奏法モジュールとリリース部あるいはジョイント部の奏法モジュールとの接続）において、波形の部分的間引きにより滑らかに接続する方法を提案する。波形と波形とを接続する際に、クロスフェード合成することはよく知られている。しかし、図9の例の場合のように、接続時点から最初のループ波形L1の開始位置までの時間tが短い場合、短い時間t内で急なクロスフェード合成をしなければならなくなる。そのような急なクロスフェード波形合成、つまり接続する波形と波形との間の時間が非常に接近している場合に当該波形間でクロスフェード波形合成を行うと、それに伴って大きなノイズを発生する波形を生ずることになり、好ましくない。

#### 【0048】

そこで、波形の一部を間引き（削除）して接続する波形と波形との時間間隔を広げることにより、急なクロスフェード波形合成を行わないようにする。この場合に、アタック部やリリース部あるいはジョイント部における波形は1つの塊であって、波形を間引くことができないので、この場合はボディ部側のループ波形の間引きを行う。図9及び図10では、黒く塗りつぶした長方形で示したループ波形L1、L6'を間引きする。例えば、図9

10

20

30

40

50

では接続時点からの時間差が比較的長い2番目のループ波形L2とアタック部波形の末尾波形とをクロスフェード合成し、最初のループ波形L1は使用しない。同様に、図10ではループ波形L5'とリリース部波形との間でクロスフェード合成を行い、波形L6'は使用しない。

なお、ジョイント部とは音と音の間（又は音部分と音部分の間）を任意の奏法でつなぐ波形区間のことである。

#### 【0049】

また、アタック部の奏法モジュールとリリース部あるいはジョイント部の奏法モジュールとの接続を波形間引きにより滑らかにする。図11及び図12は、アタック部波形とリリース部波形とを接続する場合における波形の間引きを説明するための概念図である。

この場合には、アタック部あるいはリリース部等の奏法モジュールが波形間引きできる場合とできない場合とがある。アタック部の奏法モジュールが波形間引きできる例としてはベンドアタック部（後半にいくつかのループ波形を持つ）がある。また、前半にいくつかのループ波形を持つリリース部の場合も波形間引きが行える。このように、波形間引きできる側の奏法モジュールを波形間引きする。例えば、ベンドアタック部とリリース部とを接続する場合には、図11に示すようにベンドアタック部側のループ波形を間引きする（図11では、ベンドアタック部側の黒く塗りつぶした長方形で示したループ波形を1つ間引きする）。また、ノーマルアタック部とループ波形を有するリリース部とを接続する場合には図12に示すようにリリース部側のループ波形を間引きする（図12では、リリース部側の黒く塗りつぶした長方形で示したループ波形を1つ間引きする）。

なお、間引く対象とするループ波形は奏法モジュールと奏法モジュールとの接続部に最も近いループ波形（先頭あるいは最後に位置するループ波形）とすることに限らず、複数ループ波形から所定の優先順位に従って間引く対象とするループ波形を特定するようにしてもよい。

#### 【0050】

このように、ある奏法モジュールの組み合わせにおいて、ある奏法パラメータの範囲で接続がうまく行かない場合に波形を間引くが、これを実現するために、例えば「奏法モジュール組み合わせテーブル」と、これから参照される「間引き実行パラメータ範囲テーブル」と、更にこれから参照される「間引き時間テーブル」を用意する。「奏法モジュール組み合わせテーブル」は、接続する前後の奏法モジュールの組み合わせにより所定のパラメータを決定するためのテーブルである。「間引き実行パラメータ範囲テーブル」は、上記パラメータ毎に間引きを行う時間の範囲を決定するためのテーブルである。「間引き時間テーブル」は、間引き時間を決定するためのテーブルである。接続時点と最初の（又は最後の）ループ波形L1（又はL6'）との時間差（図9～図12に示す時間t）が基準の間引き時間より短い場合に、該ループ波形を間引く。

#### 【0051】

さらに、奏法モジュールのサンプル長が短く、当該奏法モジュールの後に続く奏法モジュールが開始されるよりも前に終了してしまう場合の波形接続について図13を用いて説明する。ただし、図13では、図の左側から右側に時系列にA.Sax[BendUpAttack]、A.Sax[NormalShortBody]、A.Sax[VibratoBody]、A.Sax[NormalRelease]の4つの奏法モジュールで波形（Timbre）要素のパケットストリームを形成している場合について説明する。各奏法モジュールのサンプル長（区間長）は、“length”で示す長さで表せられる。図13において、最上段に記載されている「ノートオン」と「ノートオフ」は、MIDIデータのイベントタイミングである。また、中段に記載されているA.Sax[BendUpAttack]等はそれぞれ奏法IDの発生タイミングであり、note、dynamics、depth等はそれぞれ奏法パラメータの発生タイミングを示す。

#### 【0052】

A.Sax[BendUpAttack]モジュールは、時刻t0から開始される。また、時刻t1は当該モジュール内のノートオンのタイミングであり、指示されたノートオ

10

20

30

40

50

ンタイミングにあわせる。また、当該モジュールのパケットストリームの内容は、上記 `note`、`dynamic s`、`depth` 等の奏法パラメータに基づいて制御される。`A . Sax [ Normal Short Body ]` モジュールは、アタックモジュール直後の時刻 `t 2` から開始される。時刻 `t 3` は、接続部において、その途中からビブラート奏法がスタートしているタイミングである。このタイミングは、例えば、曲データに付与されたビブラート記号の開始タイミングに基づいて決定される。時刻 `t 5` は、`A . Sax [ Normal Release ]` モジュール内のノートオフタイミングであり、指示されたノートオフタイミングにあわせる。`A . Sax [ Normal Release ]` モジュールの始まりの時刻 `t 4` はそれに応じて特定される。

#### 【 0 0 5 3 】

すなわち、時刻 `t 1` においてノートオンされ、時刻 `t 5` においてノートオフされることから、実際に当該パケットストリームから生成される波形に従って発音される時間は時刻 `t 1` から時刻 `t 5` までの時間である。このようなパケットストリームの場合に、時刻 `t 2` から時刻 `t 4` までの時間長と、その間の `A . Sax [ Normal Release ]` モジュールと `A . Sax [ Vibrato Body ]` モジュールの各サンプル長 `length` の合計が合わないことが多く、適切に対処する必要が生ずる。このような場合、同じ奏法モジュールを繰り返すことによってサンプル長 `length` の合計を前記時間長にあわせるか、奏法モジュールのサンプル長を可変して前記時間長に合わせるか、あるいは前記両方を組み合わせて用いて前記時間長を合わせる。このようにして、各モジュール間で調節して波形接続を行うようになっている。上述の例では、`A . Sax [ Normal Short Body ]` モジュールを繰り返すことにより、その後続く `A . Sax [ Vibrato Body ]` モジュールと波形接続を行っている。同様に、`A . Sax [ Vibrato Body ]` モジュールを繰り返すことにより、その後続く `A . Sax [ Normal Release ]` モジュールとの波形接続を行っている。

#### 【 0 0 5 4 】

上記のように、奏法モジュールを複数回繰り返して波形接続を行う場合、複数繰り返す側の奏法モジュールの時間長を可変する。この時間長の可変制御は、上述した例では `A . Sax [ Normal Short Body ]` モジュールあるいは `A . Sax [ Vibrato Body ]` モジュールにおける代表点を移動することによってなされる。すなわち、`A . Sax [ Normal Short Body ]` モジュールあるいは `A . Sax [ Vibrato Body ]` モジュールを構成する複数のループ波形間のクロスフェード接続の時間を変化するなどの適宜の方法によって実現する。ループ波形の場合は、ループ回数乃至ループ継続時間を可変することによって、比較的簡単に、ループ再生波形全体の時間長を可変制御することができる。一方、ノンループ波形の場合は、時間軸上におけるその存在長を可変制御することはそれほど簡単ではない。従って、上記のように、ノンループ波形とループ波形とからなる一連の音の波形において、ループ読出区間の波形データの時間軸を伸縮可変制御することで全体の発音時間長を可変制御するように工夫することは、時間伸縮制御を容易にするので極めて好ましい。そのために、特開平 1 0 3 0 7 5 8 6 号で本出願人が先に提案した「`Time Stretch & Compress`」制御（略して「`TSC` 制御」）を用いるとよい。特に、特殊な奏法に対応するノンループ波形において時間軸の長さを可変するために、上記「`TSC` 制御」は好ましく応用できる。

#### 【 0 0 5 5 】

このようにして作成されるパケットストリームの一例を概念的に示すと、図 1 4 のようである。図 1 4 では、上から順に調和成分の振幅（`Amplitude`）要素、ピッチ（`Pitch`）要素、波形（`Timbre`）要素、調和外成分の振幅（`Amplitude`）要素、波形（`Timbre`）要素における各パケットストリームを示す。また、調和成分の振幅（`Amplitude`）要素、ピッチ（`Pitch`）要素、調和外成分の振幅（`Amplitude`）要素において、黒く塗りつぶした正方形は代表点であり、これらを結ぶ曲線はパケットストリームの中のパケットに含まれるベクトル ID で示されたベクトルの形状を示す。調和成分の波形（`Timbre`）要素、調和外成分の波形（`Timbre`

10

20

30

40

50

要素において、白抜きの長方形Lはループ波形を示すものであり、その他の長方形NLはノンループ波形を示すものである。なお、ノンループ波形のうち、斜線を引いたものは特に特徴のあるノンループ波形を示すものである。さらに、この実施形態では、NormalAttackモジュールにおける調和成分と調和外成分の波形(Timbre)要素をそれぞれ2個のベクトルで構成し、調和成分の振幅(Amplitude)要素とピッチ(Pitch)要素及び調和外成分の振幅(Amplitude)要素をそれぞれ1個のベクトルで構成している。

#### 【0056】

本実施形態では、調和成分、調和外成分とも、波形(Timbre)要素がノンループ波形となっている部分については、振幅(Amplitude)要素及びピッチ(Pitch)要素がベクトルを持たないようになっている。しかしながら、波形(Timbre)要素がノンループ波形であるところでも、振幅(Amplitude)要素、ピッチ(Pitch)要素のベクトルを持ち生成波形を制御するようにしてもよい。VibratoBodyモジュールでは、調和成分の波形(Timbre)要素を5個のベクトルで構成し、調和成分の振幅(Amplitude)要素とピッチ(Pitch)要素及び調和外成分の波形(Timbre)要素と振幅(Amplitude)要素をそれぞれ1個のベクトルで構成している。ここで、VibratoBodyモジュールは3回繰り返されているが、各繰り返し毎に各ベクトルの形状が異なっていることに注意してほしい。これは各繰り返し毎に異なる奏法パラメータが指定されたからである。異なる奏法パラメータに応じて異なる要素データが選択されたり、異なる奏法パラメータに応じて異なるレベル制御、時間軸制御が行われたりする。NormalJointモジュールでは、調和成分と調和外成分の波形(Timbre)要素をそれぞれ3個のベクトルで構成し、調和成分の振幅(Amplitude)要素とピッチ(Pitch)要素及び調和外成分の振幅(Amplitude)要素をそれぞれ2個のベクトルで構成している。なお、NormalBodyモジュールの説明は省略する。

#### 【0057】

奏法合成部101Cは、上述したようにパケットストリームを各成分(調和成分及び調和外成分)毎に生成する。これらのパケットストリームは複数個のパケットで構成されてなり、1個1個のパケットはベクトルIDとパケットの時刻情報とを具える。それに加えて、調和成分の振幅(Amplitude)要素、ピッチ(Pitch)要素、調和外成分の振幅(Amplitude)要素の場合には各代表点の確定値などを具える。勿論、これに限られるものではなく、ベクトルIDとパケットの時刻情報に加えて他の情報を具えてよい。このような1つ1つのパケットの内容に従って、各成分毎にパケットストリームが構成されている。このように、パケットストリームには複数のパケット及びその各パケットの時刻情報(開始時刻)が含まれる。

