SATA-IP ホスト向けリファレンス・デザイン説明書

Rev1.4　2009/06/05

このドキュメントは Xilinx 製 ML506/505 評価ボードで動作する SATA-IP ホスト向けのリファレンス・デザインに関して説明したものです。

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。 また、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-II 規格の 3.0Gbps 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

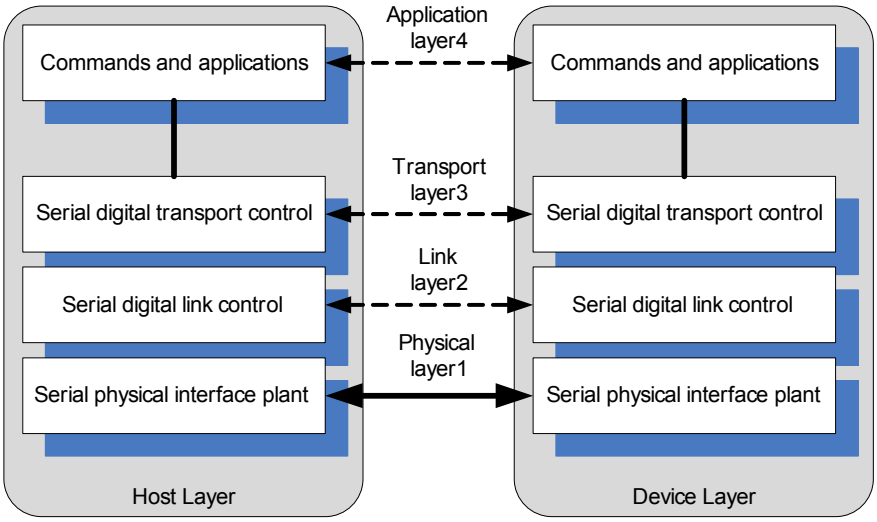

図 1: SATA のレイヤ構造

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、外付けの SATA-II ハードディスクを使って構築されます。 SATA-IP コアは Virtex-5 デバイスの GTP トランシーバと組み合わせて動作し、本リファレンス・デザインは Xilinx 製 ML506/505 評価ボード上で実装されます。 より詳細については以下で説明します。

2. 動作環境

本リファレンス・デザインは図 2 に示される以下の環境で動作します。

- Xilinx 製 ML506/505 評価ボード(ML505 で動作するためにはデバイスを変更しての再コンパイルが必要な場合があります。)
- ISE 10.1.03 / EDK 10.1.03
- SATA-II ハードディスク (ML506/505 ボードの J40 で SATA ケーブル接続します)
- シリアル(RS232C)ターミナル (ML506/505 ボードの P3 で RS232C ケーブルと接続します) ターミナルの設定は、ボーレート=115,200 / データ=8bit / Non-Parity / Stop ビット=1bit としてください。

