Altera V/10 シリーズ SATA-IP ホスト・リファレンス・デザイン説明書

Rev1.4J 2016/07/11

1. SATA について

シリアル ATA (SATA)は従来のパラレル ATA(PATA)に替わる革新的なストレージ・インターフェイスです。 最新の SATA インターフェイスにおける転送速度は、SATA-I 規格の 1.5Gbps から SATA-III 規格の 6.0Gbps に高速化されています。 SATA プロトコルによる通信システム全体としては、図 1 に示すように、アプリケーション・レイヤ、トランスポート・レイヤ、リンク・レイヤ、物理(PHY)レイヤ、の4レイヤにより実装されるアーキテクチャとなります

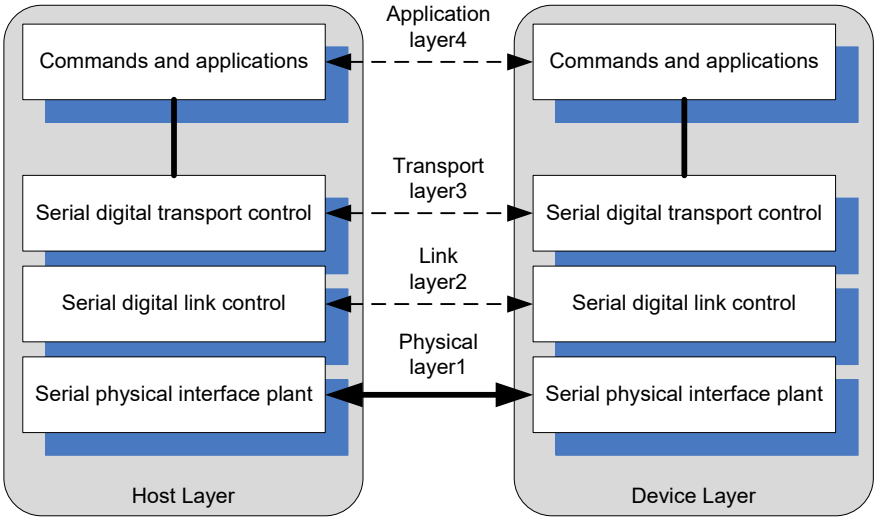

図 1 SATA のレイヤ構造

アプリケーション・レイヤはコマンド・ブロック・レジスタの制御を含む ATA コマンドの実行を担当します。トランスポート・レイヤではパケットや FIS(Frame Information Structure)と呼ばれるフレームによってホスト～デバイス間で転送される制御情報やデータを管理します。 リンク・レイヤにおいては、生成されたフレームをもとにバイトごとの 8b/10b エンコード/デコードの実行や、10 ビットのデータ・ストリームが受信側で正しくデコードされるよう制御キャラクタの挿入を行います。PHY(物理)レイヤは、シリアル・データとして外部信号線上に流れるエンコード情報を送受信します。

本リファレンス・デザインでは、ホスト側において SATA-IP を含めた全 SATA 通信レイヤの具体的な実装方法例を紹介します。 この評価システムは Altera 製各種 V シリーズおよび 10 シリーズのデバイス・ファミリにおいて評価キットと外付けの SATA デバイスを使い接続デバイスへのリード・ライト動作を実行します。 SATA-IP コアは FPGA デバイスのトランシーバと組み合わせて SATA チャネルを構築します。 より詳細については以下で説明します。

