

NITRO-Composer

サウンドシステムマニュアル

2008-05-30

任天堂株式会社発行

このドキュメントの内容は、機密情報であるため、厳重な取り扱い、管理を行ってください。

目次

1	はじめに	6
2	ハードウェア	6
2.1	サウンド回路	6
2.2	チャンネル	6
2.2.1	ADPCM/PCM	6
2.2.2	PSG矩形波	7
2.2.3	ノイズ	7
2.3	チャンネル番号	7
2.4	キャプチャ	7
3	プレイヤー	8
3.1	プレイヤーとは	8
3.2	プレイヤー数	8
3.3	シーケンス最大同時再生数	8
3.3.1	システム全体の制限	8
3.3.2	プレイヤー毎の制限	8
3.4	トラック	9
3.4.1	トラック数	9
3.5	シーケンスのチャンネル制限	9
3.5.1	チャンネル制限の設定方法	10
3.5.2	トラック毎のチャンネル制限	10
3.6	サウンドドライバのプレイヤーとの違い	10
3.6.1	サウンドドライバのプレイヤー併用の禁止	10
4	プライオリティ	11
4.1	プライオリティとは	11
4.2	プレイヤープライオリティ	11
4.3	発音プライオリティ	12
4.3.1	発音プライオリティとは	12
4.3.2	発音プライオリティの設定	12
4.3.3	リリース	12
5	メモリ管理	13
5.1	メモリ管理の必要性	13
5.2	2種類のヒープ	13
5.3	サウンドヒープ	13
5.3.1	サウンドヒープとは	13
5.3.2	サウンドヒープの用法	14
5.3.3	グループロード	14
5.4	プレイヤーヒープ	14
5.4.1	プレイヤーヒープとは	14

5.4.1.1	シーケンスアーカイブ再生時の注意	15
5.4.2	プレイヤーヒープの用法	15
5.4.3	プレイヤーヒープの作成	15
5.5	サウンドヒープとプレイヤーヒープの比較	15
5.6	メモリ管理の例	16
5.6.1	仕様	16
5.6.2	準備	16
5.6.2.1	グループの作成	16
5.6.2.2	プレイヤーヒープの設定	17
5.6.2.3	設定の確認	17
5.6.3	起動時の処理	17
5.6.3.1	サウンドヒープの作成	17
5.6.3.2	サウンドアーカイブの初期化	17
5.6.3.3	プレイヤーヒープの作成	17
5.6.3.4	常駐グループのロード	17
5.6.4	ゲーム中の処理	17
5.6.4.1	入れ替え領域へのロード	17
5.6.4.2	BGMの再生	17
5.6.4.3	場面切り替え	17
5.6.5	メモリの使用状況	17
6	ストリーム再生	19
6.1	ストリーム再生とは	19
6.2	ストリーム再生の仕組み	19
6.2.1	チャンネルの確保	19
6.2.2	データの準備	19
6.2.3	再生開始	19
6.2.4	必要に応じてデータロード	19
6.3	シーケンス再生との違い	20
7	出力エフェクト	21
7.1	出力エフェクトとは	21
7.2	出力エフェクト使用時の注意	21
7.2.1	同時発音数の減少	21
7.2.2	処理負荷	21
7.2.3	音の歪み	21
7.3	サラウンド使用時の注意	21
7.3.1	音量の増大	21
7.3.2	左右バランスの相違	22
7.4	ノーマルとモノラルについて	22
7.4.1	全て出力エフェクトを使う場合	22
7.4.2	サラウンドやヘッドフォンのみ出力エフェクトを使う場合	22
8	サウンド処理	23
8.1	ARM7	23

8.2	メインメモリアクセス.....	23
8.2.1	波形データの転送.....	23
8.2.2	シーケンスデータの処理.....	23
8.3	サウンド処理単位.....	23
8.3.1	4分音符分解能	24

表

表 2-1	チャンネル番号別機能	7
表 5-1	サウンドヒープとプレイヤーヒープの比較.....	16
表 6-1	シーケンス再生とストリーム再生の特徴	20
表 7-1	出力エフェクトタイプ	21

図

図 2-1	サウンド回路.....	6
図 3-1	プレイヤー概念図	8
図 3-2	トラック概念図.....	9
図 4-1	プレイヤープライオリティの例.....	11
図 5-1	メモリの確保と解放	14
図 5-2	プレイヤーヒープの動作	15
図 5-3	メモリ管理の例.....	16
図 5-4	メモリの使用状況	18

改訂履歴

改訂日	改訂内容
2008-05-30	1. NITRO-System の名称変更による修正 (NITRO-System を TWL-System に変更)。
2008-04-08	1. 改訂履歴の書式を変更 2. ページのヘッダを修正
2005-09-01	1. 誤記修正
2005-03-28	1. シーケンスアーカイブの再生に必要なバンク及び波形データを、プレイヤーヒープへロードできるようになったことに伴い、プレイヤーヒープの説明修正。 2. サウンドドライバのプレイヤーに関する説明追加 3. メモリ管理の実践例で、設定の確認方法の説明修正 4. 出力エフェクトのノーマルとモノラルに関する説明追加
2005-01-31	1. シーケンスのチャンネル制限に関する説明追加 2. "NITRO"を"ニンテンドーDS"に変更
2004-12-06	1. 出力エフェクトに関する説明追加
2004-08-10	1. チャンネル番号に関する説明追加 2. ストリーム再生に関する説明追加 3. サウンド回路での"サウンド"の名称を"チャンネル"に変更
2004-07-20	1. スタイルの調整
2004-06-01	1. メモリ管理の章を追加 2. 1つのプレイヤーで複数のシーケンスが再生できるようになったことに伴う修正 3. プレイヤープライオリティの扱い変更に伴う修正 4. "最大同時シーケンス再生数"を"シーケンス最大同時再生数"に変更
2004-04-01	初版