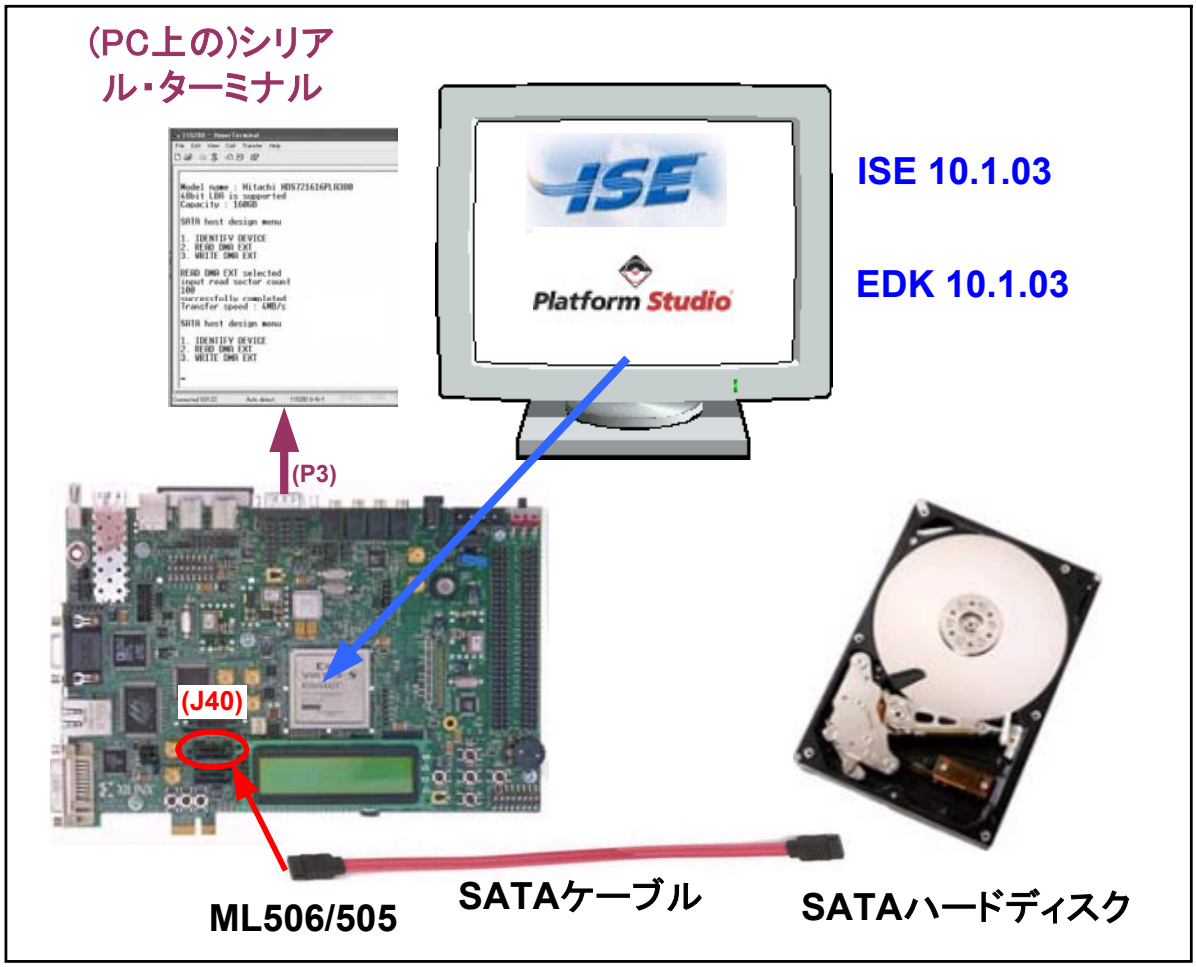

図 2: リファレンス・デザインの動作環境

本リファレンス・デザインの具体的な操作手順につきましては、”SATA_IP ホスト・デモ手順書”を参照してください。 また、リファレンス・デザインの SATA-IP コアは動作制限があり1時間経過後にデータ転送を停止します。

3. ハードウエアの説明

- Virtex5 FPGA 上で実装される SATA IP ホストデザイン

下図 3 のブロック接続図に示される通り、SATA-IP コアはトランスポート・レイヤの一部を含みますがほとんどがリンク・レイヤで構成されます。 従って、PHY（物理）レイヤとトランスポート・レイヤはユーザ自身で用意しなくてはなりません。 本リファレンス・デザインは Xilinx 製 ML506/505 評価ボードを対象として、トランスポート・レイヤと PHY レイヤの実装例を解説したものです。

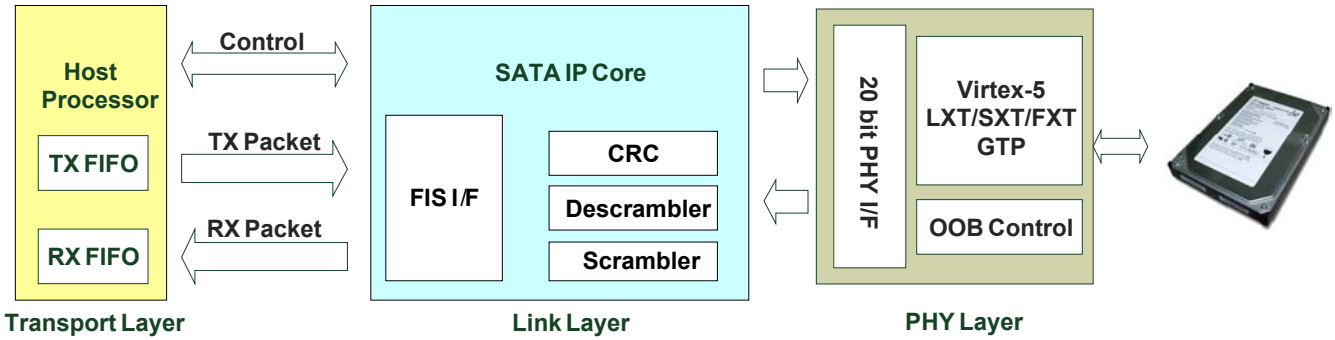

図 3: リンク・レイヤとトランスポート/PHY レイヤのブロック接続図

- PHY レイヤ

Virtex-5 LXT/SXT には GTP ブロック内に高速シリアル通信回路が内蔵されており、SATA の PHY レイヤはこの GTP ブロックで実装されます。 PHY レイヤは OOB(Out-of-Band)回路と 20 ビット幅でリンク・レイヤと通信するインターフェイスが含まれます。

本リファレンス・デザインは Xilinx のウェブサイトからダウンロードできるアプリケーション・ノート(XAPP870)をもとに編集されたため、PHY レイヤは Virtex5 の GTP リソースに最適化されております。 なお、SATA の特性情報については同じく Xilinx のウェブサイトより特性評価レポート(RPT087)がダウンロードできます。

