

GDFS (GD ファイルシステム) Library ver 1.00

1998/09/28

制限事項：

変更履歴：

1998/09/28 ver 1.00

- ドライブのメディアチェック時に GDD_ERR_CHECKBUSY を返すように修正。
- トラック内経過時間を取得する関数を追加。(gdFsDaGetInfo)

1998/09/15 ver 0.54

- Drive Status を更新させる関数を追加。(gdFsReqDrvStat)
- System ハンドルを取得する関数を追加。(gdFsGetSysHn)

1998/08/28 ver 0.53

- DA 系の関数を即時復帰型に変更。(API は従来型を使用)

1998/07/06 ver 0.52

- A0000000 番地がバッファとして渡された場合、キャッシュをインバリデートしないように修正。
- EOF と BUSY の時、ReqRd32 関数の戻り値を 0 に変更。
- gdFsReinit 関数を追加。
- gdFsReqGdRd 関数、gdFsTrans32 関数を公開。
- エラー定数 GDD_ERR_SIZEOVER を追加。

1998/06/26 ver 0.51

- gdFsGetDirrecSize マクロ関数が正しい値を返していなかった不具合を修正。
- GDC の仕様変更に伴い gdFsDaPlay 関数の引数変更。
- gdFsGetErrStat 関数を追加。
- gdFsIsTransReady 関数の廃止。(gdFsGetTransStat 関数に変更)
- gdFsGetWorkHn 関数を追加。(現在処理中のハンドルを取得できる)
- gdFsIsEof 関数を廃止。(gdFsCheckEof 関数に変更)
- gdFsGetFile* 関数の戻り値を Bool に変更。
- GDD_STAT_PLAY を廃止。

1998/06/01 ver 0.50

- Set5 版 リリース (ver 0.49 準拠)

1. 概要	4
1.1 基本機能	4
1.2 モジュール構成	4
2. 機能説明	5
2.1 用語	5
2.2 初期化	5
2.3 ディレクトリ操作	6
2.4 ファイル操作	8
2.5 ファイルへのアクセス	8
2.6 ドライブ情報取得	12
2.7 コールバック登録	13
2.8 Digital-Audio 再生	14
2.9 サーバー関数	14
3. データ仕様	15
3.1 定数	15
3.2 データ型	17
4. 関数仕様	18
4.1 関数一覧	18
4.2 関数 (API)	20
ライブラリ初期化・終了関数	20
ディレクトリ操作関数	22
ファイル操作関数	24
完了復帰関数	28
即時復帰関数	29
ドライブ情報取得関数	33
コールバック登録関数	34
Digital-Audio 再生関数	35
サーバー関数	37

1. 概要

1.1 基本機能

本ライブラリは以下の機能を提供します。

- ・ GD-ROM の高密エリアのアクセス機能
主な機能は以下の通りです。
 - a) ディレクトリのアクセス
 - b) ファイルのオープン / クローズ
 - c) ファイルの同期、非同期読み込み
- ・ GDDA の再生機能
主な機能は以下の通りです。
 - a) トラック指定再生
 - b) セクタ指定再生

1.2 モジュール構成

ファイルシステムを含む、GD アクセスに関するモジュール構成は以下に示します。

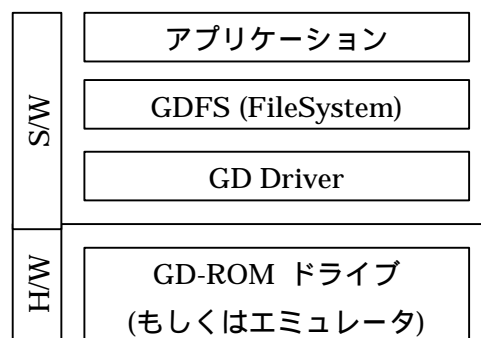


図 1 モジュール構成

2. 機能説明

2.1 用語

本ドキュメントで使用されている用語を説明します。

用 語	意 味
FAD	フレームアドレス。GD-ROM 上の物理セクタです。
ファイルハンドル	ファイルを操作するために必要な情報。 (実際にはそれを指し示すもの)
GD バッファ	GD ドライブが読み込んだデータを一時的にたくわえておく場所。
コールバック	条件が満たされたときに、登録してある処理を呼び出す仕組み。
セクタ	GD-ROM に格納されているデータを管理する最小単位。 (GD-ROM では 2048 バイト固定)
ディレクトリレコード	GD-ROM のディレクトリ構造を記録してある場所。
ディレクトリレコード ハンドル	ディレクトリレコードからファイル情報を取り出し、格納した 情報。(実際にはそれを指し示すもの)
ライブラリワークエリア	ライブラリの必要とする作業領域。

表 1 用語説明

2.2 初期化

本ライブラリを使用する前に必ず `gdFsInit` 関数を呼び出さなければなりません。
この関数は以下のことを行います。

- ライブラリの作業領域等の初期化
- デバイスの初期化 (GD-ROM)
- マウント処理

ライブラリの初期化時にはライブラリワークエリアとカレントディレクトリの情報を格納するための領域が必要ですが、これらはライブラリ利用者が用意する必要があります。

ライブラリワークエリアは、同時にオープンする最大ファイル数によって、また、カレントディレクトリの情報を格納する領域は、格納するファイル数によって必要サイズが変化します。

これらの必要サイズは、次のマクロで求めることができます。

- `gdFsGetWorkSize(max_open)`
- `gdFsGetDirrecSize(max_dirent)`

ライブラリの初期化は次のようにして行うことができます。

初期化手順例:

```
Uint32 gdfswork[gdFsGetWorkSize(8)/4];  
Uint32 gdfscurdir[gdFsGetDirrecSize(64)/4];
```

```
gdFsInit(8, gdfswork, 64, gdfscurdir);
```

但し、この場合 gdfswork は 32 バイト境界にある必要があります。

初期化後に、ライブラリ終了処理を行う前に、再度初期化処理を行うことはできません。

2.3 ディレクトリ操作

本ライブラリでは、ディレクトリの操作にはディレクトリレコードハンドルを使用します。このディレクトリレコードハンドルとは、GD-ROM 内のディレクトリのファイル情報を格納し、ファイル名、ファイルサイズ等の必要な情報をファイルシステムで使えるようにするためのものです。