1 はじめに

本ドキュメントでは NITRO-Composer を使うに当たっての基礎知識について説明しています。

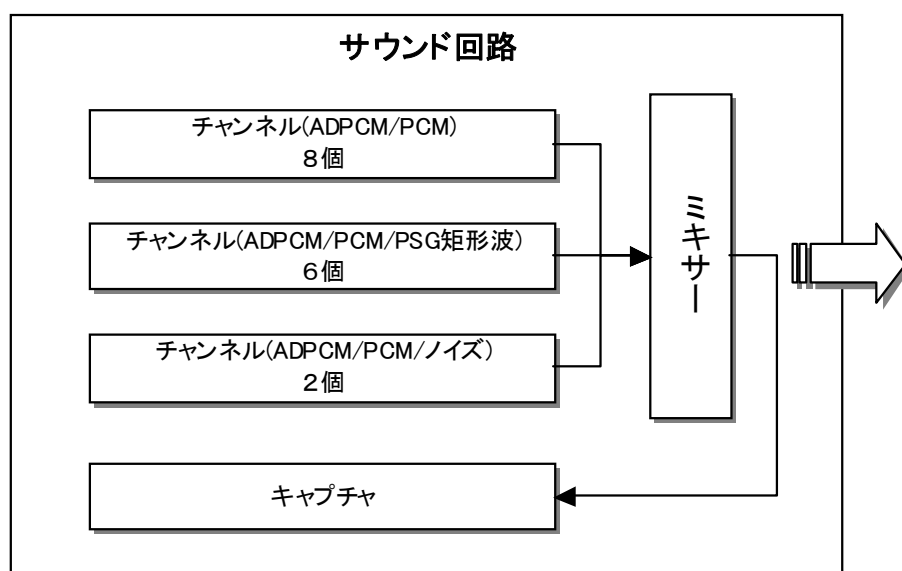
2 ハードウェア

まず、TWL およびニンテンドーDS ハードウェアのサウンド回路について説明します。

2.1 サウンド回路

TWL およびニンテンドーDS ハードウェアのサウンド回路は、下記のようになっています。

図 2-1 サウンド回路



2.2 チャンネル

上の図でチャンネルとは、1度に1つの音を発音できる回路を表しています。

ADPCM/PCM 再生専用のチャンネルが8個、PSG 矩形波も再生できるものが6個、ノイズも再生できるものが2個の合計、16チャンネルあります。すなわち、最大同時に16音まで再生できることになります。

ただし、PSG 矩形波とノイズは、再生できるチャンネルが限定されていますので、それだけで16音同時に鳴らすことはできません。

2.2.1 ADPCM/PCM

ADPCM または PCM の波形データの再生が行えます。PCM は8ビットまたは16ビットで再生できます。

2.2.2 PSG矩形波

PSG 矩形波の再生が行えます。デューティ比を選択することができます。

2.2.3 ノイズ

ホワイトノイズの再生が行えます。

2.3 チャンネル番号

16個あるチャンネルには、それぞれ 0～15 までの番号が振られています。また各番号によって、機能が少し異なります。

表 2-1 チャンネル番号別機能

チャンネル番号	機能
0, 2	ADPCM/PCM を再生できます。このチャンネルの出力をサウンドキャプチャの入力とすることもできます。
1, 3	ADPCM/PCM を再生できます。サウンドキャプチャとタイマーを共用しているため、サウンドキャプチャを使うときは、サウンドキャプチャの出力チャンネルとしてしか使えません。
4 ～ 7	ADPCM/PCM を再生できます。
8 ～ 13	ADPCM/PCM 及び PSG 矩形波を再生できます。
14, 15	ADPCM/PCM 及びホワイトノイズを再生できます。

2.4 キャプチャ

サウンド回路図のキャプチャとは、サウンド出力をとりこんで、メモリに書き戻す回路を表しています。上図では、ミキサーの出力をキャプチャしていますが、チャンネル 0 またはチャンネル 2 の出力を取り込むこともできます。

ミキサーの出力を取り込んで、再びサウンドから出力させると、時間が遅れてふたたびミキサーに送られることになります。このようにすることで、リバーブ効果(遅延効果)が得られます。

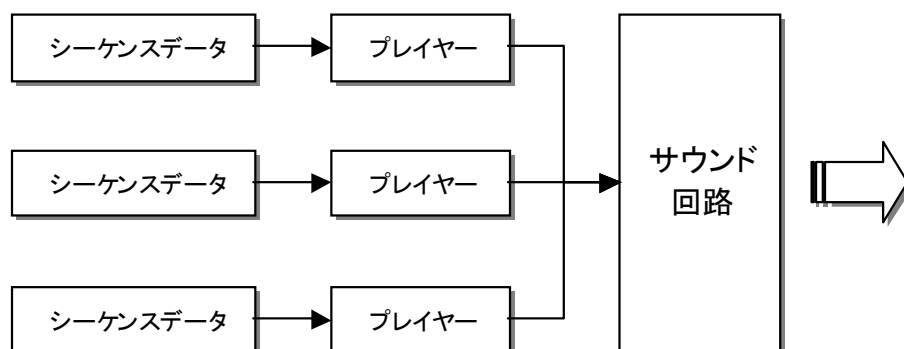
NITRO-Composerではこのサウンドキャプチャを利用して、音声出力にサラウンド効果などのエフェクトをかける機能を提供しています。詳しくは、「7章 出力エフェクト」をご覧ください。

