SATA-IP exFAT リファレンス・デザイン説明書

Rev1.2J 209/13/2013

本書は DesignGateway 社製 SATA-IP コアのオプション製品である exFAT リファレンス・デザインに関する説明書となります。 本 exFAT デザインは Altera 向けおよび Xilinx 向けの両方で共通したものとなっております。

1 exFAT 概要

exFAT ファイル・システムは FAT32 に続く次世代の FAT システムです。 FAT32 よりいくつかの点で改良が加えられており、4GB 以上のファイルをサポートします。

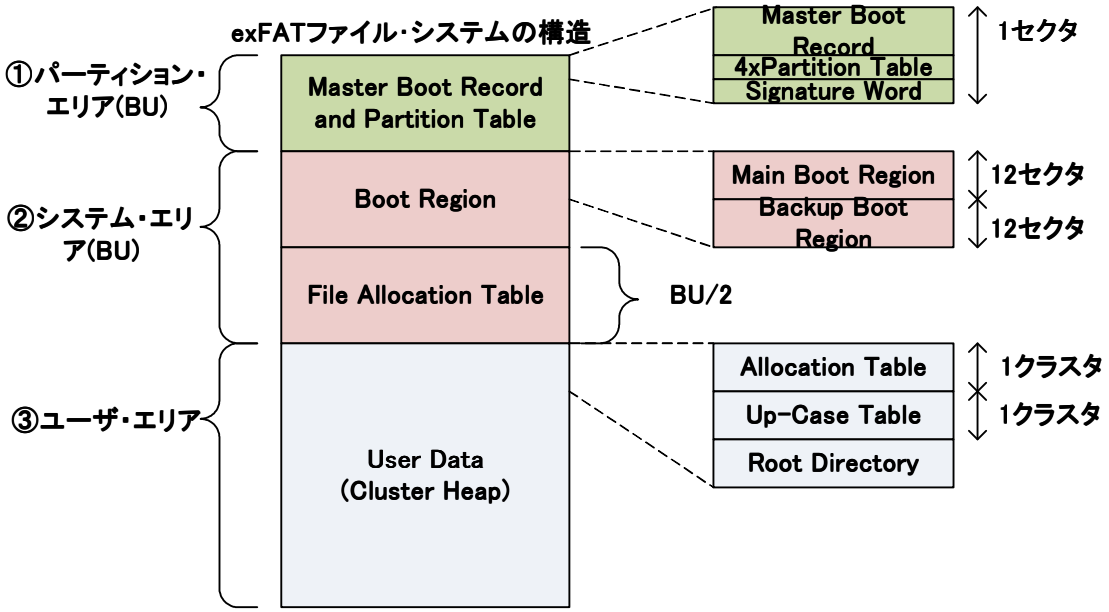

図 1-1: exFAT ファイル・システムのフォーマット・レイアウト一例

exFAT は図 1-1 に示すように単一のパーティション・テーブルをベースとしたフォーマットとなります。 ディスク領域は3つのゾーンに分けられ、①パーティション・エリア、②システム・エリア、③ユーザ・エリアとなります。 パーティション・エリアおよびシステム・エリアは等しいサイズで、バウンダリ・ユニット(BU)と呼ばれます。 バウンダリ・ユニットとして推奨されるサイズを表 1-1 に示します。 ファイルあるいはディレクトリに保存するデータの最小単位はクラスタと呼ばれますが、そのサイズはディスクの全容量に依存します。 表 1-1 から明らかですがバウンダリ・ユニットはクラスタ・サイズの 128 倍となります。

ディスク全容量	1 クラスタ当たりのセクタ数	バウンダリ・ユニット(セクタ数)
32 ~ 128 GB	256	32768
~ 512 GB	512	65536
~ 2 TB	1024	131072

表 1-1: 推奨されるバウンダリ・ユニットとクラスタ・サイズ

パーティション・エリア

パーティション・エリアの先頭セクタはマスター・ブート・レコード(MBR)で、CPU の実行コードとパーティションを定義するパーティション・テーブルを格納します。 MBR サイズは 446 バイトでパーティション・テーブルは1個当たり 16 バイトです。FAT の2バイト・シグネチャは AA55h です。パーティション・テーブル16バイトの詳細を図 1-2 に示します。パーティション・エリア内のそれ以外のセクタはアライメントをバウンダリ・ユニット・サイズに調整するためのものです。

	Byte0	Byte1	Byte2	Byte3
0	Boot Indicator	Starting Head	Starting Sector/Cylinder	
1	System ID	Ending Head	Ending Sector/Cylinder	
2	Relative Sector			
3	Total Sector			