ユーザ基板を新規に設計する場合、基板での高速シリアル特性を維持するため、UG196(Virtex-5 FPGA RocketIO GTP Transceiver User Guide)に記載のデザインガイドラインを遵守する必要があります。 リファレンス・デザインにおいて PHY レイヤ回路のソースコードは“oob_control.v”モジュールおよび”speed_neg_control.v”のインスタンスを含む“sata2phy_ml505.v”内に記載されています。

この“sata2phy_ml505.v”は、自動ネゴシエーション機能、SATA-II 固定、SATA-I 固定、の3つの動作モードから選択できるよう設計されています。 動作モードは、“sata2phy_ml505.v”の 30～32 行目の3つの”`define”文のうち、どれか1行を有効とし他の2行を”//”によるコメントアウトとすることで選択できます。 リファレンス・デザインのデフォルトは下図 4 に示すように自動ネゴシエーション機能を有効としています。

```
24:
25: // Select
26: // - FIXI     for Fixed-SATAI speed PHY design
27: // - FIXII    for Fixed-SATAII speed PHY design
28: // - AUTONEG for Auto speed negotiation PHY design
29:
30: //`define FIXI
31: //`define FIXII
32: `define AUTONEG // Auto speed negotiation
33:
34: module sata2phy_ml505 #
35: (
```

図 4: “sata2phy_ml505.v”での PHY モード設定方法

自動ネゴシエーション機能を有効とした場合、J40 の SATA コネクタには SATA-I あるいは SATA-II のどちらの HDD も接続可能ですが、SATA-I 固定あるいは SATA-II 固定とした場合は接続可能な HDD が限定されます。

Virtex5 デバイスでは1個の GTP_tile 内に1個の共有 PLL と2個の GTP チャネルを格納します。自動ネゴシエーション機能を使う場合、ネゴシエーション中に発行される PLL リセット信号が他方の GTP チャネルに影響を与えるため、GTP_tile 内では1つの GTP しか SATA チャネルとして使うことができません。従って例えば ML-506/505 ボードでは、同一の GTP_tile にアサインされた J40 と J41 の SATA チャネルを両方とも使うデザインにおいては、自動ネゴシエーション機能は使えません。

一方 SATA-I 固定あるいは SATA-II 固定の場合、共有 PLL へのリセットは発行されないため同一 GTP_tile 内の2個の GTP とも SATA チャネルとして使うことが可能です。 このため、固定モードを活用することで、多数の SATA チャネルを必要とする RAID アプリケーションにも有効です。 さらに SATA-I あるいは SATA-II 固定モードの場合、PHY デザイン中で消費する DCM 数が1個で済むため、DCM が 2 個必要な自動速度ネゴシエーションより DCM リソースを節約できる利点があります。

(注意) PHY パラメータの変更はリファレンス・デザインのハードウエアを変更して再コンパイルする必要があるため、製品版でのみ可能です。

- トランスポート・レイヤ

トランスポート・レイヤの構造はハードウエアのアーキテクチャやユーザのアプリケーションに依存するため、ユーザ自身で設計する必要があります。 本リファレンス・デザインにおいては、トランスポート・レイヤに MicroBlaze で動作させるために NPI インターフェイスを使っています。 一般的な SATA コントローラと同様に、本リファレンス・デザインもメインメモリ上に FIS データを作成し、DMA によってリンク・レイヤと通信します。

本リファレンス・デザインにおいてトランスポート・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を含む“npi_sata.vhd”内に記載されています。

- RAID アプリケーションに柔軟に対応できる MPMC インターフェイス

本リファレンス・デザインは MicroBlaze をホストプロセッサとして使っており、メインメモリのコントローラとして MPMC(Multi-port memory controller)を使います。 MPMC は最大8チャネルまでのメモリアクセス・ポートをサポートし、それぞれのポートは PowerPC や MicroBlaze と接続するための PLB(Processor Local Bus)か、あるいはユーザロジックと接続するための NPI (Native Port Interface)に設定できます。 MicroBlaze, PLB, NPI, MPMC 等の詳細に関しては Xilinx の技術ドキュメントを参照してください。

本リファレンス・デザインにおいては SATA-IP は MPMC の NPI ポートとダイレクトに接続し、メインメモリ間と DMA 転送を実行します。 本デザインは EDK (Embedded Development Kit)ツール上での周辺 IP コアとして設計されています。 従って EDK 上の NPI ポート数の設定を変更するだけで簡単に SATA チャネル数を増減できるため、容易に RAID システムが構築できます。 ただし評価版においてはハードウエアのデザインは変更できないため、EDK 上でのソフトウエア(MicroBlaze のファームウエア)のみ編集が可能です。 RAID システムなど複数の SATA チャネルを設計するためには、製品版のご購入が必要となります。

