(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3725247号
(P3725247)
(45)発行日　平成17年12月7日(2005.12.7)　(24)登録日　平成17年9月30日(2005.9.30)

(51)Int.Cl.[7]	F I
G10H 1/02	G10H 1/02
G10H 7/02	G10H 7/00 521Z

請求項の数 6　(全 18 頁)

(21)出願番号　特願平8-147820
(22)出願日　平成8年5月20日(1996.5.20)
(65)公開番号　特開平9-311681
(43)公開日　平成9年12月2日(1997.12.2)
審査請求日　平成15年4月23日(2003.4.23)

(73)特許権者　000116068
ローランド株式会社
静岡県浜松市細江町中川２０３６番地の１
(74)代理人　100087000
弁理士　上島　淳一
(72)発明者　山中　正
大阪府大阪市北区堂島浜１丁目４番１６号
ローランド株式会社内
(72)発明者　片山　忍
大阪府大阪市北区堂島浜１丁目４番１６号
ローランド株式会社内

審査官　江嶋　清仁

最終頁に続く

(54)【発明の名称】楽音生成処理方法

(57)【特許請求の範囲】

【請求項１】

複数の楽音の生成態様のうちで予め設定された所定の態様で楽音を生成するようにするとともに、サンプリング周期よりも長い所定時間毎に、前記所定時間に含まれる複数のサンプリング時点の楽音のデータを一度にまとめて生成し、時間的な順序付けをして記憶する処理を行う楽音データ生成処理と、

前記楽音データ生成処理により記憶した各サンプリング時点の楽音のデータを、サンプリング周期毎に前記時間的な順序付けに従って順次読み出して出力する処理を行う楽音データ出力処理と、

前記楽音データ生成処理にかかる負荷量を計測によって検出する処理と、

前記検出する処理により検出された負荷量に応じて、前記楽音データ生成処理における楽音の生成態様の設定を変更する処理と

を行うことを特徴とする楽音生成処理方法。

【請求項２】

前記楽音の生成態様の設定を変更する処理は、前記楽音データ生成処理の許容負荷量と前記検出する処理により検出された負荷量とに基づいて、楽音の生成態様の設定を変更する

ことを特徴とする請求項１記載の楽音生成処理方法。

【請求項３】

複数の楽音の生成態様のうちの所定の態様で楽音を生成するようにするとともに、サン

プリング周期よりも長い所定時間毎に、前記所定時間に含まれる複数のサンプリング時点の楽音のデータを一度にまとめて生成し、時間的な順序付けをして記憶する処理を行う楽音データ生成処理と、

前記楽音データ生成処理により記憶した各サンプリング時点の楽音のデータを、サンプリング周期毎に前記時間的な順序付けに従って順次読み出して出力する処理を行う楽音データ出力処理と、

前記楽音データ生成処理にかかる負荷量を自動的に刻々と計測によって検出する処理と、

前記検出する処理により検出された負荷量に応じて、前記楽音データ生成処理における楽音の生成態様の設定を自動的に刻々と変化させる処理と

を行うことを特徴とする楽音生成処理方法。

【請求項４】

前記検出する処理は、前記楽音データ生成処理に費やされる時間を計測し、前記計測した時間と前記楽音データ生成処理の実行時間間隔とに基づいて、前記負荷量を検出する

ことを特徴とする請求項１、２または３のいずれか１項に記載の楽音生成処理方法。

【請求項５】

前記楽音の生成態様を決定する処理により決定される前記楽音の生成態様は、楽音の同時発音数、効果付与手段の効果付与態様あるいはサンプリング周波数の少なくともいずれか１つである

ことを特徴とする請求項１、２または３のいずれか１項に記載の楽音生成処理方法。

【請求項６】

前記楽音の生成態様を決定する処理により決定される前記楽音の生成態様は、楽音の同時発音数、効果付与手段の効果付与態様あるいはサンプリング周波数の少なくともいずれか２つであり、使用者の指示により前記楽音の生成態様のいずれかを優先させ、前記楽音の生成態様を決定する処理は、優先されない前記楽音の生成態様を前記検出する処理により検出された前記負荷量に基づいて決定する

ことを特徴とする請求項１、２または３のいずれか１項に記載の楽音生成処理方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、楽音生成処理方法に関し、さらに詳細には、パーソナルコンピュータやワークステーションなどの汎用のコンピュータ上でソフトウェアによりシンセサイザを構築するための楽音生成処理方法に関する。

【０００２】

【発明の背景および発明が解決しようとする課題】

従来、楽音を生成するためには、楽音生成用の専用のハードウェア、所謂、電子楽器を用いることが必要であった。

【０００３】

しかしながら、近年、汎用のコンピュータに高性能のＣＰＵおよびＣＰＵが処理したデータをアナログ信号として外部へ出力するためのＤ／Ａ変換器（デジタル－アナログ変換器）が搭載されるようになってきたことに伴い、専用のハードウェアを用いることなしに、汎用のコンピュータ上で楽音をリアルタイムに生成することができる可能性が検討されてきている。

【０００４】

具体的には、楽音のデータは正確にサンプリング周期毎にＤ／Ａ変換器に出力しなければならないため、タイマに応じてサンプリング周期毎に楽音のデータを生成する処理を行うことが検討されている。

【０００５】

この処理を実現するためには、サンプリング周期毎に発生するタイマインタラプトに応じて、インタラプトルーチンの中で楽音のデータを生成する処理を行うことが考えられるが

、楽音のデータを生成する処理は複雑であり、こうした複雑な処理をインタラプトルーチンの中で行うのは困難であるという問題点が指摘されていた。

【０００６】

あるいはまた、上記した処理を実現するためには、サンプリング周期毎に起動されるタスクで楽音のデータを生成する処理を行うことも考えられるが、これによると、サンプリング周期毎という複雑なタイミングでタスクを切り換える処理を行う必要があり、タスク切り換えのためのオーバーヘッドが増大することになり、ＣＰＵが実際に処理することができる時間が短くなってしまうという問題点が指摘されていた。

【０００７】

ところで、一般に、一つのソフトウェアを様々な汎用のコンピュータで使用することが可能であるが、これら様々な汎用のコンピュータにそれぞれ搭載されているＣＰＵの能力は区々であり、低い能力のＣＰＵに合わせてソフトウェアを作成すると、能力の高いＣＰＵが搭載されたコンピュータではＣＰＵの能力を使いきることができないという問題点があった。

【０００８】

また、汎用のコンピュータにおいては、複数のソフトウェアを同時に実行させるという使用方法が一般的に行われているが、楽音を生成する処理とともに、楽音を生成する処理以外の他の処理も同時に行う場合には、他の処理の処理状況に応じてＣＰＵの負荷は刻々と変化することとなる。このため、ＣＰＵの負荷が最も重い状態に合わせてソフトウェアを作成すると、負荷が軽い場合にＣＰＵの能力を使いきることができないという問題点があった。

【０００９】

さらに、一般に、ＣＰＵによる楽音を生成する処理の少なくとも一部は、一定の時間間隔で繰り返し行われることになるが、ノートオン（発音開始）／ノートオフ（発音終了）を示す指示は、使用者（ユーザー）の所望のタイミングに応じてこの繰り返し間隔とは非同期で発生する。即ち、ノートオン／ノートオフを示す指示に応じて楽音を生成する処理にデータを受け渡し、データを受け渡された楽音を生成する処理においては、受け渡されたデータに基づいて楽音を生成する処理を行うこととなるが、ノートオン／ノートオフを示す指示は上記繰り返し間隔とは非同期であるため、楽音を生成する処理をしている最中に、ノートオン／ノートオフを示す指示に応じてデータを書き換えてしまったり、あるいはノートオン／ノートオフを示す指示に応じてデータを書き換えている最中に楽音を生成する処理を行うと、生成中の楽音にノイズが生じるという問題点があり、データの受け渡し方法を改良する必要性が指摘されていた。