3 プレイヤー

3.1 プレイヤーとは

NITRO-Composer を使う場合、通常、サウンド回路にあるチャンネルを直接操作することはありません。音の再生は、シーケンスデータ単位で行います。シーケンスデータは、プレイヤーを使って再生します。

図 3-1 プレイヤー概念図



なお、デフォルトでは1つのプレイヤーで1つのシーケンスしか再生できませんが、1つのプレイヤーで複数のシーケンスを再生できるように変更することもできます。

3.2 プレイヤー数

プレイヤーは 32 個用意されています。個々のプレイヤーには番号が振られ、それぞれ0から 31 までの値が割り当てられています。これをプレイヤー番号と呼びます。

シーケンスデータを再生するときには、どのプレイヤーで再生するのかを、プレイヤー番号で指定する必要があります。

3.3 シーケンス最大同時再生数

同時に再生できるシーケンスの数には、2重の制限があります。1つは、システム全体で再生できる数による制限で、もう1つは、プレイヤー毎に再生できる数による制限です。

3.3.1 システム全体の制限

システム全体で、シーケンスは最大同時に16個まで再生することができます。これは、BGM と効果音の区別はありません。例えば、BGMを1つ鳴らしている間は、効果音を同時に15 個まで鳴らすことができます。

3.3.2 プレイヤー毎の制限

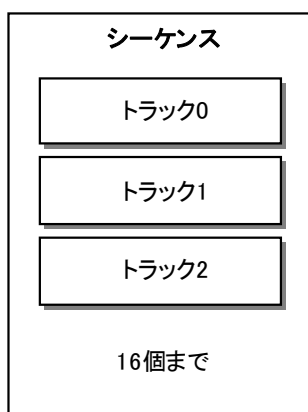
1つのプレイヤーで再生できるシーケンスの数には制限があります。デフォルトでは、1つのプレイヤーで1つのシーケンスしか再生できません。ただし、この制限は自由に変更することができ、16にして事実上無制限とすることもできます。

この制限は、システム全体の制限と違って、故意に設定する制限です。例えば、主人公の台詞を再生するためのプレイヤーを1つ用意し、同時に再生できるシーケンスの数を1つだけに制限しておけば、間違っただけで主人公が2つの台詞を同時にしゃべることがなくなります。また、敵の効果音など、複数同時に鳴らしたいけれど、上限を設けたい場合などに有効です。

3.4 トラック

MIDI シーケンスデータには、トラックという概念があります。1つのシーケンスデータには、複数のトラックがあり、各トラック毎に別々のシーケンスを再生することで、1つの曲を構成しています。NITRO-Composer でも、これと同じような仕組みのトラックがあります。

図 3-2 トラック概念図



シーケンスコマンドや、シーケンス関連の関数には、シーケンス全体に対するものと、個々のトラックに対するものがありますので、注意してください。

3.4.1 トラック数

1つのシーケンスに対し、トラックは最大16個まで使うことができます。ただし、システム全体としてトラックは32個しかありません。また、1つのシーケンスを再生するためには、最低1つのトラックが必要です。

例えば、2つのシーケンスでそれぞれ16個ずつトラックを使っていると、シーケンスは16個まで同時に再生できるという条件は満たしていますが、トラックの残りがありませんので、それ以上シーケンスを再生することはできません。

3.5 シーケンスのチャンネル制限

シーケンスで発音する際には、通常、全てのチャンネルの中から利用可能なチャンネルを探して、行われます。しかし、あるプレイヤーで再生するシーケンスが使用できるチャンネルを、制限することもできます。

例えば、次のような状況での利用が考えられます。

- シーケンス再生途中で、サウンドキャプチャを使い始めると、チャンネル1及びチャンネル3で発音中の音が強制的に止められてしまう。そのため、止められて問題のあるシーケンスは、あらかじめチャンネル1及びチャンネル3を使わないように制限をかけておく。
- SE と BGM の発音をそれぞれ、互いにうち消し合わないようにしたい。そのため、SE 用のプレイヤーが利用できるチャンネルと、BGM 用のプレイヤーが利用できるチャンネルを、互いに重ならないように設定しておく。

3.5.1 チャンネル制限の設定方法

シーケンスのチャンネル制限の設定は、通常サウンドデザイナーがプレイヤー毎に行います。明示的に設定していない場合は、チャンネル制限無しとして扱われます。

なお、サウンドデザイナーが設定したプレイヤー毎のチャンネル制限は、実行時にプログラマーが上書きで書き換えることができます。

3.5.2 トラック毎のチャンネル制限

プレイヤー毎のチャンネル制限は、そのプレイヤーで再生するシーケンス全体に対して有効になります。あるシーケンス中の特定のトラックのみチャンネル制限をかけたい(かけたくない)場合は、実行時にプログラマーがそのように設定することができます。

これは、プレイヤーとは無関係に、指定したシーケンスの、指定したトラックに対してのみ有効になります。プレイヤー毎のチャンネル制限と併用した場合は、トラック毎に設定した方が有効になります。

3.6 サウンドドライバのプレイヤーとの違い

サウンドドライバにも、プレイヤーと呼ばれるものがありますが、NITRO-Composer におけるプレイヤーは、サウンドドライバのプレイヤーを、より使いやすいように拡張したものとなっています。

例えば、サウンドドライバのプレイヤーでは、1つのプレイヤーで1つのシーケンスしか再生することができませんが、NITRO-Composer のプレイヤーを使えば、複数のシーケンスを再生することができます。