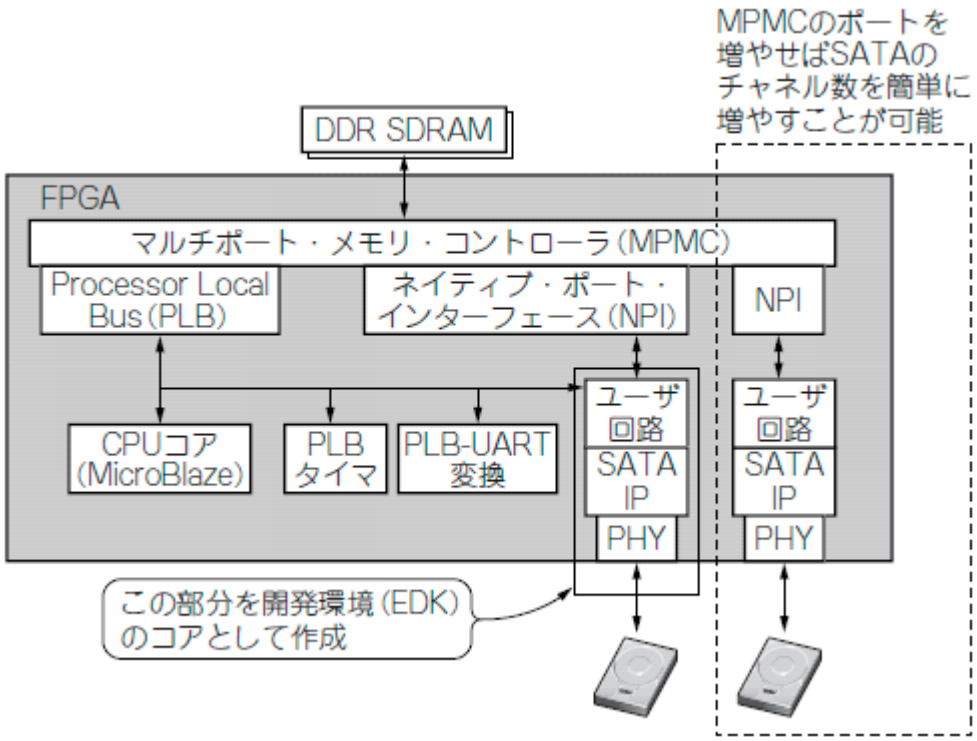

図 5: リファレンス・デザインのブロック図

- メモリコントローラ～IP コア間の接続回路

SATA-IP コアのインターフェイス信号定義を表 1 に示します。 トランスポート・レイヤとのインターフェイス信号は大きくは送信系と受信系の2グループに分かれます。 データ送信時および受信時のトランスポート信号波形をそれぞれ図 6 と図 7 に示します。 図 8 はロジック接続のブロック図です。 表2は MicroBlaze からみたレジスタ・マッピングです。

MicroBlaze 上のソフトウエアは DMA を使いメインメモリ上の FIS データを SATA-IP のリンク・レイヤに送信あるいはリンク・レイヤからのデータをメインメモリ上に受信します。 リンク･レイヤからのデータ受信時は、FIS のヘッダ情報をチェックしデータ FIS だけは別に指定したアドレスに転送するようになっています。 また、送信時もデータ FIS のヘッダをデータパケットに自動的に追加して送信します。このメカニズムにより、ユーザ回路はヘッダ情報を意識せずに送受信データを管理できます。

信号名	方向	説明
共通インターフェイス信号		
trn_reset	In	コアのロジックをリセットするリセット信号、正論理。
trn_link_up	Out	コアと SATA-PHY との通信が確立されると本トランザクション・リンク・アップ信号がアサートされる。
trn_clk	In	コアに対して供給するホストとのトランザクション・インターフェイス信号(trn_xxx)用のクロック信号。 コア外部にて GCLK によるグローバル・クロック・バッファの挿入が必要。 trn_clk の周波数は core_clk と同じかそれ以上とする必要がある。
core_clk	In	IP コアの動作クロック。(SATA-I の場合 37.5MHz で SATA-II の場合 75.0MHz) 本 core_clk は PHY レイヤ内で生成されたものを使う。
dev_host_n	In	コアが SATA Host と SATA Device のどちらとして使われるかを指定する。 SATA Host の場合'0'とし SATA Device の場合'1'とする。

信号名	方向	説明
送信トランザクション・インターフェイス信号		
trn_tsof_n	In	Transmit Start-Of-Frame (SOF): 送信 SATA FIS パケットの開始信号、負論理。
trn_teof_n	In	Transmit End-Of-Frame (EOF): 送信 SATA FIS パケットの終了信号、負論理。
trn_td[31:0]	In	Transmit Data: 送信 FIS パケットの 32 ビット・データ信号。
trn_tsrc_rdy_n	In	Transmit Source Ready: Host は trn_td[31:0]に有効な送信データを用意し本信号を Low とすることで転送を要求する、負論理。
trn_tdst_rdy_n	Out	Transmit Destination Ready: コアは Host から送られる送信データを trn_td[31:0]で受け取ることができる状態を示す信号、負論理。 trn_tsrc_rdy_n は本信号がネゲートされてから4trn_clk 期間以内にネゲートする必要がある。 すなわち IP コアは本信号をネゲートしてから 4DWORD 分までの送信データ(trn_td[31:0])を受け取ることが可能。
trn_tsrc_dsc_n	In	Transmit Source Abort: Host は現在の SATA FIS パケット送信を中断したことを示す、負論理。 SOF から EOF までの期間内であれば Host は本信号をいつアサートしても良い。 Host が本信号をアサートした場合、コアは現在の転送を中断するために SYNC プリミティブを送信する。
trn_tdst_dsc_n	Out	Transmit Destination Abort: コアは現在の SATA FIS パケットを中断していることを示す、負論理。 物理的なリンクがリセット状態となった場合にアサートされる。