【００１０】

本発明は、上記したような種々の問題点に鑑みてなされたものであり、その目的とするところは、汎用のコンピュータに適した処理を行うことにより上記した種々の問題点の解決を図り、汎用のコンピュータ上で楽音をリアルタイムで効率的に生成することのできる楽音生成処理方法を提供しようとするものである。

【００１１】

【課題を解決するための手段】

上記目的を達成するために、本発明による楽音生成処理方法は、複数の楽音の生成態様のうちで予め設定された所定の態様で楽音を生成するようにするとともに、サンプリング周期よりも長い所定時間毎に、上記所定時間に含まれる複数のサンプリング時点の楽音のデータを一度にまとめて生成し、時間的な順序付けをして記憶する処理を行う楽音データ生成処理と、上記楽音データ生成処理により記憶した各サンプリング時点の楽音のデータを、サンプリング周期毎に上記時間的な順序付けに従って順次読み出して出力する処理を行う楽音データ出力処理と、上記楽音データ生成処理にかかる負荷量を計測によって検出する処理と、上記検出する処理により検出された負荷量に応じて、上記楽音データ生成処理における楽音の生成態様の設定を変更する処理とを行うようにしたものである。

【００１５】

ここで、上記楽音の生成態様の設定を変更する処理は、上記楽音データ生成処理の許容負荷量と上記検出する処理により検出された負荷量とに基づいて、楽音の生成態様の設定を変更するものとすることができる。

【００１６】

さらに、本発明による楽音生成処理方法は、複数の楽音の生成態様のうちの所定の態様で楽音を生成するようにするとともに、サンプリング周期よりも長い所定時間毎に、上記所定時間に含まれる複数のサンプリング時点の楽音のデータを一度にまとめて生成し、時間的な順序付けをして記憶する処理を行う楽音データ生成処理と、上記楽音データ生成処理により記憶した各サンプリング時点の楽音のデータを、サンプリング周期毎に上記時間的な順序付けに従って順次読み出して出力する処理を行う楽音データ出力処理と、上記楽音データ生成処理にかかる負荷量を自動的に刻々と計測によって検出する処理と、上記検出する処理により検出された負荷量に応じて、上記楽音データ生成処理における楽音の生成態様の設定を自動的に刻々と変化させる処理とを行うようにしたものである。

【００１７】

ここで、上記検出する処理は、上記楽音データ生成処理に費やされる時間を計測し、上記計測した時間と上記楽音データ生成処理の実行時間間隔とに基づいて、上記負荷量を検出するものとすることができる。

【００１８】

また、上記楽音の生成態様を決定する処理により決定される上記楽音の生成態様は、楽音の同時発音数、効果付与手段の効果付与態様あるいはサンプリング周波数の少なくともいずれか１つであるとすることができる。

【００１９】

また、上記楽音の生成態様を決定する処理により決定される上記楽音の生成態様は、楽音の同時発音数、効果付与手段の効果付与態様あるいはサンプリング周波数の少なくともいずれか２つであり、使用者の指示により上記楽音の生成態様のいずれかを優先させ、上記楽音の生成態様を決定する処理は、優先されない上記楽音の生成態様を上記検出する処理により検出された上記負荷量に基づいて決定するものとすることができる。

【００２３】

【発明の実施の形態】

以下、添付の図面を参照しながら、本発明の楽音生成処理方法の実施の形態を詳細に説明する。

【００２４】

図１は、本発明の楽音生成処理方法によるプログラムを搭載可能な、典型的な汎用のコンピュータのハードウェア構成を示すブロック図である。

【００２５】

この図１に示す汎用のコンピュータは、極めて一般的な構成を備えており、バス１０を介して、ＣＰＵ１２と、メモリ１４と、ディスクコントローラ１６と、ビデオカード１８と、Ｉ／Ｏカード２０と、サウンドカード２２とが接続されており、さらに、ディスクコントローラ１６にはハードディスク２４が接続され、ビデオカード１８にはディスプレイ装置２６が接続され、Ｉ／Ｏカード２０にはキーボード２８およびマウス３０が接続されている。そして、このコンピュータにおいては、サウンドカード２２から楽音信号が出力される。

【００２６】

また、この汎用のコンピュータにおいては、本発明の楽音生成処理方法によるプログラムの他に、シーケンサ（自動演奏）のプログラムも同時に起動され、シーケンサのプログラムから本発明の楽音生成処理方法によるプログラムへＭＩＤＩ情報が引き渡され、本発明の楽音生成処理方法によるプログラムにおいては、引き渡されたＭＩＤＩ情報に応じて楽音信号を生成して出力する。

【００２７】

図２は、本発明の楽音生成処理方法によるプログラムの全体構成を示し、制御タスク、音

源タスクおよび出力タスクより構成されている。
【００２８】
即ち、制御タスクは制御プログラムを実行し、シーケンサのプログラムとの間でＭＩＤＩ情報のやりとりを行うとともに、音源タスク側へイベントフラグ、発音パラメータ、定常処理パラメータ、エフェクトパラメータを引き渡し、音源タスク側からボイス解放情報を受け取る。
【００２９】
音源タスクは、複数のボイス（楽音生成チャンネル）ならびにエフェクトに関する処理を行い、ウェーブバッファへ生成した楽音信号を記憶させて出力タスクへ引き渡す。即ち、複数のボイスのそれぞれの処理を行い、後述する発音パラメータのエフェクト送り量に応じて各ボイスの出力をエフェクトへ引き渡し、エフェクトを付加されていない各ボイスの出力とエフェクトを付加された各ボイスの出力とを最終的にミキシングしてウェーブバッファへ記憶し、出力タスクへ引き渡すことになる。
【００３０】
出力タスクにおいては、ウェーブバッファを用いてウェーブ出力処理を行い、外部へ楽音信号を出力する。そして、出力タスクにおいて使用されたウェーブバッファは、音源タスク側へ戻される。
【００３１】
上記したように、制御タスクと音源タスクとがやりとりする情報には、イベントフラグ、発音パラメータ、定常処理パラメータ、エフェクトパラメータおよびボイス解放情報がある。以下、これらの情報に関して説明する。
【００３２】
（１）イベントフラグ
イベントフラグはボイス数分用意されていて、音源タスクは制御タスクからイベントフラグにより各ボイスの発音開始および発音終了の情報（フラグ）を受ける。イベントフラグは、制御タスクと音源タスクとの両方からアクセス可能な共有メモリに記憶される。
【００３３】
（２）発音パラメータ
後述するように、音源タスクは起動される度にイベントフラグの内容をサーチし、発音開始フラグを見つけると、発音パラメータに従って発音を開始するが、発音パラメータには、ボイス毎に以下の（ａ）乃至（ｉ）に示す情報が含まれている。
（ａ）波形読み出し関係パラメータ
（波形開始アドレス、ループ関係アドレス、ピッチ）
（ｂ）ピッチＬＦＯ関係パラメータ
（波形、周期、遅延など）
（ｃ）フィルタＬＦＯ関係パラメータ
（波形、周期、遅延など）
（ｄ）アンプＬＦＯ関係パラメータ
（波形、周期、遅延など）
（ｅ）ピッチエンベロープ関係パラメータ
（アタック、ディケイ、サスティン、リリースなどの各時間やレベル）
（ｆ）ＴＶＦ（Ｔｉｍｅ　Ｖａｒｉａｂｌｅ　Ｆｉｌｔｅｒ：時変フィルタ）エンベロープ関係パラメータ
（アタック、ディケイ、サスティン、リリースなどの各時間やレベル）
（ｇ）ＴＶＡ（Ｔｉｍｅ　Ｖａｒｉａｂｌｅ　Ａｍｐｌｉｆｉｅｒ：時変アンプ）エンベロープ関係パラメータ
（アタック、ディケイ、サスティン、リリースなどの各時間やレベル）
（ｈ）パン
（ｉ）エフェクト送り量
これらの情報を含む発音パラメータは、制御タスクと音源タスクとの両方からアクセス可