3.6.1 サウンドドライバのプレイヤー併用の禁止

NITRO-Composer のプレイヤーを使いつつ、サウンドドライバのプレイヤーを直接操作することはできません。

また、NITRO-Composer のプレイヤー番号と、サウンドドライバのプレイヤー番号は、別の意味で使用されているので、あるプレイヤー番号を NITRO-Composer で使っていないという理由で、そのプレイヤー番号をサウンドドライバで使うということもできません。

4 プライオリティ

4.1 プライオリティとは

これまでの説明にあったように、同時に再生できるシーケンスやチャンネルの数は限られています。しかし、必ずその数以内に収まるように、データを作ったり、再生したりすることは不可能です。また、数がオーバーしたときに、必ず再生に失敗するようだと、思ったように再生することができなくなります。

このような問題を解決するために、プライオリティというパラメータを使います。プライオリティとは優先度を表し、値が大きいほど優先して再生できるようになります。例えば、すでに全てのチャンネルが使用中だったとしても、より優先度の高い音を鳴らそうとすると、再生中の音を止めて、新しい音を鳴らします。

このようにプライオリティを使うことで、より聞かせたい音と、補助的な音を区別し、より聞かせたい音の方を優先的に鳴らすことができるようになります。

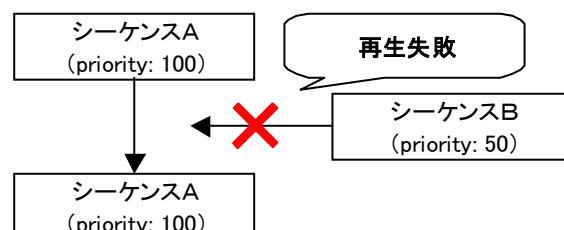
4.2 プレイヤープライオリティ

プレイヤープライオリティとは、あるプレイヤーでシーケンスを再生しようとしたときに、上で説明した最大同時再生数に関する2つの制限に引っかかった場合に使用される優先度です。

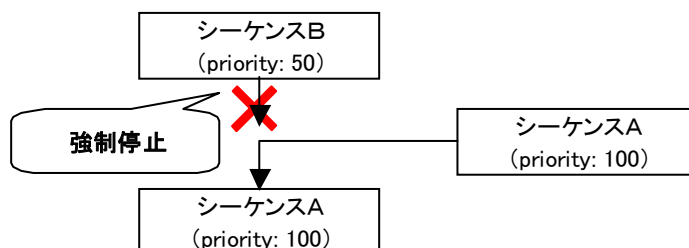
シーケンスの再生時には、まずプレイヤー毎の最大同時再生数のチェックが行われます。制限に引っかかった場合は、そのプレイヤーで再生中のシーケンスの内、一番プレイヤープライオリティの値が小さいものと比較されます。その結果、値の大きい方が優先され、値の小さい方は再生失敗または強制停止となります。値が同じ場合は、後から再生しようとした方が優先されます。

図 4-1 プレイヤープライオリティの例

●シーケンスA再生中にシーケンスBを再生



●シーケンスB再生中にシーケンスAを再生



もしこのチェックに通った場合には、次に、システム全体での最大同時再生数のチェックが行われます。チェックに引っかかった場合には、先ほどと同様に、システム全体で再生中のシーケンスの内、一番プレイヤープライオリティの小さいものと比較されます。値の大きい方が優先され、値の小さい方は再生失敗または強制停止となります。

なお、フェードアウト中のシーケンスは、プレイヤープライオリティが最低の 0 になります。

4.3 発音プライオリティ

4.3.1 発音プライオリティとは

発音プライオリティとは、16チャンネル全て使用中の状態、さらに発音しようとしたときに使用される優先度です。

新しく発音しようとした音のプライオリティと、既に再生されている16個の音を合わせた合計17個のプライオリティを比較して、一番小さい値の音が止められます。それが、新しく発音しようとした音だった場合は、発音に失敗します。一番小さい値の音が複数存在した場合には、そのうちのどの音が止められるかは、決まっています。

4.3.2 発音プライオリティの設定

発音プライオリティの値を設定するパラメータは2種類あります。1つは、あるシーケンス全体の優先度を決定するシーケンスの発音プライオリティで、もう1つはあるシーケンス中のトラック毎の優先度を決定するトラックの発音プライオリティです。この2つのパラメータは、それぞれ足し合わされて、最終的な発音プライオリティの値となります。

たとえば、BGMのシーケンス発音プライオリティを下げておくと、効果音の方が優先して発音することができるようになります。また、さらにBGMのあるトラックの発音プライオリティを大きく上げて、発音プライオリティ値の合計を効果音より大きくなるようにすると、BGMのあるトラックだけは、効果音に消されずに発音することができます。

4.3.3 リリース

リリース中の音は、設定したパラメータにかかわらず、発音プライオリティが最低になります。そのためリリース中の音は、プライオリティ値にかかわらず、優先的に消されていきます。

リリース中の音の中で、消されていく順番は、現在のボリューム値に依存します。ボリューム値が小さい音、すなわち、まもなくリリースが完了しそうなものから順に消されていきます。

5 メモリ管理

5.1 メモリ管理の必要性

サウンドデータは、TWL およびニンテンドーDS の ROM に格納されます。ただし、そのままではサウンドデータを使うことはできません。サウンドデータは、ROM からメインメモリ上へロードして始めて、使用可能になります。

メインメモリは、ROM に比べると小さいです。そのため、ROM に格納したサウンドデータを全てメインメモリへロードすることはできません。従って、必要なサウンドデータのみロードし、不要になったら削除する、というメモリの管理が必要になります。