図 1-2: パーティション・テーブル

システム・エリア

システム・エリアはブート領域とファイル・アロケーション・テーブル(FAT)の2種類の情報を含みます。 ブート領域はメインおよびバックアップそれぞれで 12 セクタ分の領域を持ちますがその詳細を図 1-3 に示します。 最初のセクタはクラスタ・サイズやパーティション・サイズなどのパラメータ情報を格納します。 12 番目のセクタはブート領域の4バイト・チェックサム結果の繰り返しパターンです。 バックアップのブート領域はメインのブート領域が破損した場合に回復するためのバックアップです。パーティション・エリアと同様ブート領域内のそれ以外のセクタはアライメントをバウンダリ・ユニット・サイズに調整するため未使用です。

exFAT において FAT はフラグメント(分断) 化したファイルのクラスタ・チェイン情報を維持するためだけに使われます。FAT テーブルは従ってフラグメントされていないファイルにおいては更新する必要はありません。 フォーマット後の FAT の例を図 1-4 に示します。 4 バイトの FAT エントリはクラスタ・チェインにおいて現在のクラスタの次となるクラスタ番号を示します。 最初およびその次の FAT エントリはリザーブされその値はそれぞれ 0xFFFFFFF8h および FFFFFFFFh の固定値となります。 一般的には3番目から5番目のエントリの値は FFFFFFFFh となりますが、それはすなわちクラスタ#2-#4 はそれぞれアロケーション・ビットマップ・テーブル、アップケース・テーブル、ルートディレクトリにコピーされていることを意味します。 各エントリ値の意味については表 1-2 を参照してください。

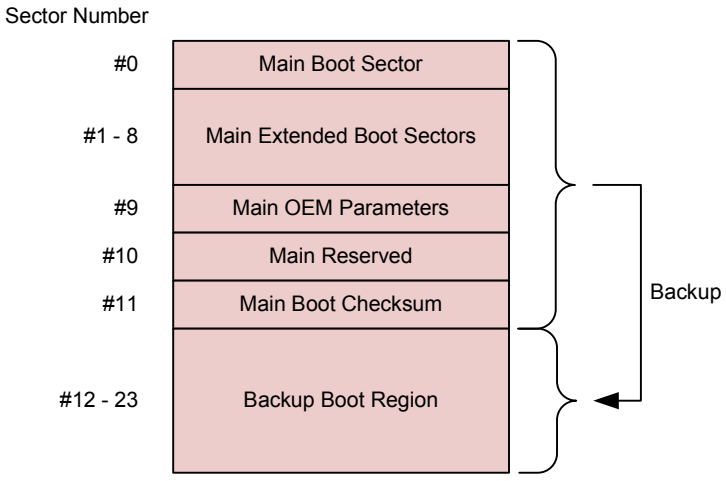

図 1-3: ブート領域

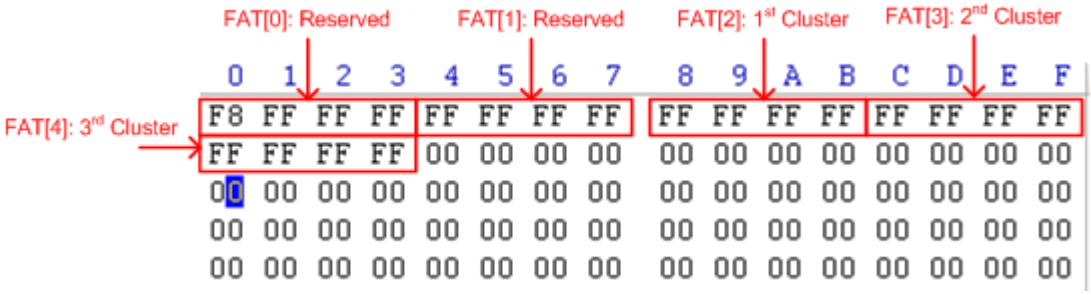

図 1-4: ファイル・アロケーション・テーブルの例

FAT エントリ値	説明
00000000h	クラスタは未使用。フラグメント(分断)されていないファイルやディレクトリに使用可能
00000002h to ClusterCount + 1	クラスタはすでに使用済み。 この値はクラスタ・チェインにて次のクラスタ番号を示す。
FFFFFFF7h	欠陥のある(不良)クラスタ
FFFFFFFFh	クラスタはすでに使用済みでかつクラスタ・チェインにて本クラスタが最終クラスタである。
上記以外	未使用

表 1-2: FAT エントリ値

ユーザ・エリア

ユーザ・エリアで先頭のクラスタ番号はクラスタ#2 であり、アロケーション・ビットマップ・テーブルを格納します。 1ビットが1クラスタ・データに相当するため、先頭バイトはクラスタ#2-#9 の利用ステータスを示します。 クラスタが開放されると、ビット・アロケーション・テーブルの該当クラスタが' 0' となります。 よってビット・アロケーション・テーブルにて' 1' はそのクラスタがすでに使われていることを示します。

クラスタ#3 は小文字を大文字に変換するための大文字変換テーブルを格納しますが、ファイル名の大文字・小文字の区別をしないで処理する exFAT ファイルシステムにおいてユニコードを使ったファイル名ディレクトリ・エントリでは重要です。