通常の OS でいうところのディレクトリキャッシュに相当します。

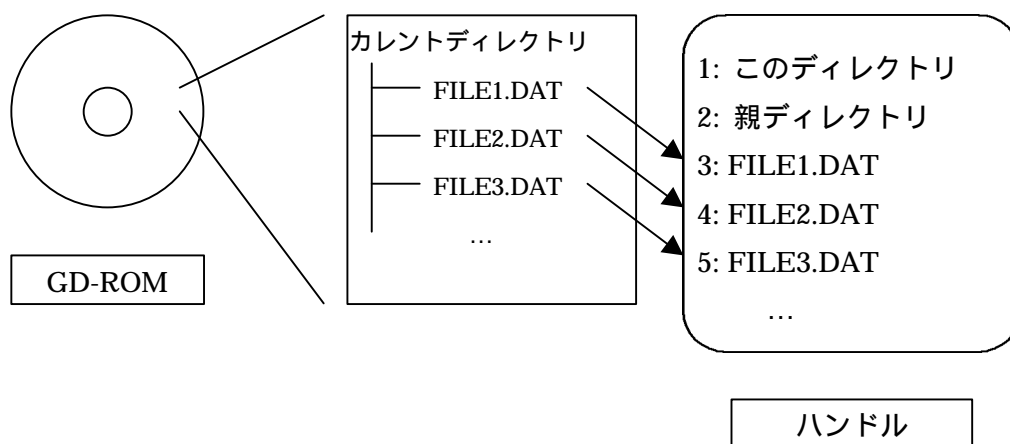


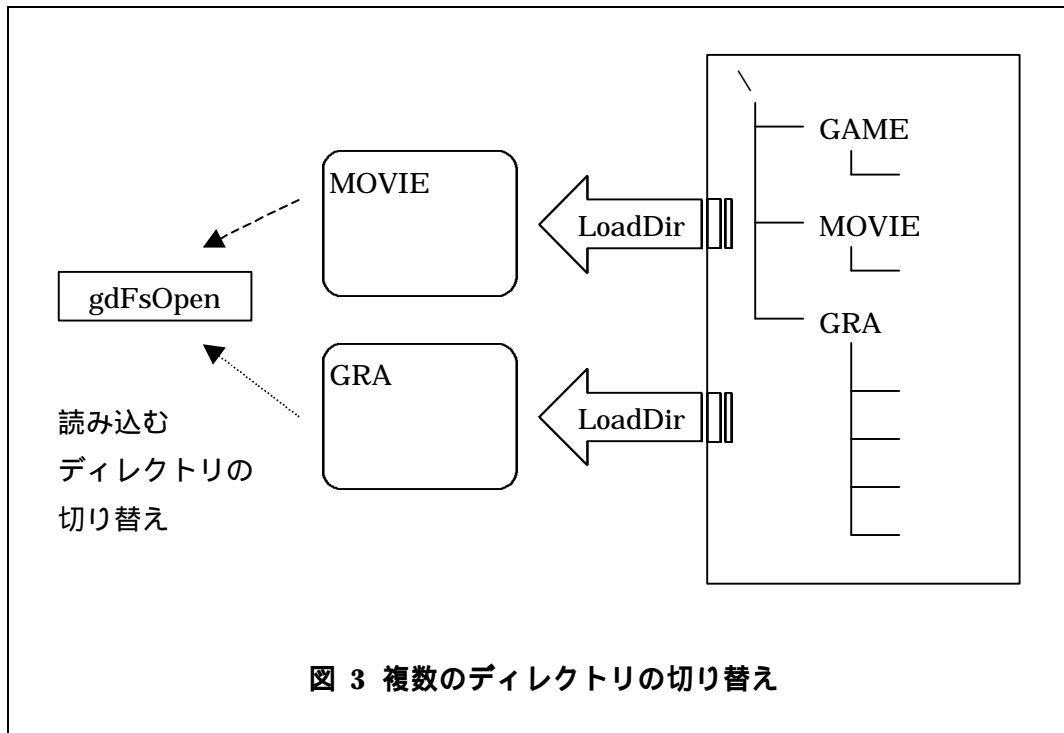
図 2 ディレクトリレコードハンドル

ライブラリ初期化時に、カレントディレクトリの情報を格納するためのディレクトリレコードハンドルが自動的に生成されています。

カレントディレクトリを作業ディレクトリに移動して作業する場合は、他にハンドルを生成する必要は特にありません。

カレントディレクトリの移動には gdFsChangeDir を使用します。

複数のディレクトリにあるファイルに、交互に違うディレクトリのファイルにアクセスするような場合には、GD-ROM にその度にアクセスすることになるのでパフォーマンスが著しく低下する場合があります。



こうした時には、ディレクトリレコードハンドルを生成し、頻繁にアクセスする必要のあるディレクトリの情報をあらかじめ読み取っておけば回避することができます。

複数ディレクトリ切り替え例

```

Uint32 dirbuf1[gdFsGetDirrecSize(64)];
Uint32 dirbuf2[gdFsGetDirrecSize(64)];
GDFS_DIRREC gf_moviedir;
GDFS_DIRREC gf_gradir;
GDFS gdfs1, gdfs2;

gf_moviedir = gdFsCreateDirhn(dirbuf1, 64);
gf_gradir = gdFsCreateDirhn(dirbuf2, 64);

gdFsLoadDir("MOVIE", gf_moviedir);
gdFsLoadDir("GRA", gf_gradir);

gdfs1 = gdFsOpen("ABC.MOV", gf_moviedir);
gdfs2 = gdFsOpen("DEF.GRA", gf_gradir);

/* ファイルの読み込み等の処理 */

gdFsClose(gdfs1);
gdFsClose(gdfs2);

```

2.4 ファイル操作

読み込み等のファイル操作を行うためには、ファイルハンドルが必要となります。
ファイルハンドルは `gdFsOpen` 関数で生成することができます。

ファイルハンドルからファイルサイズ等の情報を取得することができます。
(`gdFsGetFileSize`, `gdFsFileSctSize`, `gdFsGetFad`)

また、そのファイルの読み出し位置を変更したり、取得したり、ファイルの終わりにさしかかったかどうかを調べることができます。
(`gdFsSeek`, `gdFsTell`, `gdFsCheckEof`)

また、必要が無くなったファイルハンドルは、`gdFsClose` 関数で、開放しなければなりません。

2.5 ファイルへのアクセス

ファイルへのアクセス方式には以下の二つの方法があります。

- 完了復帰型アクセス
処理が完了するまで制御をブロックします。
- 即時復帰型アクセス
要求を受け付けたらすぐにアプリケーションへ制御を移します。

2.5.1 完了復帰型アクセス

完了復帰型アクセス関数は、制御方法は簡単ですが、処理が完了するまでブロックされるので、ゲーム等のリアルタイム制を要求される場合には、パフォーマンスを低下させる原因となる場合があります。

完了復帰型アクセス例

```
GDFS gf;  
Uint32 buf[2048/4];  
  
gf = gdFsOpen("ABC.DAT", NULL);  
  
gdFsRead(gf, 1, buf);  
  
gdFsClose(gf);
```

2.5.2 即時復帰型アクセス

即時復帰型アクセス関数は、制御方法は完了復帰型よりも多少複雑ですが、完了するまで待たされることがないので、読み込み中でも他の動作を平行して行うことができます。

即時復帰型アクセス例

```
GDFS gf;  
Uint32 buf[2048/4];  
  
gf = gdFsOpen("ABC.DAT", NULL);  
  
gdFsReqRd32(gf, 1, buf);  
  
do {  
    stat = gdFsGetStat(gf);  
} while (stat == GDD_STAT_READ);  
  
gdFsClose(gf);
```