表 1: SATA IP コアのインターフェイス信号

信号名	方向	説明
受信トランザクション・インターフェイス信号		
trn_rsof_n	Out	Receive Start-Of-Frame (SOF): 受信 SATA FIS パケットの開始信号、負論理。
trn_reof_n	Out	Receive End-Of-Frame (EOF): 受信 SATA FIS パケットの終了信号、負論理。
trn_rd[31:0]	Out	Receive Data: 受信 FIS パケットの 32 ビット・データ信号。
trn_rsrc_rdy_n	Out	Receive Source Ready: コアが有効な受信データを trn_rd[31:0]に出力されている状態を示す、負論理。
trn_rdst_rdy_n	In	Receive Destination Ready: Host が trn_rd[31:0]で受信データを受け取ることができる状態を示す信号、負論理。 trn_rsrc_rdy_n は本信号がネゲートされてから 4trn_clk 期間以内にコアによってネゲートされる。 従って Host は本信号をネゲートして以降にコアから送られてくる最大 4DWORD 分の受信データ(trn_rd[31:0])を受け取ることが可能な回路を実装しなくてはならない。
trn_rsrc_dsc_n	Out	Receive Source Abort: コアは現在の SATA FIS パケットを中断したことを示す、負論理。 SOF から EOF までの期間内であればいつでもアサートされる可能性がある。
trn_rdst_dsc_n	In	Receive Destination Abort: Host は現在の SATA FIS パケット受信を中断したことを示す、負論理。 Host が本信号をアサートした場合、コアは現在の転送を中断するために SYNC プリミティブを送信する。

信号名	方向	説明
Virtex5 GTP の SATA PHY インターフェイス信号		
PHYRESET	In	コアに対する PHY のリセット信号、正論理。
PHYCLK	In	Virtex5 GTP の SATA-PHY と通信する 16 ビット・データのリファレンスクロック。 1. SATA-I の場合 75MHz 1. SATA-II の場合 150MHz Virtex5 GTP の SATAI/II PHY との通信には送信(TX)と受信(RX)では単一のクロックを共通して使うことができる。 その理由として TX/RX データは実転送速度とリファレンス・クロック周波数の誤差を吸収するために GTP 内部に設けられている elastic バッファが使えるためである。本クロックは Virtex5 内部 DCM によって生成する必要があり SATA 速度ネゴシエーション・ロジックにより動作周波数が選択される。
TXDATA[15:0]	Out	コアから GTP に対して出力される16 ビットの送信データ
TXDATAK[1:0]	Out	送信データのデータ/制御信号の認識シンボルとして使われる 2 ビット信号。 この信号が 0 の場合はデータを、1の場合は制御バイトが TXDATA[15:0]上に出力されていることを示す。
RXDATA[15:0]	In	GTP からコアに出力される 16 ビットの受信データ
RXDATAK[1:0]	In	受信データのデータ/制御信号の認識シンボルとして使われる 2 ビット信号。 この信号が 0 の場合はデータを、1の場合は制御バイトが TXDATA[15:0]上に出力されていることを示す。
RXDATAVALID	In	RXDATA[15:0]と RXDATAK[1:0]上のデータや認識シンボルが有効であることを示す。
LINKUP	In	SATA リンクの通信が確立されていることを示す、正論理。
PLLLOCK	In	GTP 用の DCM がロックできていることを示す、正論理。

表 1: SATA IP コアのインターフェイス信号(続き)