クラスタ#4 はルート・ディレクトリです。 各ディレクトリは一連のディレクトリ・エントリから構成されるルート・ディレクトリを含みます。 一つのエントリ・サイズは 32 バイトです。 図 1-5 にディレクトリをひとつ持つルート・ディレクトリの例を示します。それぞれのエントリで最初のバイトはエントリ・タイプを示しますが、この例では6種類のエントリ・タイプがあります。 各エントリ・タイプを表 1-3 で説明します。

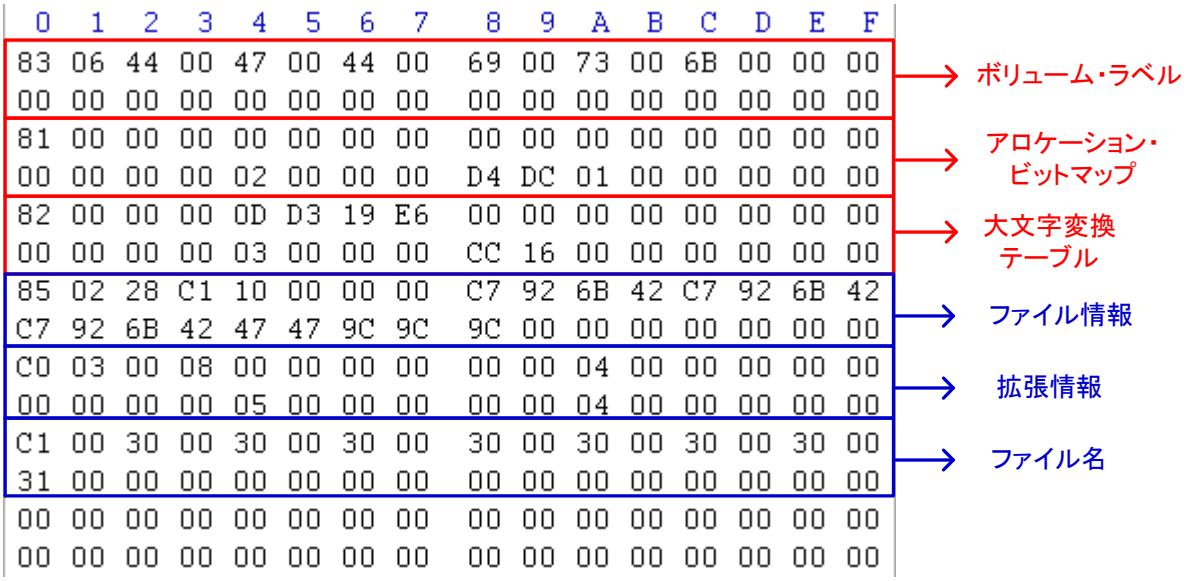

図 1-5: ルート・ディレクトリの各ディレクトリ・エントリ例

エントリ・タイプ	定義	説明
81h	アロケーション・ビットマップ	アロケーション・ビットマップ・テーブルの先頭クラスタ番号と有効長を示す。
82h	大文字変換テーブル	大文字変換テーブルの先頭クラスタ番号と有効長を示す。
83h	ボリューム・ラベル	このボリュームのラベルをユニコード文字列で示す。 ボリュームラベルが定義されていない場合エントリ·タイプは 0x03 にセットされる。
85h	ファイル情報	ファイルとディレクトリを示し、生成日時と編集日時を保持する。 またこのファイル/ディレクトリに続く拡張情報/ファイル名のエントリ・カウントを示す。
C0h	拡張情報	名前の文字数、開始クラスタ番号、このファイル/ディレクトリの有効長を示す。
C1h	ファイル名	ファイルまたはディレクトリの名前を示す。 1エントリ当たり最大 15 ユニコード文字まで保持できるので、255 文字のファイル名をサポートするために各ファイル/ディレクトリの最大ファイル名エントリ数は 17 となる。

表 1-3: FAT エントリ・タイプ

アロケーション・ビットマップ、大文字変換テーブル、ボリュームラベル・ディレクトリはルート・ディレクトリ内にのみ存在する一方、ファイル情報、拡張情報、ファイル名は全てのディレクトリで存在します。

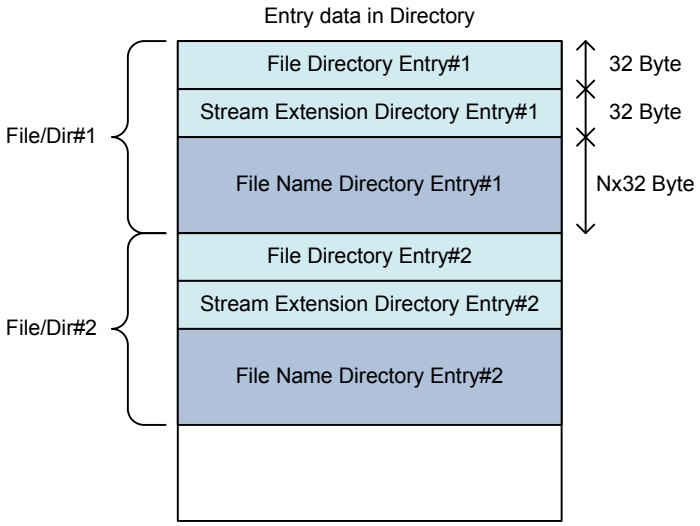

図 1-6: ディレクトリの例

図 1-6 に示すように、あるひとつのファイル又はディレクトリを作成すると、ファイル情報ディレクトリ・エントリを1つ、拡張情報ディレクトリ・エントリを1つ、そして最大17のファイル名ディレクトリ・エントリが親ディレクトリに作成されます。各エントリの詳細を図 1-7～図 1-9 に示します。