読み込み中の状態変化は以下ようになります。

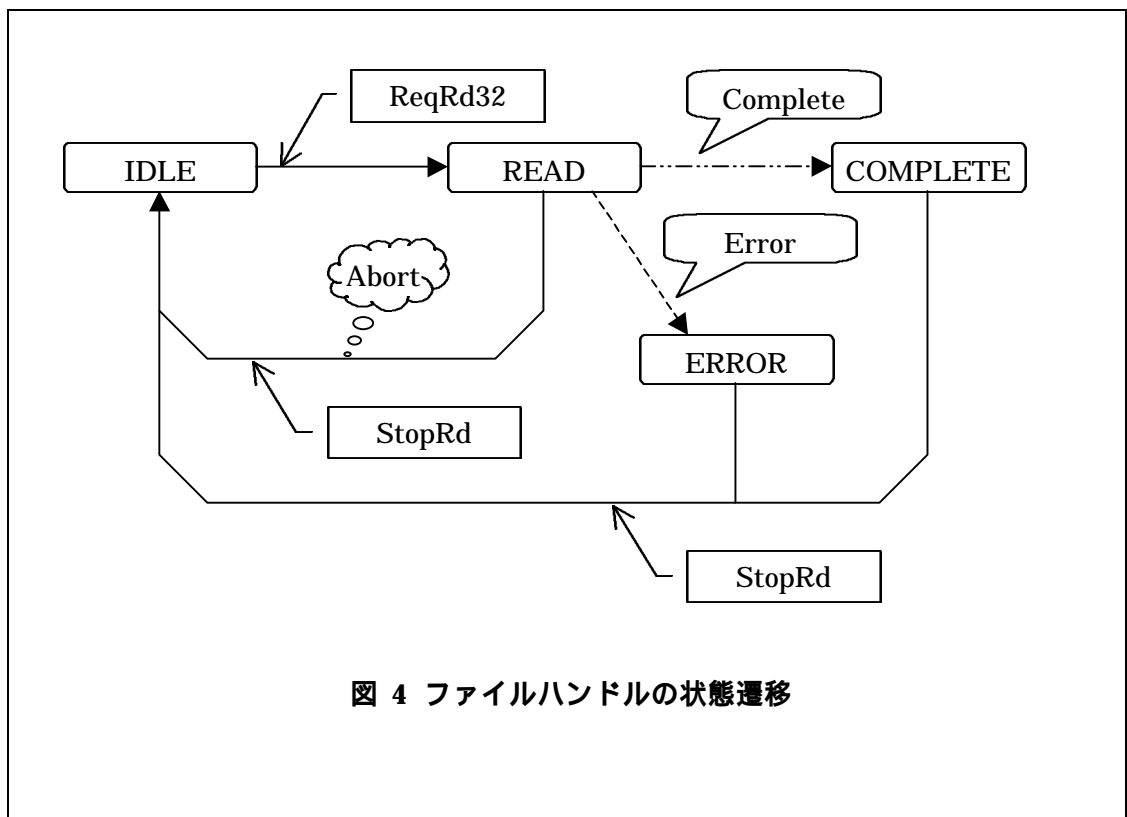
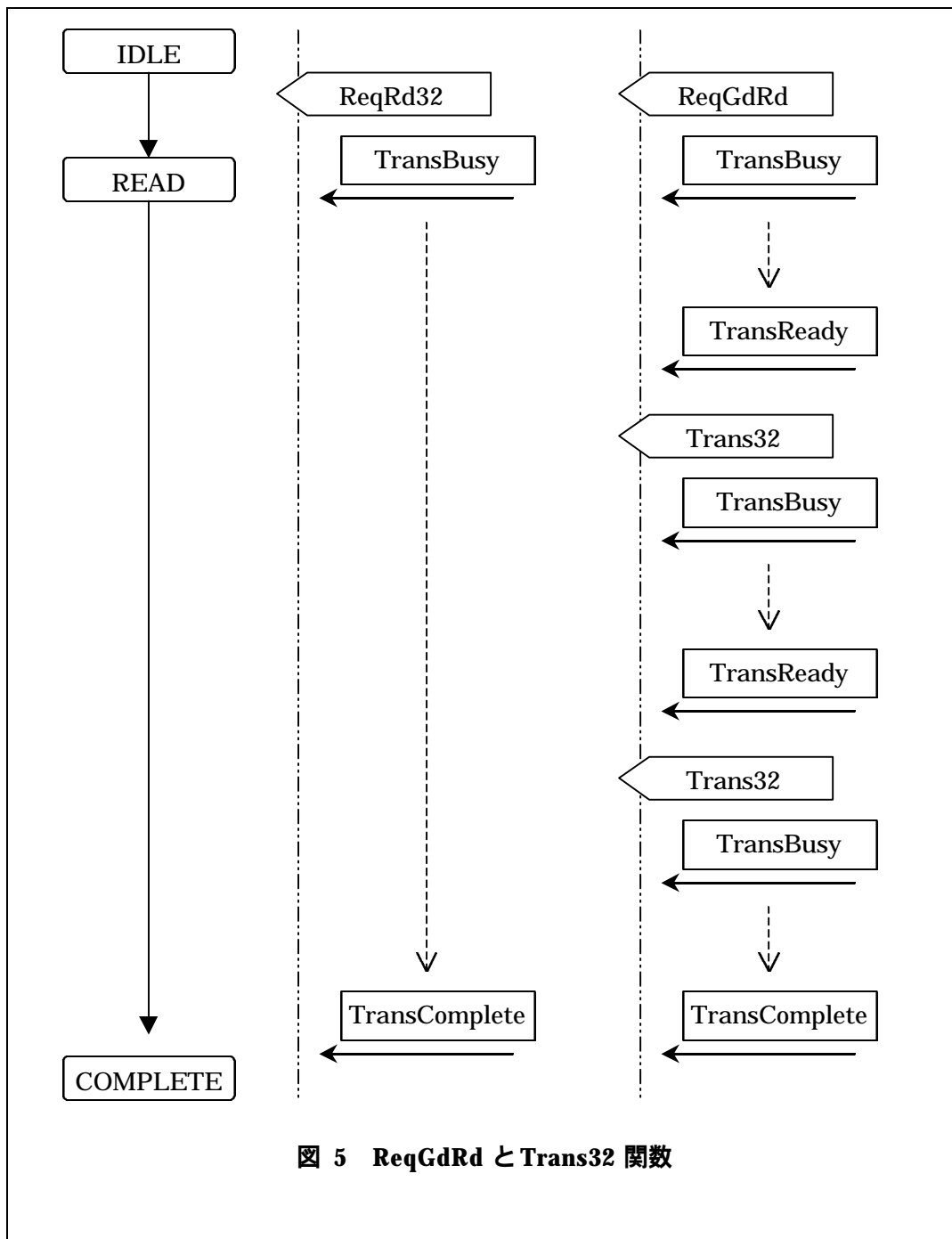


図 4 ファイルハンドルの状態遷移

また、主にストリーミング等の用途で、読み込みと転送を分割して行うことができる関数が用意されています。(gdFsReqGdRd 関数と gdFsTrans32 関数)



RegGdRd アクセス例:

```
gdFsReqGdRd(gf, 128);

while ((stat = gdFsGetStat(gf)) == GDD_STAT_READ) {
    if (gdFsGetTransStat(gf) == GDD_TRANS_READY)
        gdFsTrans32(gf, 2048, buf);
}
```

2.5.3 アクセス中の注意点

2.5.3.1 キャッシュについて

Katana には、16KB のオペランドキャッシュ(OC)と 8KB のインストラクションキャッシュ(IC)が搭載されています。

これらが有効であるとき、次のようなキャッシュ問題が発生する場合があります。

- オーバーレイモジュールなどのように、プログラムを同一アドレスにモジュールを読み直す場合、メモリ上に次のモジュールを読み込んでも、以前のコードがインストラクションキャッシュ(IC)上にあるため、誤動作を起こすことがある。
- データを同一アドレスに読み直す場合、たまたまオペランドキャッシュ(OC)上に残ったデータがあると、メモリ上のデータを書き換えても、キャッシュ上のデータが読めてしまうことがある。