能な共用メモリに記憶される。そして、発音パラメータを記憶するメモリ空間は、ボイス毎に２面づつ設けられている。即ち、音源タスクがイベントフラグに発音開始のフラグを見つけると、音源タスクはこれ以後に音源タスクが参照するメモリ空間を今まで参照していた面からもう一つの面に切り換えるとともに、今まで参照していた面を制御タスクに渡す。従って、制御タスクは、同じボイス番号の発音パラメータを設定する場合でも、音源タスクから渡された現在処理中の面とは異なるもう一面の新しいメモリ空間を用いることによって、音源タスク側で発音時に参照される現在処理中の面のパラメータエリアを壊すことなしに、音源タスクへ発音パラメータの引き渡し行うことができる。逆に、制御タスク側で書き換えている途中のパラメータエリアを音源タスクが参照するという事態の発生を防ぐこともできる。

【００３４】

（３）定常処理パラメータ

定常処理パラメータは、ボイス毎に以下の（ａ）乃至（ｃ）に示す情報が含まれている。

（ａ）ピッチオフセット

（ｂ）フィルタオフセット

（ｃ）アンプオフセット

これらの定常処理パラメータは、制御タスクと音源タスクとの両方からアクセス可能な共用メモリに記憶され、ピッチベンダやホイルなどの操作子などから得られた情報をサービスするためのものである。

【００３５】

これらの定常処理パラメータは、音源タスクが起動される度に読み込まれ、発音中に音源タスク内部で生成されるエンベロープやＬＦＯなどと合成されることになり、こうして合成されたデータを発音サービスパラメータと称する。

【００３６】

発音サービスパラメータは、波形読み出しのピッチや、ＴＶＦのカットオフ周波数、ＴＶＡのアンプリチュード（増幅度）を制御する。

【００３７】

（４）エフェクトパラメータ

エフェクトパラメータは、制御タスクと音源タスクとの両方からアクセス可能な共用メモリに記憶されたエフェクト制御用の情報であり、エフェクトに渡される。このエフェクトパラメータは、音源タスクが起動される度に読み込まれ、前回読み込んだ内容と異なるパラメータがあれば、それをエフェクトに反映することになる。

【００３８】

なお、エフェクトパラメータは、エフェクトの具体的内容によってパラメータが異なるものであり、エフェクトの具体的内容は本発明の要旨から外れるものであるので、その詳細な説明は省略する。

【００３９】

（５）ボイス開放情報

ボイス開放情報は、音源タスクから制御タスクへ返されるものであり、発音が終了したボイスの番号を示す。なお、各ボイスの発音終了は、ＴＶＡのエンベロープにより判断する。このボイス開放情報はメモリ空間を用いるのではなく、イベントによって制御タスクへ渡される。

【００４０】

ここで、イベントフラグ、発音パラメータ、定常処理パラメータおよびエフェクトパラメータが共有メモリを用いて受け渡しが行われるのに対して、ボイス開放情報がイベントで渡されるというタスク間の通信方法の違いは、音源タスクはタイマにより起動されるため、制御タスク側から非同期なイベント情報を受け取ることができないためである。

ところで、上記した制御タスク、音源タスクおよび出力タスクには優先度（プライオリティ：ｐｒｉｏｒｉｔｙ）が設けられていて、他のタスクなどにより処理を中断されることが好ましくないタスクは、優先度を高くするようになされている。なお、この実施の形態

においては、出力タスクの優先度が最も高く、次に音源タスクの優先度が高く、制御タスクの優先度が最も低くなされている。

【００４１】

以下、図２および図３を参照しながら、制御タスク、音源タスクおよび出力タスクを詳細に説明する。

【００４２】

（１）制御タスク

制御タスクはＭＩＤＩ情報を得て音源タスクを制御するものであり、シーケンサのプログラムからのＭＩＤＩ情報や音源タスクからのボイス解放情報を受け取ることにより起動される。

【００４３】

ここで、制御タスクは、各ボイスに関して、当該ボイスが発音中か空いているかを示すボイス情報を常に管理しているものであり、ＭＩＤＩ情報としてノートオンを受け取ると、ボイス情報を参照して適当な空いているボイスを指定し、当該ボイスを用いて当該ノートオンを発音するように、当該ボイスに対して当該ノートオンに必要な発音パラメータを設定するとともにその後にイベントフラグに発音開始を設定し、指定したボイスのボイス情報を更新する処理を行う。そして、音源タスクにおいては、上記した制御タスクの処理に応じて、指定されたボイスで新たな発音を開始する。

【００４４】

また、制御タスクは、ＭＩＤＩ情報としてノートオンを受け取ったときに、ボイス情報を参照して空きボイスが無い場合には、発音中のレベルが小さいボイスを選択したり、あるいは最も過去に発音開始されたボイスを選択したりするなど所定の選択基準により不必要と思われるボイスを選択し、当該ボイスを用いて当該ノートオンを発音するように、当該ボイスに対して当該ノートオンに必要な発音パラメータを設定するとともにその後にイベントフラグに発音開始を設定する処理を行う。そして、音源タスクにおいては、上記した制御タスクの処理に応じて、すばやく選択されたボイスの現在の発音をミュートしてから、選択されたボイスにより新たな発音を開始する。

【００４５】

上記したように、制御タスクにおいては、ＭＩＤＩ情報としてノートオンを受け取ると、ボイスの割り当て、所謂、アサイン処理を行うことになる。

【００４６】

一方、制御タスクは、ＭＩＤＩ情報としてノートオフを受け取ると、当該ノートオフに対応する発音中のボイスに対して発音を終了するように、イベントフラグに発音終了を設定する処理を行う。そして、音源タスクにおいては、上記した制御タスクの処理に応じて、当該ノートオフに対応する発音中のボイスを消音する。

【００４７】

なお、後述するように、音源タスク中のボイスの処理に関しては、発音していないボイスに対しては行われない。従って、発音ボイス数に応じて、音源タスクの演算量が増減することになる。このため、本発明においては、最大発音可能な発音数は、コンピュータの能力に応じて予め設定できるとともに、適宜変更することができるようになされている。

【００４８】

また、制御タスクは、ＭＩＤＩ情報として定常処理パラメータ、エフェクトパラメータの値を変更するメッセージを受け取ると、対応する定常処理パラメータ、エフェクトパラメータを音源タスクに引き渡す処理を行う。

【００４９】

さらに、制御タスクは、音源タスクから制御タスクへボイス解放情報が渡された場合には、渡されたボイス解放情報に応じて指定したボイスのボイス情報を更新する。

【００５０】

（２）音源タスク

音源タスクは１０ｍｓ（ミリ秒）毎にタイマによって起動され、１０ｍｓ分の波形読み出

し、ＴＶＦ、ＴＶＡ、パン、エンベロープ、ＬＦＯ、エフェクト処理をまとめてなるべく短時間で行う。上記したように、音源タスクは１０ｍｓ分の処理を行うものであるので、サンプリング周波数を３２ＫＨｚとすると、３２０サンプル分の処理を行うことになる。

