TECH I Bus Interface
Technology Interface
Vol.54

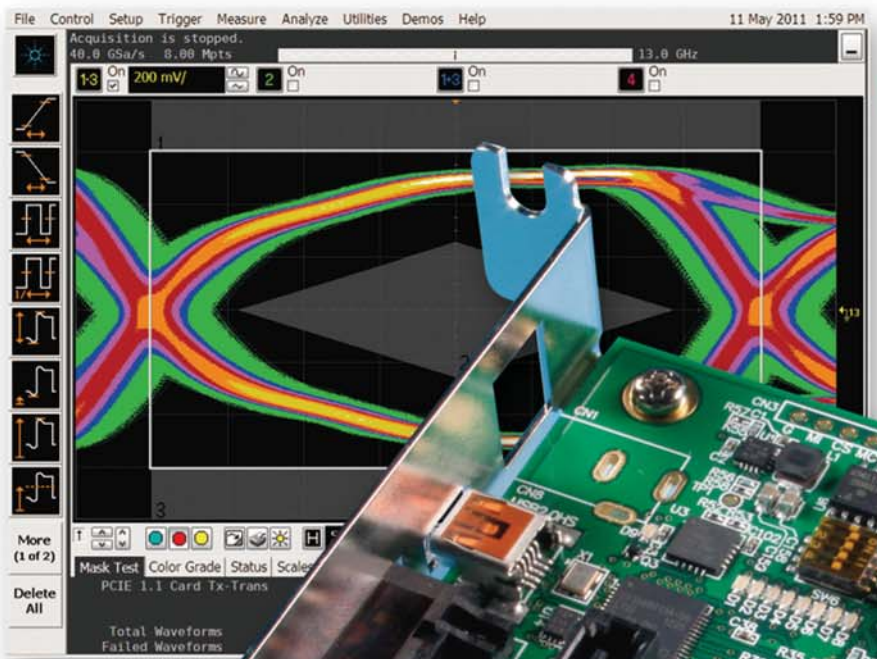
FPGAでゼロから作る PCI Express

見本

PC拡張用の定番バスはこうやって動かす

内藤 竜治●著

通信メカニズムのすべてがわかる!