これらのことを防ぐには、読み込みをする前にインストラクションキャッシュ(IC)、またはオペランドキャッシュ(OC)の該当部分をクリアする必要がある。

しかし、ファイルシステムでは、読むべきファイルが、プログラムであるか、データであるかは、判断がつかねます。

そこで、ファイルシステムではオペランドキャッシュ(OC)のコヒーレンシのみを保証するように設計されています。

具体的には、ファイルの読み込みリクエストが受理された時点で、該当メモリ空間のキャッシュをインバリデート(invalidate)します。

しかし、インバリデートしたあと、実データがメモリ上に転送される前に、該当メモリにアクセスした場合、データがキャッシュに読み込まれてしまいます。

この理由により、読み込みを始めてから、完了するまでは該当メモリのアクセスを禁止とさせていただきます。

2.6 ドライブ情報取得

ドライブ情報取得関数を使用することによって、ドライブ状態を取得したり、TOC を取得することができます。

GD トレイが空いているかどうかは、このドライブ状態取得関数で取得することができます。

ドライブ状態取得例:

```
Sint32 dstat;  
  
dstat = gdFsGetDrvStat();  
  
if (dstat == GDD_DRVSTAT_OPEN) {  
    /* abort 処理 */  
}
```

ただし、ドライブ状態はドライブにアクセスするコマンドを発行した後に更新されます。長期間ドライブにアクセスしていない場合は、ドライブ情報更新関数 (gdFsReqDrvStat)を使用して、状態を更新した後でドライブ状態を取得してください。

また、TOC を取得することによって、各トラックの情報を取得することができます。これを調べると DA(Digital-Audio)トラックを識別することができます。

TOC 取得例:

```
Sint32 tocbuf[102];  
  
gdFsGetToc(1, &tocbuf);
```

TOC は GD の単密エリア、高密エリアとそれぞれ別々に取得できます。通常ゲームで使う DA トラックは高密エリアのものをしますが、特殊アプリケーション等で単密の DA を再生することは可能です。以下にその構造を示します。

バイト	内 容
0-3	トラック 1 情報
4-7	トラック 2 情報
	...
392-395	トラック 99 情報
396-399	先頭トラック情報
400-403	最終トラック情報
404-407	リードアウト情報

図 6 TOC データフォーマット

上位	下位			
1 st Byte		2 nd byte	3 rd byte	4 th byte
Ctr	0x1	トラック開始 FAD		
Ctr	0x1	先頭 トラック番号	0x00	0x00
Ctr	0x1	終了 トラック番号	0x00	0x00
Ctr	0x1	リードアウト開始 FAD		

図 7 TOC データフォーマットの各情報の詳細

Control Field	Description
00x0	2 Audio without Pre-emphasis
00x1	2 Audio with Pre-emphasis
0x0x	Copy Prohibited
0x1x	Copy Permitted
01x0	Digital Data
1xxx	Broadcast Data

図 8 Control(Ctr)ビットの値

2.7 コールバック登録

本ライブラリでは、以下のコールバックを用意しています。

- 読み込み完了コールバック
- 転送終了コールバック
- エラー発生コールバック

それぞれ、ライブラリ利用者が用意したコールバック関数を登録する関数として、gdFsEntry~関数を用意しています。

これらのコールバック関数は割り込みの中から呼ばれるため、処理はなるべく単純、簡潔なものにしてください。

コールバック関数が長い時間処理を占有してしまうと、他の処理に悪影響を与える可能性があります。十分注意してください。

なお、転送終了コールバック内では、gdFsTrans32 関数および、gdFsStopRd 関数以外のアクセス関数は使用できません。

2.8 Digital-Audio 再生

DA (Digital-Audio)再生に関しては次の機能を用意しています。

- トラック指定再生
- セクタ指定再生
- 再生停止
- 再生ポーズ
- 再生再開

これらはそれぞれ gdFsDa~関数に該当します。

DA 再生中にファイルの読み込み等の処理を行った場合は、再生が強制中断されます。
また、ファイルの読み込み中に DA を再生した場合は無効となります。

2.9 サーバー関数

本ライブラリでは、リクエストされた処理をバックグラウンド実行するために、サーバー関数を用意しています。

しかし、このサーバー関数はライブラリ初期化時に自動的に VblankIn 割り込みに登録されます。
したがって、ライブラリ利用者は、特に必要が無い限り使用する必要はありません。

3. データ仕様

3.1 定数

定数	GDD_SEEK_~	シークモード
----	------------	--------

定 数 名	説 明
GDD_SEEK_SET	ファイルの先頭
GDD_SEEK_CUR	ファイルの現在の読み込み位置
GDD_SEEK_END	ファイルの終端

定数	GDD_STAT_~	ステータス
----	------------	-------

定 数 名	説 明
GDD_STAT_IDLE	アイドル
GDD_STAT_COMPLETE	動作完了
GDD_STAT_READ	読み込み中
GDD_STAT_ERR	エラー発生
GDD_STAT_FATAL	致命的エラー発生

定数	GDD_DRVSTAT_~	ドライブステータス
----	---------------	-----------

定 数 名	説 明
GDD_DRVSTAT_BUSY	稼働中
GDD_DRVSTAT_PAUSE	ポーズ状態
GDD_DRVSTAT_STANDBY	スタンバイ状態
GDD_DRVSTAT_PLAY	再生中
GDD_DRVSTAT_SEEK	シーク中
GDD_DRVSTAT_SCAN	スキャン再生中
GDD_DRVSTAT_OPEN	トレイオープン
GDD_DRVSTAT_NODISC	ディスクがない
GDD_DRVSTAT_RETRY	リトライ
GDD_DRVSTAT_ERROR	エラー

定数	GDD_FS_TRANS_~	転送状態
----	----------------	------

定 数 名	説 明
GDD_FS_TRANS_READY	転送可能
GDD_FS_TRANS_BUSY	転送中
GDD_FS_TRANS_COMPLETE	転送完了
GDD_FS_TRANS_ERROR	転送エラー

定数	GDD_ERR_~	エラーコード
----	-----------	--------

定 数 名	説 明
GDD_ERR_OK	正常終了
GDD_ERR_INIT	ライブラリ初期化に失敗しました
GDD_ERR_RESET	ドライブの初期化に失敗しました
GDD_ERR_LIBOV	オーバーレイの初期化に失敗しました
GDD_ERR_MOUNT	マウントできませんでした
GDD_ERR_DISC	不正なディスクを使用しました
GDD_ERR_DIRREC	不正なディレクトリレコードハンドルを使用しました
GDD_ERR_CANTOPEN	ファイルがオープンできませんでした
GDD_ERR_NOTFOUND	ファイルが見つかりませんでした
GDD_ERR_NOHNDL	未使用のハンドルが見つかりませんでした
GDD_ERR_ILLHNDL	不正なハンドルを使用しました
GDD_ERR_NOTDIR	ディレクトリではありません
GDD_ERR_DIROVER	格納できるエントリー数を超過しました
GDD_ERR_BUSY	実行中です
GDD_ERR_32ALIGN	32 バイト境界からはずれています
GDD_ERR_SIZE	32 バイト単位ではありません
GDD_ERR_SEEK	シークの指定が不正です
GDD_ERR_OFS	指定された位置が不正です
GDD_ERR_ILLTMODE	転送モードが正しくありません
GDD_ERR_READ	読み込みに失敗しました
GDD_ERR_NOTREAD	読み込み中ではありません
GDD_ERR_TOUT	タイムアウトしました
GDD_ERR_EOF	ファイルの終端に達しました
GDD_ERR_TRAYOPEND	トレイがオープンされています
GDD_ERR_SIZEOVER	要求サイズが大きすぎます
GDD_ERR_FATAL	致命的なエラーが発生しました
GDD_ERR_UNDEF	未定義なエラーです
GDD_ERR_NOERR	エラーはありませんでした
GDD_ERR_RECOVER	エラーが発生しましたが復歸しました
GDD_ERR_NOTREADY	ドライブの準備ができていません
GDD_ERR_MEDIA	メディアエラー
GDD_ERR_HWARE	ハードウェアエラー
GDD_ERR_ILLREQ	不正なリクエストを発行しました
GDD_ERR_UNITATTENT	媒体の交換を検出しました
GDD_ERR_PROTECT	プロテクトされています
GDD_ERR_ABORT	中断されました
GDD_ERR_NOREADABLE	読み込みできません