【００５１】

そして、音源タスクは起動される度にイベントフラグの内容をサーチし、発音開始フラグを見つけると、発音パラメータに従って発音を開始する。この際に、発音パラメータを記憶するメモリ空間は、ボイス毎に２面づつ設けられており、音源タスクがイベントフラグに発音開始のフラグを見つけると、音源タスクはこれ以後に音源タスクが参照するメモリ空間を今まで参照していた面からもう一つの面に切り換えるとともに、今まで参照していた面を制御タスクに渡す。従って、制御タスクは、同じボイス番号の発音パラメータを設定する場合でも、音源タスクから渡された現在処理中の面とは異なるもう一面の新しいメモリ空間を用いることによって、音源タスク側で発音時に参照される現在処理中の面のパラメータエリアを壊すことなしに、音源タスクへ発音パラメータの引き渡し行うことができる。

【００５２】

ここで、各ボイスにおける処理を説明すると、音源タスクにおいては、共有のメモリに記憶される発音パラメータおよび定常処理パラメータは、音源タスクが起動される度に取り込まれ、エンベロープやＬＦＯの演算およびそれに定常処理パラメータを反映し、発音サービスパラメータを生成する演算を行う。こうして生成された発音サービスパラメータにより、波形読み出しのピッチや、ＴＶＦのカットオフ周波数、ＴＶＡのアンプリチュード（増幅度）が制御され、発音パラメータのエフェクト送り量に応じてボイスの出力をエフェクトへ引き渡し、エフェクトを付加されていないボイスの出力は発音パラメータのパンによりパンポットを制御されてステレオ出力される（図３（ａ））。

【００５３】

そして、エフェクトパラメータによって制御されたエフェクトを付加された各ボイスの出力とエフェクトを付加されていない各ボイスの出力とが、ミキシングされてステレオ出力され（図３（ｂ））、最終的にウェーブバッファへ記憶されて出力タスクへ引き渡される。

【００５４】

図４には、発音サービスパラメータの生成の処理を示すフローチャートが示されており、最初のボイスからボイス毎に順次処理が行われる。

【００５５】

まず、イベントフラグが発音開始を示しているかチェックし（ステップＳ４０２）、イベントフラグが発音開始を示していない場合には、イベントフラグが発音終了を示しているかチェックする（ステップＳ４０４）。

【００５６】

ステップＳ４０４でイベントフラグが発音終了を示していない場合には、発音中を示すフラグであるｖｏｉｃｅＡｃｔｉｖｅに「１」がセットされていて、発音中であることを示しているかをチェックする（ステップＳ４０６）。

【００５７】

一方、ステップＳ４０４でイベントフラグが発音終了を示している場合には、それを受けてエンベロープをリリース処理へ進め（ステップＳ４０８）、前回処理の際の発音サービスパラメータの計算値ｎ１をｎ０にセット（ステップＳ４１０）した後に、エンベロープ、ＬＦＯおよび定常処理パラメータを反映させた演算を行い発音サービスパラメータを生成して計算値ｎ１にセットする（ステップＳ４１２）。即ち、前回処理からの１０ｍｓの時間経過に応じてエンベロープ、ＬＦＯの値を変化させ新たなエンベロープ、ＬＦＯおよび定常処理パラメータを合成（加算あるいは乗算）して発音サービスパラメータを生成する。なお、発音サービスパラメータは、波形読み出しのピッチ、ＴＶＦ、ＴＶＡそれぞれについて独立に演算する。

【００５８】

また、ステップＳ４０６でｖｏｉｃｅＡｃｔｉｖｅに「１」がセットされていて、発音中である場合にも、前回処理の際の発音サービスパラメータの計算値ｎ１をｎ０にセット（ステップＳ４１０）した後に、エンベロープ、ＬＦＯおよび定常処理パラメータを反映させた演算を行い発音サービスパラメータを生成して計算値ｎ１にセットする（ステップＳ４１２）。なお、発音サービスパラメータは、波形読み出しのピッチ、ＴＶＦ、ＴＶＡそれぞれについて独立に演算する。
ところで、ステップＳ４０２でイベントフラグが発音開始である場合には、制御タスク、音源タスクの使用する発音パラメータエリアを示すポインタを、もう一面の発音パラメータエリアを示すものに更新し（ステップＳ４１４）、ｖｏｉｃｅＡｃｔｉｖｅに「１」がセットされていて、発音中であることを示しているかをチェックする（ステップＳ４１６）。

【００５９】
ステップＳ４１６でｖｏｉｃｅＡｃｔｉｖｅに「１」がセットされていて、発音中である場合には、現在発音中のボイスを急速にミュートしてから（ステップＳ４１８）、エンベロープ、ＬＦＯおよび定常処理パラメータを反映させた演算を行い発音開始時の発音サービスパラメータを生成して計算値ｎ０にセットする（ステップＳ４２０）。なお、発音サービスパラメータは、波形読み出しのピッチ、ＴＶＦ、ＴＶＡそれぞれについて独立に演算する。

【００６０】
一方、ステップＳ４１６でｖｏｉｃｅＡｃｔｉｖｅに「１」がセットされておらず、発音中でない場合には、そのままステップＳ４２０へ進み、エンベロープ、ＬＦＯおよび定常処理パラメータを反映させた演算を行い発音開始時の発音サービスパラメータを生成して計算値ｎ０にセットする。

【００６１】
上記のようにして、ステップＳ４２０でエンベロープ、ＬＦＯおよび定常処理パラメータを反映させた演算を行い発音開始時の発音サービスパラメータを生成して計算値ｎ０にセットし、それからｖｏｉｃｅＡｃｔｉｖｅに「１」をセットして発音中であることを示し（ステップＳ４２２）た後に、発音開始からの１０ｍｓの時間経過に応じてエンベロープ、ＬＦＯおよび定常処理パラメータを反映させた演算を行い発音サービスパラメータを生成して計算値ｎ１にセットする（ステップＳ４１２）。なお、発音サービスパラメータは、波形読み出しのピッチ、ＴＶＦ、ＴＶＡそれぞれについて独立に演算する。

【００６２】
そして、ステップＳ４１２でエンベロープ、ＬＦＯおよび定常処理パラメータを反映させた演算を行い発音サービスパラメータを生成して計算値ｎ１にセットすると、必要に応じて補間処理を行う際に用いるために、今回の発音サービスパラメータの計算値ｎ１と前回の発音サービスパラメータの計算値ｎ０との差分ｄを求め（ステップＳ４２４）、ＴＶＡのエンベロープが最後まで来ているかをチェックする（ステップＳ４２６）。

【００６３】
ステップＳ４２６でＴＶＡのエンベロープが最後まで来ていない場合には、そのままイベントフラグをクリアし（ステップＳ４２８）、ステップＳ４２６でＴＶＡのエンベロープが最後まで来ている場合には、制御タスクにボイス解放情報を返し（ステップＳ４３０）、ｖｏｉｃｅＡｃｔｉｖｅに「０」をセットして発音中でないことを示してから（ステップＳ４３２）、イベントフラグをクリアする（ステップＳ４２８）。

【００６４】
また、ステップＳ４０６でｖｏｉｃｅＡｃｔｉｖｅに「１」がセットされておらず発音中でない場合には、そのままイベントフラグをクリアする（ステップＳ４２８）。