	Byte0	Byte1	Byte2	Byte3
0	Entry Type	SecondaryCount	SetChecksum	
1	FileAttributes		Reserved1	
2	CreateTimestamp			
3	LastModifiedTimestamp			
4	LastAccessedTimestamp			
5	Create10msInc	LastMod10msInc	CreateUtcOff	LastModUtcOff
6	LastAccUtcOff	Reserved2		
7				

図 1-7: ファイル情報ディレクトリ・エントリ

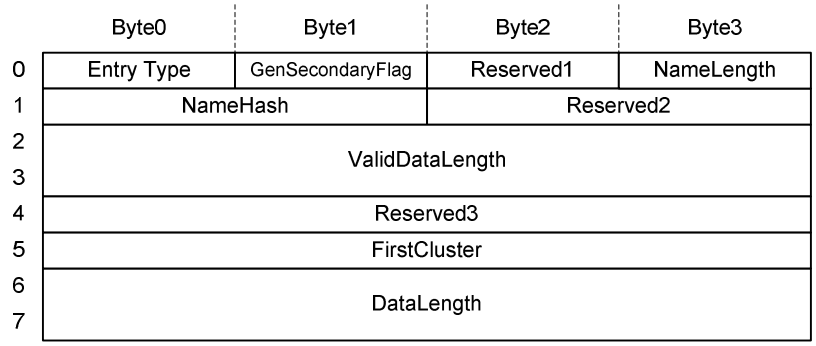

図 1-8:拡張情報ディレクトリ・エントリ

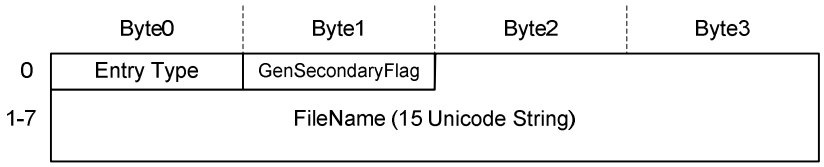

図 1-9: ファイル名ディレクトリ・エントリ

- ファイル情報ディレクトリ・エントリ内の SecondaryCount は拡張情報とファイル名ディレクトリ・エントリの総数を示す。
- 拡張情報ディレクトリ・エントリ内の DataLength はファイル/ディレクトリのサイズをバイト単位で示す。
- 拡張情報ディレクトリ・エントリ内の FirstCluster はファイル/ディレクトリの先頭クラスタ番号を示す。

2　ハードウエアについて

exFAT リファレンス・デザインのハードウエアは Altera 向け/Xilinx 向けとも 1 チャネル SATA ホスト・デザインと完全に同一です。ハードウエアの詳細については、各コアに対応した下表 2-1 のリファレンス・デザイン説明書を参照してください。1 チャネル SATA ホスト・デザインと本 exFAT リファレンス・デザインの違いは CPU のファームウエアのみとなります。

評価ボード	リファレンス・デザイン説明書の参照先 URL
KC-705	http://www.dgway.com/products/IP/SATA-IP/dg_sata_ip_refdesign_host_kt7_jp.pdf
ZC-706	http://www.dgway.com/products/IP/SATA-IP/dg_sata_ip_refdesign_host_z7_jp.pdf
Arria V GX	http://www.dgway.com/products/IP/SATA-IP/Altera/dg_sata3_host_refdesign_ar5_jp.pdf

表 2-1: ホスト・リファレンス・デザイン説明書の参照先 URL リスト

3　ソフトウエアについて

exFAT のソフトウエアは 1 チャネル SATA ホスト・デザインで SATA-IP を通して接続 SATA デバイスを制御する低レベル・ファンクションにて動作します。 本 exFAT デザインを理解するために、1 チャネル SATA ホスト・デザイン説明書を参照することを推奨します。

本リファレンス・デザインは以下の3種類のコードで構成されています。

(1) “sata_host_ctrl.c/.h”: 低レベルで動作する write や read 関数を格納します。
(2) “sata_host.h”: ハードウエアで異なるプラットフォームを吸収するためのパラメータ情報を格納します。
(3) “sata_exfat.c”: exFAT 管理ファンクションおよび、本デモのメイン関数（例えばフォーマットやファイル/ディレクトリの消去・作成・リードなどの機能）を格納します。