定数	GDD_FF_~	ファイルフラグ
----	----------	---------

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Multi Extent	****	****	Protection	Record	Associated	Directory	Existence

3.2 データ型

データ型	GDFS	ファイルハンドル
------	------	----------

データ型	GDFS_DIRREC	ディレクトリレコードハンドル
------	-------------	----------------

データ型	GDFS_DIRINFO	ディレクトリ情報
------	--------------	----------

	型	名 前	説 明
	Sint32	fad	先頭 FAD
	Sint32	fsize	ファイルサイズ
	Uint8	flag	ファイルフラグ
	Uint8	pad[3]	予約

データ型	GDFS_FUNC	コールバック関数
------	-----------	----------

	型	名 前	説 明
	void *	obj	コールバック関数に渡される第一引数
関数例	void gdfs_func(void *obj)		

データ型	GDFS_ERRFUNC	エラーコールバック関数
------	--------------	-------------

	型	名 前	説 明
	void *	obj	コールバック関数に渡される第一引数
	Sint32	errcode	エラーコード
関数例	void gdfs_errfunc(void *obj, Sint32 errcode)		

データ型	GDFS_DAINFO	DA 再生情報
------	-------------	---------

	型	名 前	説 明
	Sint32	track	トラック番号
	Sint32	min	経過時間(分)
	Sint32	sec	経過時間(秒)
	Sint32	frame	経過時間(フレーム)
	Sint32	fad	フレームアドレス

4. 関数仕様

4.1 関数一覧

ライブラリ初期化・終了関数		
	gdFsInit	ライブラリの初期化
	gdFsReinit	ライブラリの再初期化
	gdFsFinish	ライブラリの終了
	gdFsGetWorkSize	ライブラリワークサイズ取得 (macro)
	gdFsGetDirrecSize	ディレクトリレコードバッファサイズ取得 (macro)
ディレクトリ操作関数		
	gdFsCreateDirhn	ディレクトリハンドルの生成
	gdFsLoadDir	ディレクトリの読み取り
	gdFsSetDir	カレントディレクトリを設定
	gdFsChangeDir	カレントディレクトリの変更
	gdFsGetDirInfo	ファイルの情報を取得
ファイル操作関数		
	gdFsOpen	ファイルのオープン
	gdFsOpenRange	ファイルのセクタ指定オープン
	gdFsClose	ファイルのクローズ
	gdFsSeek	ファイルの読み出し位置変更
	gdFsTell	ファイルの次の読み出し位置を取得
	gdFsCheckEof	ファイルが EOF に到達したかを調べる
	gdFsGetFileSize	ファイルサイズの取得
	gdFsGetFileSctSize	ファイルのセクタサイズ取得
	gdFsGetFad	ファイルの FAD を取得
	gdFsCalcSctSize	バイト数からセクタ数を算出 (macro)
完了復帰関数		
	gdFsRead	ファイルの読み込み
即時復帰関数		
	gdFsReqRd32	ファイルの読み込み
	gdFsGetStat	ファイルの状態取得
	gdFsGetErrStat	ファイルのエラー状態取得
	gdFsGetNumRd	ファイルの読み込み済みサイズの取得
	gdFsStopRd	ファイルの読み込み中断
	gdFsMovePickup	ピックアップ (読取装置) の移動
	gdFsReqGdRd	GD バッファへの先読みのリクエスト発行
	gdFsTrans32	GD バッファからの転送指定
	gdFsGetTransStat	転送状態を取得
	gdFsGetWorkHn	現在実行中のハンドルの取得
	gdFsGetSysHn	システムハンドルの取得

ドライブ情報取得関数		
	gdFsGetToc	TOC の取得
	gdFsGetDrvStat	ドライブの状態の取得
	gdFsReqDrvStat	ドライブの状態を更新させる

コールバック登録関数		
	gdFsEntryRdEndFunc	読み込み終了時のコールバック関数を登録
	gdFsEntryTrEndFunc	転送終了時のコールバック関数を登録
	gdFsEntryErrFunc	エラー発生時のコールバック関数を登録

Digital-Audio 再生関数		
	gdFsDaPlay	DA のトラック再生
	gdFsDaPlaySct	DA のセクタ指定再生
	gdFsDaStop	DA の再生停止
	gdFsDaPause	DA の再生の一時停止
	gdFsDaRelease	DA の再生の一時停止解除
	gdFsDaGetInfo	DA の再生情報を取得

サーバー関数		
	gdFsExecServer	サーバー関数

4.2 関数 (API)

ライブラリ初期化・終了関数

関数 (D) ¹	gdFsInit	ライブラリの初期化
関数	Sint32 gdFsInit(Sint32 max_open, void *gdfs_work, Sint32 max_dirent, void *dirbuf)	
入力	max_open gdfs_work max_dirent dirbuf	同時にオープンできるファイル数 ワークエリアのポインタ (ユーザー領域から提供) カレントディレクトリのエントリ数 カレントディレクトリのバッファ (ユーザー領域から提供)
出力	なし	
戻り値	エラーコード この関数で返される主なエラーコードは以下の通りです。 GDD_ERR_OK GDD_ERR_32ALIGN GDD_ERR_RESET GDD_ERR_TRAYOPEND GDD_ERR_DISC GDD_ERR_MOUNT GDD_ERR_DIROVER	初期化完了 gdfs_work が 32 バイト境界にない ドライブのリセットに失敗した GD トレイが空いている このディスクは扱えません マウントに失敗した ルートディレクトリのエントリー数が多すぎる
機能	ライブラリを初期化します	
使用例	Uint32 gdfswork[gdFsGetWorkSize(8)/4]; Uint32 gdfscurdir[gdFsGetDirrecSize(64)/4]; gdFsInit(8, gdfswork, 64, gdfscurdir);	
備考	gdfs_work は 32 Byte align でなければなりません。 (dirbuf は 4 Byte align で結構です。) 一度初期化が完了した後、gdFsFinish 関数を呼び出さずに gdFsInit 関数を呼び出しても再初期化されません。 gdFsInit 関数で処理される主な内容は、以下のとおりです。 1. GD Driver の初期化 2. ワークエリアの初期化 3. デバイスの初期化 4. マウント処理	

¹ (D) ドライブに対して実際にアクセスがある関数を示す

関数 (D)	gdFsReinit	ライブラリの再初期化
関数	Sint32 gdFsReinit(void)	
入力	なし	
出力	なし	
戻り値	エラーコード	
機能	ライブラリの再初期化をします。 デバイスの初期化からマウント処理までを行います。 メディアの交換時などに使用してください。	
使用例	gdFsReinit();	