2. ハードウエアの説明

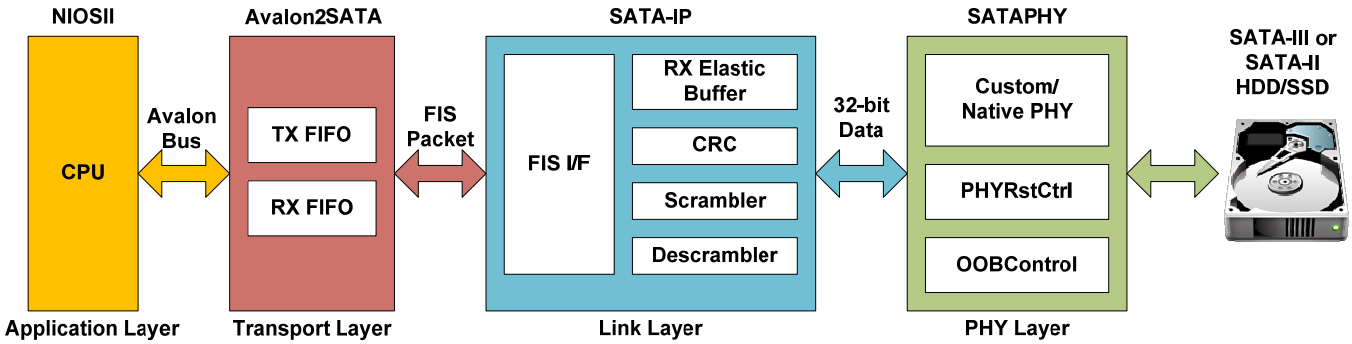

図 2-1 リファレンス・デザインの各レイヤ構成

本リファレンス・デザインのハードウエアは、図 2-1 に示すように各 SATA レイヤを実装する4モジュールに分かれたアーキテクチャとなっています。SATA-IP コアはリンク・レイヤと一部のトランスポート・レイヤを内蔵するため、ユーザはコア以外の PHY レイヤやトランスポート・レイヤならびにアプリケーション・レイヤを構築する必要があります。 本リファレンス・デザインはユーザがデザインする各レイヤにおいて DesignGateway 社製 SATA アダプタを装着した Altera 各種評価キットで実機動作する実装例を示したものです。

● PHY レイヤ

このレイヤは Altera IP ウィザードの Custom PHY または Native PHY コンポーネントを使って SATA パラメータを設定したビルト・イン高速シリアル・ブロックを、OOB(Out-of-Band)信号を制御するロジックと組み合わせて構築します。SATA プロトコルの OOB シーケンスはステートマシンで制御され、COMRESET, COMINIT, COMWAKE の生成・受信を行う OOBComCtrl サブ・モジュールと一緒に動作します。 PHYRstCtrl はリセット・シーケンスを制御し SATA として動作する適切な初期パラメータをセットします。

PHY レイヤは”hdl”ホルダ内に格納された”sata2phy.vhd”または”sata3phy.vhd”にて、サブ・モジュールを格納する“OOBComCtrl.vhd”, “PhyRstCtrl.vhd”, “OOBMainCtrl.vhd”の各ファイルと合わせて実装されます。これらPHYレイヤの全モジュールはコア製品に同梱したリファレンス・デザイン・プロジェクトに VHDL ソース・コードで提供されます。

● SATA-IP コアによる LINK レイヤ

SATA-IP はリンク・レイヤ全てとトランスポート・レイヤの一部を含み、リンク・レイヤのデザインは CRC、デスクランブル、スクランブルモジュールを含みます。 FIS インターフェイスはユーザ回路と簡単に接続できるように設計されユーザ回路側のクロックで動作します。 SATA-IP コアのインターフェイス信号詳細仕様や信号波形については、SATA-IP 紹介 Web ページに掲載されるデータシートを参照してください。

SATA-IP 紹介ページ URL:
http://www.dgway.com/SATA-IP_A.html

● トランスポート・レイヤ

本リファレンス・デザインにてトランスポート・レイヤは下図 2-2 に示すように Avalon2SATA モジュールで構築されますが、このモジュールについて以下に説明します。