【００６５】
そして、イベントフラグをクリアすると（ステップＳ４２８）、それから全ボイスの処理が終わったかチェックする（ステップＳ４３４）。ステップＳ４３４でのチェックにより全ボイスの処理が終わったならば、このフローチャートの処理を終了するが、ステップＳ

４３４でのチェックにより全ボイスの処理が終わっていないならば、ステップＳ４０２へ戻り以降の処理を繰り返して次のボイスの処理を行う。即ち、全ボイスの処理が終わるまで、このフローチャートの処理を繰り返すことになる。

【００６６】

なお、発音サービスパラメータを受けて行われる波形読み出し、ＴＶＦ、ＴＶＡ、パンおよびエフェクトへの送り量の処理は、音源タスクが起動される度にこれらの処理をまとめて１０ｍｓ分行い、その後に、エフェクト処理を１０ｍｓ分まとめて行うものであり、図５にはこの処理のフローチャートが示されている。

即ち、図５のフローチャートにおいては、最初のボイスからボイス毎に順次処理が行われる。

【００６７】

まず、ｖｏｉｃｅＡｃｔｉｖｅに「１」がセットされていて、発音中であることを示しているかをチェックし（ステップＳ５０２）、ステップＳ５０２でｖｏｉｃｅＡｃｔｉｖｅに「０」がセットされていて、発音中でない場合には、全ボイスの処理が終わったかをチェックするステップＳ５１４へジャンプして進み、次のボイスの処理へ移る。つまり、このフローチャートにおいては、発音されていないボイスの処理は行わないことになる。

【００６８】

一方、ステップＳ５０２でｖｏｉｃｅＡｃｔｉｖｅに「１」がセットされていて、発音中である場合には、発音サービスパラメータの示すピッチで発音するように波形読み出しを行う（ステップＳ５０４）。ここで、発音サービスパラメータは、図４に示すフローチャートのステップＳ４２４で求めた差分ｄを用いて、必要な精度まで直線補間される。即ち、サンプリング周波数３２ＫＨｚとすると、図４に示すフローチャートのステップＳ４１０あるいはステップＳ４２０で求めた計算値ｎ０にセットされた発音サービスパラメータとステップＳ４２４で求めた差分ｄを用いて、

ｎ０＋ｄ×ｋ／３２０

なる演算を行い補間値とする。なお、３２０はサンプリング周波数３２ＫＨｚ時の１０ｍｓ分のサンプル数であり、ｋは３２０サンプルのうちの何番目のサンプルであるかを示しており、最初の処理では‘０’、以後処理の度に‘１’、‘２’、・・・とされる値である。

【００６９】

次に、ステップＳ５０４の波形読み出しの出力をＴＶＦによりフィルタリングする（ステップＳ５０６）。ここで、ＴＶＦのカットオフ周波数は発音サービスパラメータによって制御され、発音サービスパラメータは、図４に示すフローチャートのステップＳ４２４で求めた差分ｄを用いて、必要な精度に直線補間して用いる。

【００７０】

さらに、ステップＳ５０６のフィルタリングされた出力のアンプリチュードを、ＴＶＡにより発音サービスパラメータに従って制御する（ステップＳ５０８）。ここで、発音サービスパラメータは、図４に示すフローチャートのステップＳ４２４で求めた差分ｄを用いて、必要な精度に直線補間して用いる。

【００７１】

さらに次に、発音パラメータによって、パンおよびエフェクトへの送り量を制御するとともに図３（ｂ）に示される他のボイスの楽音信号との加算混合処理を行い（ステップＳ５１０）、その後に、１０ｍｓ分の処理を終えたかチェックされ（ステップＳ５１２）、１０ｍｓ分の処理を終わっていなければ、ステップＳ５０４へ戻って次のサンプルに関してステップＳ５０４以降の処理を行う。即ち、ステップＳ５０４乃至ステップＳ５１０の処理は、サンプル毎の処理であり、１０ｍｓ分のサンプル個数の処理が行われる。

【００７２】

一方、１０ｍｓ分の処理を終わったならば、全ボイスの処理が終わったかチェックする（ステップＳ５１４）。このチェックにより、全ボイスの処理が終わったならば、エフェクトの処理を１０ｍｓ分まとめて行い（ステップＳ５１６）、上記した処理の行われた１０

ｍｓ分の楽音信号を記憶したウェーブバッファを出力タスクに渡し（ステップＳ５１８）、このフローチャートの処理を終了する。ここで、上記した処理の行われた楽音信号を記憶したウェーブバッファを出力タスクに渡すと同時に、新しいウェーブバッファを受け取り、新しいウェーブバッファは、次回のステップＳ５０２乃至ステップＳ５１６の処理に用いる。

【００７３】

一方、ボイスの処理が終わっていないならば、ステップＳ５０２へ戻り以降の処理を繰り返して次のボイスの処理を行う。即ち、全ボイスの処理が終わるまで、このフローチャートの処理を繰り返すことになる。

【００７４】

（３）出力タスク

出力タスクは、音源タスクより受け取ったウェーブバッファの記憶内容を、オーディオデータ出力デバイス（図１に示すコンピュータにおいては、サウンドカード２２である。）にＤＭＡ転送する。このＤＭＡ転送は、オーディオデータ出力デバイスからの割り込みによって開始される。

【００７５】

ここで、サウンドカード２２などのオーディオデータ出力デバイス（以下、「出力デバイス」と称す。）は、出力するオーディオデータを一時記憶するためのバッファを内蔵しており、サンプリング周期毎に一時記憶しているオーディオデータを１サンプリング分づつ出力する。そして、内蔵バッファに残っているオーディオデータが所定量より少なくなると出力タスクに割り込みをかけ、この割り込みに応じて、出力タスクは出力デバイスが必要とする所定サンプル分のオーディオデータを出力デバイスにＤＭＡ転送するものである。

【００７６】

なお、上記した出力タスクへの割り込みは、通常、約１０ｍｓ程度の周期でかかる。

【００７７】

即ち、図１に示すコンピュータにおいては、出力タスクは、サウンドカード２２からの割り込みと音源タスクからのウェーブバッファを受け取るときに起動される。

【００７８】

そして、出力タスクは、音源タスクから１０ｍｓ分のウェーブバッファを受け取ると、空いているウェーブバッファを音源タスクに返す。ウェーブバッファは４本以上あり、常に１つは出力タスクが出力デバイスに出力を行い、１つは音源タスクが処理に用い、残りは空いているというローテーションを繰り返すようになされている。

【００７９】

以上において説明したように、本発明による楽音処理方法においては、エンベロープやＬＦＯの演算はサンプリング周波数よりも長い時間単位で定期的に行い、その演算結果を用いる際には、所望の精度となるように当該演算結果を補間するようにしている。

【００８０】

これによって、演算に複雑な条件判断を要し、比較的演算ステップ数の多いエンベロープやＬＦＯの演算を軽減することが可能となる。

【００８１】

具体的には、上記した処理においては、エンベロープ、ＬＦＯおよび操作子などによって与えられるような情報を反映する演算は、音源タスクの処理中に一度だけしか行われていない。つまり、音源タスクが１０ｍｓに一度起動されるものとすれば、これらの演算は１０ｍｓに一度しか行われない。そして、こうした演算結果を用いる箇所（波形読み出しピッチ、ＴＶＦ、ＴＶＡなど）では、必要に応じて当該演算結果を補間して用いている。