関数	gdFsFinish	ライブラリの停止
関数	void gdFsFinish(void)	
入力	なし	
出力	なし	
戻り値	なし	
機能	ライブラリの使用を停止します	
使用例	<pre> Uint32 gdfswork[gdFsGetWorkSize(8)/4]; Uint32 gdfscurdir[gdFsGetDirrecSize(64)/4]; gdFsInit(8, gdfswork, 64, gdfscurdir); gdFsFinish(); </pre>	

マクロ	gdFsGetWorkSize	ワークエリアのサイズを取得
マクロ	gdFsGetWorkSize(max_open)	
引数	max_open	同時にオープンできるファイル数
マクロ値	必要なライブラリワークエリアのバイト数	
機能	ライブラリのワークエリアの必要とするバイト数を求めます(マクロ関数)	
使用例	<pre> Uint32 gdfswork[gdFsGetWorkSize(8)/4]; </pre>	

マクロ	gdFsGetDirrecSize	ディレクトリバッファのサイズを取得
マクロ	gdFsGetDirrecSize(max_dirent)	
引数	max_dirent	ディレクトリの最大エントリ数
マクロ値	必要なディレクトリバッファのバイト数	
機能	ディレクトリバッファのバイト数を求めます(マクロ関数)	
使用例	<pre> Uint32 gdfscurdir[gdFsGetDirrecSize(64)/4]; </pre>	

ディレクトリ操作関数

関数	gdFsCreateDirhn	ディレクトリハンドルを生成
関数	GDFS_DIRREC gdFsCreateDirhn(void *dirbuf, Sint32 max_dirent)	
入力	dirbuf	ディレクトリのバッファ
	max_dirent	ディレクトリのエントリ数
出力	なし	
戻り値	ディレクトリレコードハンドル	
機能	ディレクトリハンドルを生成します	
使用例	<pre> Uint32 dirbuf[gdFsGetDirrecSize(64)]; GDFS_DIRREC g_dir; g_dir = gdFsCreateDirhn(dirbuf, 64); </pre>	

関数 (D)	gdFsLoadDir	ディレクトリレコードを読み込む
関数	Sint32 gdFsLoadDir(const char *dirname, GDFS_DIRREC gf_dirrec)	
入力	dirname	ディレクトリ名
出力	gf_dirrec	ディレクトリレコードハンドル (NULL の場合はカレントディレクトリへ)
戻り値	エラーコード	
機能	ディレクトリレコードを読み込みます(大小文字同一視)	
使用例	<pre> /* サンプル 1 */ /* MOVIE directory を g_dir へ読み取る */ Uint32 dirbuf[gdFsGetDirrecSize(64)]; GDFS_DIRREC g_dir; g_dir = gdFsCreateDirhn(dirbuf, 64); gdFsLoadDir("MOVIE", g_dir); /* サンプル 2 */ /* カレントディレクトリを DATA directory へ移動する */ gdFsLoadDir("DATA", NULL); </pre>	

関数	gdFsSetDir	カレントディレクトリを設定
----	------------	---------------

関数 Sint32 gdFsSetDir(GDFS_DIRREC gf_dirrec)
 入力 gf_dirrec ディレクトリレコードハンドル
 出力 なし
 戻り値 エラーコード

機能 カレントディレクトリを設定します。

使用例

```
/* MOVIE directory を g_dir へ読み取る */
Uint32 dirbuf[gdFsGetDirrecSize(64)];
GDFS_DIRREC g_dir;

g_dir = gdFsCreateDirhn(dirbuf, 64);
gdFsLoadDir("MOVIE", g_dir);
gdFsSetDir(g_dir);
```

関数 (D)	gdFsChangeDir	カレントディレクトリを変更
--------	---------------	---------------

関数 Sint32 gdFsChangeDir(const char *dirname)
 入力 dirname ディレクトリ名
 出力 なし
 戻り値 エラーコード

機能 カレントディレクトリを変更します (大小文字同一視)

使用例

```
/* カレントディレクトリを DATA directory へ移動する */
gdFsChangeDir("DATA");
```

関数	gdFsGetDirInfo	ファイルの情報を取得
----	----------------	------------

関数 Sint32 gdFsGetDirInfo(const char *name, GDFS_DIRINFO dirinfo)
 入力 name ファイル名 / ディレクトリ名
 出力 dirinfo ファイル情報
 戻り値 エラーコード

機能 ファイルの情報を取得します

使用例

```
GDFS_DIRINFO dinfo;
gdFsGetDirInfo("ABC.DAT", &dinfo);
```

ファイル操作関数

関数	gdFsOpen	ファイルのオープン
関数	GDFS	gdFsOpen(const char *fname, GDFS_DIRREC gf_dirrec)
入力	fname	ファイル名
	gf_dirrec	ファイル名を検索するディレクトリレコードハンドル (NULL の場合はカレントから検索)
出力	なし	
戻り値	NULL	... 失敗
	それ以外	... ファイルハンドル
機能	ファイルをオープンします	
使用例	<pre> /* サンプル 1 */ GDFS gf; gf = gdFsOpen("A.BIN", NULL); /* サンプル 2 */ GDFS gf; Uint32 dirbuf[gdFsGetDirSize(64)]; GDFS_DIRREC g_dir; g_dir = gdFsCreateDirhn(dirbuf, 64); gdFsLoadDir("MOVIE", g_dir); gf = gdFsOpen("SMP.MOV", g_dir); </pre>	

関数	gdFsOpenRange	セクタ指定オープン
関数	GDFS	gdFsOpenRange(Sint32 stsct, Sint32 nsct)
入力	stsct	開始セクタ
	nsct	セクタサイズ
出力	なし	
戻り値	NULL	... 失敗
	それ以外	... ファイルハンドル
機能	セクタを指定してファイルをオープンします。	
使用例	<pre> GDFS gf; gf = gdFsOpenRange(0xB555, 0x10); gdFsClose(gf); </pre>	
注意	単密領域のファイルをオープンする事はできません。	

関数	gdFsClose	ファイルのクローズ
----	-----------	-----------

関数 void gdFsClose(GDFS gdfs);
 入力 gdfs ファイルハンドル
 出力 なし
 戻り値 なし

機能 ファイルをクローズします

使用例 GDFS gf;

 gf = gdFsOpen("TEST.BIN", NULL);
 gdFsClose(gf);

関数	gdFsSeek	読み込みポインタの移動
----	----------	-------------

関数 Sint32 gdFsSeek(GDFS gdfs, Sint32 sctno, Sint32 type)
 入力 gdfs ファイルハンドル
 sctno セクタ番号
 type GDD_SEEK_SET, _CUR, _END
 出力 なし
 戻り値 エラーコード

機能 ファイルの読み込みポインタを移動します

使用例 GDFS gf;

 gf = gdFsOpen("TEST.BIN", NULL);

 /* 先頭から 5 セクタ目にシークする */
 gdFsSeek(gf, 4, SEEK_SET);

関数	gdFsTell	読み込みポインタの位置を取得
----	----------	----------------

関数 Sint32 gdFsTell(GDFS gdfs);
 入力 gdfs ファイルハンドル
 出力 なし
 戻り値 読み込み位置

機能 ファイルの読み込みポインタの位置を取得します（セクタ単位）

使用例 Sint32 pos;
 GDFS gf;

 gf = gdFsOpen("TEST.BIN", NULL);

 pos = gdFsTell(gf);

関数	gdFsCheckEof	ファイルの終端のチェック
----	--------------	--------------

関数 Sint32 gdFsCheckEof(GDFS gdfs, Bool *iseof)
 入力 gdfs ファイルハンドル
 出力 iseof
 戻り値 エラーコード