本リファレンス・デザインにおいてリード/ライトの各トランザクションでの最大転送サイズは、フォーマット・コマンド実行時に検出したバウンダリ・ユニット・サイズと同じとなっています。 ファイルおよびディレクトリはクラスタ単位でアクセスされます。 本デザインにおけるバウンダリおよびクラスタ・サイズは表 1-1 に示す通りなので、SATA-IP をアクセスする低レベル・ファンクション(“sata_host_ctl.c 内の低レベル関数)は 64MB（すなわちバウンダリ・ユニット・サイズ）転送をサポートするよう編集されています。 表 1-1 に示す値をサポートするため、本デザインは 2TB までの容量を持つドライブをサポートしますが 8GB 以下のドライブはサポートされません。

exFAT デザインにおいても 1 チャネル SATA ホスト・デザインと同様に各 FPGA 評価ボード上の DDR3 はデータ・バッファとして使われ、CPU や SATA-IP を相手にデータを転送します。 図 3-1 に示すように、TX/RX_SATA_FIS エリアは非データ FIS パケットを保存し DATA_FIS エリアはデータ FIS を保存します。 また、2 つのエリアが DDR3 メモリ内に追加して実装され、一つは TMP_DATA エリアで一時的にデータを保存し、もう一つは TEST_PATTERN エリアでファイル内データの生成やベリファイ用としてテストパターンを保存します。

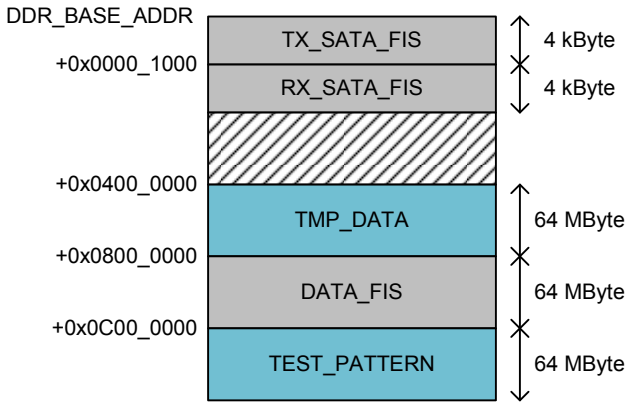

図 3-1: DDR メモリ・マップ

本デザインは全部で 5 種類のコマンドをサポートし、フォーマット、ファイル/ディレクトリの新規作成、ファイル/ディレクトリの読出し、消去、カレント・ディレクトリの移動です。 各コマンドの詳細を以下に説明します。

- フォーマット

SATA デバイスへの exFAT ファイルシステムでのフォーマットは以下の手順で実行します。

(1) ドライブ容量から表 1-1 のようにクラスタ・サイズとバウンダリ・ユニット・サイズを設定します。
(2) マスタ・ブート・レコードの1セクタ・データを TMP_DATA エリアに用意しセクタ#0 に対してライト関数を呼び出します。
(3) 12セクタ分のブート領域データを TMP_DATA エリアに用意し12セクタのライト関数をセクタ・アドレス=BU と(バックアップ用の)BU+12 に分けて2回呼び出します。
(4) BU/2 サイズの FAT テーブル・データを TMP_DATA エリアに用意しセクタ・アドレス=3*BU/2 としてライト関数を呼び出します。
(5) アロケーション・ビットマップ・テーブル、大文字変換テーブル、ルートディレクトリを TMP_DATA エリアに用意しクラスタ・アドレス=2 より3連続クラスタへ書き込むライト関数を呼び出します。
(6) その結果ファイル・システムのレイアウトは図 1-1 と同じになるようフォーマットされます。