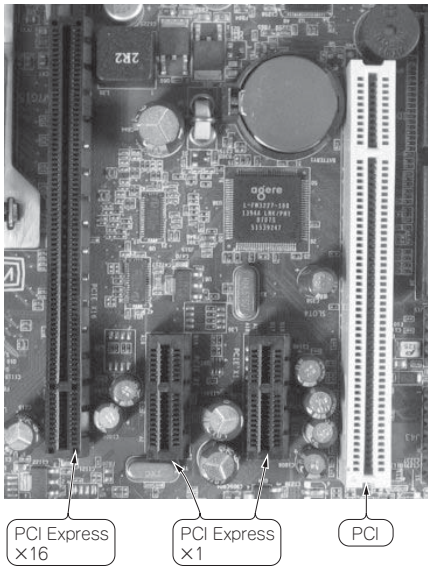


CQ出版社

Chapter 1

バス接続からポイント-ポイント接続へ
PCIバスとの比較から見る

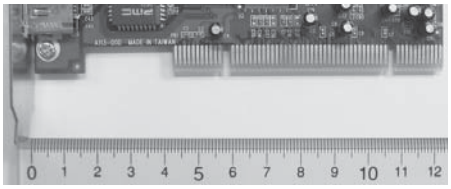
PCI Expressの基礎知識



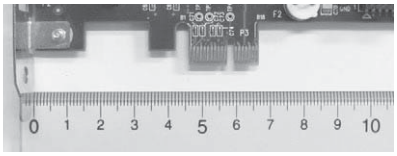
（a）PCIとPCI Expressを備えたマザーボードのコネクタ

1．似て非なるPCIとPCI Express

PCIとPCI Expressは名前は似ていますが，実は全く異なる規格のバスです．双方の主な特徴を**表1**に示します．



（b）PCIアドイン・カードの端子部分



（c）PCI Expressアドイン・カードの端子部分（×1）

写真1
PCIとPCI Expressのコネクタ

表1
PCIとPCI Expressの特徴

	PCI/PCI-X	PCI Express
トポロジ	バス型	ポイント・ツー・ポイント型
プロトコル	多数の制御信号を用いる	パケット処理
伝送方式	パラレル（32/64ビット），シングルエンド信号	シリアル，差動信号
伝送速度	33MHz～1066MHz	2.5Gbps（1レーン当たり）
高速化の方法	バス幅を広げる， クロック速度を上げる， DDRやQDRを採用する，など	レーンを増やす （×1，×2，×4，×8，×12，×16，×32） 伝送速度を上げる〔Gen2（5Gbps），Gen3（8Gbps）〕
エラー・ チェック方法	●パリティ・エラーの検出（PCI） ●ECCチェックの検出（PCI-X） ●検出したエラーの処理方法については規格では定めていない	●2段階のCRCチェック ●無効なシンボルのチェック ●パケットの形式チェック ●自動的な再送 ●エラー・レポート
通信の品質	システムの設計者による	●フロー・コントロールによって自動的に速度を制限 ●QoS（Quality of Service）によって重要な通信の品質を確保
耐故障性	1本でも信号が切れたら全く通信できない	一部のレーンが故障しても残ったレーンを用いて動作できる

Chapter 2

NXP 社製 PX1011A & TI 社製 XIO1100 と FPGA の接続

PIPE インターフェースと PHY チップの使い方

1. PIPE とは

● 物理層と論理層を接続する重要なインターフェース

PCI Express は，2.5Gbps という伝送速度をもつ高速シリアル・インターフェースです．一般に，この速度の信号を直接扱うことができる FPGA はピン数も多く高価で，複数の大電流と高精度な電源が必要になるなど，手軽に扱えない場合も多いでしょう．

×1 の速度でもよいから，とにかく PCI Express に対応させたいという場合には，物理層（PHY）チップを使うと便利です．PHY チップは，シリアル-パラレル変換や弾性バッファなど，PCI Express の物理層の高速で難しい処理を行ってくれる（**図 1**）ので，FPGA は PCI Express のリンク状態を管理する LTSSM（リンク・トレーニング・ステートマシン）やレーン間デスキューといった，論理層（MAC）より上の層の処理に専念することができます．

PHY チップと MAC 層の間のインターフェースの方法は，Intel 社がパイプ（PIPE：PHY Interface for the PCI Express Architecture）という規格を策定しています．

本章では，PHY チップと PIPE の使い方について解説します．

● PHY チップを使うメリット

PCI Express のアドイン・カードを実現するには，高速シリアル・インターフェースを内蔵した FPGA を用いる方法のほか，専用の外付けチップ（PHY チップ）を使う方法があります．PHY チップを使うと，低価格な FPGA を利用できること以外にもいくつかのメリットがあります．

例えば，PCI Express のクロックはスペクトラム拡散が施されているので，FPGA でダイレクトに実現するならば，スペクトラム拡散に対応した PHY を持った FPGA でなければなりません．また，PCI Express の高速差動信号は，電気的アイドルの送信や検出，レシーバの検出など，データ転送以外にもいろいろなことをやらなければなりません．

FPGA に内蔵されている PHY 機能は汎用的なものなので，こういう特殊なステートを実現できるかどうかはケース・バイ・ケースですが，PHY チップを使うと確実に実現できます．

また，FPGA に実装する IP コアの入出力を PIPE に適合させると，以下のようなメリットもあります．

- 複数のメーカの PHY チップの中から最も用途に適合するものを選択できる
- 125MHz 程度で動作する安価な FPGA でよい
- 電気的アイドルやスペクトル拡散，レシーバの検出などの特殊な状態に対応できる
- 弾性バッファのように，複数のクロック・ドメインを

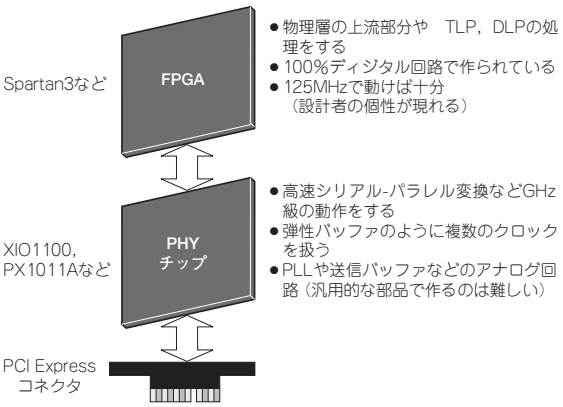


図 1　PHY チップを使うと PCI Express が簡単に作れる

Chapter 3

信号遅延の要因とタイミング調整，Double Data Rate 動作の活用

200MHz を超える I/O を扱うロジックを FPGA に実装する方法

最近の FPGA は，当たり前のように 100MHz 以上の速度で動作するようになってきました．しかし，プリント基板上の配線で 100MHz を超える信号を扱うことは，それほど簡単ではありません．