なお、楽器の種類等によってパケットストリームの数が異なっていてよいことは言うまでもない。

#### 【0058】

### 3.4. 波形合成部101D

#### 3.4.1. 波形合成部101Dの全般動作

波形合成部101Dは、奏法合成部101Cから供給される各成分毎のパケットストリーム(ベクトルID、時刻情報、補正情報等を含む複数のパケットの列)に基づいて波形を合成する。図15は、波形合成部101Dにおける動作を説明するために全体構成を示した概念図である。図16～図19、図22、図23は、波形合成部101Dにおける各動作を詳細に説明するための図面である。図16は、波形合成の全体の流れを簡単に示すブロック図である。図17は、ベクトルロードを説明するためのブロック図である。図18は、ベクトルオペレータを説明するためのブロック図である。図19は、ベクトルデコーダを説明するためのブロック図である。図22は、奏法合成部101Cから波形合成部101Dに各パケットが供給されるタイミングを示す図であり、図23はキャッシュ制御部40を説明するためのブロック図である。

10

20

30

40

50

## 【 0 0 5 9 】

図 1 5 において、奏法合成部（アーティキュレーター）1 0 1 C で作成された各成分要素毎のパケットストリームは、波形合成部 1 0 1 D における各成分要素毎に対応して設けられる所定のパケットキューバッファ 2 1 ~ 2 5 に順次にパケット入力（つまり、パケット単位での入力）される。入力されたパケットはパケットキューバッファ 2 1 ~ 2 5 に蓄積され、順次所定の順番でベクトルローダ 2 0 に送られる。ベクトルローダ 2 0 ではパケット内のベクトル I D を参照して、当該ベクトル I D に対応するオリジナルのベクトルデータをキャッシュ制御部 4 0 を介してコードブック 2 6 から読み出し（ロード）する。読み出されたベクトルデータは、各成分要素毎に対応して設けられる所定のベクトルデコード 3 1 ~ 3 5 へ送られ、ベクトルデコード 3 1 ~ 3 5 で各成分要素毎の波形が生成される。

10

## 【 0 0 6 0 】

さらに、ベクトルデコード 3 1 ~ 3 5 では、生成された各成分要素毎の波形を各ベクトルデコード 3 1 ~ 3 5 間で同期しながら各成分毎（調和成分及び調和外成分）の波形を生成する。こうして生成された各成分毎の波形は、ミキサ 3 8 に送られる。奏法合成部（アーティキュレーター）1 0 1 C では、パケットキューバッファ 2 1 ~ 2 5 に対してパケットを入力する他、ストリーム管理（個々のベクトルデータの生成や削除あるいはベクトルデータ間の接続に関する管理）や再生コントロール（所望の波形生成の実行あるいは生成された所望の波形の再生 / 停止などのコントロール）などの各種の制御を波形合成部 1 0 1 D に対して実行する。

## 【 0 0 6 1 】

20

上述したように、ベクトルローダ 2 0 にはパケットキューバッファ 2 1 に蓄積されたパケットストリームを構成するパケットを順次に入力され、ベクトルローダ 2 0 は各パケット内のベクトル I D に基づいて、キャッシュ制御部 4 0 を介してコードブック 2 6 から当該ベクトル I D に対応するベクトルデータを読み出し、読み出したベクトルデータをベクトルデコード 3 1 に送る（図 1 6 参照）。この際に、読み出した各パケット内に補正情報（例えば、代表点に関する補正情報）が付されている場合がある。このような場合、ベクトルローダ 2 0 では読み出したオリジナルのベクトルデータを補正情報に従って変形し、変形したベクトルデータ（これをオリジナルのベクトルデータと区別するためにベクトル情報データと呼ぶ）を情報として持つパケット（これを奏法合成部 1 0 1 C から入力されたパケットと区別するためにベクトルパケットと呼ぶ）をベクトルデコード 3 1 ~ 3 5 へ出力する。このように、ベクトルローダ 2 0 は奏法合成部（アーティキュレーター）1 0 1 C から入力されたパケットのベクトル I D に基づきオリジナルのベクトルデータをコードブック 2 6 から読み出し、必要に応じて補正情報でベクトルデータの補正を行った上でベクトルデコード 3 1 ~ 3 5 にベクトルパケットを渡す（図 1 7 参照）。上述したようなベクトルデータの代表点に関する補正情報には、例えば乱数に基づいて時刻情報をずらすための補正情報等いろいろな補正情報がありうる。

30

## 【 0 0 6 2 】

図 1 8 に示すように、ベクトルデコード 3 1 ~ 3 5 は入力されたベクトルパケットを処理するためのベクトルオペレータの生成や破棄、ベクトルオペレータ間の接続・同期管理、時刻管理、他ベクトル I D ストリームからの入力のベクトルオペレータ毎のパラメータへの変換設定などの各種のオペレータ動作管理を行うものである。ベクトルオペレータ 3 6 及び 3 7 では、ベクトル情報データの読み出し、あるいはベクトル情報データの読み出し位置制御（S p e e d 入力）やゲイン制御（G a i n 入力）などが行われる。このベクトルオペレータ 3 6 及び 3 7 に設定される各種のパラメータは、ベクトルデコード 3 1 ~ 3 5 で管理される。各成分要素毎に対応するようにベクトルデコード 3 1 ~ 3 5 が設けられており、当該ベクトルデコード 3 1 ~ 3 5 がベクトルパケット内のベクトル情報データを読み出して所望の波形の時系列的生成を行う。

40

## 【 0 0 6 3 】

例えば、図 1 9 に示すように、ベクトルデコード 3 1 は調和成分の振幅（A m p l i t u d e）要素のエンベロープ波形を生成し、ベクトルデコード 3 2 は調和成分のピッチ（P

50

i t c h) 要素のエンベロープ波形を生成し、ベクトルデコーダ 33 は調和成分の波形 (T i m b r e) 要素の波形を生成し、ベクトルデコーダ 34 は調和外成分の振幅 (A m p l i t u d e) 要素のエンベロープ波形を生成し、ベクトルデコーダ 35 は調和外成分の波形 (T i m b r e) 要素のエンベロープ波形を生成する。ベクトルデコーダ 33 は、ベクトルデコーダ 31 及び 32 で生成された調和成分の振幅 (A m p l i t u d e) 要素のエンベロープ波形と調和成分のピッチ (P i t c h) 要素のエンベロープ波形を付与した調和成分の波形を生成してミキサ 38 へ出力する。すなわち、ベクトルデコーダ 33 に対して、調和成分の振幅 (A m p l i t u d e) 要素のエンベロープ波形をゲイン制御 (G a i n 入力) を行うためのベクトルオペレータとして、調和成分のピッチ (P i t c h) 要素のエンベロープ波形をベクトル情報データの読み出し位置制御 (S p e e d 入力) を行うためのベクトルオペレータとして入力する。また、ベクトルデコーダ 35 では、ベクトルデコーダ 34 で生成された調和外成分の振幅 (A m p l i t u d e) 要素のエンベロープ波形を付与した調和外成分の波形を生成してミキサ 38 へ出力する。すなわち、ベクトルデコーダ 35 に対しては、調和外成分の振幅 (A m p l i t u d e) 要素のエンベロープ波形をゲイン制御 (G a i n 入力) を行うための制御命令として入力する。

10

#### 【0064】

さらに、各成分、要素における波形の時系列的生成の際には、各ベクトルデコーダ 31 ~ 35 間で波形の同期を行いながら波形生成が行われる。例えば、波形 (T i m b r e) 要素のベクトルパケットと振幅 (A m p l i t u d e) 要素のベクトルパケットを入力した場合、波形 (T i m b r e) 要素のベクトルパケットに基づく波形合生成の時間を基準として、それに同期して振幅 (A m p l i t u d e) 要素のベクトルパケットに基づく振幅波形を生成する。該振幅波形により、波形 (T i m b r e) 要素のベクトルパケットに基づいて生成される波形の振幅が制御される。波形 (T i m b r e) 要素のベクトルパケットとピッチ (P i t c h) 要素のベクトルパケットを入力した場合、波形 (T i m b r e) 要素のベクトルパケットに基づく波形生成の時間を基準として、それに同期してピッチ (P i t c h) 要素のベクトルパケットに基づくピッチ波形を合成する。

20

#### 【0065】

該ピッチ波形により、波形 (T i m b r e) 要素のベクトルパケットに基づいて生成される波形のピッチが制御される。調和成分の波形 (T i m b r e) 要素のベクトルパケットと調和外成分の波形 (T i m b r e) 要素のベクトルパケットを入力する場合、調和成分の波形 (T i m b r e) 要素のベクトルパケットに基づく調和成分合成の時間を基準として、それに同期して調和外成分の波形 (T i m b r e) 要素のベクトルパケットに基づく調和外成分が合成される。合成された調和成分と調和外成分の波形を混合することにより、所望の楽音波形を生成する。

30

なお、この実施形態において、調和成分と調和外成分の同期または非同期を選択できるように構成し、同期が選択された場合にのみにおいて、上述の調和成分の波形 (T i m b r e) 要素のベクトルパケットに基づいて生成される調和成分の波形合成の時間を基準として、それに同期して調和外成分の波形 (T i m b r e) 要素のベクトルパケットに基づいて生成される調和外成分の波形を合成するようにしてもよい。

#### 【0066】

40

上述したように、パケットストリームは複数パケット列で構成されており、ベクトルパケットのパケットストリームの場合、各パケットには各々ベクトルデータが含まれている。すなわち、パケットストリームとはベクトルデータが時間方向に並んだものであり、振幅 (A m p l i t u d e) 要素のベクトルデータもピッチ (P i t c h) 要素のベクトルデータも波形 (T i m b r e) 要素のベクトルデータもデータ構造や意味は違うが、ベクトルオペレータ 36 及び 37 から見た場合は基本的に同じものである。

#### 【0067】

##### 3.4.2. ベクトルデータの構造

図 20 は、ベクトルデータのデータ構造の一実施形態を概念的に示す概念図である。例えば、ベクトルデータの読み出し位置の単位は [ S E C ] で、読み出し速度を等速とした場

50

合、ベクトルデータ上での1サンプルは出力波形の1サンプルと一致する。また、読み出し速度の単位は $1/1200[\text{cent}]$ ( $=2$ の $n$ 乗)で、指数 $n$ が0だと等速、 $-1$ だと2倍(例えば、波形(Timbre)要素の場合には1オクターブ上がる)、 $-1$ だと0.5倍(例えば、波形(Timbre)要素の場合には1オクターブ下がる)となる(図20の上段図参照)。また、コードブック26には実際のベクトルデータが格納される。例えば、振幅(Amplitude)要素のベクトルデータあるいはピッチ(Pitch)要素のベクトルデータはVECTORPOINT構造体の配列と、代表点データとからなっている。

#### 【0068】

VECTORPOINT構造体の配列には各点のサンプル位置と値が順番に入っており、例えば振幅(Amplitude)要素のベクトルデータの値は[db]単位であり、ピッチ(Pitch)要素のベクトルデータの値はMIDIのノートナンバ0を0.0としたときの $1/1200[\text{cent}]$ 単位である。また、代表点データはWORD型の配列で、代表点となるVECTORPOINT構造体の配列のインデックス番号が格納されている(図20の下段図参照)。勿論、上述した例に限られるものでないことは言うまでもない。

#### 【0069】

### 3.4.3. キャッシュ制御部40の詳細

#### (1) キャッシュ制御部40の全体構成

次に、キャッシュ制御部40の全体構成を図23を参照し説明するが、このキャッシュ制御部40を設けた意義について説明しておく。コードブック26はハードディスク109内に格納されているため、ベクトルデコード31~35が必要とするベクトルデータはハードディスク109から読み出されることになる。しかし、ハードディスク等はアクセス時間が遅く、かつ一定していないため、ベクトルデコード31~35が当該ベクトルデータを加工するタイミングで即座に読み出すことは不可能である。そこで、本実施形態においては、波形合成部101D内にキャッシュ制御部40を設け、将来使用される(または使用が予測されている)ベクトルデータを予めキャッシュメモリにロードすることとしている。

#### 【0070】

図23において42は先読みプリフェッチ部であり、奏法合成部101Cから波形合成部101Dに供給されるパケットからベクトルIDを抽出し、コードブック26からこのベクトルIDに係るベクトルデータをプリフェッチ(先読み)するようにハードディスク109に読み出し制御を行う。上述したように、これらパケットはパケットキューバッファ21~25内のパケットストリームを構成し、やがてベクトルローダ20によって読み出されることになるが、これと並行してプリフェッチ処理が行われることになる。