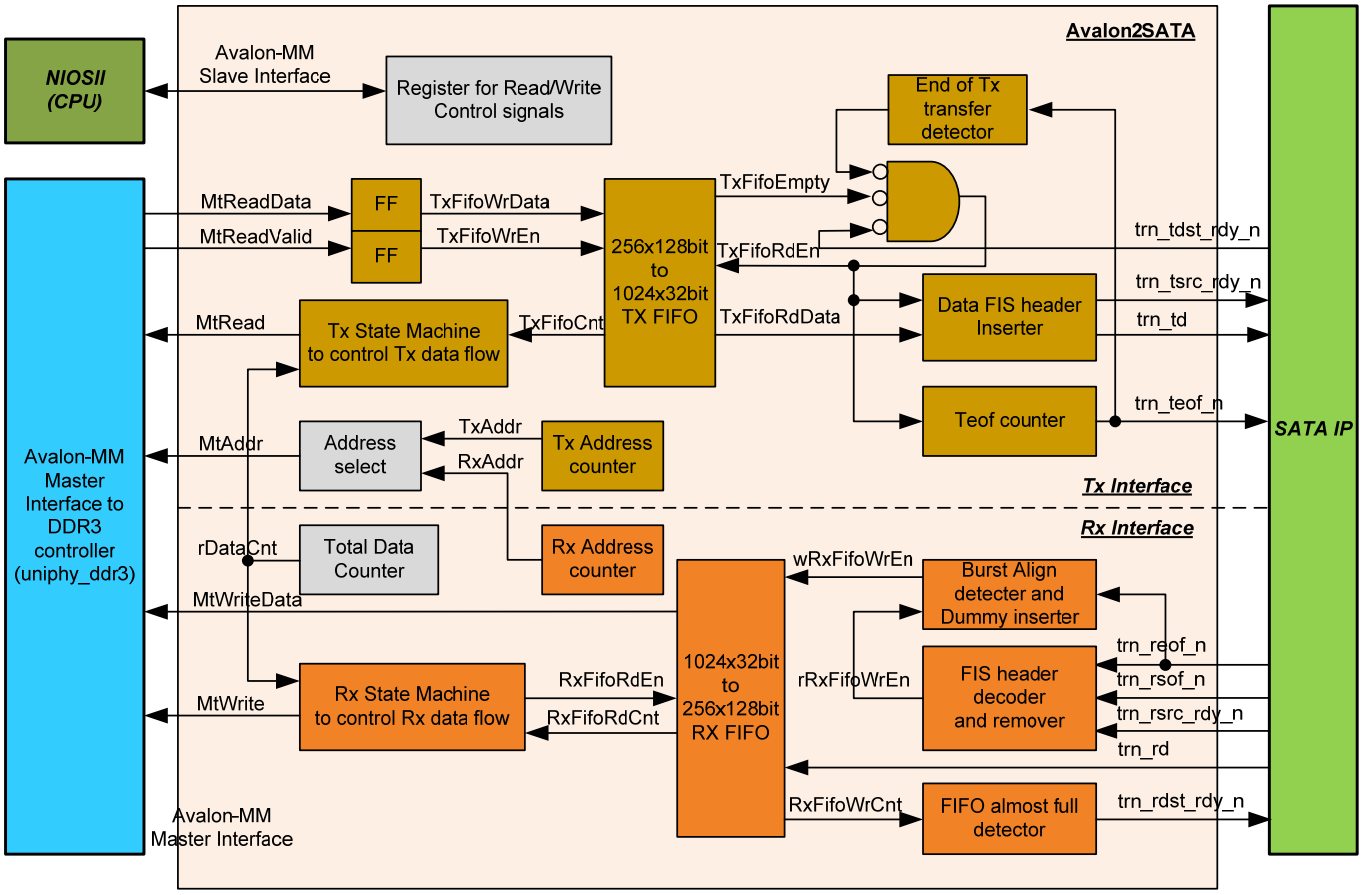

図 2-2 Avalon2SATA によるトランスポート・レイヤのブロック図

本デザインのトランスポート・レイヤは一般的な SATA コントローラと同様、CPU が DDR3/DDR4 のメイン・メモリ上に FIS データを構築してから下位レイヤ内の DMA に対して転送開始信号を出力します。 データ転送を DMA によって実行するため、Avalon2SATA モジュールには図 2-2 左下側に示す Avalon-MM マスター・インターフェイスを実装します。一方 NiosII による CPU とは図 2-2 左上側の Avalon-MM スレーブ・インターフェイスで接続します。 Avalon-MM マスター・インターフェイスは高い転送パフォーマンスを維持するために 512 バイトの固定サイズにて設計されています。バス幅は DDR コンポーネントに依存するため評価キットのプラットフォームによって異なります。 リファレンス・デザインのデータ・バスは 512 バイト単位での転送を実行するため、①8 ビートの 512 ビット幅データ・バス、②16 ビートの 256 ビット幅データ・バス、③32 ビートの 128 ビット幅データ・バス、の 3 種類の実装スタイルがあります。

SATA-IP コアに対して FIS パケットを送信する場合、転送カウンタ数がゼロにデクリメントされるまで送信データを 512 バイト単位で TXFIFO に格納する必要があります。送信パケットがデータ FIS であった場合はデータ FIS ヘッダを自動的に生成します。 