【００８２】

また、補間は、エンベロープ、ＬＦＯ、操作子などによって与えられる情報を合成した後に、その合成値に対して行っているので、エンベロープ、ＬＦＯ、操作子などの各情報毎に補間を行う場合に比べて処理量が低減されている。

【００８３】
また、本発明による楽音処理方法においては、波形読み出し、補間、エンベロープ、エフェクトなどそれぞれの演算を、サンプリング周波数間隔で処理するのではなくて、処理毎にまとめて長い時間スパンで処理を行うようにしている。
【００８４】
具体的には、１０ｍｓ毎に波形を読み出して、１０ｍｓ分まとめて処理するようにしている。このときに、他の処理に影響しないように、なるべく短時間で処理を完了させるようにしている。
【００８５】
このように、ある一つの処理をまとめた時間で行わせることによって、処理から処理へ移るためのオーバーヘッドを軽減させることができるようになる。
【００８６】
また、処理自体を細かく分けることにより、処理ルーチンを小さく作り、それをループ処理することによって、当該処理ルーチンがＣＰＵ上のキャッシュメモリにのりやすくなり、高速に処理される可能性が高くなるものである。
【００８７】
さらに、処理毎に優先度をつけて、重要な処理は他の処理に妨げられることなく、途切れることの無く処理を行うようにしている。具体的には、出力タスクの優先度が最も高く、次に音源タスクの優先度が高く、制御タスクの優先度が最も低くなされており、出力タスクが他の処理に妨げられることが無いようにしている。このため、出力タスクにより発音中の楽音が、他の処理により途切れることはない。
【００８８】
さらに、本発明による楽音処理方法においては、発音しないボイスの処理を行わないようにした。
【００８９】
これによって、発音数が少ない場合はＣＰＵ１２の処理を軽減できる。即ち、プロセッサ資源を他のプログラムに解放することができるようになる。
【００９０】
さらに、本発明による楽音処理方法においては、音源タスクとそれを制御する制御タスクとの通信をメモリ空間を共有することで実現するようにしている。
【００９１】
即ち、図６に示すように、音源タスクと制御タスクとが共有するメモリ空間は２面（図６におけるメモリ空間Ａおよびメモリ空間Ｂ）設けられていて、それらを交互に用いるようになされている。
【００９２】
具体的には、制御タスク側は発音に必要なパラメータをメモリ空間Ａの発音パラメータエリアに用意し、発音パラメータが用意できた旨をフラグを立てることによって音源タスク側に知らせる。そして、音源タスク側においては、メモリ空間Ａの発音パラメータエリアを取り込むと、もう一面（メモリ空間Ｂ）の発音パラメータエリアを制御タスク側に渡す。制御タスク側は、新たに同じボイスで発音する場合には、新しく渡されたメモリ空間Ｂの発音パラメータエリアに対してパラメータを用意する。
【００９３】
これによって、音源タスク側が発音中で発音パラメータを使用していても、制御タスク側はそれを気にすることなく新しい発音パラメータを設定することができる。
【００９４】
なお、２面のメモリ空間の管理方法として１つのポインタのみを設けるようにした場合には、制御タスクはこのポインタの指し示す片方のメモリ空間に新たなパラメータを記憶するとともに、音源タスクは他方のメモリ空間に記憶されたパラメータを参照する。あるいは、制御タスクがパラメータを書き込むメモリ空間を指し示すポインタと、音源タスクがパラメータを参照するメモリ空間を指し示すポインタとを別々に設けるようにした場合に

は、これらのポインタの指し示すメモリ空間を互いに逆とすればよい。

【００９５】

また、本発明による楽音処理方法においては、ＣＰＵ１２の負荷の具合をみて、発音可能ボイス数およびエフェクトの有無またはサンプリング周波数を決定するようにしてもよい。これにより、ＣＰＵ１２の能力に応じた楽音生成処理のための構成を構築できる。

【００９６】

即ち、本発明の楽音生成処理方法によるプログラムの起動時や、あるいは本発明の楽音生成処理方法によるプログラムを実行している間の使用者により指定される任意のタイミングで、適当な波形を所定のボイス数発音してみて、音源タスクのＣＰＵ消費率を計測することによって、発音可能ボイス数およびエフェクトの有無またはサンプリング周波数を決定する。

【００９７】

なお、音源タスクのＣＰＵ消費率の計測には、内部タイマを用いればよい。この内部タイマは、ハードウェアタイマからの割り込みによって、定期的（例えば、１ｍｓ毎）にインクリメントされるだけのカウンタである。

【００９８】

音源タスクでは、タスクの開始時と終了時に、即ち、発音処理の前と発音処理の終了時に内部タイマの値を読み出してその差を求めることによって、音源タスクによる発音処理に消費された時間を算出できる。ここで、音源タスクは１０ｍｓ毎に呼ばれるので、そのうちどれくらいの時間を音源タスクの処理が消費しているかが、音源タスクのＣＰＵ消費率となる。

【００９９】

音源タスクのＣＰＵ消費率をどのくらい許すか予め決めておいたり、あるいはユーザーが任意に設定できるようにしておく。実際に計測されたＣＰＵ消費率と許容されるＣＰＵ消費率との比較結果に応じて、発音可能ボイス数およびエフェクトの有無またはサンプリング周波数を決定することになる。

【０１００】

即ち、実際のＣＰＵ消費率が許容されるＣＰＵ消費率よりも大きかったら、例えば、発音可能ボイス数を計測時の所定数よりも減らし、実際のＣＰＵ消費率が許容されるＣＰＵ消費率と同じになるようにボイス数を調整する。実際のＣＰＵ消費率が許容されるＣＰＵ消費率よりも小さい場合は、例えば、発音可能ボイス数を計測時の所定数よりも増やし、実際のＣＰＵ消費率が許容されるＣＰＵ消費率と同じになるようにボイス数を調整する。

【０１０１】

ここで、発音可能ボイス数、エフェクトの有無、サンプリング周波数を変更することにより、ＣＰＵ消費率がどの程度変化するかは予め求めることができるので、実際のＣＰＵ消費率と許容されるＣＰＵ消費率とから、これらの発音態様を一度に決めることも可能であるが、発音態様を変化させながら発音を繰り返し、実際のＣＰＵ消費率と許容されるＣＰＵ消費率とが一致するようにすることも可能である。

【０１０２】

また、このとき、サウンドの質（サンプリング周波数やエフェクトの有無）に重点をおくのか、ボイス数に重点をおくのかを、使用者が選択可能とすることができる。この場合には、重点をおいた項目に関しては固定とし、重点がおかれない項目に関してＣＰＵ消費率に応じて変更したり、あるいは、重点をおいた項目に関してはとりあえず固定とし、重点がおかれない項目をＣＰＵ消費率に応じて許容できる範囲内で変更してみて、それでも実際のＣＰＵ消費率が許容できるＣＰＵ消費率を越える場合には、重点をおいた項目に関しても変更するようにすることができる。

【０１０３】

また、実際に計測されたＣＰＵ消費率をディスプレイに表示し、使用者に知らせるようにしてもよい。

【０１０４】

なお、可能ボイス数の変更の処理は、以下のようにして行えばよい。即ち、制御タスクにおいて、各ボイス毎にそのボイスの使用を許可するか許可しないかを示す許可情報を管理するようにする。そして、制御タスクは、ＭＩＤＩ情報としてノートオンを受け取り、そのノートオンをアサインするボイスを選択するときに、許可情報を参照して使用が許可されているボイスのなかから選択するようにする。