- ファイル/ディレクトリの新規作成

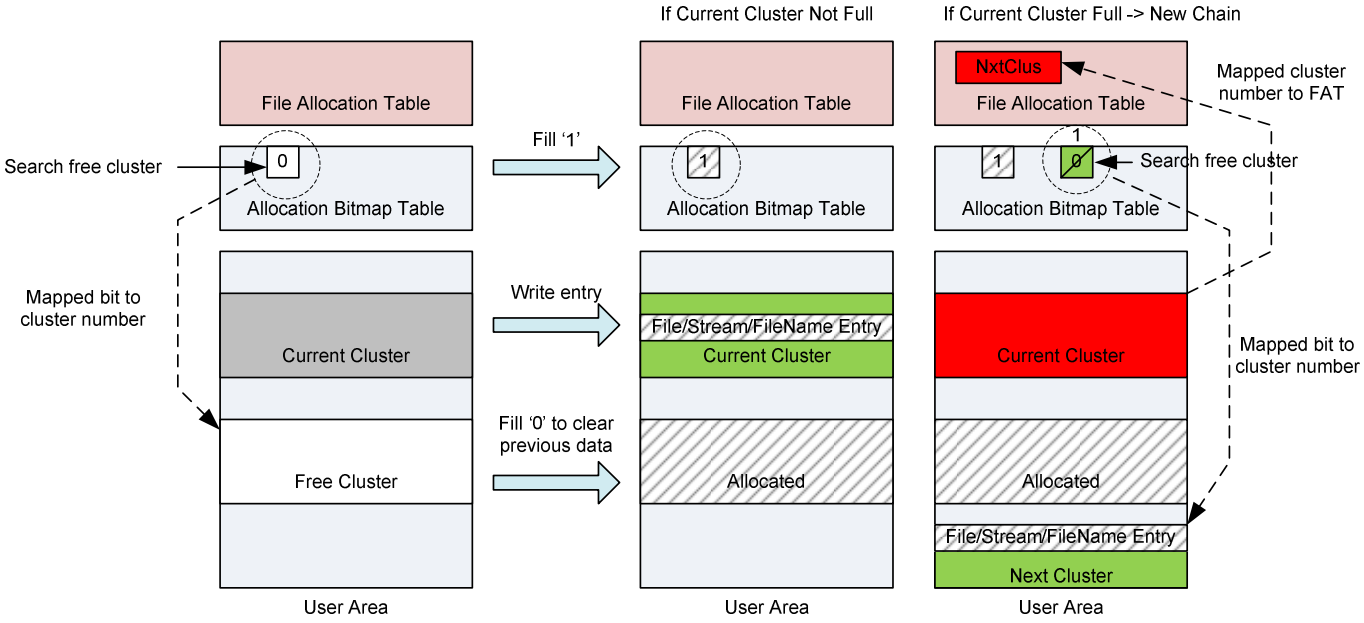

図 3-2: ディレクトリ新規作成の手順

ディレクトリの新規作成は以下の手順となります。
(1) アロケーション・ビットマップ・テーブル(クラスタ・アドレス=2)から空きクラスタを探します。
(2) アロケーション・ビットマップ・テーブルの該当クラスタ相当ビットに'1'をセットします。
(3) カレント・ディレクトリのエントリを読み出しファイル・ディレクトリ・エントリを追加する空き空間を探します。
(4) カレント・ディレクトリ内に十分な空き空間がなかった場合、別の空きクラスタを探しその空きクラスタを FAT テーブルに書き込むことでカレント・ディレクトリに FAT チェインを構成します。
(5) ファイル情報、拡張情報、ファイル名の各ディレクトリ・エントリを確保した空きクラスタへ書き込みカレント・ディレクトリのエントリ更新ファンクションを呼び出します。

ファイルの新規作成は以下の手順となります。
(1) DDR メモリの TEST_PATTERN エリアに書き込むデータのテストパターンを準備します。
(2) アロケーション・ビットマップ・テーブルから空きクラスタを探します。 探す空きクラスタのサイズはユーザが入力したファイルサイズと同じです。
(3) a. 空きクラスタが連続しておりフラグメント(分断) 化されていない場合、TEST_PATTERN エリアから SATA デバイスへの転送データはバースト転送で実行されます。 そしてアロケーション・ビットマップで書き込まれたエリアに該当するビットに'1'をセットします。
b. 空きクラスタが非連続でフラグメント化していた場合、各トランザクションで1クラスタづつのシングル転送で実行されます。 FAT テーブルにて FAT チェインが作成され次のクラスタ番号がセットされます。 そして各トランザクションごとにアロケーション・ビットマップ・テーブルの該当するクラスタに'1'がセットされます。
(4) ファイル情報、拡張情報、ファイル名の各ディレクトリ・エントリがカレント・ディレクトリに追加されます。
(5) ディレクトリの新規作成手順と同様、カレント・ディレクトリにエントリを追加するのに十分な空間がない場合は新しい FAT チェインが作成されます。