送信データ数が丁度 512 バイトの倍数であった場合は TXFIFO 内の全データが SATA-IP コアに転送されますが、そうでなかった場合は必要数だけのデータが SATA-IP コアに転送されます。 この場合に余ったデータは TXFIFO 内に残っているため、送信後に TXFIFO をフラッシュします。

SATA-IP コアからの FIS パケットを受信する場合、FIS ヘッダがデコードされます。デコード結果がデータ FIS であった場合 FIS ヘッダは自動的に削除されるので RXFIFO にはヘッダ以外の純粋なデータのみが格納されます。SATA-IP コアからの受信データ数が 512 バイトの倍数でなかった場合は倍数となるようにダミー・データがデータ末尾の後に自動的に追加されてから DDR メモリに転送されます。Avalon2SATA モジュールにおいて送信側と受信側のステートマシンはそれぞれ別個に用意され独立して動作します。

トランスポート・レイヤのソース・コードは SATA-IP コアと PHY レイヤのインスタンスとあわせて"MMMt2SATA128.vhd"または"MMMt2SATA256.vhd"または"MMMt2SATA512.vhd"に格納されます。

- **NiosII システムによるアプリケーション・レイヤ**

本リファレンス・デザインにおいてアプリケーション・レイヤは NiosII システムで構築されます。 下図 2-3 に NiosII システムを含むデザイン全体のブロック図を示し以下に説明します。