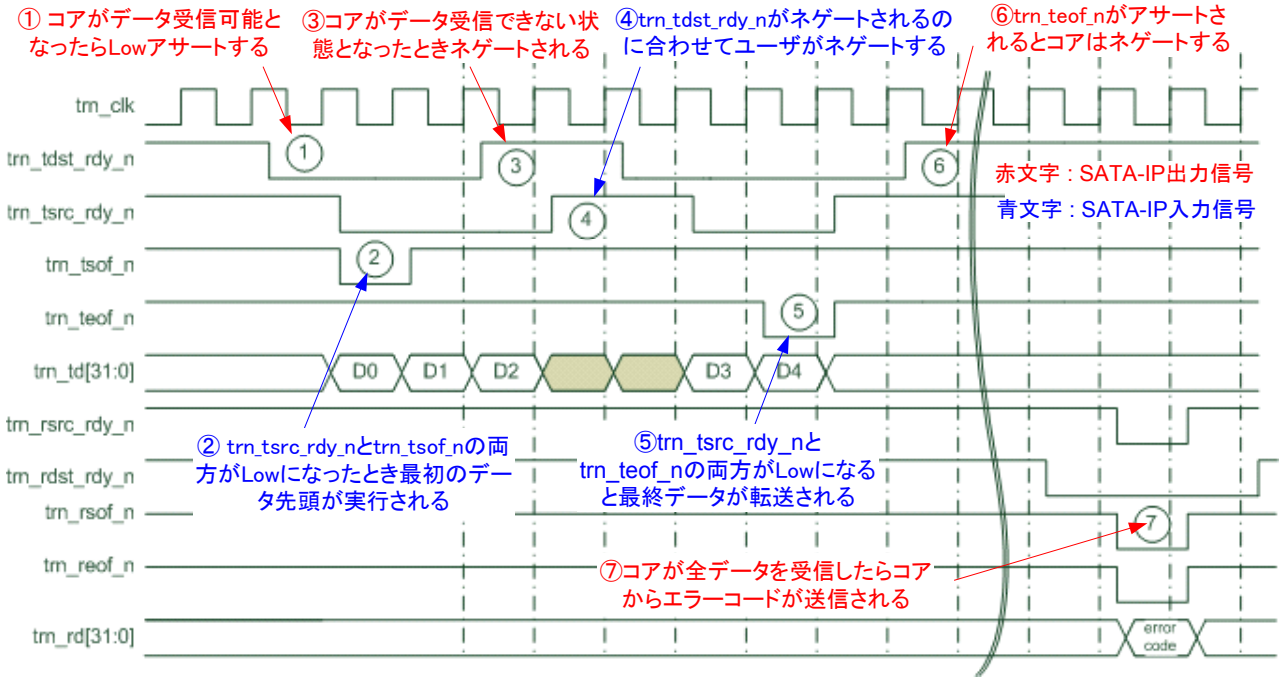

図 6: 送信トランザクションの信号波形

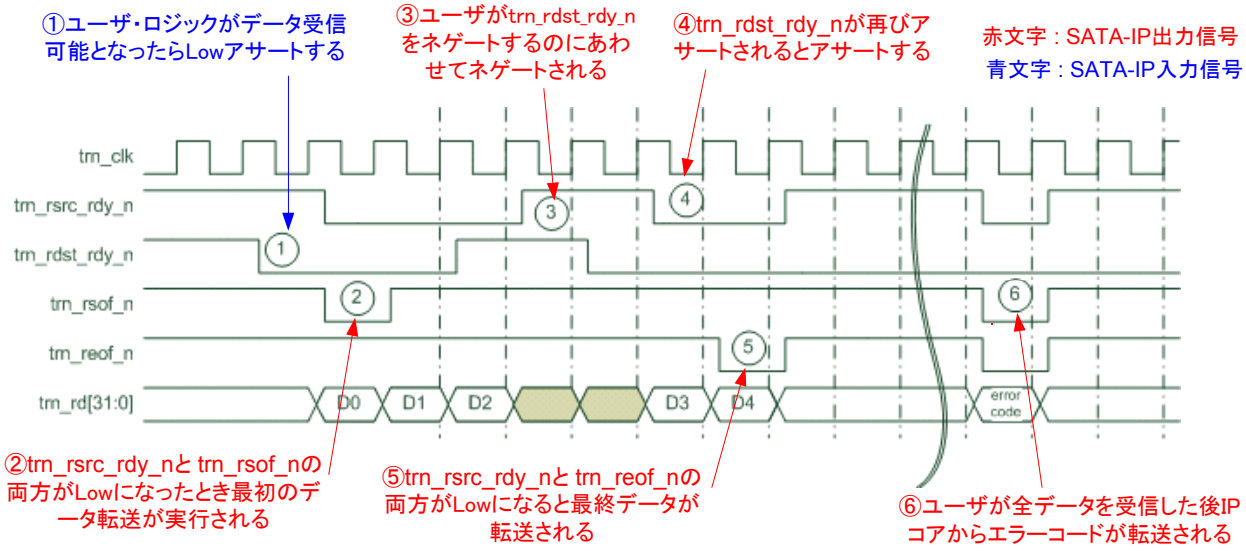

図 7: 受信トランザクションの信号波形

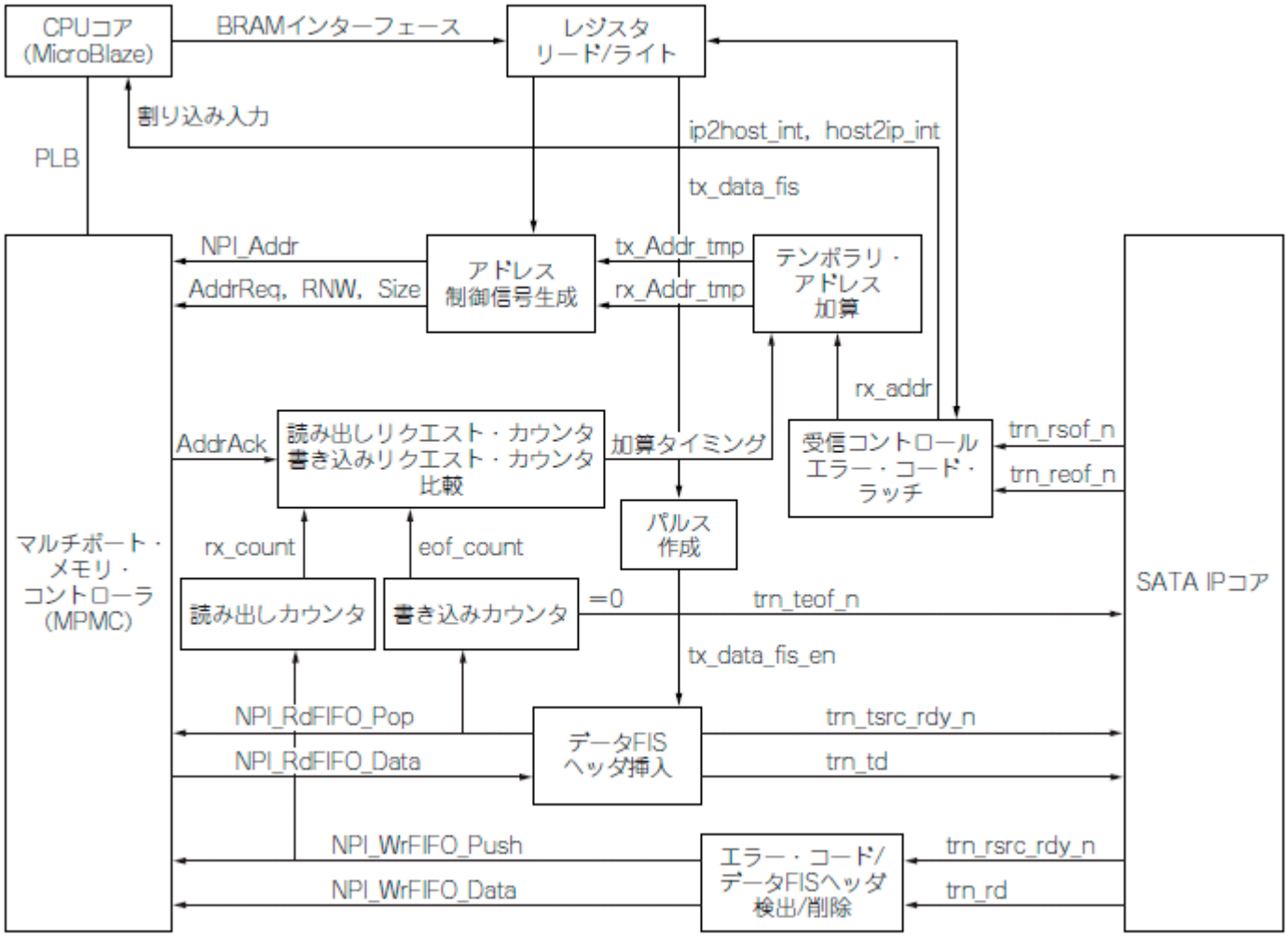

図 8: ロジック接続のブロック図

アドレス Rd/Wr	レジスタ名 (“sata_host.c” でのラベル名)	説明 (Bit ならびはリトル・エンディアンで表記)
BA+0x00 Rd	ステータス・レジスタ (STATUS)	[8]MPMC 準備完了 [7:4]OOB 状態データ [3]GTP PLL がロック [1]Gen2 でリンク [0]SATA IP コアがリンク
BA+0x04 Rd	エラー・コード・レジスタ (ERROR_CODE)	SATA IP のエラーコード。 送信完了および受信完了時にセットされる。 CRC エラーや FIS エラーはここで判別できる。
BA+0x08 Rd	割り込みクリア・レジスタ (INT_CLEAR)	空読みすると割り込みが解除される。
BA+0x0C Rd	受信ワード数レジスタ (RX_COUNT)	受信したワード数、クリアするまで全ての FIS データの受信数が積算される。
BA+0x00 Wr	送信データ格納アドレス (TX_ADDR)	送信するデータを格納している先頭のアドレスを指定する。
BA+0x04 Wr	受信データ格納アドレス 1 (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アドレス。 1つ受信するたびに＋100h アドレスが進む。
BA+0x08 Wr	コントロール・レジスタ (CONTROL)	[31]SATA リセット [30]送信要求 [29]データ FIS を送る [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)	データ FIS を受信したときに格納される先頭アドレス。

(BA : ベース・アドレス)

表 2: MicroBlaze 側からのレジスタ・マップ

4. 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 MicroBlaze 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 MicroBlaze 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, DMA READ EXT, DMA WRITE EXT となります。 デバイスとして接続する HDD は 48 ビット LBA(LogicalBlock Address)かつ SATA-II 対応のものを使う必要があります。

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 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: IDENTIFY COMMAND の FIS 構造

- DMA READ EXT

表 4 は SATA デバイスからデータを読み出す DMA READ EXT 命令の FIS 構造です。 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 DMAREAD の方が手順が簡単なので、こちらを使います.