機能 ファイルが EOF に到達したかを調べます。

使用例 GDFS gf;
 Bool flag;
 Sint32 ret;

 gf = gdFsOpen("TEST.BIN", NULL);

 /* 読み込み処理の後 */

 ret = gdFsCheckEof(gf, &flag);
 if (ret == GDD_ERR_OK && flag == TRUE) {
 gdFsClose(gf);
 return;
 }

関数	gdFsGetFileSize	ファイルサイズを取得
----	-----------------	------------

関数 Bool gdFsGetFileSize(GDFS gdfs, Sint32 *fsize)
 入力 gdfs ファイルハンドル
 出力 fsize ファイルサイズ (バイト)
 戻り値 エラーコード

機能 ファイルサイズを取得します

使用例 GDFS gf;
 Sint32 flen;

 gf = gdFsOpen("TEST.BIN", NULL);
 gdFsGetFileSize(gdfs, &flen);

関数	gdFsGetFileSctSize	ファイルのセクタサイズを取得
----	--------------------	----------------

関数 Bool gdFsGetFileSctSize(GDFS gdfs, Sint32 *fsctsize)
 入力 gdfs ファイルハンドル
 出力 fsctsize ファイルサイズ (バイト)
 戻り値 エラーコード

機能 ファイルのセクタサイズを取得します

使用例 GDFS gf;
 Sint32 fslen;

 gf = gdFsOpen("TEST.BIN", NULL);
 gdFsGetFileSctSize(gdfs, &fslen);

関数	gdFsGetFad	ファイルの先頭セクタの取得
----	------------	---------------

関数 Bool gdFsGetFad(GDFS gdfs, Sint32 *fad)
 入力 gdfs ファイルハンドル
 出力 fad フレームアドレス (物理セクタ)
 戻り値 エラーコード

機能 ファイルの FAD を取得します

使用例 Sint32 fad;
 Sint32 gf;

 gf = gdFsOpen("TEST.BIN", NULL);
 gdFsGetFad(gf, &fad);

マクロ	gdFsCalcSctSize	バイト数からセクタ数の算出
-----	-----------------	---------------

マクロ gdFsCalcSctSize(bytes)
 入力 bytes ファイルのバイトサイズ
 マクロ値 セクタサイズ

機能 バイト数からセクタ数を算出します (マクロ関数)

使用例 GDFS gf;
 Sint32 flen;
 Sint32 fslen;

 gf = gdFsOpen("TEST.BIN", NULL);
 gdFsGetFileSize(gdfs, &flen);
 fslen = gdFsCalcSctSize(flen);

完了復帰関数

関数 (D)	gdFsRead	ファイルの読み込み
関数	Sint32	gdFsRead(GDFS gdfs, Sint32 nsct, void *buf)
入力	gdfs	ファイルハンドル
	nsct	読み込むセクタ数
出力	buf	格納バッファ
戻り値		エラーコード
機能	ファイルを読み込みます（完了復帰）	
使用例	<pre>GDFS gf; Sint32 buf[32*2048/4]; gf = gdFsOpen("TEST.BIN", NULL); gdFsRead(gf, 32, buf); gdFsClose(gf);</pre>	

即時復帰関数

関数 (D)	gdFsReqRd32	ファイルの読み込み
関数	Sint32	gdFsReqRd32(GDFS gdfs, Sint32 nsct, void *buf)
入力	gdfs	ファイルハンドル
	nsct	読み込むセクタ数
出力	buf	格納バッファ
戻り値	実際にリクエストされたセクタ数 (> 0)。負のときはエラーコード	
機能	ファイルの読み込みリクエストします (即時復帰)	
注意	buf は 32 bytes align にする必要があります	
備考	リクエストの受付は同時に 1 つのみ	
使用例	<pre>GDFS gf; Uint32 buf[32*2048/4]; gf = gdFsOpen("TEST.BIN", NULL); gdFsReqRd32(gf, 32, buf); while (gdFsGetStat(gf) == GDD_STAT_READ); gdFsClose(gf);</pre>	

関数	gdFsGetStat	ハンドルの状態取得
関数	Sint32	gdFsGetStat(GDFS gdfs)
入力	gdfs	ファイルハンドル
出力	なし	
戻り値	ステータス	
機能	ハンドルの状態を取得します	
使用例	<pre>GDFS gf; Uint32 buf[32*2048/4]; gf = gdFsOpen("TEST.BIN", NULL); gdFsReqRd32(gf, 32, buf); while (gdFsGetStat(gf) == GDD_STAT_READ); gdFsClose(gf);</pre>	

関数	gdFsGetErrStat	ハンドルのエラー状態取得
----	----------------	--------------

関数 Sint32 gdFsGetErrStat(GDFS gdfs)
 入力 gdfs ファイルハンドル
 出力 なし
 戻り値 エラーコード

機能 ハンドルのエラー状態を取得します

使用例 GDFS gf;
 Uint32 buf[32*2048/4];
 Sint32 stat;
 Sint32 err;

 gf = gdFsOpen("TEST.BIN", NULL);
 gdFsReqRd32(gf, 32, buf);

 while ((stat = gdFsGetStat(gf)) == GDD_STAT_READ);

 if (stat == GDD_STAT_ERR) {
 err = gdFsGetErrStat(gf);
 /* error 処理 */
 }
 gdFsClose(gf);

関数	gdFsGetNumRd	読み込んだバイト数取得
----	--------------	-------------

関数 Sint32 gdFsGetNumRd(GDFS gdfs);
 入力 gdfs ファイルハンドル
 出力 なし
 戻り値 0 以上 読み込んだバイト数

機能 読み込んだバイト数取得します

使用例 GDFS gf;
 Uint32 buf[32*2048/4];

 gf = gdFsOpen("TEST.BIN", NULL);
 gdFsReqRd32(gf, 32, buf);

 while (gdFsGetStat(gf) != GDD_STAT_COMPLETE) {
 readnum = gdFsGetNumRd(gdfs);
 }

 gdFsClose(gf);

関数 (D)	gdFsStopRd	動作を中断します
--------	------------	----------

関数 Sint32 gdFsStopRd(GDFS gdfs)
 入力 gdfs ... ファイルハンドル
 出力 なし
 戻り値 エラーコード

機能 リクエストを中断します

使用例 gdFsStopRd(gf);

関数 (D)	gdFsMovePickup	ピックアップを移動する
--------	----------------	-------------

関数 Sint32 gdFsMovePickup(GDFS gdfs)
 入力 gdfs ファイルハンドル
 出力 なし
 戻り値 エラーコード

機能 次の読み込み位置に、ピックアップを移動する

使用例 gdFsMovePickup(gf);

関数 (D)	gdFsReqGdRd	読み込みリクエスト
--------	-------------	-----------

関数 Sint32 gdFsReqGdRd(GDFS gdfs, Sint32 nsct)
 入力 gdfs ... ファイルハンドル
 nsct ... 読み込みセクタ数
 出力 なし
 戻り値 実際にリクエストされたセクタ数 (> 0)。負のときはエラーコード

機能 GD バッファへの先読みリクエスト発行

使用例 gdFsReqGdRd(gf, 128);

関数	gdFsTrans32	転送指定
----	-------------	------

関数 Sint32 gdFsTrans32(GDFS gdfs, Sint32 nbytes, void *buf)
 入力 gdfs ... ファイルハンドル
 nbytes ... 転送するバイト数 (32 バイト単位)
 出力 buf ... 転送先アドレス
 戻り値 エラーコード