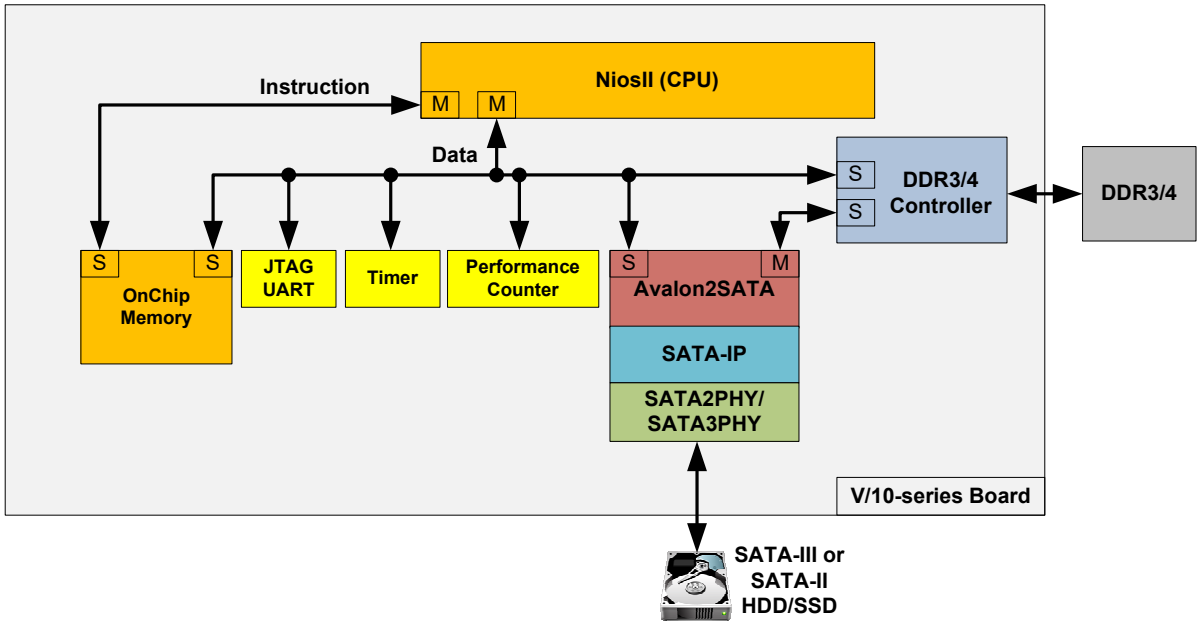

図 2-3 リファレンス・デザインのブロック図

本リファレンス・デザインはホスト・プロセッサとして NiosII を使い、NiosII および SATA-IP と接続したユーザ・ロジックの双方からアクセスできる DDR をメイン・メモリとして使います。 DDR メモリは図 2-4 に示すように4種類の空間に分割されリードあるいはライト動作でのデータ FIS あるいはその他の FIS 格納用として用いられます。

NiosII のプログラムおよびデータは FPGA 内部メモリ(OnChipMemory)を使います。ユーザからのコマンド指示や結果は NiosII により提供される JTAG UART 経由にて行います。コマンドの実行結果としてデータ転送速度を表示するためタイマーを用意します。 本デザインの制御信号となるレジスタ・マップを表 2-1 に示します。 NiosII ファームウエアは DMA を使ってメイン・メモリ上の FIS データを SATA-IP に送信し、また SATA-IP からの受信データをメイン・メモリに転送します。 転送先のメモリ・アドレスや転送サイズは NiosII ファームウエアで制御します。 データ FIS ヘッダは Avalon2SATA モジュールにより自動的に追加/削除されるため、NiosII ファームウエアは送受信データについてのヘッダ情報を管理する必要はありません。 各転送が終わると NiosII に通知するため転送完了フラグがセットされ、次の転送を開始できます。 ファームウエアの詳細については次のトピックで説明します。