#### 【0071】

41は予測制御部であり、楽譜解釈部101Bから供給される予測データ(プログラムチェンジ等)と、先読みプリフェッチ部42から供給される予測条件とに基づいて、将来使用される可能性の高いベクトルデータを予測し、予測したベクトルデータに係るベクトルIDを先読みプリフェッチ部42に供給する。ここで、「予測条件」とは、使用されることが確定した(すなわち奏法合成部101Cから供給された)ベクトルID等である。このように、先読みプリフェッチ部42においては、奏法合成部101Cおよび予測制御部41の双方からベクトルIDが供給される。そして、先読みプリフェッチ部42は、使用されることが確定しているベクトルデータのロード(すなわち奏法合成部101Cから指定されたベクトルデータのロード)を優先しつつ、双方のベクトルIDに係るベクトルデータについてプリフェッチ処理を行う。以下、使用されることが確定しているベクトルデータのロードを「指定ロード」と呼び、使用されることが確定していない(単に予測されたものの)ベクトルデータのロードを「予測ロード」と呼ぶ。44はキャッシュメモリであり、プリフェッチされたベクトルデータを格納する。43は読み出し制御部であり、ベクトルローダ20からベクトルIDを受信すると、このベクトルIDに対応するベクトルデ

10

20

30

40

50

ータを主としてキャッシュメモリ 4 4 から読出し、ベクトルローダ 2 0 に供給する。4 5 は時刻管理部であり、プリフェッチ等のタイミング制御を行う。

#### 【 0 0 7 2 】

##### ( 2 ) 予測制御部 4 1 の動作

次に、予測制御部 4 1 における処理内容を図 2 4 の状態遷移図を参照し説明する。この予測制御部 4 1 の状態は、ベクトルデコーダ 3 1 ~ 3 5 において波形合成が行われているか否か、および合成が行われている場合にはどのモジュールに係るものかに応じて決定される。初期状態においては、ベクトルデコーダ 3 1 ~ 3 5 において何ら波形合成が行われていないため、予測制御部 4 1 においてはステップ S 3 0 の処理が行われる。ここでは、アタック部のベクトルデータの候補が予測され、予測されたベクトル ID が先読みプリフェッチ部 4 2 に逐次供給される。但し、ステップ S 3 0 においては、楽譜解釈部 1 0 1 B から予測データとして「プログラムチェンジ」が供給されていない限り、アタック部の予測は実行されない。プログラムチェンジが未定であれば、指定される可能性のあるアタック部のベクトルデータは膨大な数になるからである。従って、初期状態においては、楽譜解釈部 1 0 1 B からプログラムチェンジが供給されると、直ちに該プログラムチェンジに応じたアタック部のベクトルデータの予測ロードが開始されることになる。

#### 【 0 0 7 3 】

例えば、該プログラムチェンジとして「ピアノ」が指定され、「ピアノ」に係るアタック部のベクトルデータが 1 0 0 種類あれば、1 0 0 種類のベクトルデータの予測ロードが開始されることになる。アタック部に係るパケットが確定されると、やがてベクトルローダ 2 0 およびベクトルデコーダ 3 1 ~ 3 5 においてアタック部の波形合成処理が開始される。その際、予測制御部 4 1 の状態はステップ S 3 1 に遷移する。

#### 【 0 0 7 4 】

ステップ S 3 1 においては、確定されているアタック部のベクトル ID に基づいて、ボディ部のベクトルデータの候補について予測ロードが行われる。すなわち、アタック部のピッチは既知であるから、候補となるベクトルデータはアタック部と同一のピッチに対応するものに限られる。さらに、確定されているアタック部のエンベロープ波形等に対応して、これに繋がる可能性があるボディ部のベクトルデータは、さらに限定される。ステップ S 3 1 においては、このような条件に応じて絞りこまれた候補が予測ロードされるのである。ここで、ボディ部に係るパケットが実際に奏法合成部 1 0 1 C から供給され、当該ボディ部のパケットが確定されると、予測制御部 4 1 の状態はステップ S 3 3 に遷移する。

#### 【 0 0 7 5 】

ステップ S 3 3 においては、確定されているボディ部のベクトル ID に基づいて、次のベクトルデータの候補について予測ロードが行われる。上述したように、ボディ部に繋がるモジュールは、他のボディ部、ジョイント部あるいはリリース部のうち何れかである。従って、予測制御部 4 1 においては、これらのベクトルデータが予測ロードされる。なお、予測ロードされるベクトルデータの候補は、確定されたボディ部のピッチやエンベロープ波形等に応じて絞りこまれることはステップ S 3 1 と同様である。

#### 【 0 0 7 6 】

ステップ S 3 3 の実行中に実際に奏法合成部 1 0 1 C からパケットが供給されると、当該パケットに応じて予測制御部 4 1 の状態が遷移する。まず、供給されたパケットがボディ部のパケットであった場合には、状態はステップ S 3 3 に留まり、再び当該ボディ部に基づいて、他のボディ部、ジョイント部あるいはリリース部の予測ロードが実行される。また、供給されたパケットがジョイント部のパケットであった場合には、予測制御部 4 1 の状態はステップ S 3 2 に遷移する。

#### 【 0 0 7 7 】

ステップ S 3 2 においては、ボディ部のベクトルデータの候補が予測ロードされる。ここで、予測ロードされるベクトルデータの候補は、確定されているジョイント部に繋がる可能性のあるボディ部のベクトルデータに絞りこまれる。ここで、奏法合成部 1 0 1 C から次のボディ部に係るパケットが供給されると、予測制御部 4 1 の状態は再びステップ S 3

3に戻り、他のボディ部、ジョイント部あるいはリリース部のベクトルデータが予測ロードされる。ここで、奏法合成部101Cからリリース部のパケットが供給されると、予測制御部41の状態はステップS30に遷移する。

【0078】

ステップS30においては、上述したように、アタック部のベクトルデータの候補が予測ロードされる。但し、その直前のリリース部のピッチが既に確定されているならば、次に来るアタック部のピッチはあまり大きく外れない可能性が高い。従って、リリース部の合成中において、予測ロードされるアタック部のベクトルデータは、当該リリース部のピッチの周辺（例えば±1オクターブの範囲）に対応するものに限定してもよい。

【0079】

(3) 先読みプリフェッチ部42の動作

(3.1) ロード処理(図25)

次に、先読みプリフェッチ部42の動作を説明する。まず、先読みプリフェッチ部42においては、所定周期毎に図25に示すロード処理が実行される。図において処理がステップS41に進むと、未だ実行されていない指定ロード要求を受けているか否かが判定される。ここで「YES」と判定されると、処理はステップS42に進み、要求を受けたベクトルデータの指定ロードが実行される。これにより、暫くの読出し時間が経過すると、ハードディスク109からこれらベクトルデータが読出されキャッシュメモリ44に格納されることになる。

【0080】

一方、ステップS41において「NO」と判定されると、処理はステップS43に進み、予測ロードが実行される。すなわち、予測ロード要求のあったベクトルデータのうち、キャッシュメモリ44内への読出し指令が行われているもの（既にキャッシュメモリ44内にロードされているものと、他のロード要求に応じてロードが予定されているもの）と、未だハードディスク109に読出し指令が行われていないものが検出され、後者のみについてハードディスク109に読出し指令が供給される。なお、予測ロードされるベクトルデータが多い場合には、予測ロードの途中で再び本ルーチンが呼び出される場合もある。かかる場合にも、要求された指定ロードが存在する限りは、当該予測ロードが中断され、ステップS42を介して指定ロードが実行されることになる。このように、先読みプリフェッチ部42においては、予測ロードよりも指定ロードの方が優先的に実行される。

【0081】

(3.2) パケット受信処理(図26)

また、先読みプリフェッチ部42においては、奏法合成部101Cからパケットが供給される毎に図25に示すパケット受信処理が実行される。図において処理がステップS51に進むと、供給されたパケットからベクトルIDが抽出され、該ベクトルIDに対応するベクトルデータがヒットしているか否かが判定される。ここで、「ヒットした場合」とは、(1)既にキャッシュメモリ44内に予測ロードされたベクトルデータのうち何れかが的中した場合、(2)未だキャッシュメモリ44にロードされいないが、当該ベクトルデータの予測ロードが予定されていた場合、(3)過去にロードされ使用済みになった(USED状態の)ベクトルデータがそのまま使用できる場合、のうち何れかである。

【0082】

なお、本実施形態においては、使用済みのベクトルデータは直ちに解放されるのではなく、「USED状態」のベクトルデータとして一応キャッシュメモリ44内に残される。そして、他のベクトルデータを記憶する記憶容量が足りなくなった場合等において、USED状態のベクトルデータの領域が解放され、新たなベクトルデータが記憶されるのである。なお、その詳細については後述する。

【0083】

さて、ヒットするベクトルデータが存在しなかった場合は、ステップS51において「NO」と判定され、処理はステップS53に進む。ここでは、上記パケットに含まれていたベクトルIDについて、指定ロード要求が発生する。従って、次に上述したパケット受信

10

20

30

40

50

処理（図 2 6）が実行される際に、該ベクトルデータが指定ロードされることになる。

#### 【 0 0 8 4 】

一方、ヒットしたベクトルデータが存在する場合は「YES」と判定され処理はステップ S 5 2 に進む。ここでは、該ベクトルデータに係る予測ロードが完了していなければ、該予測ロード要求が指定ロード要求に変更される。これは、その後にパケット受信処理が実行される際、優先的にロードされるからである。次に、処理がステップ S 5 4 に進むと、奏法合成部 1 0 1 C に対して、指定ロード要求されたベクトルデータのハンドルが返される。このハンドルは、ベクトルデータに対して一意に対応するものである。仮に指定ロード要求されたベクトルデータが既にキャッシュメモリ 4 4 に格納されていればそのベクトルデータのハンドルが返され、格納されていなければ、新たなハンドルが生成され奏法合成部 1 0 1 C に返されることになる。

10

#### 【 0 0 8 5 】

次に、処理がステップ S 5 5 に進むと、先に発生した予測ロード要求のうち外れたものがキャンセルされる。ここで、「キャンセル」の具体的内容については後述する。次に、処理がステップ S 5 6 に進むと、予測制御部 4 1 における予測条件が変更される。すなわち、奏法合成部 1 0 1 C から供給されたパケットに応じて、図 2 4 の各状態における予測ロードの候補が絞りこまれ、あるいは予測制御部 4 1 における状態が遷移される。

#### 【 0 0 8 6 】

このパケット受信処理は、奏法合成部 1 0 1 C 側から見れば図 2 8 に示す「Get Vector 指令」になる。すなわち、奏法合成部 1 0 1 C が波形合成部 1 0 1 D に対してパケットを供給する動作は、該パケットに含まれるベクトル ID に基づいて「ベクトルデータを得よ」という意味の Get Vector 指令を送信することに他ならない。このベクトルデータは後にベクトルローダ 2 0 において読み出されるまでに準備しなければならないが、その時点において多数のベクトルデータの中から所要のベクトルデータを特定するためにハンドルが必要なのである。

20

#### 【 0 0 8 7 】

（ 4 ）キャッシュメモリ 4 4 内のデータ構造

（ 4 . 1 ）キャッシュページ（図 2 7）

キャッシュメモリ 4 4 に格納されるベクトルデータのうち同時に使用されるものは、パッキングされ、1 ファイル（あるいはベクトルデータの総数以下の複数ファイル）に変換される。この変換は、ユーザの指示あるいは作成された曲データに応じて、演奏に先立って自動的に実行される。その例を図 2 7 に示す。図において、コードブック 2 6 から読み出されるベクトルデータのうち指定ロードされるものは、時刻情報が含まれている。従って、同時に使用される複数のベクトルデータを抽出することができる。コードブック 2 6 から読み出された各ベクトルデータは個々にヘッダを有している。同時に使用されるベクトルデータは、例えば 1 つのファイルにまとめられ、このファイルに対して共通のヘッダが付与される。これにより、ベクトルローダ 2 0 が各ベクトルデータにアクセスする時間を短縮することができる。

30

#### 【 0 0 8 8 】

このファイルのヘッダには、以下のような情報が記憶される。

40

データ ID：ここには、ファイルの種類を識別する 4 バイトの識別文字「PACK」が記憶される。

Data Size：ファイルのデータサイズを示す

VQ Type：記憶されているベクトルデータの種別を示す。

Version：ファイルフォーマットのバージョンを示す。

なお、パッキングされる前のベクトルデータについても同様のヘッダが設けられているが、データ ID は「PACK」ではなく、個々のベクトルデータの種別を識別するデータ ID が付与される。