FPGA が出力する信号は，何も工夫しないと**図 1**（**a**）のように，バラバラのタイミングで出力されます．このような波形では 100MHz を超えたあたりからその動作がだんだん怪しくなってきます．できれば，出力は**図 1**（**b**）のように，すべての信号が同じタイミングで遷移するのが理想です．このような波形ならば，I/O の周波数はどこまでも上げることができるでしょう．

本章では，FPGA の入出力パッドのタイミングを自由自在に操り，200MHz を超える I/O でも安定して動作する回路の作り方について考えます．

1．FPGA の実力

● PERIOD 制約とクロック周波数

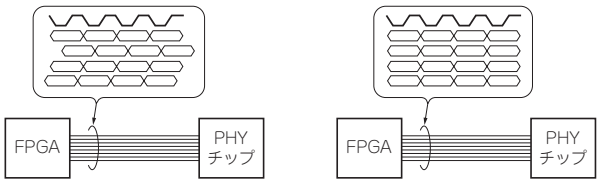
Xilinx 社の FPGA を使って回路設計をする場合，UCF ファイルに PERIOD という制約を書くと，最小のクロック周期（すなわち最大のクロック周波数）を指定することができます．配置配線ツールは，この指定を守るように最大限の努力を払います．

```
NET "clk" TNM_NET = "clk";
TIMESPEC "TS_clk" = PERIOD "clk" 5 ns
   HIGH 50 %;
```

● FPGA の最大動作周波数は何で決まるか

Xilinx 社の無償開発ツール ISE WebPACK 9.1i を用いて，**リスト 1** のような 32 ビット・カウンタを作ってみました．デバイスは，Spartan-3E/1200 を指定しました．論理合成後，レポート中のタイミング・サマリを見ると 192 MHz という結果が表示されていました．これは論理合成時点での予測値なので，実際のデバイスに配置配線した後の結果とは若干異なります．UCF ファイルで制約をかけて配置配線すると，4.823ns（207MHz）まで上がりました．

では，なぜ 192MHz や 207MHz という結果が出たのでしょうか．FPGA の動作タイミングは，**図 2** のように考えることができます．最大動作周波数は，FPGA 内のフリップフロップのセットアップ・タイムと，組み合わせ回路の遅延時間，配線の遅延，フリップフロップの出力遅延時間の和の逆数となります．



（**a**）普通に設計した場合，出力される信号のタイミングがばらついてしまい高速動作ができないため，100MHz 程度が上限になる

（**b**）FPGAのI/Oを工夫した場合，タイミングがそろうようになり，数百MHzの速度でも扱えるようになる

図 1　パラレル信号はタイミングをそろえるのが難しい

リスト 1　32 ビット・カウンタの HDL ソース

```
Port ( clk    : in  STD_LOGIC;

    ～中略～
    signal count     : std_logic_vector(31 downto 0);
    ～中略～

    process(clk) begin
         if(clk'event and clk = '1') then
              count <= count + 1;
         end if;
    end process;

    count_op <= count;
```

Chapter 4

PIPEのアクセス方法と物理層リンクアップの流れ

PCI Expressの物理層

FPGAとPHYチップをPIPEでつなぐと，どのようなデータが送られてくるのでしょうか．**図1**は，実際に送られてきたデータをキャプチャしたものです．こういったデータを解析して処理するのが物理層の仕事です．そこで本章では，物理層の作り方について考えます．

1．物理層は何をやっているか

PCI Expressの物理層は，電気サブブロックと論理サブブロックに分けられます(**図2**)．PHYとFPGA間を接続するインターフェースはPIPE(パイプ)と呼ばれます．PHYは第2章で解説したとおり，電気サブブロックを担当します．本書で設計するIPコアは，論理サブブロックより上の層を実現します．

論理サブブロックの役割は，次のとおりです．

- **オーダード・セット**(物理層パケット)を処理する
- リンクの確立とネゴシエーションを行う
- 電気的アイドル状態への遷移を管理する
- 必要に応じてスクランブルを行う
- 受信したDLLP(データリンク層パケット)とTLP(トランザクション層パケット)をDLL層へ渡す
- TLP，DLLP，オーダード・セットを送信する
- ×2以上の構成ではByte Stripingとその復元を行う