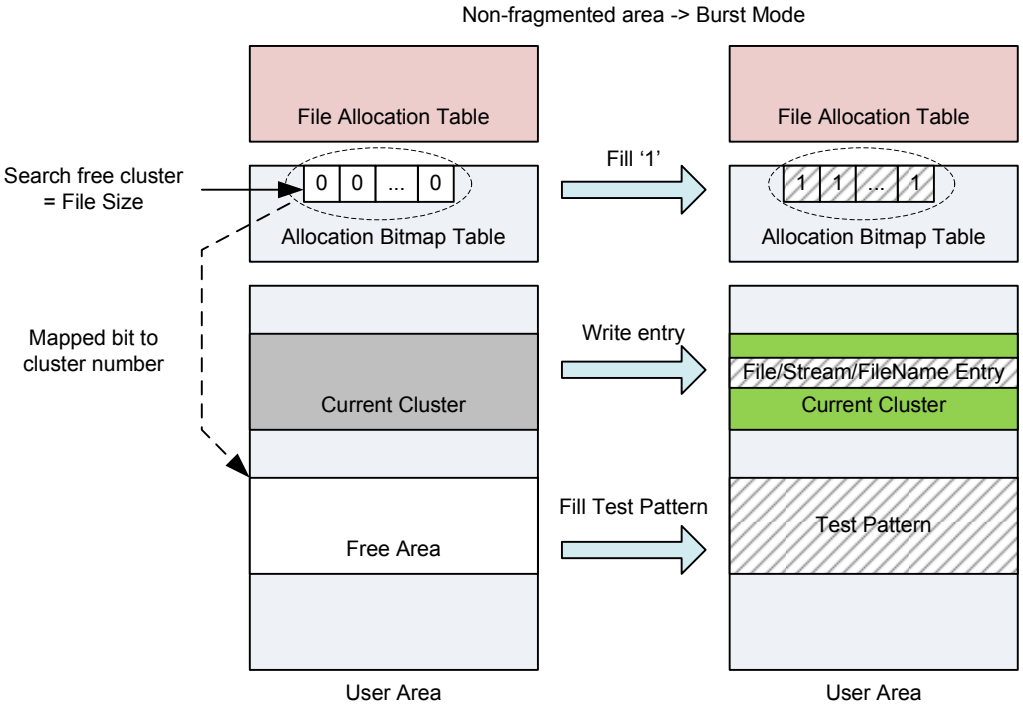

図 3-3: フラグメント(分断)化していない場合のファイルの新規作成

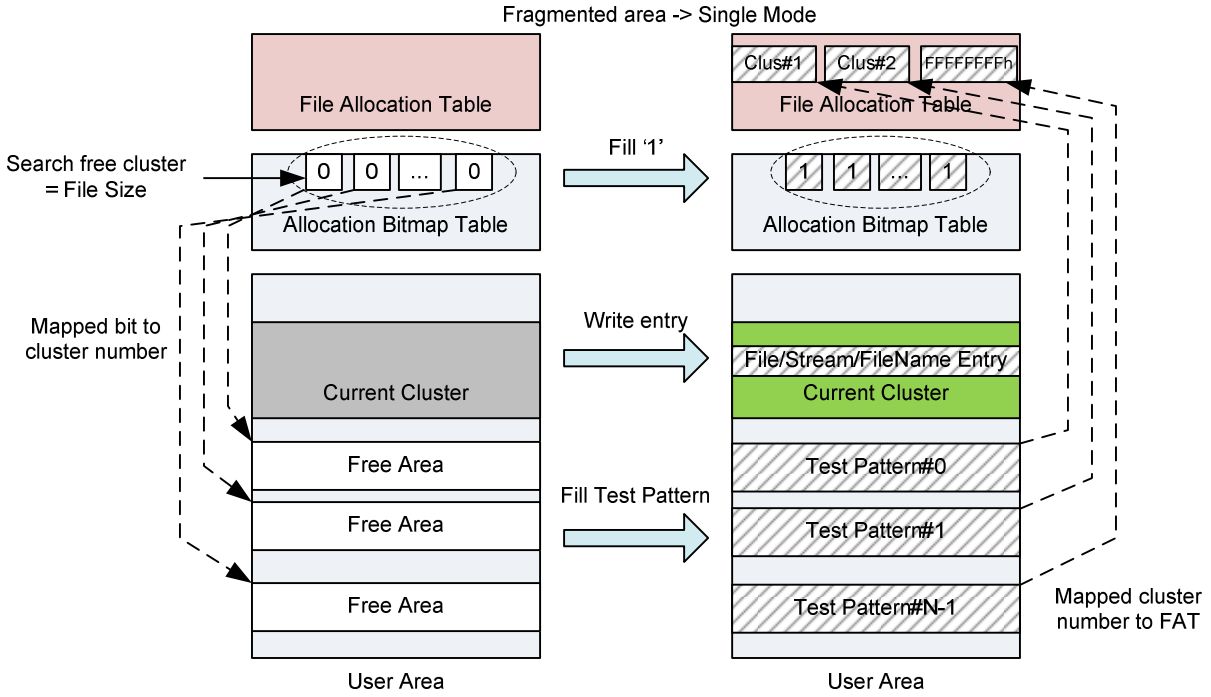

図 3-4: フラグメント(分断)化していた場合のファイルの新規作成

● ファイル/ディレクトリの読出し

カレント・ディレクトリの読出しは以下の手順で実行します。

(1) カレント・クラスタの全データを DATA_FIS エリアへ読み出します。
(2) カレント・ディレクトリの中のエントリをクラスタの最後まで全てスキャンし、有効なファイル/ディレクトリのみ画面に出力します。
(3) カレント・クラスタの末尾に到着し他に FAT チェインがなくなると動作を官僚します。 FAT チェインが見つかった場合は FAT テーブルから次のクラスタ・アドレスを読出しステップ(1)に戻ります。

ファイルの読出しは以下の手順で実行します。

(1) カレント・クラスタの全データを DATA_FIS エリアへ読み出します。
(2) ユーザから指定されたファイル名と合致するエントリをカレント・クラスタのエントリから探します。
(3) 該当するファイル名がカレント・クラスタで見つからないままクラスタの最後に到達し、FAT チェーンがそれ以上ない場合は動作を中断し終了します。 FAT チェーンがある場合は次のクラスタを探します。
(4) 合致したファイル・ディレクトリ・エントリから最初のクラスタ情報を読出しそのデータを DATA_FIS エリアへ読み出します。
(5) 読出しコマンドの終了を検出するかファイルの最後に到達するまで DATA_FIS 内のデータを表示します。