【０１０５】

このため、全てのボイスの使用が許可されていれば、最大の発音数となる。発音可能なボイス数を減らす場合には、許可情報を書き換え、使用するボイス数を減らすようにする。

【０１０６】

従って、使用が許可されていないボイスは発音せず、音源タスクでは発音していないボイスに関しては処理が行われないので、発音可能なボイス数を減らすことにより、ＣＰＵ消費率を下げることができる。

【０１０７】

このように、アサイン処理を行う制御タスクにおいて、ボイスの使用を許可するか許可しないかを示す許可情報を管理するようにし、この許可情報を書き換えるだけという簡単な処理で、発音可能ボイス数を調整することができる。

【０１０８】

また、サンプリング周波数は、出力デバイス（図１のコンピュータにおけるサウンドカード）のオーディオデータの出力周期と、音源タスクの１回の処理時の処理サンプル数を調整することによって変化することができる。

【０１０９】

さらに、エフェクト付加の有無は、音源処理の図５のステップＳ５１６の処理を行うかあるいは行わないかを、切り換えにより選択できるようにすることにより実現できる。

【０１１０】

また、上記したように、本発明の楽音生成処理方法によるプログラムの起動時や、あるいは本発明の楽音生成処理方法によるプログラムを実行している間の使用者により指定される任意のタイミングで、適当な波形を所定のボイス数発音してみて、音源タスクのＣＰＵ消費率を計測することによって、発音可能ボイス数およびエフェクトの有無またはサンプリング周波数を決定するようにした場合には、本発明の楽音生成処理方法によるプログラムの動作時にはそれに見合った処理負荷量（ボイス数、サンプリング周波数、エフェクトの有無）で動き続ける。即ち、処理負荷量は、再度設定されるまで変更されずに一定である。

【０１１１】

ところが、実際にコンピュータを使用する場合には、本発明の楽音生成処理方法によるプログラム以外の他のプログラムが同時に並行して動かされることがある。この場合には、各プログラムは各タイムスロットに割り当てられ、タイムシェアリングで処理されることになるので、本発明の楽音生成処理方法によるプログラムの各タスクの実行中に、他のプログラムが割り込んで実行されることがある。

【０１１２】

例えば、音源タスクの１回にかかる処理時間が、他のプログラムの動作状態によって変動することとなる。従って、上記したように、本発明の楽音生成処理方法によるプログラムの処理負荷量を一定とすると、他のプログラムによるＣＰＵの処理負荷量の変動には追従できなくなってしまう。このため、場合によっては音源タスクの処理が間に合わずに音声が途切れたり、ＣＰＵ１２の処理能力に余裕ができてもそれをうまく活かすことができなくなってしまう。

【０１１３】

こうした欠点を解消するためには、常に音源タスクのＣＰＵ消費率を監視して、使用者の設定した音源タスクのＣＰＵ消費率に追従するようにダイナミックに発音可能ボイス数の増減を行うようにすればよい。

【０１１４】

具体的には、音源タスクが起動される度に、音源タスクの開始時と終了時との内部タイマの値の差からＣＰＵ消費率を計測し、これに応じてその都度発音可能ボイス数およびエフェクトの有無またはサンプリング周波数を決定する。

【０１１５】

なお、このようにダイナミックに発音可能ボイス数の増減を行う場合には、発音中に発音可能なボイス数が変動することになるが、この場合には次に示すような処理を行う。

【０１１６】

即ち、ボイス数を減らす場合には、現在使用が許可されているボイスの中で発音していないボイスがあれば、減らす分のボイスの許可情報を使用不許可にする。もし、現在使用が許可されているボイスの中で発音していないボイスがなかったり、あるいは発音していないボイスを全て使用不許可としても、まだボイス数を減らす必要がある場合には、発音しているボイスの中から減らす分だけのボイスを選択し、このボイスを急速にミュートし、以後の使用を不許可とする。なお、ボイスを選択するには、発音中のレベルが小さいボイス、最も過去に発音開始されたボイスなど、アサイン処理と同様の選択基準で選択することができる。

【０１１７】

あるいは、上記とは別の方法とし、発音しているボイスを急速にミュートする代わりに、そのボイスがノートオフによって自動的に消音されるのを待って、使用不許可とするようにしてもよい。

【０１１８】

一方、ボイス数を増やす場合には、使用が許可されていないボイスの中から増やす分だけのボイスを使用許可するようにすればよい。

【０１１９】

【発明の効果】

以上において説明したように、本発明による楽音生成処理方法は、汎用のコンピュータ上で楽音をリアルタイムで生成する際に、汎用のコンピュータに適した処理を行うように構成したので、従来より指摘されていた種々の問題点の解決を図ることができるとともに、汎用のコンピュータ上で楽音をリアルタイムで効率的に生成することができるという優れた効果を奏する。

【図面の簡単な説明】

【図１】本発明の楽音生成処理方法によるプログラムを搭載可能な、典型的な汎用のコンピュータのハードウェア構成を示すブロック図である。

【図２】本発明の楽音生成処理方法によるプログラムの全体構成を示す概念図である。

【図３】音源タスクにおける処理を示す概念図であり、（ａ）は１ボイス分の処理を示し、（ｂ）はボイスおよびエフェクトのミキシングを示す。

【図４】音源タスクにおける発音サービスパラメータの生成の処理を示すフローチャートである。

【図５】音源タスクにおける発音サービスパラメータを受けて行われる波形読み出し、ＴＶＦ、ＴＶＡ、パンおよびエフェクトへの送り量の処理を示すフローチャートである。

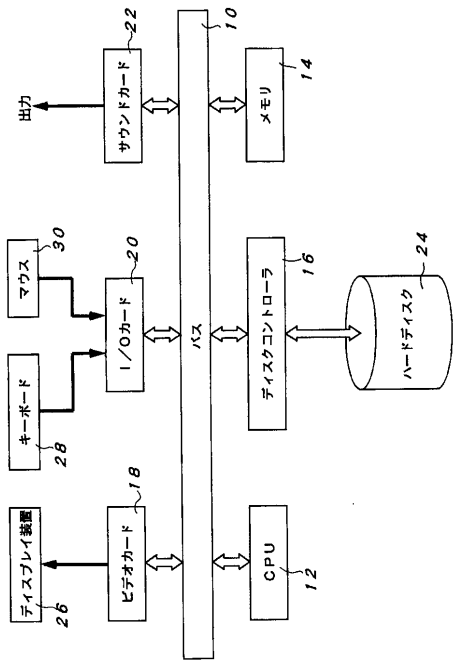
【図６】音源タスクと制御タスクとが共有するメモリ空間が２面設けられていて、それを交互に使用する状態を示した説明図である。

【符号の説明】

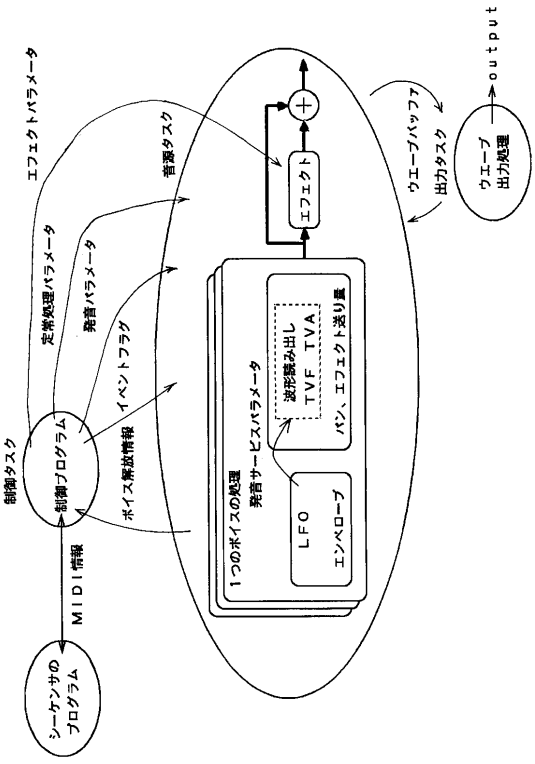
１０	バス
１２	ＣＰＵ
１４	メモリ
１６	ディスクコントローラ
１８	ビデオカード
２０	Ｉ／Ｏカード
２２	サウンドカード
２４	ハードディスク