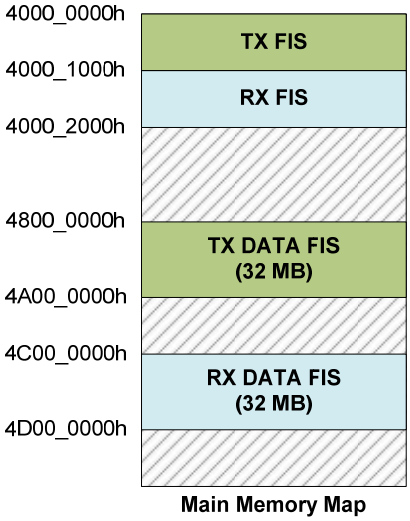

図 2-4 DDR メモリのメモリ・マップ

アドレス Rd/Wr	レジスタ名	説明
BA+0x00 Rd	PHY ステータス・レジスタ (PHY_STATUS)	bit[0] : SATA PHY リンクアップ ('0' : リンクしていない, '1' : SATA PHY 初期化完了 (注) CycloneV SX SoC 開発キットの場合 SATA-Ⅱ固定でその他の場合は SATA-Ⅲ固定となります。 bit[1] : SATA 速度 ('1' : SATA-Ⅲ, '0' : SATA-Ⅱ) CycloneV
BA+0x04 Rd	エラー・コード・レジスタ (ERROR_CODE)	SATA IP エラー・コード、送受信完了時にコアが出力するエラー・コードを示す。 (詳細は SATA-IP データシートのエラー・コード表を参照のこと)
BA+0x0C Rd	受信ワード・カウント・レジスタ (RX_COUNT)	bit[31] : 割込み発生時の受信 FIS タイプ、' 1' は非データ FIS、' 0' はデータ FIS。 このビットは INT_CLEAR レジスタの bit[29]に' 1' をライトすることでクリアされる。 bit[23:0]: 受信した FIS データの総ワード数。 コントロール･レジスタにて次の転送開始が指示されるとゼロクリアされる。
BA+0x10 Rd	割込みクリア・レジスタ (INT_CLEAR)	bit[31] 受信完了フラグ、パケット受信完了時に IP コアによりセットされる bit[30] 送信完了フラグ、パケット送信完了時に IP コアによりセットされる
BA+0x00 Wr	送信データ・アドレス (TX_ADDR)	送信する DDR データを格納している先頭のアドレスを指定する。 本レジスタの Bit[8:0]はアライメント調整のため必ずゼロにセットすること。
BA+0x04 Wr	受信データ・アドレス (RX_ADDR)	受信したデータのうち、データ FIS 以外のものが格納される DDR のアドレス。本レジスタの Bit[8:0]はアライメント調整のため必ずゼロにセットすること。
BA+0x08 Wr	コントロール・レジスタ (CONTROL)	bit[31] SATA ハードウエア・リセット bit[30] 転送方向('1' :送信, '0' :受信) bit[29] FIS タイプ(データ FIS を送る場合' 1' それ以外の FIS タイプの場合' 0') bit[15:0] 送信データワード数。 本レジスタの書き込みで RX_COUNT レジスタがリセットされる。
BA+0x0C Wr	受信データ格納アドレス 2 (RX2_ADDR)	受信したデータのうち、データ FIS が格納される DDR のアドレス。 本レジスタの Bit[8:0]はアライメント調整のため必ずゼロにセットすること。
BA+0x10 Rd	割込みクリア・レジスタ (INT_CLEAR)	bit[31] '1' をセットすることで受信完了フラグをクリア bit[30] '1' をセットすることで送信完了フラグをクリア bit[29] '1' をセットすることで受信 FIS タイプ(RX_COUNT bit[31])をクリア

表 2-1 NiosII のレジスタ・マップ

(BA : ベース・アドレス = 0x0100_0000)

3. ソフトウエアの説明

● FIS を介した SATA デバイスへのアクセス

SATA によるホストとデバイス間の通信は FIS(Frame Information Structure)データ構造によって実行されます。 ホストデザインの NiosII はメイン・メモリ上に FIS データを構築し、バス・マスタとなる DMA によってデバイスに転送されます。 また、デバイスからの FIS データも同じように DMA によってメイン・メモリに転送されます。

従って NiosII は以下の手順で SATA デバイスへのアクセスを実行します。

(1) FIS データ・ストラクチャを作成します。 最初の FIS コマンドは RegH2D FIS とする必要があります。
(2) FIS データを転送します。
(3) デバイスからの FIS データ受信を待ちます。
(4) 受信した FIS データを読み取り、解析します。
(5) 必要に応じて追加の FIS データの送受信を行います。

プロトコルによって送信する FIS の数や受信する FIS の数は異なってきますが、おおむねこのような流れになります。

● リファレンス・デザインのソフトウエア

本リファレンス・デザインのソフトウエアは一般的な3コマンドを実装しており、それは IDENTIFY DEVICE, READ DMA EXT/READ DMA, WRITE DMA EXT/WRITE DMA となります。 本リファレンス・デザインは 48 ビット LBA (LogicalBlock Address)モードと 28 ビット LBA モードの両方をサポートしいます。

デバイスがパワーON したとき, デバイスは必ず Register –Device to Host FIS を最初に送ります。 従って、ホストは最初のコマンドを発行する前にデバイスからの RegD2H FIS を待つ必要があります。

● IDENTIFY DEVICE

表 3-1 は SATA デバイスからデバイス情報を取得するための IDENTIFY COMMAND の FIS 構造です。 コマンドは ECh で、あとはデバイス番号を設定するだけで実行できます。 SATA の場合、デバイス番号は通常'0'になります。