#### 【 0 0 8 9 】

本実施形態の波形生成装置は、パーソナルコンピュータ上のアプリケーションプログラム

50

によって実現されており、そのキャッシュはシステムメモリ上に存在する。ベクトルデータをキャッシュに記憶する場合、一つのベクトルデータ単位（サイズが不均一）で制御するのではなく、所定サイズのキャッシュページを単位としてキャッシュ制御が行われる。つまり、一つのキャッシュデータは複数のキャッシュページに分割され、キャッシュページ単位でキャッシュに記憶され管理される。また、システムメモリには、各キャッシュページに対応するキャッシュ管理データ（ページヘッダ）が記憶されている。なお、ハードディスクへのデータ記憶は、通常、固定サイズのクラスタ単位で行われているので、キャッシュページのサイズは、そのサイズと同じか、あるいはその整数倍にするとよい。ベクトルデータの容量等を考慮すると、キャッシュページのサイズは1k～10kバイトにすることが好適である。

10

#### 【0090】

##### （4.2）ページヘッダのデータ構造

次に、各キャッシュページに付与されているヘッダのデータ構造について説明する。各ヘッダは、「VDDLCSPAGE」なる構造体によって構成されており、該構造体のメンバは以下の通りである。

`dwPage`：当該キャッシュページに対して一意に付与されるページ番号である。

`dwID`：このキャッシュページに含まれるベクトルデータのベクトルIDである。

`dwSize`：このキャッシュページのデータサイズである。

`dwCount`：プリフェッチ回数を示す。このメンバ`dwCount`は、指定ロードによってプリフェッチされる毎に「1」ずつインクリメントされ、ベクトルロード20によって読み出される毎に「1」ずつデクリメントされる。

20

`dwStatus`：当該キャッシュページの状態を示す。キャッシュページの状態は、`FREE`、`ALLOCATED`、`USED`、`FILLED`、`LOCKED`のうち何れかである。

`lpBuf`：当該キャッシュページにおけるベクトルデータの実体（ヘッダ以外の部分）の先頭アドレスを指標するポインタである。

`lpForward / lpBackward`：本実施形態においては、複数のVDDLCSPAGE構造体によって双方向リンクリストが形成される。メンバ`lpForward`は、リンクの前方向に存在する他のVDDLCSPAGE構造体へのポインタであり、メンバ`lpBackward`は、リンクの後方向に存在する他のVDDLCSPAGE構造体へのポインタである。

30

`lpNext`：上述したように、波形生成装置を専用のハードウェア上で実現する場合等においては、同時の使用される複数のベクトルデータが複数のキャッシュページに分割される。これら複数のキャッシュページは「グループ」を構成しており、メンバ`lpNext`は同一グループに属する他のVDDLCSPAGE構造体を順次指標してゆくポインタである。

#### 【0091】

ここで、上記メンバ`dwCount`の役割について述べておく。例えば、一連のパケットストリームにおいて、同一のベクトルデータが2回使用されるならば、該ベクトルデータについて2回の指定ロード要求が発生することになる。しかし、ベクトルロード20によって1回目に読出された際に該キャッシュページを`USED`状態に設定してしまうと、2回目の読出しができなくなる。そこで、該キャッシュページがあと何回読み出されるのかをカウントすることにより、かかる不具合を防止したものである。

40

ところで、ベクトルデータが複数回用いられる場合としては、同一のイベントデータに対する波形合成処理において同一のベクトルデータが複数回用いられる場合と、複数のイベントデータに対する波形合成処理において同一のベクトルデータが複数回用いられる場合とが考えられる。本実施形態においては、何れの場合においてもメンバ`dwCount`によって残りの使用回数が判別されるから、キャッシュメモリ44内のキャッシュページに対して統一的な取扱が可能になる。

#### 【0092】

50

## ( 4 . 3 ) ページヘッダのリンク構造 ( 図 2 9 )

次に、ポインタ `l p F o r w a r d` とポインタ `l p B a c k w a r d` によって実現される双方向リンクドリストの構造を図 2 9 に示す。本実施形態においては、データの並べ替えや、追加 / 削除を自由にかつ高速に実行するため、図示のようなリスト構造を採用した。図において A - 1 は双方向リンクドリストの先頭に位置するページヘッダであり、その先頭は所定のポインタ `l p T o p` によって指標され、末尾はポインタ `l p T a i l` によって指標されている。A - 2 , A - 3 はページヘッダ A - 1 と同一のグループに属するページヘッダである。すなわち、この「グループ」は、一つのファイルが分割されて形成された個々のキャッシュページのデータに相当する。ページヘッダ A - 2 の先頭アドレスはページヘッダ A - 1 のメンバ `l p N e x t` によって指標され、ページヘッダ A - 3 の先頭アドレスはページヘッダ A - 2 のメンバ `l p N e x t` によって指標されている。

10

## 【 0 0 9 3 】

また、B - 1 はページヘッダ A - 1 の次段にリンクされているページヘッダであり、ページヘッダ B - 1 の先頭アドレスはページヘッダ A - 1 のメンバ `l p F o r w a r d` によって指標されている。そして、このページヘッダ B - 1 のメンバ `l p N e x t` によって、同一グループに属するページヘッダ B - 2 の先頭アドレスが指標されている。また、ページヘッダ B - 1 のメンバ `l p F o r w a r d` によって、次段にリンクされているページヘッダ C - 1 の先頭アドレスが指標され、ページヘッダ C - 1 , C - 2 のメンバ `l p N e x t` によって、各々ページヘッダ C - 2 , C - 3 の先頭アドレスが指標されている。ここで、F R E E 状態のキャッシュページも 1 グループを構成するため、例えばグループ A , B は F I L L E D 状態、グループ C は F R E E 状態に設定される。新しいベクトルデータをキャッシングしようとするときは、F R E E 状態あるいは U S E D 状態のグループからキャッシュページが取得され、そのキャッシュページが新たなベクトルデータのキャッシュページに設定されることになる。

20

## 【 0 0 9 4 】

また、システムメモリ上に新たなキャッシュページおよび対応するページヘッダを作成し、これを新たなベクトルデータのキャッシュページにしてもよい。但し、その場合には、キャッシュページの総量を指定値に制限し、メモリ資源を消費しすぎないようにしておくことが好適である。この指定値は、システムメモリ量に応じて自動設定してもよく、ユーザが手動設定してもよい。また、F R E E 状態のグループのメモリ量としてある一定値を常に確保するように制御すると、新たなベクトルデータのキャッシュページへの割当を高速化することができる。

30

## 【 0 0 9 5 】

図 2 9 の双方向リンクドリストによれば、メンバ `l p F o r w a r d` を参照してゆくことにより、リストを構成する各グループの先頭のページヘッダ A - 1 , B - 1 , C - 1 を順番に辿ることができる。また、メンバ `l p B a c k w a r d` を参照してゆくことにより、該双方向リンクドリストを逆方向に辿ることができる。従って、あるベクトルデータがキャッシュされているか否かを調査する場合は、リストを構成する先頭のページヘッダを順次辿ってゆき、ページヘッダ内のメンバ `d w I D` が該ベクトルデータと同一のベクトル I D と一致するか否かを判定してゆくことがよい。もし一致するものが存在すれば、そのページヘッダに係るキャッシュページが、該ベクトルデータをキャッシングしているグループの先頭のキャッシュページになる。なお、同時に使用される全ベクトルデータが必ず 1 ページに格納される場合には、上記双方向リンクドリストはページヘッダ A - 1 , B - 1 , C - 1 のみによって構成されることになり、これらページヘッダのメンバ `l p N e x t` には、ヌルデータが格納されることになる。

40

## 【 0 0 9 6 】

なお、上記メンバ `d w S t a t u s` で示される各状態を重要度順に配列すると、「L O C K E D > F I L L E D > A L L O C A T E D > U S E D > F R E E」の順になる。そこで、この重要度順でキャッシュページの順序を設定しておけば、新たなベクトルデータに対するキャッシュページの割当を高速化することができる。さらに、メンバ `d w C o u n t`

50

の値をそのまま重要度に反映させてもよい。すなわち、メンバ `d w C o u n t` が大きいほど重要度を高くするとよい。また、`U S E D` 状態のキャッシュページ間では、先に `U S E D` 状態に遷移したものの重要度がより低くなるように設定しておく为好適である。

#### 【0097】

(5) 読出し制御部 43 における動作

次に、読出し制御部 43 の動作を再び図 28 を参照し説明する。

ベクトルローダ 20 においては、パケットストリームの内容すなわち各パケット内のベクトル ID に基づいて、キャッシュメモリ 44 を読み出すように読出し制御部 43 にコマンドが送信される。このコマンドを `L o c k V e c t o r` 指令と呼ぶ。この `L o c k V e c t o r` 指令には、先に `G e t V e c t o r` 指令に応じて返したハンドルが付随されている。通常動作状態であれば、該ベクトルデータはキャッシュメモリ 44 内に格納されている筈であるから、該ベクトルデータに係るキャッシュページの先頭アドレスのポインタがベクトルローダ 20 に返される。

10

#### 【0098】

これにより、ベクトルローダ 20 およびベクトルデコーダ 31 ~ 35 によって、キャッシュメモリ 44 の内容が適宜読み出され、波形合成処理が実行される。このように、ベクトルローダ 20 等によって読み出され得るキャッシュページは、`L O C K E D` 状態(詳細は後述する)に設定され、他の処理によって内容が改変されないよう設定されている。

#### 【0099】

ところで、状況によっては、ベクトルローダ 20 から `L o c k V e c t o r` 指令を受信した際にキャッシュメモリ 44 上に必要なベクトルデータが未だロードされていない場合がある。特に本実施形態の波形生成装置をパーソナルコンピュータのアプリケーションプログラムによって実現した場合には、パーソナルコンピュータのオペレーティングシステムの都合によって比較的長時間ハードディスク 109 が占有されることがあり、かかる状況がしばしば発生し得る。この場合には、波形生成装置の動作モードによって異なる処理が実行される。

20

#### 【0100】

まず、ノンリアルタイムで波形合成を行っている場合は、指定されたベクトルデータがロードされるまで以降の処理を停止することが好適である。そこで、読出し制御部 43 においては、ハードディスク 109 に対する同期読出しが実行される。この同期読出しは、所望のデータが読み出されるまで他の処理を行わないことを意味する。

30

#### 【0101】

一方、波形生成装置がリアルタイムで動作している場合には、同期読出しを実行することは楽音が途切れることになりかねないため、代替ページのポインタがベクトルローダ 20 に返される。本実施形態においては、様々な音色変化を忠実に表現するために多種類のベクトルデータを用いているが、プログラムチェンジで選択された音色について、奏法表現等を伴わない基本的な発音を行うためのベクトルに限定すれば、その容量はあまり大きくない。そのようなベクトル(複数)が、その音色のデフォルトのベクトルとして予め `R A M 103` 内に格納されており、キャッシュメモリ 44 内に準備できなかったベクトルデータの代替として用いられる。

40

#### 【0102】

なお、リアルタイム合成を行う場合であっても、波形合成処理 `S 14` (波形合成部 101D)における時間遅延が充分確保されており、その範囲内で必要なベクトルのロードおよび波形合成が可能な場合は、ノンリアルタイムで波形合成を行う場合と同様に、必要なベクトルデータがロードされていないことが判明した段階で、該ベクトルデータを緊急に読み出して波形合成を行うことができる。ベクトルローダ 20 およびベクトルデコーダ 31 ~ 35 による使用が終了したキャッシュページについては、ベクトルローダ 20 から読出し制御部 43 にハンドルを伴った `R e l e a s e` 指令が供給される。この `R e l e a s e` 指令が供給されると、該キャッシュページの `L O C K E D` 状態が解消され、ハンドルが解放される。