● カレント・ディレクトリの移動

カレント・ディレクトリからサブ・ディレクトリへの移動は以下の手順で実行します。

(1) ユーザから指定されたディレクトリ名と合致するものを「ファイルの読出し」手順のステップ(1)～(3)と同じように探します。
(2) システムのカレント・クラスタを、ファイル・ディレクトリ・エントリで合致した最初のクラスタ値に変更します。 この後ユーザは新しいディレクトリ内の全エントリに対してアクセスが可能となります。

注意: 本デザインはグローバル・パラメータの制約から最大 15 レベルまでのサブ・ディレクトリまでをサポートします。

カレント・ディレクトリを親(上位)ディレクトリに移動する場合、ファームウエア内のグローバル・パラメータから親ディレクトリのクラスタ・アドレスを読出しクラスタ・アドレス値を変更します。

● 消去

ファイルの消去は以下の手順で実行します。

(1) ユーザから指定されたファイル名と合致するものを「ファイルの読出し」手順のステップ(1)～(3)と同じように探します。
(2) 親ディレクトリのファイル・エントリから、エントリ・タイプの 85h,C0h,C1h から 05h,40h,41h となるよう Bit[7]を' 1' から' 0' に変更します。
(3) 編集したエントリを DDR バッファから SATA ドライブへ書き込みます。

ディレクトリの消去は以下の手順で実行します。

(1) ユーザから指定されたディレクトリ名と合致するものを「ファイルの読出し」手順のステップ(1)～(3)と同じように探します。
(2) システムのカレント・クラスタを、ファイル・ディレクトリ・エントリで合致した最初のクラスタ値に変更します、そしてそのクラスタ末尾および FAT チェーンの最後までの各エントリを消去します。 エントリ内にサブ・ディレクトリが見つかった場合は、カレント・ディレクトリをサブ・ディレクトリに移動して全エントリを消去し、その後ディレクトリをもとしてエントリの消去を継続します。

- 本デザインの制約事項

- 本 exFAT ソフトウエアはイリーガルな動作や予期されない動作などに対するエラー・チェックやリカバリ機能は一切含まれておりません。
- 本デザインではサブ・ディレクトリは 15 レベルまでサポートしています。 より深いレベルでサブ・ディレクトリを構築したい場合はユーザにて編集する必要があります。
- DesignGateway 社は本リファレンス・デザインに対して、FPGA 評価ボードをターゲットとし、かつ、ソフトウエアに一切手を加えない状態でのみサポートを提供します。ターゲットがユーザ設計のオリジナル基板など FPGA 評価ボードでない場合あるいはオリジナルの状態からソフトウエアが編集された場合はサポートの対象外となります。

下図 3-5 に本デザインの実行サンプルを表示します。

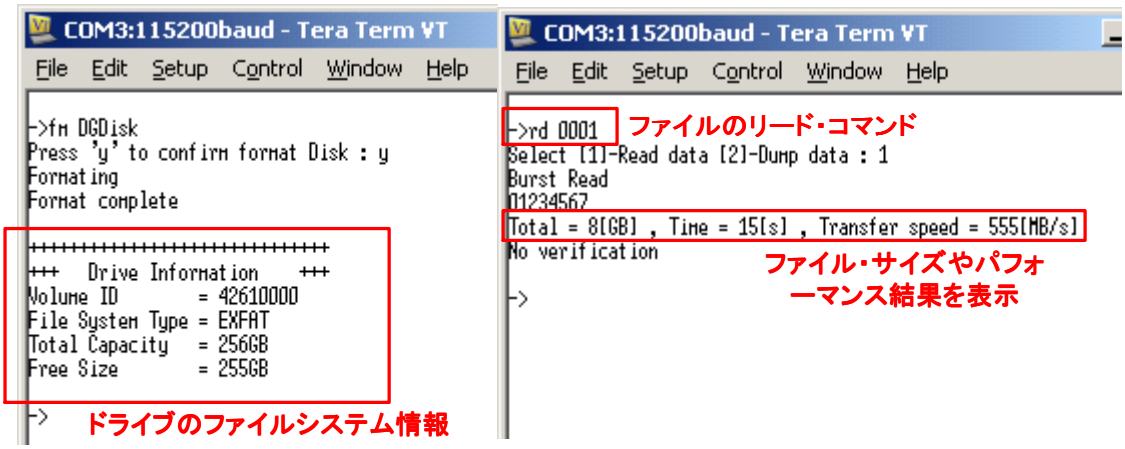

図 3-5: 実行サンプル画面

4 更新履歴

リビジョン	日時	説明
1.0J	2013/04/8	日本語初版作成
1.1J	2013/04/24	図および説明文の修正
1.2J	2013/09/11	マルチ・プラットフォーム対応に伴う修正