機能: GD バッファからの転送指定

使用例

```
while ((stat = gdFsGetStat(gf)) == GDD_STAT_READ) {
    if (gdFsGetTransStat(gf) == GDD_TRANS_READY)
        gdFsTrans32(gf, 2048, buf);
}
```

関数	gdFsGetTransStat	転送状態の取得
----	------------------	---------

関数 Sint32 gdFsGetTransStat(GDFS gdfs)
 入力 gdfs ファイルハンドル
 出力 なし
 戻り値 転送状態

機能 転送状態を取得

使用例 gdFsGetTransStat(gf);

関数	gdFsGetWorkHn	動作中のハンドルを取得
----	---------------	-------------

関数 GDFS gdFsGetWorkHn(void)
 入力 なし
 出力 なし
 戻り値 現在実行中のハンドル。なければ NULL。

機能 動作中のハンドルを取得

使用例 GDFS gf;
 gf = gdFsGetWorkHn();

関数	gdFsGetSysHn	システムハンドルを取得
----	--------------	-------------

関数 GDFS gdFsGetSysHn(void)
 入力 なし
 出力 なし
 戻り値 システムハンドル。

機能 システムハンドルを取得
 備考 システムハンドルが実行中かどうかを調査するために使用します。

使用例 GDFS sys;
 sys = gdFsGetSysHn();
 stat = gdFsGetStat(sys);

ドライブ情報取得関数

関数 (D)	gdFsGetToc	TOC を読み込む
--------	------------	-----------

関数	Sint32 gdFsGetToc(Sint32 type, void *buf)
入力	type ... TOC のタイプ (0: 単密, 1: 高密)
出力	buf ... TOC を読み込むためのバッファ (必要バイト数 408 バイト)
戻り値	エラーコード
機能	TOC の取得します
使用例	gdFsGetToc(1, &buf);
注意	単密の TOC をアクセスする機能は無くなりました。

関数	gdFsGetDrvStat	ドライブの状態を取得
----	----------------	------------

関数	Sint32 gdFsGetDrvStat(void)
入力	なし
出力	なし
戻り値	ドライブステータス
機能	ドライブ状態の取得します
使用例	Sint32 dstat; dstat = gdFsGetDrvStat();

関数 (D)	gdFsReqDrvStat	ドライブの状態を更新させる
--------	----------------	---------------

関数	Sint32 gdFsReqDrvStat(void)
入力	なし
出力	なし
戻り値	エラーコード
機能	ドライブ状態を更新させます
使用例	Sint32 dstat; gdFsReqDrvStat(); /* リクエスト動作完了の後 */ dstat = gdFsGetDrvStat();

コールバック登録関数

関数	gdFsEntryRdEndFunc	読み込み完了時のコールバックを指定
----	--------------------	-------------------

関数 void gdFsEntryRdEndFunc(GDFS gdfs, GDFS_FUNC func, void *obj);
入力 gdfs ファイルハンドル
 func コールバック関数
 obj コールバック関数に渡す第 1 引数
出力 なし
戻り値 なし

機能 読み込み完了時のコールバック関数を登録します

使用例 void rdend_callback(void *obj)
 { /* 処理 */
 }

 gdFsEntryRdEndFunc(gf, rdend_callback, (void *) 0x1234);

関数	gdFsEntryTrEndFunc	転送完了時のコールバックを指定
----	--------------------	-----------------

関数 void gdFsEntryTrEndFunc(GDFS gdfs, GDFS_FUNC func, void *obj)
入力 gdfs ファイルハンドル
 func コールバック関数
 obj コールバック関数にわたす第一引数
出力 なし
戻り値 なし

機能 転送完了時のコールバック関数を登録します

使用例 void trend_callback(void *obj)
 { /* 処理 */
 }

 gdFsEntryTrEndFunc(gf, trend_callback, (void *) 0x1234);

関数	gdFsEntryErrFunc	エラー発生時のコールバックを指定
----	------------------	------------------

関数 void gdFsEntryErrFunc(GDFS gdfs, GDFS_ERRFUNC func, void *obj)
入力 gdfs ファイルハンドル
 func コールバック関数
 obj コールバック関数にわたす第一引数
出力 なし
戻り値 なし

機能 エラー発生時のコールバック関数を登録します

使用例 void err_callback(void *obj, Sint32 errcode)
 { /* 処理 */
 }

 gdFsEntryErrFunc(gf, err_callback, (void *) 0x1234);

Digital-Audio 再生関数

関数 (D)	gdFsDaPlay	DA 再生開始
--------	------------	---------

関数	Sint32 gdFsDaPlay(Sint32 st_track, Sint32 end_track, Sint32 reptime)
入力	st_track 開始トラック番号 end_track 終了トラック番号 reptime 繰り返し回数 (0-14: 繰り返し回数, 15: 無限ループ)
出力	なし
戻り値	エラーコード
機能	DA 再生を開始します
使用例	gdFsDaPlay(4, 4, 0);
注意	単密領域のオーディオトラック(CD-DA)は再生できません。

関数 (D)	gdFsDaPlaySct	セクタ指定DA 再生開始
--------	---------------	--------------

関数	Sint32 gdFsDaPlaySct(Sint32 st_sct, Sint32 end_sct, Sint32 reptime)
入力	st_sct 開始セクタ end_sct 終了セクタ reptime 繰り返し回数 (0-14: 繰り返し回数, 15: 無限ループ)
出力	なし
戻り値	エラーコード
機能	DA 再生をセクタ指定で開始します
使用例	gdFsDaPlaySct(0xC000, 0xD000, 2);
注意	単密領域のオーディオトラック(CD-DA)は再生できません。

関数 (D)	gdFsDaStop	DA 再生停止
--------	------------	---------

関数	Sint32 gdFsDaStop(void)
入力	なし
出力	なし
戻り値	エラーコード
機能	DA 再生を停止させます
使用例	gdFsDaStop();

関数 (D)	gdFsDaPause	DA 再生一時停止
--------	-------------	-----------

関数 Sint32 gdFsDaPause(void)
 入力 なし
 出力 なし
 戻り値 エラーコード

機能 DA 再生の一時停止

使用例 gdFsDaPause();

関数 (D)	gdFsDaRelease	DA 再生一時停止解除
--------	---------------	-------------

関数 Sint32 gdFsDaRelease(Sint32 reptime)
 入力 reptime くり返し回数
 出力 なし
 戻り値 エラーコード

機能 DA 再生の一時停止解除

使用例 gdFsDaRelease(0);

備考 現在 reptime は無効です。

関数 (D)	gdFsDaGetInfo	DA 再生情報を取得
--------	---------------	------------

関数 Sint32 gdFsDaGetInfo(GDFS_DAINFO *dainfo)
 入力 なし
 出力 dainfo DA 再生情報
 戻り値 エラーコード

機能 DA 再生情報を取得

使用例 gdFsDaRelease(dainfo);

備考 システムハンドルを取得しステータスを監視してください。
 GDD_STAT_COMPLETE になった時だけ情報が格納されます。

サーバー関数

関数	gdFsExecServer	サーバー関数
----	----------------	--------

関数 void gdFsExecServer(void)

入力 なし

出力 なし

戻り値 なし

機能 サーバー関数

使用例 gdFsExecServer();

備考 ライブラリ初期化時に VBlank In 割り込みに自動的に登録されます。

処理内容は主に次の通りです。

1. GD Driver 関数の起動
(ドライブに対する実際の処理を行います)
2. ステータスの取得