50

## 【 0 1 0 3 】

## ( 6 ) キャッシュページに対する状態遷移動作

次に、図 3 0 の状態遷移図を参照し、各キャッシュページの状態を遷移させてゆく動作について説明する。

まず、初期状態においては、全キャッシュページは F R E E 状態 ( S 6 1 ) に設定される。すなわち該キャッシュページのヘッダのメンバ d w S t a t u s が ' F R E E ' に設定される。これは、未使用状態であって、内容が保証されていないことを意味する。このキャッシュページに対して、ベクトルデータのロードが行われる場合は、該キャッシュページが A L L O C A T E D 状態 ( S 6 2 ) に設定される。A L L O C A T E D 状態は、未だデータの格納は完了していないが、コードブック 2 6 からのデータ読出しのために予約されていることを意味する。その際、メンバ d w C o u n t に「 1 」が格納される。

10

## 【 0 1 0 4 】

その後、ベクトルデータが該ページに格納されると、キャッシュページの状態は F I L L E D 状態に遷移する。A L L O C A T E D 状態または F I L L E D 状態において、同一のベクトルデータについて再び指定ロード要求が発生すると、その度にメンバ d w C o u n t が「 1 」づつインクリメントされる。また、A L L O C A T E D 状態または F I L L E D 状態において、予測ロード要求がキャンセルされると ( ロード処理 ( 図 2 5 ) のステップ S 5 5 )、メンバ d w C o u n t が「 1 」づつデクリメントされる。そして、これらの状態においてメンバ d w C o u n t の値が「 0 」になると、該キャッシュページは U S E D 状態 ( S 6 3 ) に設定される。

20

## 【 0 1 0 5 】

キャッシュページが F I L L E D 状態に遷移した後、ベクトルローダ 2 0 から L o c k V e c t o r 指令が供給されると、該キャッシュページが L O C K E D 状態 ( S 6 5 ) に設定される。そして、ベクトルローダ 2 0 およびベクトルデコーダ 3 1 ~ 3 5 による当該キャッシュページの使用が終了し R e l e a s e 指令が供給されると、L O C K E D 状態が解消され ( U n L o c k され )、該キャッシュページが再び F I L L E D 状態 ( S 6 4 ) に設定される。その際、該キャッシュページのメンバ d w C o u n t が「 1 」だけデクリメントされる。その際、メンバ d w C o u n t が「 0 」になった場合には、該キャッシュページの状態は直ちに U S E D 状態 ( S 6 3 ) に設定される。

## 【 0 1 0 6 】

また、キャッシュページが U S E D 状態に設定された後、該キャッシュページに対して再び予測ロード要求または指定ロード要求 ( プリフェッチ ) が発生すると、該キャッシュページは再び F I L L E D 状態に戻される。以上のような処理が続行すると、やがてキャッシュメモリ 4 4 内に U S E D 状態のキャッシュページが増加してゆき、F R E E 状態のキャッシュページが減少してゆく。そして、F R E E 状態のキャッシュページが無くなった後に、新たなベクトルデータについてプリフェッチが発生すると、U S E D 状態であるキャッシュページのうち最も古く U S E D 状態になったものから順にキャンセル ( S w a p O u t ) され、F R E E 状態 ( S 6 1 ) に戻される。そして、この F R E E 状態に戻されたキャッシュページが、上記新たなベクトルデータのロードのために A L L O C A T E D 状態に設定される。

30

40

## 【 0 1 0 7 】

ここで、U S E D 状態であるキャッシュページのうち最も古いものを速やかに特定する方法として、先に図 2 9 において説明した双方向リンクドリストを用いるとよい。すなわち、何れかのキャッシュページが U S E D 状態に設定されると、リンクドリストの先頭に該キャッシュページを追加し、F R E E 状態のキャッシュページが不足する場合には、リンクドリストの末尾のキャッシュページから順次リンクドリストから外し、F R E E 状態に変更してゆくとよい。

## 【 0 1 0 8 】

## ( 7 ) 時刻管理部 4 5

時刻管理部 4 5 は波形合成部 1 0 1 D 全体のタイミングを制御する。ここで、本実施形態

50

におけるタイミング制御の概要を図 3 1 を参照し説明する。

図において、時刻  $t_{30}$  に再生開始が指示された（例えばユーザにより再生ボタンが押下された）こととし、実際に楽音出力が開始される時刻を  $t_{40}$  とする。この時刻  $t_{30}$  から  $t_{40}$  に至るまでの時間をレーテンシ と呼ぶ。レーテンシ は、例えば  $2000\text{ msec}$  程度に設定されるが、特に設定しなくてもよい。

#### 【0109】

再生開始が指示されると、CPU 101 において再生スレッドが所定時間毎に生成される。この再生スレッドにおいては曲データが読み出され、その内容に応じてハードディスク 109 等に対するプリフェッチ処理が実行される。ある曲データが読み出される時刻  $t_{32}$  から実際の発音開始時刻  $t_{40}$  までの時間を先送り時間 と呼ぶ。先送り時間 の基準値は例えば  $4000\text{ msec}$  程度にしておき、処理状況に応じて  $1000 \sim 10000\text{ msec}$  程度の範囲で伸縮できるようにしておくことよい。

10

#### 【0110】

このように、先送り時間 をある程度確保しておくことにより、再生スレッドを断続的に発生させ曲データの読出しを断続的に（ある程度づつまとめて）行うことが可能になる。再生スレッドが発生する間隔を再生スレッド発生間隔 と呼ぶ。再生スレッド発生間隔の基準値は例えば  $20\text{ msec}$  にしておき、処理状況に応じて  $5 \sim 100\text{ msec}$  程度の範囲で伸縮できるようにしておくことよい。

#### 【0111】

時刻  $t_{32}$  に曲データが読み出されると、上述したようにステップ S 11 において曲データの一部が取り出され、ステップ S 12 において楽譜が解釈され、しかる後にステップ S 13 において奏法合成が実行される。この結果、時刻  $t_{34}$  においてベクトルデータのプリフェッチが行われる。プリフェッチ時刻  $t_{34}$  から発音開始時刻  $t_{40}$  までの時間をプリフェッチ時間 と呼ぶ。

20

#### 【0112】

ハードディスク 109 から読み出されたベクトルデータは、やがてキャッシュメモリ 44 に書き込まれる。その後、時刻  $t_{36}$  において、ベクトルロード 20 によって読出し制御部 43 を介してキャッシュメモリ 44 から該ベクトルデータが読み出され、波形合成が開始される。この波形合成開始時刻  $t_{36}$  から発音開始時刻  $t_{40}$  までの時間を出力レーテンシ と呼ぶ。出力レーテンシ の基準値は例えば  $300\text{ msec}$  程度にしておき、処理状況に応じて  $10 \sim 1000\text{ msec}$  程度の範囲で伸縮できるようにしておくことよい。

30

#### 【0113】

なお、かかる波形合成処理についても、再生スレッドの場合と同様に、ある程度の時間範囲をまとめて断続的に実行すると好適である。この波形合成処理の起動頻度間隔の基準値は例えば  $50\text{ msec}$  にしておき、処理状況に応じて  $10 \sim 500\text{ msec}$  程度の範囲で伸縮できるようにしておくことよい。

#### 【0114】

図 3 1 から明らかなように、先送り時間 、プリフェッチ時間 および出力レーテンシ には、「 $>$ 」の関係がある。ここで、プリフェッチ時刻  $t_{34}$  から波形合成開始時刻  $t_{36}$  までの時間間隔「 $-$ 」は、ハードディスク 109 からベクトルデータをロードできる程度に設定しておく必要がある。ベクトルデータのロード量のピーク値が高い場合には、この時間間隔「 $-$ 」を長くすることにより、ノイズの発生等を防止することができる。

40

#### 【0115】

### 4. 変形例

本発明は上述した実施形態に限定されるものではなく、例えば以下のように種々の変形が可能である。

（１）上述したような波形生成装置を電子楽器に用いた場合、電子楽器は鍵盤楽器の形態に限らず、弦楽器や管楽器、あるいは打楽器等のようなタイプの形態でもよい。また、その場合に、曲データ再生部 101 A、楽譜解釈部 101 B、奏法合成部 101 C、波形

50

合成部 101D等を1つの電子楽器本体内に内蔵したものに限らず、それぞれが別々に構成され、MIDIインタフェースや各種ネットワーク等の通信手段を用いて各構成部を接続するように構成されたものにも同様に適用できることはいうまでもない。また、パソコンとアプリケーションソフトウェアという構成であってもよく、この場合処理プログラムを磁気ディスク、光ディスクあるいは半導体メモリ等の記録媒体に格納して供給したり、ネットワークを介して供給するものであってもよい。さらに、自動演奏ピアノのような自動演奏装置などにも適用してよい。

【0116】

(2) 上記実施形態においては、一つのキャッシュページに複数のベクトルデータを格納したが、一つのキャッシュページに一つのベクトルデータを格納するようにしても良いことはいうまでもない。

10

【0117】

(3) 上記実施形態においては、キャッシュページのヘッダのメンバ `d w C o u n t` をインクリメントあるいはデクリメントすることによって、`U S E D` 状態に遷移させるものを判定した。しかし、かかる方法では、未使用状態のキャッシュページが増大した時にキャッシュメモリ 44 が容量不足になる可能性がある。かかる場合には、`F I L L E D` 状態であるキャッシュページに対して「優先度」を定義し、優先度の低い `F I L L E D` 状態のキャッシュページを順次 `U S E D` 状態そして `F R E E` 状態に遷移させてもよい。この場合、`F R E E` 状態に遷移させたキャッシュページに対してベクトルロード 20 から `L o c k V e c t o r` 指令を受信すると、上述した代替ページが使用されることになる。「優先度」

20

【0118】

(4) 上記実施形態においては、`U S E D` 状態であるキャッシュページを双方向リンクドリストによって結合し、`F R E E` 状態のキャッシュページが不足する場合には、リンクドリストの末尾のキャッシュページから（最も古いものから）順次外してゆき、`F R E E` 状態に変更した。しかし、`F R E E` 状態に遷移させる順序はこれに限定されるものではない。例えば、使用された累積回数の多いキャッシュページ、メンバ `d w C o u n t` の履歴上の最大値の大きいキャッシュページ、あるいは各モジュールの先頭部分において使用されるベクトルデータについては、なるべく `F R E E` 状態に遷移しないように、換言すればキャッシュメモリ 44 内に優先的に残すように設定してもよい。また、実際に使用された使用済みのベクトルデータを、実際には使用されなかった（予測が外れた）ベクトルデータよりも優先的に残すようにしてもよい。

30

【0119】

また、モジュールの先頭部分で使用されるベクトルデータを優先的に残しておくことにより、ハードディスク 109 の動作に対する余裕が大きくなり、代替ページを使わざるを得なくなるような事態を減らすことができる。

具体的には、優先的に残すべきキャッシュページを `U S E D` 状態に遷移させる時は当該キャッシュページをリンクドリストの先頭に追加し、それ以外のキャッシュページを `U S E D` 状態に遷移させる時は該キャッシュページをリンクドリストの中間部分に挿入するようにするとよい。

40

【0120】

(5) 上記実施形態においては、音データの具体例としてベクトルデータを用いた例を示したが、該音データはベクトルデータに限定されるものではなく、楽音波形を規定する種々の波形データあるいはパラメータを音データとして用いてもよいことはいうまでもない。

【0121】

(6) 上記実施形態においては、図 15 ないし図 23 において説明したように、奏法合成部 101C からのパケットストリームが先読みプリフェッチ部 42 に直接入力された。し

50

かし、かかる動作に代えて、一旦パケットキューバッファ 2 1 ~ 2 5 に書き込まれたパケットストリームを先読みプリフェッチ部 4 2 が読み出すようにしてもよい。その場合、先読みプリフェッチ部 4 2 は、ベクトルロード 2 0 がパケットキューバッファに記憶されたパケットを読み出して処理するタイミングよりも所定時間前のタイミングで、同パケットキューバッファから同パケットを読み出して処理することが好適である。

【 0 1 2 2 】

【発明の効果】

以上説明したように本発明によれば、音色指定あるいは第 1 ~ 第 3 音データ指定を受信すると、それ以降に指定される音データの候補を予測して予め低速記憶装置から高速記憶装置に転送するから、低速記憶装置に大量の音データを蓄積しつつ第 1 ~ 第 3 音データ指定 10 に対して迅速に対応することができる。これにより、表情豊かな波形合成を行うための音データをリアルタイムで再生することができる。