ここでは、NITRO-Composer におけるメモリ管理の概要について説明します。

なお、メモリ管理が不要なストリーム再生という再生方式を使うこともできます。ストリーム再生については、6章を参照してください。

5.2 2 種類のヒープ

NITRO-Composer では 2 種類のヒープが用意されています。この 2 つを用途に応じて使い分けます。

- サウンドヒープ
- プレイヤーヒープ

サウンドヒープの方が基本となるヒープで、プレイヤーヒープはサウンドヒープの中に構築されます。まず始めにサウンドヒープを作成し、その後で必要に応じてプレイヤーヒープを作ります。

以下、それぞれについて説明します。

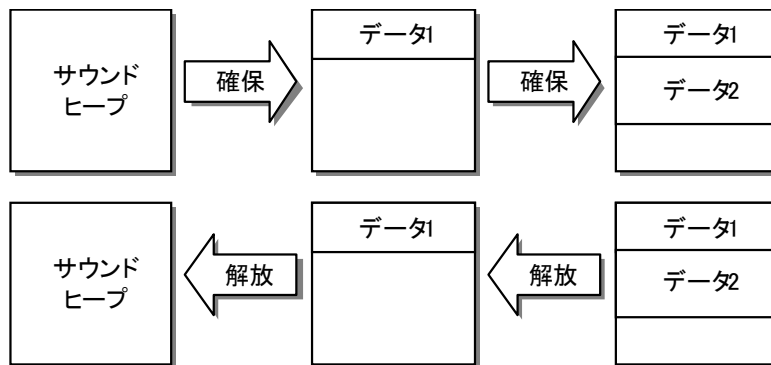
5.3 サウンドヒープ

5.3.1 サウンドヒープとは

サウンドヒープは、積み上げ方式のシンプルなヒープです。

サウンドデータをロードすると、データはサウンドヒープの上から順に格納されていきます。不要なデータを削除する時には、ヒープの下から順に行います。

図 5-1 メモリの確保と解放



5.3.2 サウンドヒープの用法

サウンドヒープへのデータのロードや削除は、主に起動時や場面切り替えなどで行います。

波形データなどの大きなデータや、効果音シーケンスデータのような小さいけど頻繁に必要となるデータなどは、ゲーム中にロードするのは非効率なため、場面切り替え時などでサウンドヒープへロードしておきます。サウンドヒープに一度ロードしておけば、削除するまで、再度ロードすることなく、そのデータを使い続けることができます。

5.3.3 グループロード

サウンドデータのロードは、サウンドデータ単位で行うものと、グループ単位で行うものがあります。グループとは、複数のサウンドデータを一括でロードするための固まりです。

グループを使うことで、複数のサウンドデータを一度にロードできます。たとえば、常駐サウンドデータのグループを作成しておいて、起動時にそのグループをロードするといった使い方をします。

どのグループにどのサウンドデータを含めるかという定義は、サウンドデザイナーがサウンドアーカイブ定義ファイルで行います。

5.4 プレイヤーヒープ

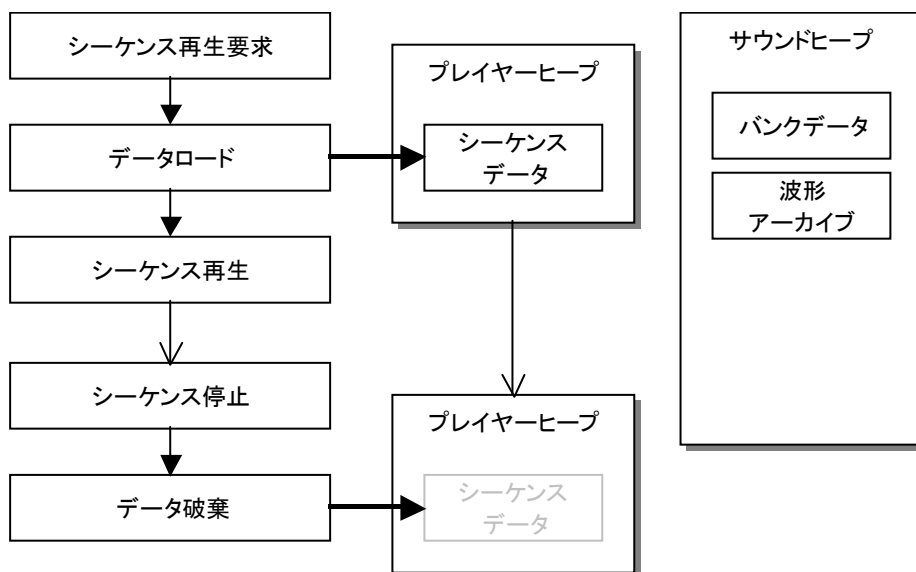
5.4.1 プレイヤーヒープとは

プレイヤーヒープは、シーケンスの再生中だけ使えるデータを置くためのヒープです。

シーケンスの再生時、サウンドヒープ上に必要なサウンドデータが無い場合には、プレイヤーヒープに自動的にデータがロードされます。ロードしたデータを使ってシーケンスは再生され、シーケンスの再生が完了すると、プレイヤーヒープは解放されます。

例えば、バンクと波形データがサウンドヒープ上にあって、シーケンスデータが無い場合、シーケンスデータをプレイヤーヒープにロードした後、シーケンスが再生されます。再生が終わると、シーケンスデータのみ解放されます。そのため、再び同じシーケンスを再生するときでも、もう一度シーケンスデータがロードされます。

図 5-2 プレイヤーヒープの動作



5.4.1.1 シーケンスアーカイブ再生時の注意

プレイヤーヒープへシーケンスアーカイブをロードすることはできません。シーケンスアーカイブの再生時には、シーケンスアーカイブをあらかじめサウンドヒープへロードしておく必要があります。ただし、シーケンスアーカイブの再生に必要なバンク及び波形データは、プレイヤーヒープへロードすることができます。