なお, Device レジスタの 5 ビットと 7 ビットは obsolete(廃止)ビットですが、慣例では常に 1 にセットするようですのでここは A0h を設定します。 また C ビットを 1 にします。コマンドを送信する場合はこれを必ずセットしますが、以後のコマンドも同様です。

これらの値を Register – Host to Device FIS に格納してリンク・レイヤに送信します。 するとデバイスから PIO SetupFIS が送られてきたあとに Data FIS が送られてきます。この中にデバイスの情報が格納されています。デバイス情報の詳細については ATA 規格書(http://www.t13.org/から入手可能)を参照してください。 本リファレンス・デザインではデバイス型番、48 ビット LBA の対応情報、ディスク容量等を表示します。

0	Features 00h	command ECh	C R R R PM Port 1 0 0 0 0h	FIS Type(27h)
1	Device A0h	LBA High 00h	LBA Mid 00h	LBA Low 00h
2	Features(exp) 00h	LBA High(exp) 00h	LBA Mid(exp) 00h	LBA Low(exp) 00h
3	Control 00h	Reserved(0)	sector Count(exp) 00h	Sector Count 00h
4	Reserved(0)	Reserved(0)	Reserved(0)	Reserved(0)

表 3-1 IDENTIFY COMMAND の FIS 構造

● READ DMA EXT

表 3-2 は SATA デバイスから 48 ビット LBA でデータを読み出す READ DMA EXT 命令の FIS 構造です。28 ビット LBA では READ DMA コマンドが使われます。 データ転送は大きく分けて PIO と DMA がありますが、SATA にとっては若干 FIS の手順が違うだけで、どちらもそれほど変わりません。 実は PIO 転送を使っても、DMA と変わらないくらいの速度が出ますが, リードに関しては READ DMA の手順が簡単なので、こちらを使います.

コマンドは 25h (28 ビット LBA の READ DMA コマンドでは C8h)、Device レジスタ 6 ビットの LBA ビットを 1 にし、あとは LBA アドレスと読み出したいセクタ数を Register – Host to Device FIS に格納して送信します。 するとデバイスから Data FIS が要求したデータ分だけ送られてきた後、Register – Device to Host FIS が送られてきて完了です.

0	Features 00h				command 25h			C 1	R 0	R 0	R 0	PM Port 0h		FIS Type (27h)	
1	Device E0h				LBA High LBA[23:16]			LBA Mid LBA[15:8]						LBA Low LBA[7:0]	
2	Features (exp) 00h				LBA High (exp) LBA[47:40]			LBA Mid (exp) LBA[39:32]						LBA Low (exp) LBA[31:24]	
3	Control 00h				Reserved (0)			sector Count (exp) sector_count[15:8]						Sector Count sector_count[7:0]	
4	Reserved (0)				Reserved (0)			Reserved (0)						Reserved (0)	

表 3-2 DMA READ EXT の FIS 構造

● WRITE DMA EXT

表 3-3 は SATA デバイスへデータを書き込む WRITE DMA EXT 命令の FIS 構造です。(28 ビット LBA では READ DMA コマンドが使われます。) コマンドは 35h(28 ビット LBA の場合 CAh)、LBA ビットや LBA アドレス、セクタ数の設定は DMA READ EXT と同じです。 その後、デバイスから DMA Activate FIS が返ってきます。 それを受けてホストは最初の Data FIS を送信します。 これを繰り返しデータをすべて送信し終わったら、デバイスから Register– Device to Host FIS が送られてきて完了です。

0	Features 00h				command 35h			C 1	R 0	R 0	R 0	PM Port 0h		FIS Type (27h)	
1	Device E0h				LBA High LBA[23:16]			LBA Mid LBA[15:8]						LBA Low LBA[7:0]	
2	Features (exp) 00h				LBA High (exp) LBA[47:40]			LBA Mid (exp) LBA[39:32]						LBA Low (exp) LBA[31:24]	
3	Control 00h				Reserved (0)			sector Count (exp) sector_count[15:8]						Sector Count sector_count[7:0]	
4	Reserved (0)				Reserved (0)			Reserved (0)						Reserved (0)	