【図面の簡単な説明】

【図 1】この発明に係る波形生成装置のハードウェア構成例を示すブロック図。

【図 2】波形生成装置において実行される「波形データベース作成処理」の一実施形態を示すフローチャート。

【図 3】奏法モジュールに対応する実波形区間を構成する各成分及び要素の一例を模式的に示す図。

【図 4】「データベースに基づく楽音合成処理」の一実施形態を示すフローチャート。

【図 5】図 4 と同様の波形合成処理を専用ハードウェアの形態で構成した場合の一実施形態を示すブロック図。 20

【図 6】上述した奏法合成部における奏法合成処理の流れを説明するためのブロック図。

【図 7】奏法合成部で行われる奏法合成処理の一実施形態を詳細に示したフローチャート。

【図 8】奏法モジュールが振幅要素又はピッチ要素に対応する場合におけるリンク処理を説明するための概念図。

【図 9】アタック波形とボディ波形とを接続した場合の波形の間引きを説明するための概念図。

【図 1 0】ボディ波形とリリース波形とを接続した場合の波形の間引きを説明するための概念図。 30

【図 1 1】ベンドアタック波形とリリース波形とを接続した場合の波形の間引きを説明するための概念図。

【図 1 2】ノーマルアタック波形とループ部を有するリリース波形とを接続した場合の波形の間引きを説明するための概念図。

【図 1 3】後に続く奏法モジュールが開始されるよりも前の奏法モジュールが終了してしまう場合の波形接続について説明するための概念図。

【図 1 4】奏法合成部で生成されるパケットストリームを説明するための概念図。

【図 1 5】波形合成部における動作を説明するために全体構成の一実施形態を示した概念図。

【図 1 6】波形合成の全体の流れを簡単に示すブロック図。 40

【図 1 7】ベクトルロードを説明するためのブロック図。

【図 1 8】ベクトルオペレータを説明するためのブロック図。

【図 1 9】ベクトルレコーダを説明するためのブロック図。

【図 2 0】ベクトルデータのデータ構造の一実施形態を概念的に示す概念図。

【図 2 1】楽譜解釈部 1 0 1 B において作成される演奏データの内容を示す図。

【図 2 2】奏法合成部 1 0 1 C から波形合成部 1 0 1 D に各パケットが供給されるタイミングを示す図。

【図 2 3】キャッシュ制御部 4 0 の全体構成を示すブロック図。

【図 2 4】予測制御部 4 1 の予測動作の状態遷移図。

【図 2 5】先読みプリフェッチ部 4 2 において実行されるロード処理のフローチャート。 50

【図 26】先読みプリフェッチ部 42 において実行されるパケット受信処理のフローチャート。

【図 27】キャッシュページの構成方法を示す動作説明図。

【図 28】奏法合成部 101C および波形合成部 101D 間のシグナルフロー図。

【図 29】キャッシュメモリ 44 内におけるページヘッダのリンク構造を示す図。

【図 30】キャッシュメモリ 44 内における各ページの状態遷移図。

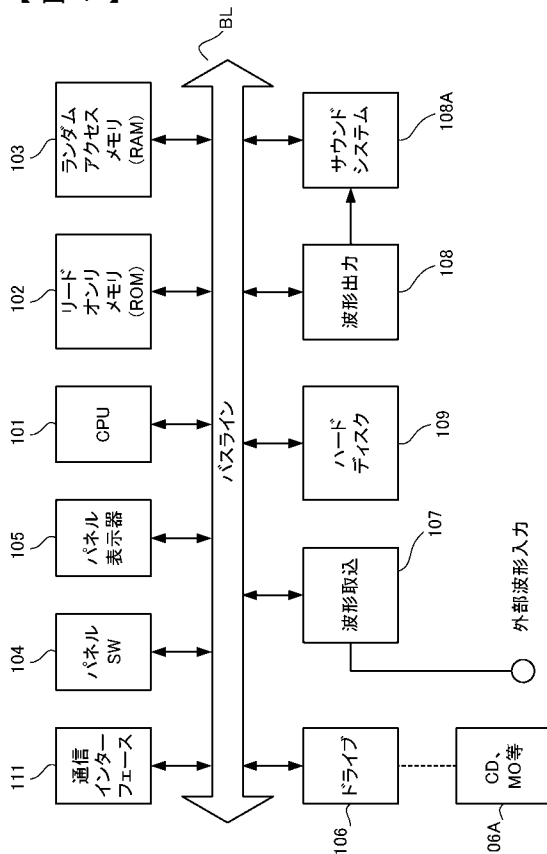
【図 31】上記実施形態のタイミング制御の概要を示すタイミングチャート。

【符号の説明】

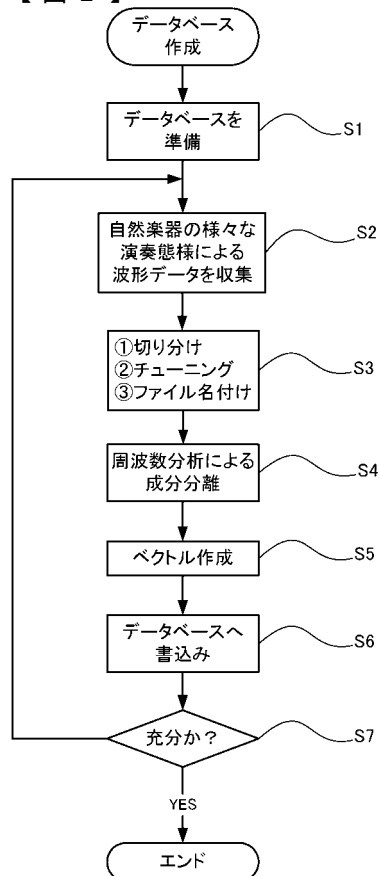
1 ~ 12 ..... データ、20 ..... ベクトルローダ、21 ~ 25 ..... パケットキューバッファ、26 ..... コードブック、31 ~ 35 ..... ベクトルデコーダ、36, 37 ..... ベクトルオペレータ、38 ..... ミキサ、40 ..... キャッシュ制御部、41 ..... 予測制御部、42 ..... 先読みプリフェッチ部、43 ..... 読出し制御部、44 ..... キャッシュメモリ（高速記憶装置）、45 ..... 時刻管理部、50 ..... アタック部奏法モジュール、101 ..... CPU、101A ..... 曲データ再生部、101B ..... 楽譜解釈部、101C ..... 奏法合成部、101D ..... 波形合成部、102 ..... ROM、103 ..... RAM、104 ..... パネルスイッチ、105 ..... パネル表示器、106 ..... ドライブ、106A ..... 前記ドライブ、106B ..... 外部記憶メディア、107 ..... 波形取込部、108 ..... 波形出力部、108A ..... サウンドシステム、109 ..... ハードディスク、110 ..... 通信インターフェース、111 ..... ハードディスク（低速記憶装置）、112 ..... 通信インターフェース。

10

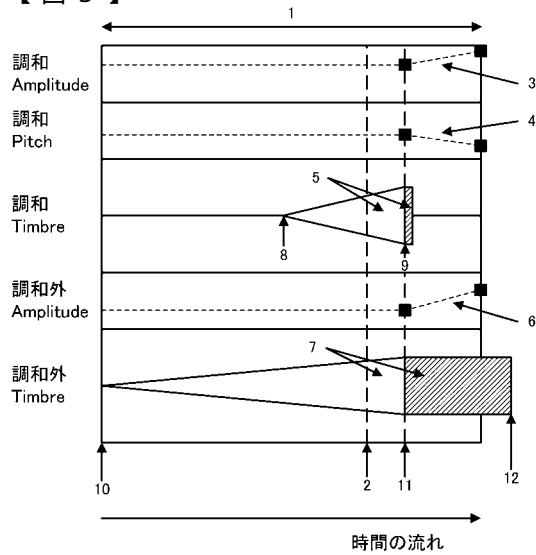
【図 1】



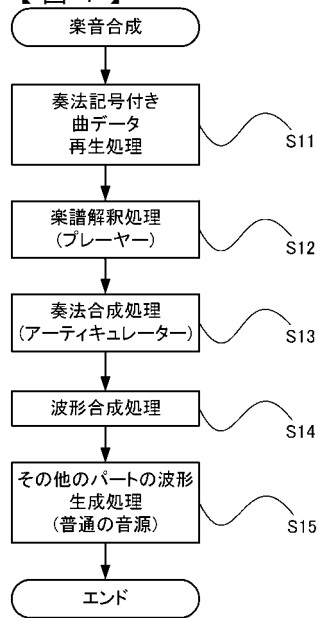
【図 2】



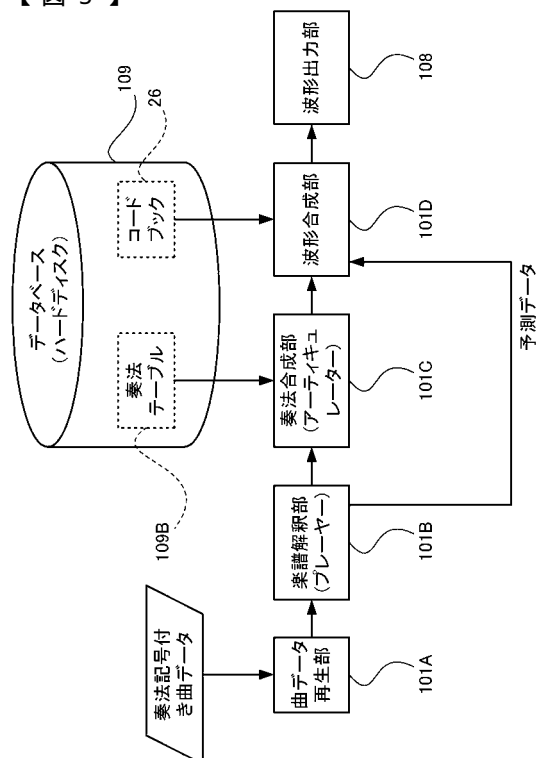
【図 3】



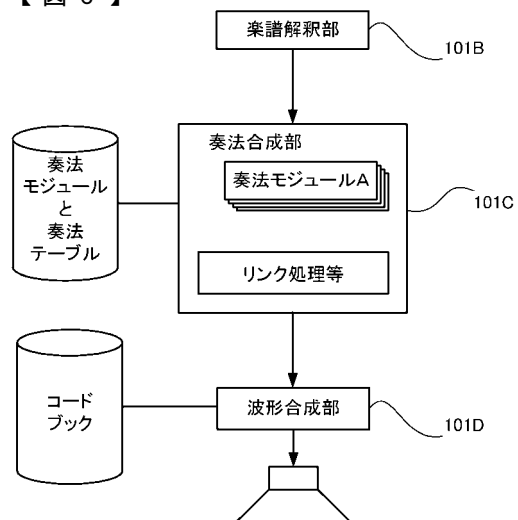
【図 4】

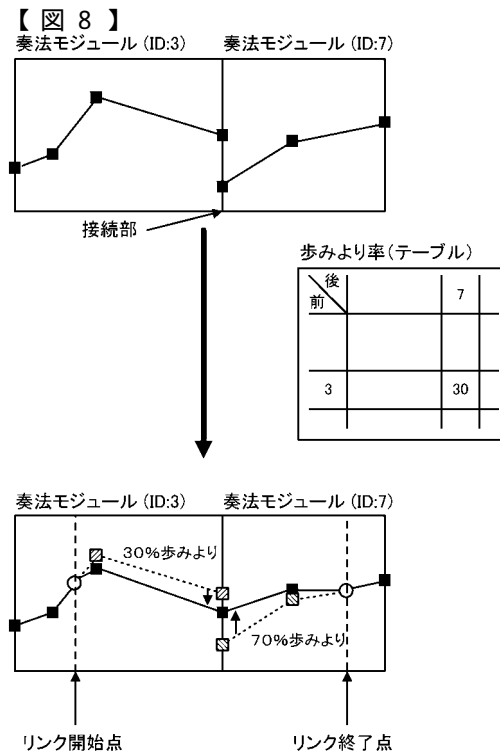
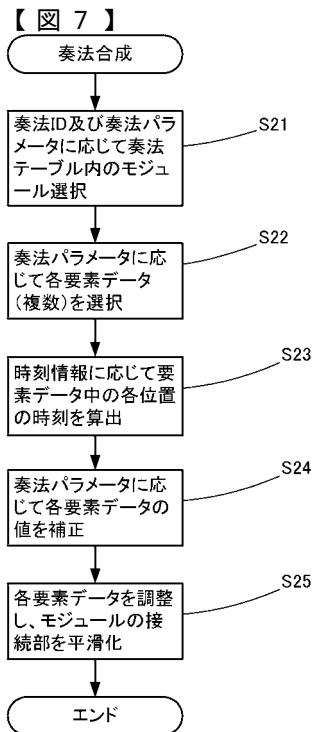