5.4.2 プレイヤーヒープの用法

プレイヤーヒープへのデータのロードは、シーケンスの再生時に自動的に行われます。そのため、あまり大きなデータや頻繁に必要なデータを、プレイヤーヒープへロードするようにしていると、処理に負担がかかります。主に、BGM シーケンスデータや、BGM 用のバンクデータなどをロードするために使います。

プレイヤーヒープにロードしないようにするためには、あらかじめサウンドヒープにロードしておく必要があります。また、シーケンスの再生に必要な全てのデータがサウンドヒープ上にあるときは、プレイヤーヒープを作成する必要はありません。

5.4.3 プレイヤーヒープの作成

プレイヤーヒープは、プレイヤー毎に作成する必要があります。また、そのプレイヤーで同時に複数のシーケンスを再生できる場合には、その数分のプレイヤーヒープを作成しなければなりません。

基本的に、プレイヤーヒープのサイズは、サウンドデザイナーがサウンドアーカイブ定義ファイルで設定します。ただし、プログラマーが設定することもできます。

プレイヤーヒープは、サウンドヒープからメモリを確保して作成されます。プレイヤーヒープのあるメモリ領域が解放されると、プレイヤーヒープは自動的に破棄されます。その後、再びメモリを確保して、プレイヤーヒープを作成することができます。

5.5 サウンドヒープとプレイヤーヒープの比較

下の表は、サウンドヒープとプレイヤーヒープの特徴や主な用法について、比較したものです。

表 5-1 サウンドヒープとプレイヤーヒープの比較

	サウンドヒープ	プレイヤーヒープ
ロードのタイミング	主に起動時や場面切り替え時	シーケンスの再生時
ロードの手続き	ロード関数呼び出し	シーケンス再生関数内で自動処理
データの破棄	メモリ解放関数呼び出し	シーケンス再生完了時に自動削除
主なロード対象	波形アーカイブ シーケンスアーカイブ 効果音用バンクデータ	シーケンスデータ BGM 用バンクデータ
長所	データのロード・削除が自由に制御でき、効率的に処理できる。全てのシーケンス再生に対し、データを共有できるため、メモリ効率が良い。	データのロード・削除が自動的に行われ、面倒なメモリ操作が不要。
短所	前もって必要なデータがわからない場合や、一時的に必要なデータの扱いが困難。	ゲーム中にロードが発生するため負荷がかかる。1つのシーケンス再生に対し、1つのプレイヤーヒープが必要なため、メモリ効率が悪い。

5.6 メモリ管理の例

メモリ管理方法の一例を紹介します。

5.6.1 仕様

サウンドヒープを大きく2つのブロックに分けて考えます。1つは、常駐領域で、もう1つは場面毎に入れ替える入れ替え領域です。

図 5-3 メモリ管理の例



常駐領域には、ゲーム起動時にデータをロードし、その後はそのまま保持し続けます。ここには、効果音データなどのゲーム全体を通していつでも使用するデータを格納します。

入れ替え領域には、ステージBGMで必要な波形データなどを格納します。この領域は、場面切り替え毎に消されます。また、ステージBGMで必要なシーケンスデータは、プレイヤーヒープにロードすることになります。

5.6.2 準備

5.6.2.1 グループの作成

サウンドデザイナーは、常駐領域に格納するグループ及び、入れ替え領域に格納するグループを作成します。

常駐領域用のグループをグループ番号 0 として定義します。ここに効果音用のシーケンスデータやバンクデータ、波形データを登録します。

また、入れ替え領域用のグループとして、グループ1, 2, 3の3つを定義します。これらは、場面に応じて3つの内1つだけをロードします。ここには、各場面で再生するBGMに必要なバンクデータ、波形データを登録します。

5.6.2.2 プレイヤーヒープの設定

プレイヤーヒープには、BGMシーケンスデータを格納しますので、BGMを再生するプレイヤーにプレイヤーヒープを作成します。サウンドデザイナーは、サウンドアーカイブ定義ファイルにて、プレイヤーヒープ作成のための設定を行います。

プレイヤーヒープのサイズには、再生するBGMシーケンスデータ、すなわち*.sseqファイルが格納できるくらいの十分な大きさを設定します。

5.6.2.3 設定の確認

サウンドデザイナーは、NITRO-Playerのヒープシミュレーションモードまたは、SoundPlayerの手動ロードモードで、以上の設定が問題ないかシミュレーションします。詳しくは、「NITRO-Player ユーザーマニュアル」または、「サウンドデザイナーガイド」をご覧ください。

5.6.3 起動時の処理

5.6.3.1 サウンドヒープの作成

まず、サウンドヒープを作成します。あらかじめサウンド用にどれくらいのメモリを使うのかを決めておき、そのサイズ分のサウンドヒープを作成します。

5.6.3.2 サウンドアーカイブの初期化

サウンドアーカイブを使うためには、サウンドアーカイブを初期化しなければなりません。サウンドアーカイブを初期化すると、サウンドヒープを消費します。

5.6.3.3 プレイヤーヒープの作成

サウンドアーカイブの設定に基づいて、プレイヤーヒープを作成します。プレイヤーヒープ用のメモリは、サウンドヒープから確保されます。

5.6.3.4 常駐グループのロード

常駐グループをロードしておきます。これで、いつでも効果音を再生できます。

5.6.4 ゲーム中の処理

5.6.4.1 入れ替え領域へのロード

場面に応じて、入れ替え領域用グループをロードします。BGM用のバンク、波形データがロードされます。

5.6.4.2 BGMの再生

BGMを再生しようとすると、BGMシーケンスデータが、プレイヤーヒープへロードされます。バンクと波形データは、サウンドヒープ上のデータを使って、シーケンスが再生されます。