表 3-3 DMA WRITE EXT の FIS 構造

● **リファレンス・デザインの動作について**

本デザインの NiosII ファームウエアのソース・コードは IP コア製品に同梱したリファレンス・プロジェクトにて、“software/Sata_host/Sata_host.c”内に格納されています。ただし本デザインはエラーチェックや異常発生時のリカバリなどの処理は含まれていません。従ってユーザが開発するソフトウエアにおいては、デバイスから Register– Device to Host FIS が送られたときにステータスやエラーをチェックし、必要な処理を追加する必要があります。

図 3-1 に本リファレンス・デザインを動作したときの PC 上のシリアル・ターミナル画面サンプルを示します。

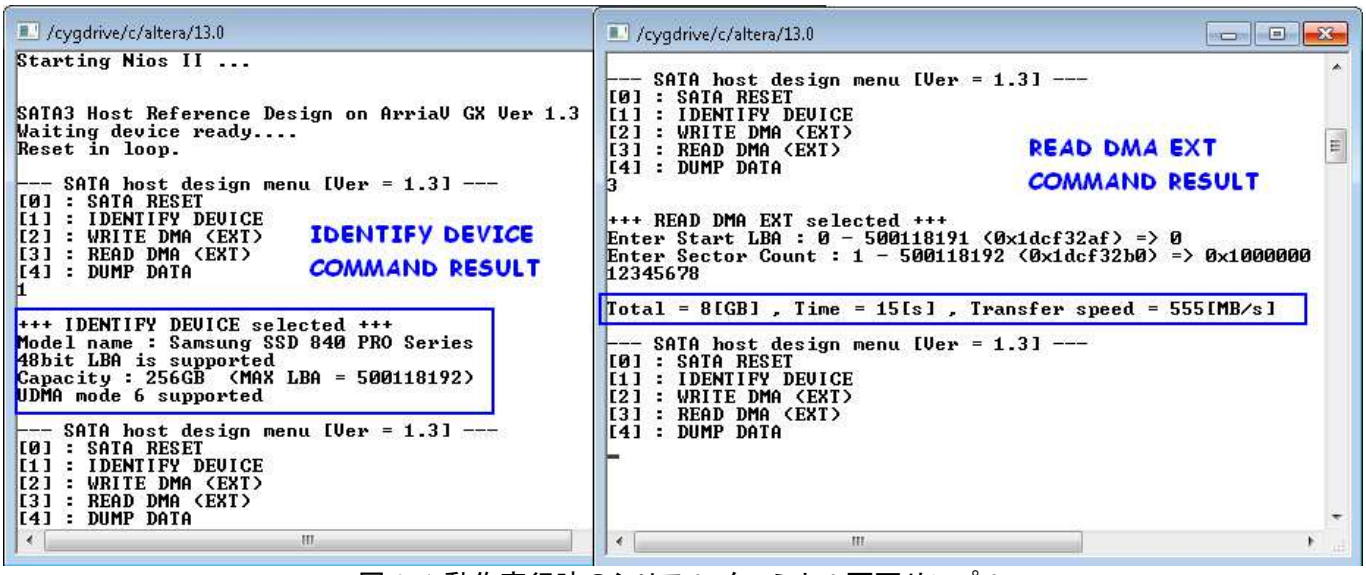

図 3-1 動作実行時のシリアル・ターミナル画面サンプル

4. 改版履歴

リビジョン	日付	説明
1.0J	2013/08/29	日本語版の初版発行
1.1J	2013/09/11	レジスタ・マッピングを変更
1.4J	2016/07/11	10 シリーズ・ファミリのサポート開始、英語版に合わせマニュアルを統合