【図 5】



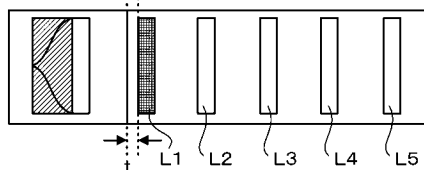
【図 6】





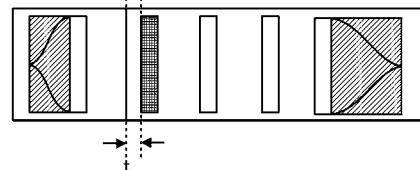
【図 9】

アタック部波形 ボディ部波形



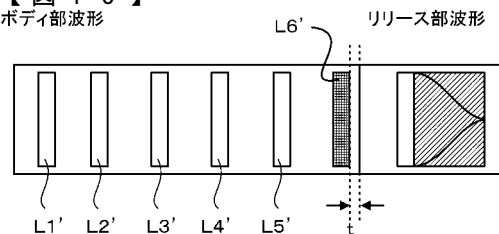
【図 12】

アタック部波形 リリース部波形



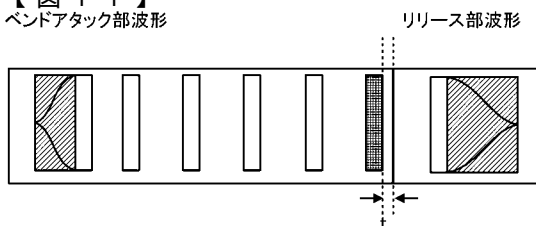
【図 10】

ボディ部波形

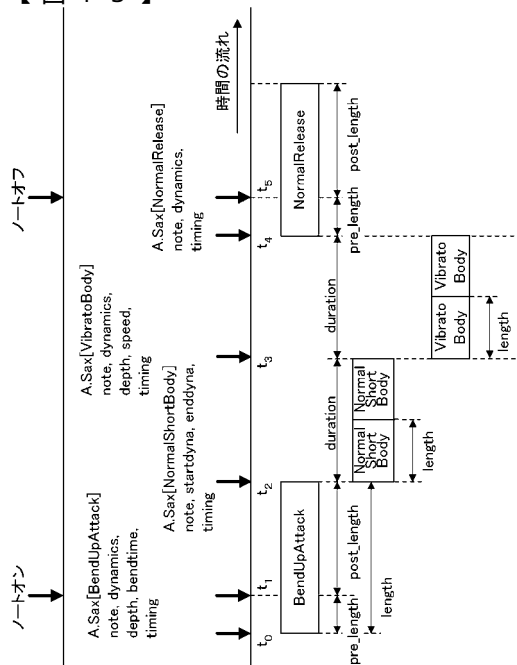


【図 11】

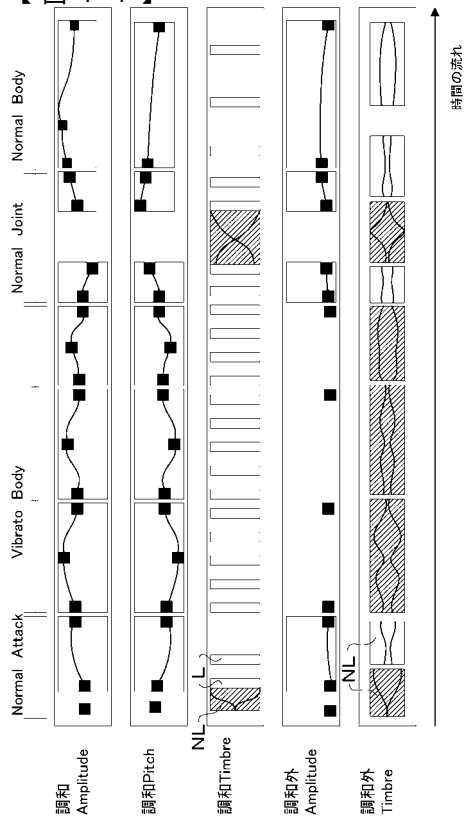
バンドアタック部波形



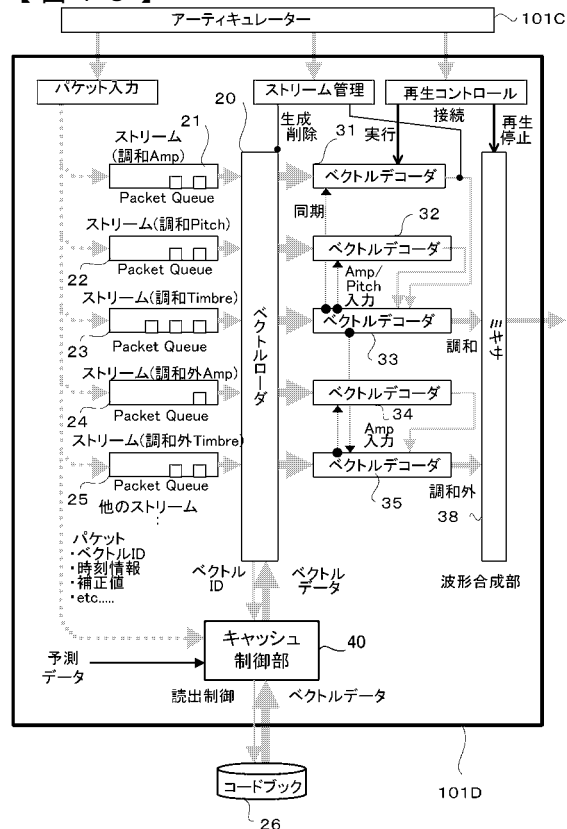
【 ㊦ 1 3 】



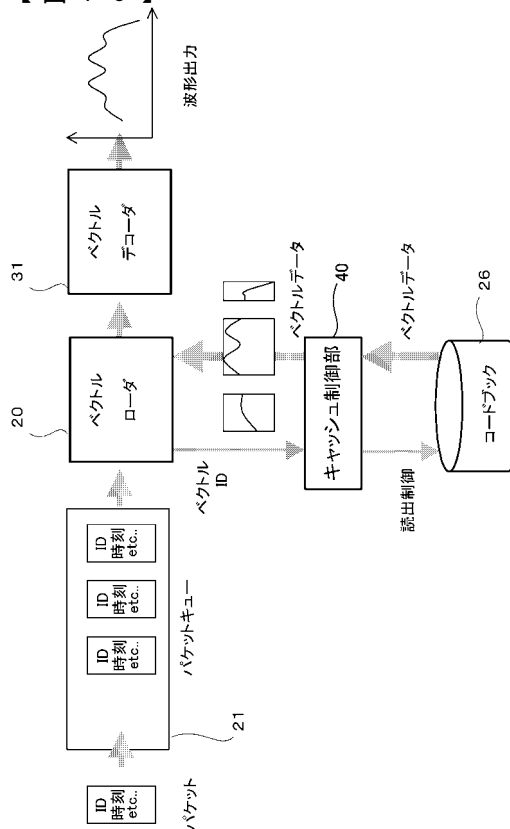
【 ㊦ 1 4 】



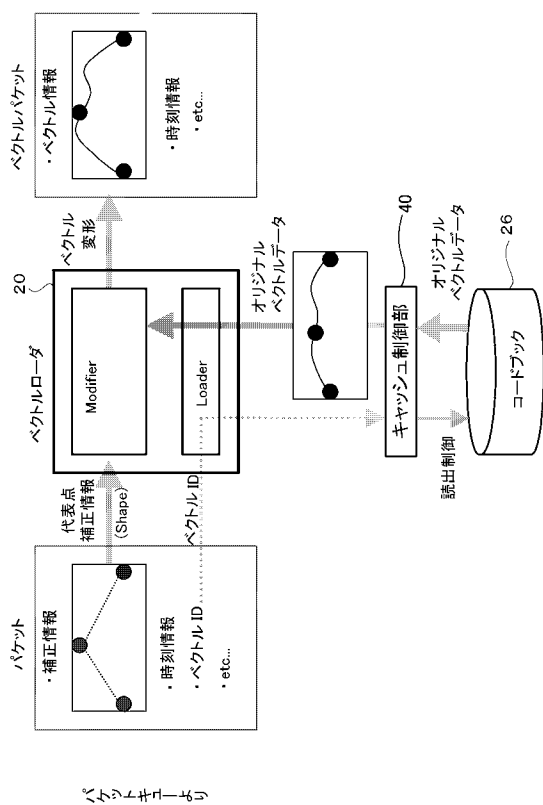
【 図 1 5 】



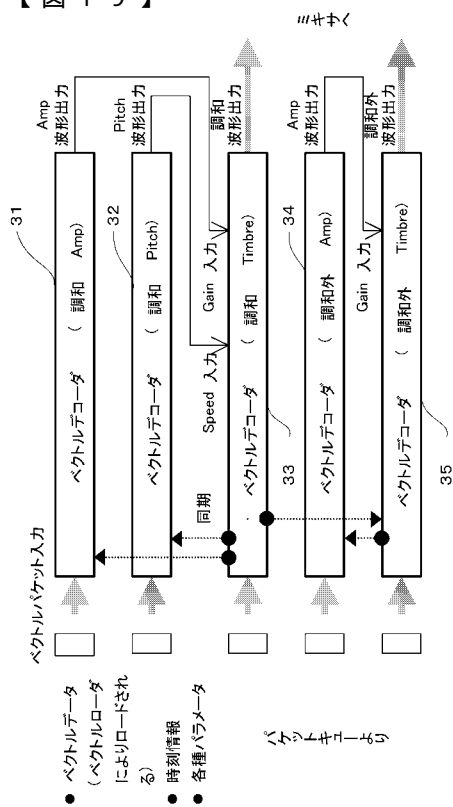
【 図 1 6 】



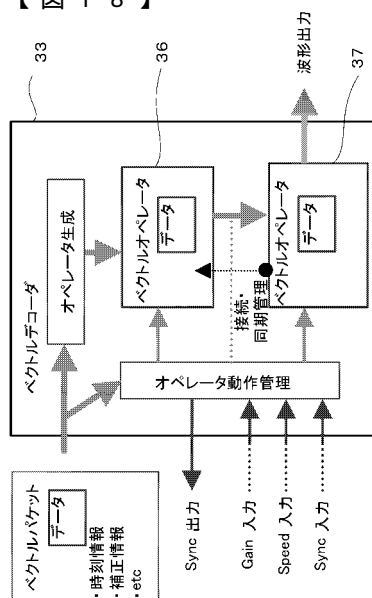
【 図 1 7 】  
くろくしつにけいけい



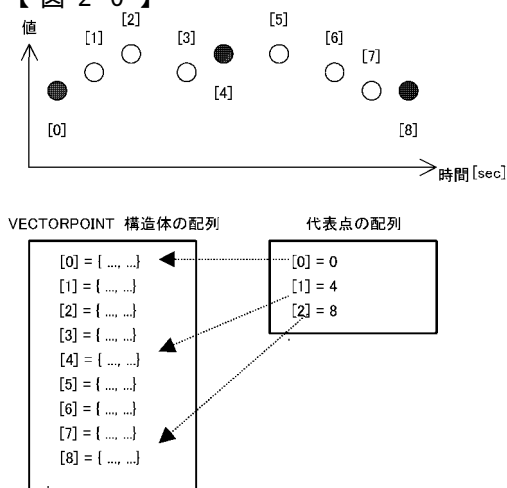
【 図 1 9 】



【 図 1 8 】

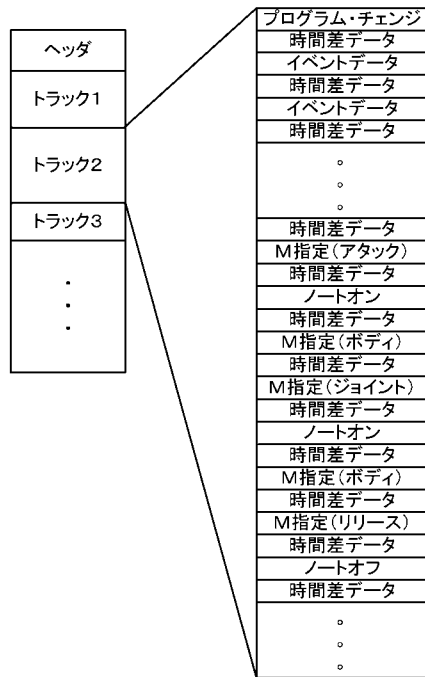


【 図 2 0 】