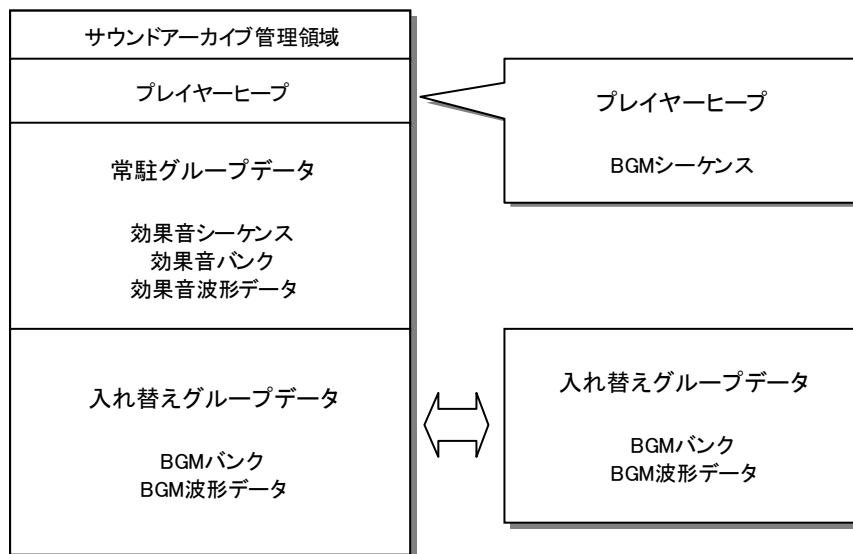
5.6.4.3 場面切り替え

場面切り替え時に、入れ替え領域をクリアします。その後、再び入れ替え領域へ、データをロードします。

5.6.5 メモリの使用状況

以上のような処理を行うと、メモリの使用状況は下図のようになります。

図 5-4 メモリの使用状況



6 ストリーム再生

6.1 ストリーム再生とは

ストリーム再生とは、再生するサウンドデータ全部を前もってロードせずに、再生しながら必要な分だけ少しずつロードする再生方式です。一方、シーケンスデータを使って再生する方式を、シーケンス再生と呼びます。

ストリーム再生では、長時間の波形データであっても、必要最低限のRAMで再生できます。また、5章で説明したようなメモリ管理の必要もありません。ただし、シーケンス再生では無かった幾つかの制限もあります。

ここでは、ストリーム再生の仕組み及び、シーケンス再生との相違点について説明します。

6.2 ストリーム再生の仕組み

ストリーム再生は次のような手順で処理されます。

1. チャンネルの確保
2. データの準備
3. 再生開始
4. 必要に応じてデータロード

6.2.1 チャンネルの確保

ストリーム再生を行うためのチャンネルを確保します。ステレオデータなら2チャンネル、モノラルデータなら1チャンネル必要です。

使用するチャンネル番号は、同時に使うチャンネルの機能と衝突しないように選択する必要があります。例えば、サウンドキャプチャを同時に使用する場合は、チャンネル 1 とチャンネル 3 は使用できません。詳しくは、「2.3チャンネル番号」をご覧ください。

6.2.2 データの準備

再生前にあらかじめ、最低限必要なデータをロードしておかなければなりません。このロードが完了するまでは、再生を開始が待たされることに注意する必要があります。

6.2.3 再生開始

実際に再生を開始します。

6.2.4 必要に応じてデータロード

再生して不要になったデータを破棄し、代わりに新しいデータをロードします。これを繰り返すことによって、延々と再生を続けることができます。

6.3 シーケンス再生との違い

シーケンス再生とストリーム再生には、それぞれ一長一短があります。下の表は、それぞれの特徴をまとめたものです。

表 6-1 シーケンス再生とストリーム再生の特徴

	シーケンス再生	ストリーム再生
データの構成	シーケンスデータ バンクデータ 波形データ	波形データのみ
データのロード	再生前に必要なデータをロードしておく必要がある	再生しながら必要なものだけ自動的にロードされる
再生時の処理負荷	非常に軽い	かなり重い
最大同時再生数	16個まで	4個まで
プログラム制御	テンポや音程の変更など、多くのパラメータを変更できる	ボリューム、パン以外の変更はできない

ストリーム再生は、再生時の処理負荷が大きいとため、気軽には使えません。仕様のには、4つまで同時に再生できるようになっていますが、処理時間の面から4つ同時に再生することは難しいです。

シーケンス再生ではメモリが足りないときや、処理時間を気にする必要がある場面を使うなど、うまく使い分ける必要があります。

7 出力エフェクト

7.1 出力エフェクトとは

サウンドキャプチャを使用して、音声出力全体にサラウンド効果などのエフェクトをかけることができる機能です。エフェクトの種類は、以下のものが用意されています。

表 7-1 出力エフェクトタイプ

タイプ	説明
ノーマル	キャプチャ音声はそのまま出力されます。単なるステレオ出力になります。
サラウンド	DS 本体スピーカー位置より広がって音を感じられるようにするエフェクトをかけます。
ヘッドフォン	ヘッドフォン使用時に、耳の負担を軽減するようなエフェクトをかけます。
モノラル	左右の音声成分を足し合わせて、モノラル出力にします。