コマンドは 25h、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 4: DMA READ EXT の FIS 構造

- DMA WRITE EXT

表 5 は SATA デバイスへデータを書き込む DMA WRITE EXT 命令の FIS 構造です。 コマンドは 35h、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 5: DMA WRITE EXT の FIS 構造

- リファレンス・デザインの動作について

本リファレンス・デザインのソフトウエアのソースコードは“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 9 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。

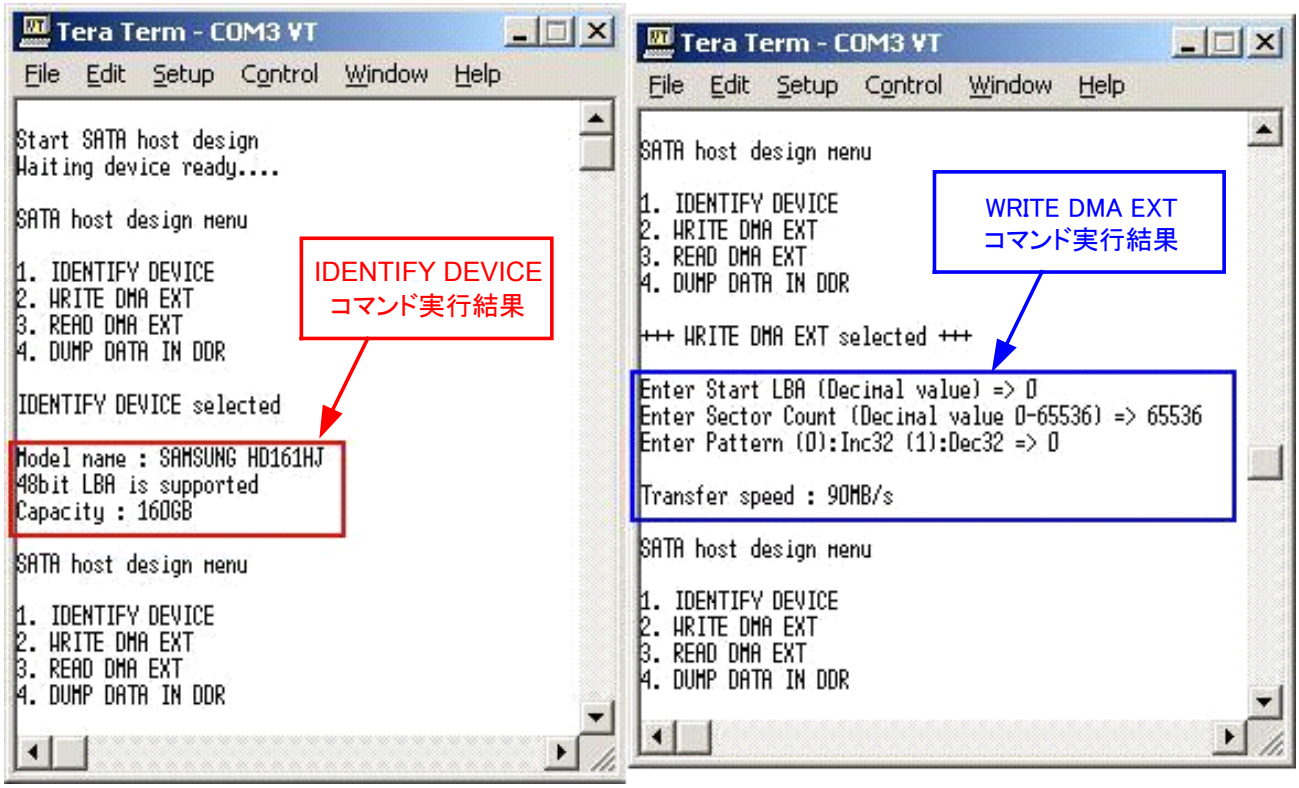

図 9: 動作実行時のシリアル・ターミナル画面サンプル

5. 改版履歴

リビジョン	日付	内容
1.0	2008/10/08	評価版の記載を含めた第 1 版のリリース
1.1	2008/11/10	SATA の紹介を追加
1.2	2008/11/14	AutoNegotiation 機能のサポート追加
1.3	2008/12/12	dev_host_n 信号の追加とテストアプリケーションメニューの更新
1.31	2009/04/18	trn_reset の論理を正論理(Active High)に修正
1.4	2009/06/05	trn_clk の周波数が core_clk 以上とする必要がある制約の記述を追加