２６　ディスプレイ装置
２８　キーボード
３０　マウス

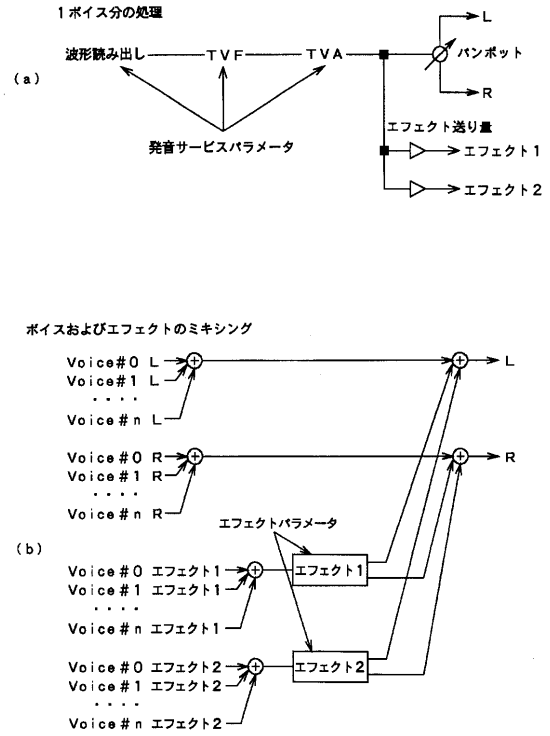
【図１】



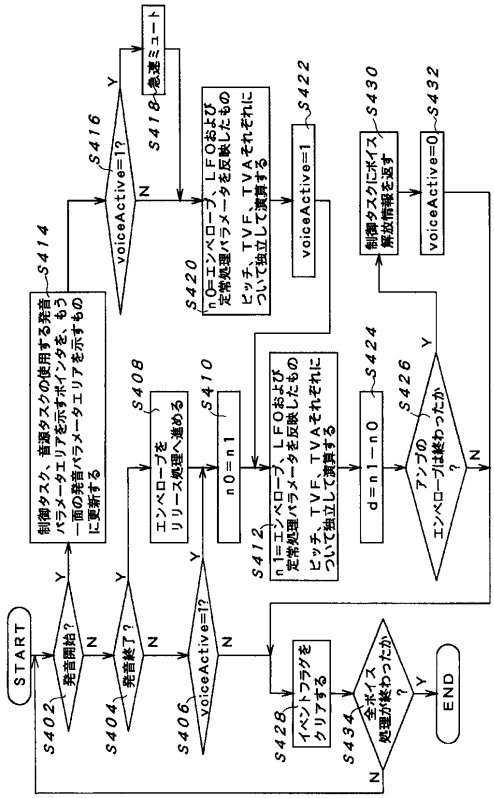
【図２】



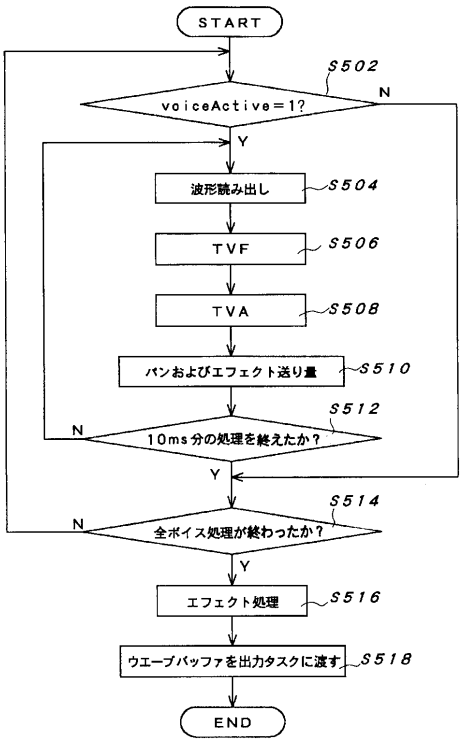
【図３】


１ボイス分の処理
(a)
波形読み出し
TVF
TVA
パンポット
L
R
発音サービスパラメータ
エフェクト送り量
エフェクト１
エフェクト２
ボイスおよびエフェクトのミキシング
(b)
Voice#0 L
Voice#1 L
Voice#n L
Voice#0 R
Voice#1 R
Voice#n R
エフェクトパラメータ
Voice#0 エフェクト1
Voice#1 エフェクト1
Voice#n エフェクト1
Voice#0 エフェクト2
Voice#1 エフェクト2
Voice#n エフェクト2
エフェクト1
エフェクト2

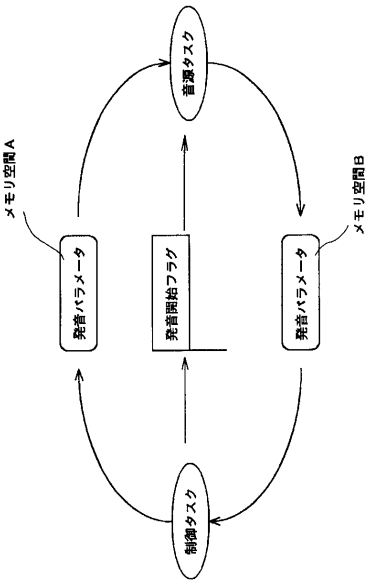
【図４】


START
S402
発音開始?
S404
発音終了?
S406
voiceActive=1?
S408
エンベロープをリリース処理へ進める
S410
n0=n1
S412
n1=エンベロープ、LFOおよび定常処理パラメータを反映したものピッチ、TVF、TVAそれぞれについて独立して演算する
S414
制御タスク、音源タスクの使用するパラメータエリアを示すポインタを、もう一面の発音パラメータエリアを示すものに更新する
S416
voiceActive=1?
S418
急速ミュート
S420
n0=エンベロープ、LFOおよび定常処理パラメータを反映したものピッチ、TVF、TVAそれぞれについて独立して演算する
S422
voiceActive=1
S424
d=n1−n0
S426
アンプのエンベロープは終わったか?
S428
イベントフラグをクリアする
S430
制御タスクにボイス解放情報を返す
S432
voiceActive=0
S434
全ボイス処理が終わったか?
END

【図５】


START
S502
voiceActive＝1?
波形読み出し
S504
TVF
S506
TVA
S508
パンおよびエフェクト送り量
S510
S512
10ms分の処理を終えたか?
S514
全ボイス処理が終わったか?
エフェクト処理
S516
ウエーブバッファを出力タスクに渡す
S518
END

【図６】


音源タスク
メモリ空間A
メモリ空間B
発音パラメータ
発音開始フラグ
発音パラメータ
制御タスク

フロントページの続き

(56)参考文献 特開平０９－１６０５５２（ＪＰ，Ａ）
特開平０９－０９７０６４（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.[7]，ＤＢ名)
G10H 1/00-7/08