7.2 出力エフェクト使用時の注意

出力エフェクトを使用するときには、以下の点に注意する必要があります。

7.2.1 同時発音数の減少

通常 TWL やニンテンドーDS では、16 音同時に再生できますが、出力エフェクト使用時にはサウンドキャプチャを使用するため、チャンネル番号 1 及び 3 の、2チャンネルを消費します。そのため、最大同時発音数は、14 に減ります。

7.2.2 処理負荷

サラウンドやヘッドフォンのエフェクトをかけるときには、処理にそれなりの負荷がかかります。おおよそ1フレームの 5% 程度の処理がかかります。

7.2.3 音の歪み

ミキサーは 24 ビットで処理されますが、サウンドキャプチャで取り込む時には、16 ビットで取り込まれます。そのため、音が重なるような状況では、音が歪んで聞こえることがあります。

7.3 サラウンド使用時の注意

サラウンドを使用するとき、特に注意する必要がある点について説明します。

7.3.1 音量の増大

サラウンドの場合、そのアルゴリズムの性質から、左右に振られた個々の音は、通常のステレオ再生より音量が大きくなります。そのため、最終的に出力される音に歪みが生じやすくなります。また、全体的な音量バランスが、通常のステレ

オ再生と異なります。この傾向は、左右に偏った音が多いときほど、顕著に現れます。

7.3.2 左右バランスの相違

ニンテンドーDS ステレオスピーカーは、左右の聞こえ方が若干異なります。これがサラウンド使用時には誇張されて、違いが目立つ場合があります。例えば、真左と真右で鳴らした音の広がり方が異なる場合があります。

7.4 ノーマルとモノラルについて

ノーマルとモノラルについては、出力エフェクトを使わなくても実現可能です。ノーマルは出力エフェクトを使わない状態に相当します。また、モノラルは `NNS_SndSetMonoFlag` 関数で実現することができます。

ノーマルとモノラルだけ使用するのであれば、出力エフェクトを使わない方法の方が適しています。一方、サラウンドなどもあわせて使用する場合は、2通りの選択があり、それぞれ長所と短所があります。

7.4.1 全て出力エフェクトを使う場合

ノーマルとモノラルでも出力エフェクトを使う場合、ノーマルとモノラルの状態でも上記の「出力エフェクト使用時の注意」に書かれているような問題点が発生します。

ただ逆に言えば、サラウンドやヘッドフォンの時と、音の聞こえ方が大きくは変わらないと言うメリットもあります。また、出力エフェクトタイプを切り替えるとき、出力エフェクトを止めずに済むため、発音中であっても大きなノイズにはなりません。

7.4.2 サラウンドやヘッドフォンのみ出力エフェクトを使う場合

サラウンドやヘッドフォンのみ出力エフェクトを使う場合、ノーマルとモノラルの状態では上記の「出力エフェクト使用時の注意」に書かれているような問題点が発生しません。

ただし、サラウンドやヘッドフォンの時と比べて、音の聞こえ方が大きく異なる場合があります。また、出力エフェクトタイプを切り替えるとき、出力エフェクトの開始または停止をする必要があるため、発音中の場合、大きな音途切れなどが発生することがあります。

8 サウンド処理

8.1 ARM7

サウンドの処理は、主に ARM7 を使って行われます。ARM7 とは、TWL やニンテンドーDS に搭載されている CPU の1つです。通常のゲーム処理やグラフィックス処理などは、ARM9 という別の CPU で処理されます。ARM7 ではサウンド処理の他に、無線処理やマイク処理などを行います。

8.2 メインメモリアクセス

上記の様に、サウンド処理と、ゲーム・グラフィックス処理は別の CPU で処理されるため、互いの処理時間に影響を及ぼしあうことは、あまりありません。ただし、サウンド側で1つ注意しなければならないのは、メインメモリアクセスです。

メインメモリとは、プログラムやサウンドデータ、グラフィックスデータなど様々なものを格納しておくメモリです。これには、ARM7、ARM9 から共にアクセスできます。ただし、ARM7 と ARM9 で同時にアクセスしようとしたときには、ARM7 が優先され、ARM9 は待たされます。これは、ARM7 のサウンド処理で、メインメモリにアクセスしている間は、ARM9 側の処理を止めることがあることを意味しています。

ARM7 のサウンド処理の内、メインメモリにアクセスするもので主なものは、次の2つです。

- 波形データの転送
- シーケンスデータの処理

8.2.1 波形データの転送

メインメモリに置かれている波形データは、TWL やニンテンドーDS のサウンド回路に転送して再生されます。この転送でメインメモリアクセスが発生します。

この転送は、サンプリングレートの高い波形データの再生または、元の波形データより高い音程での再生を行うと、より頻度が増します。このような再生を行う場合は、処理時間に注意してください。

8.2.2 シーケンスデータの処理

メインメモリに置かれているシーケンスデータを再生するときにも、メインメモリアクセスが発生します。

シーケンス再生には、通常発音と発音の間には、待ちが入りますが、この間はシーケンス処理を停止していますので、メインメモリアクセスは発生しません。逆に言うと、発音と発音の間でも、音程を常に変化し続けたりした場合には、毎回メインメモリアクセスが発生することになります。

8.3 サウンド処理単位

サウンド処理は、約 5.2ms 間隔で行われます。この1回のサウンド処理を、「サウンドフレーム」と呼んでいます。

これより細かい間隔で音を変化させることはできません。

8.3.1 4分音符分解能

シーケンスデータの4分音符分解能は、48です。テンポが 240 の時、ちょうど1ティック(すなわち4分音符の 48 分の 1)が、1サウンドフレームと同じになります。

© 2004-2008 Nintendo

任天堂株式会社の許諾を得ることなく、本書に記載されている内容の一部あるいは全部を無断で複製・複写・転写・頒布・貸与することを禁じます。