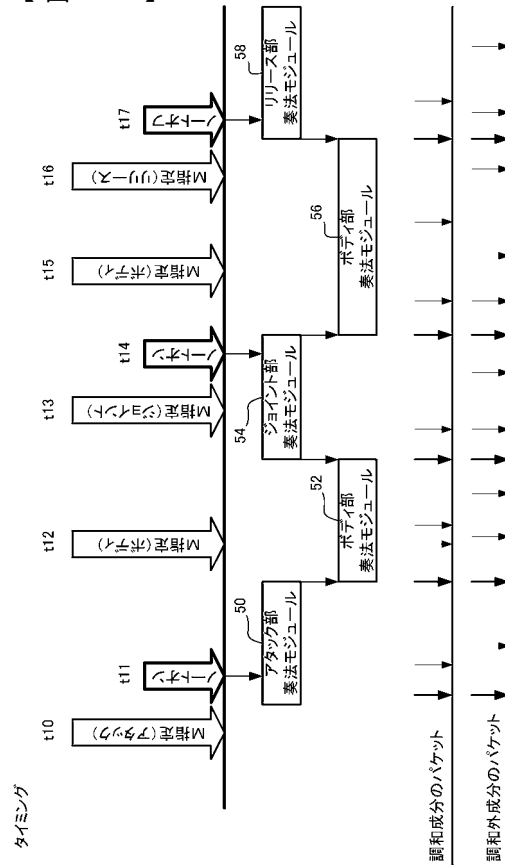


【図 2 1】

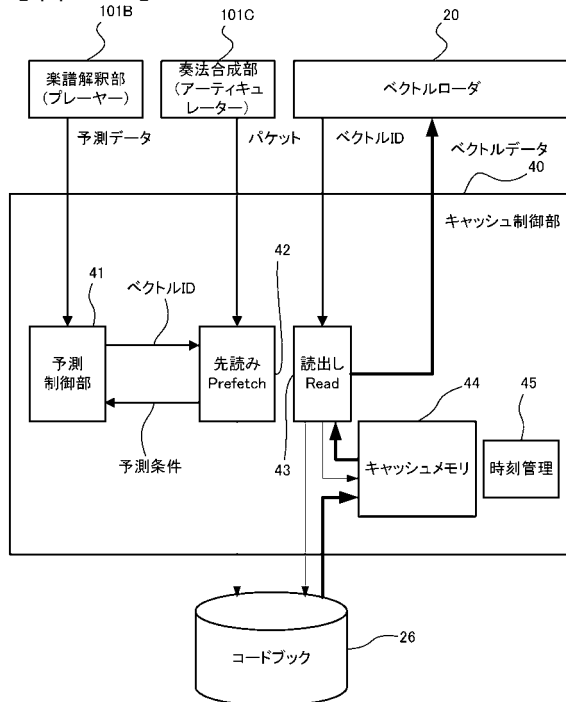
「演奏解釈処理(プレーヤー)」の処理済みの演奏データ



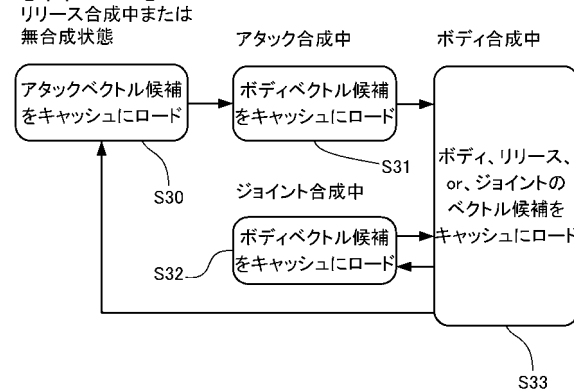
【図 2 2】



【図 2 3】

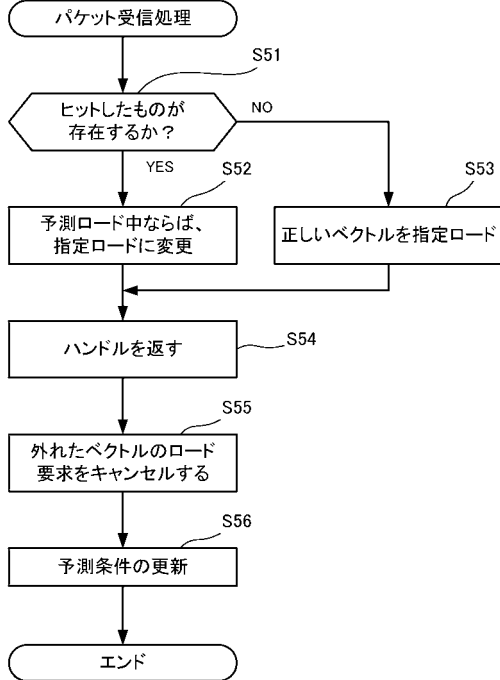


【図 2 4】

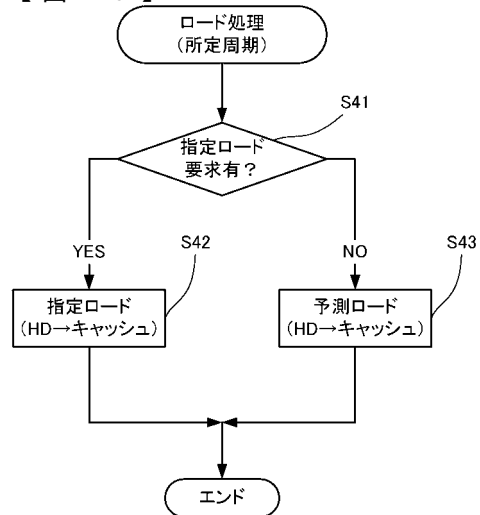


【図 25】

(Get Vector)

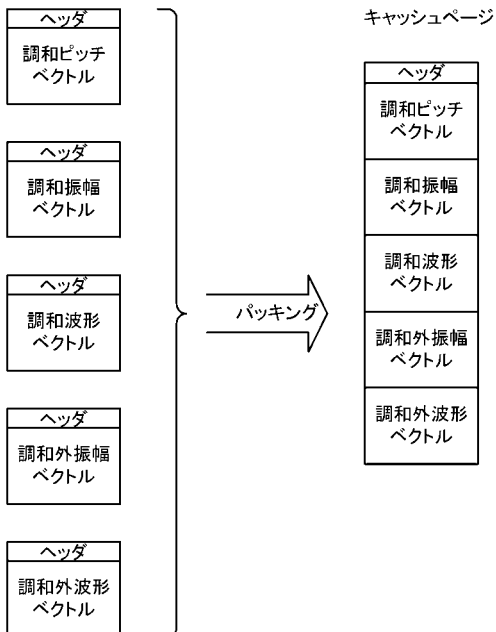


【図 26】



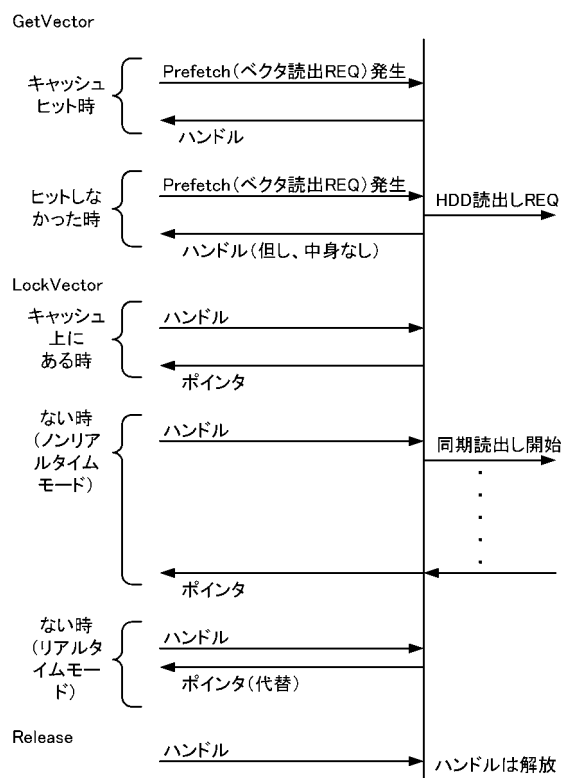
【図 27】

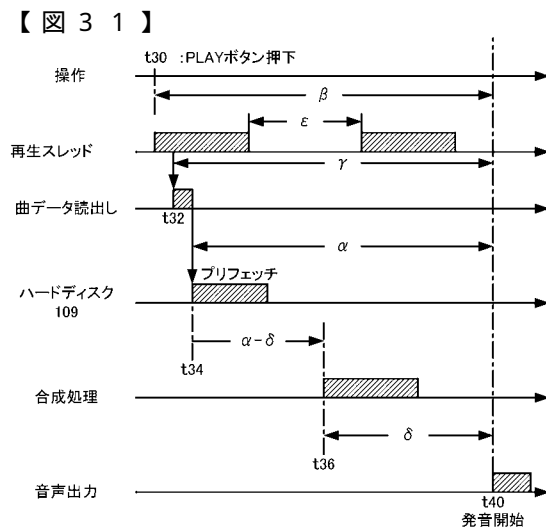
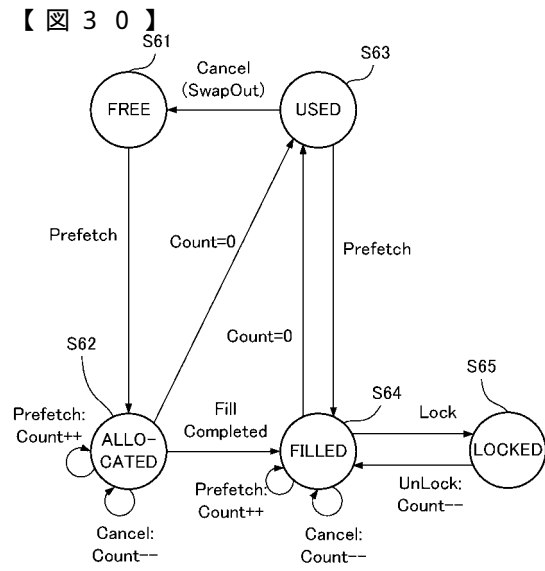
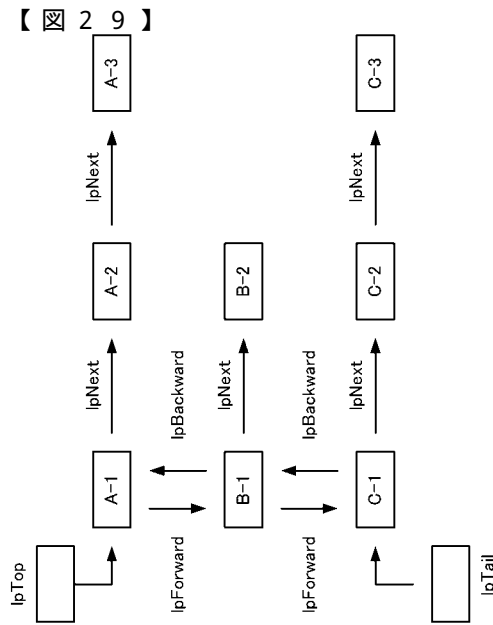
個々のベクトルデータ



【図 28】

キャッシュ管理





---

フロントページの続き

- (56)参考文献 特開平06-308964(JP,A)  
特開2000-276172(JP,A)  
特開昭58-221898(JP,A)  
特開平05-273980(JP,A)  
特公昭64-001800(JP,B1)  
特開平09-319373(JP,A)

- (58)調査した分野(Int.Cl.<sup>7</sup>, DB名)  
G10H 1/00 - 7/12