● Kコード

PIPEのデータ・バスには，8ビットのデータのほか，通常のデータとコマンドの区別を表す1ビットの符号があります．この符号が立てられると，Kコードという特殊なコマンドとして扱われます．PCI Expressで使われるKコードの一覧を**表1**に示します．通常のデータは，Dコードと呼ばれます．

● スクランブル

スクランブルというのは，伝送路上を同じデータ(例えば，通信がない場合のD0.0)が連続した場合に，特定の周波数のEMIノイズが強くなるのを避けるためにデータに乱数を足す処理です．

スクランブル回路は，**図3**のようにして作ります．これは擬似乱数を用いて，入力データのXORを取るというものです．受信側にも同じ乱数発生回路(LFSR)を用意して

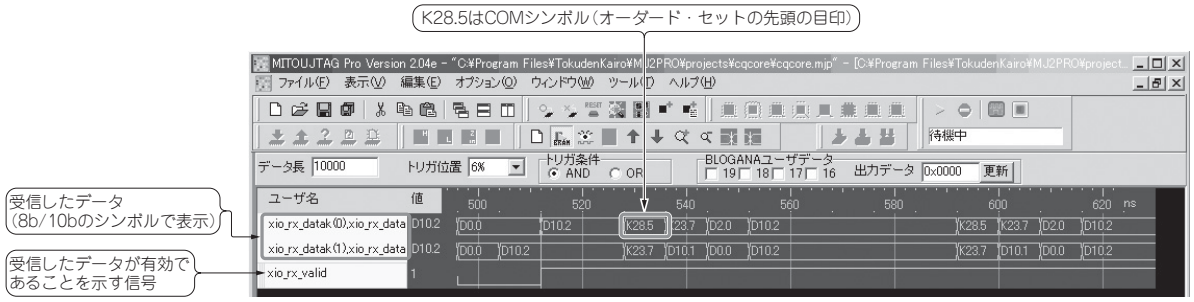


図1　PIPEから受信した最初のパケット

RX_VALIDが有効になって初めて現れたパケットは，「K28.5 K23.7 K23.7 D10.1 D2.0 D0.0 D10.2…」という並びであった．この図は筆者の開発したMITOUJTAGというソフトウェアでキャプチャしたもの．FPGA内部のブロックRAMに波形を溜め込んで，JTAGを通じてパソコンに取り込んで表示している．

Chapter 5

データリンク層の役割から
Ack/Nak プロトコル，CRC 生成回路の設計まで

データリンク層の概要

1．データリンク層の概要

● データリンク層とは

データリンク層は，トランザクション層と物理層の間にあり（**図 1**），PCI Express の 3 階層の中間部分を構成しています．この層の目的は，リンクの両端の二つのデバイス間で TLP（トランザクション層パケット）を確実に交換するしくみを提供することにあります．

データリンク層が提供するサービスには，データ交換，エラー検出と再送，フロー・コントロール，初期化とパワー・マネージメントがあります（**表 1**）．

データ交換は，トランザクション層から受け取ったデータを物理層に渡し，物理層から受け取ったデータをトランザクション層に渡す機能です．

エラー検出は，TLP の中に含まれるシーケンス番号と LCRC（Link CRC ；TLP の中にある 32 ビットの CRC）をチェックして，データが壊れていないか，あるいは途中でパケットが失われていないかどうかを調べる機能です．そして，リンクの向こう側の通信相手に Ack（肯定応答）または Nak（否定応答）を送り，正しく受け取れたかどうかを知らせます．

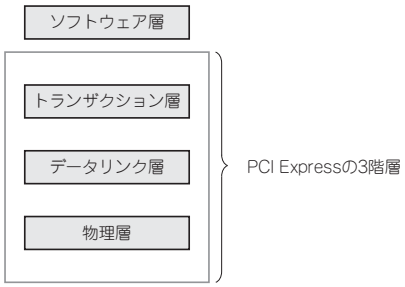


図 1
データリンク層の位置

送信側のデータリンク層は，TLP を送信した後に相手が Nak を返してきたり，規定時間内に Ack が返ってこない場合はどこかでエラーが生じたものと判断して，前に送信した TLP を再び送ります．この再送のメカニズムにトランザクション層は関与しないため，データリンク層が自発的に行わなければなりません．そのため，データリンク層はトランザクション層から受け取った TLP を一時的なバッファに保存しておき，必要になったらいつでも再送できるように備えています．

フロー・コントロールは，受信側の処理速度に合わせて送信速度を制限するためのしくみです．このしくみを実現するため，リンクの両端のデバイスは，フロー・コントロール・パケットという DLLP（データリンク層パケット）を頻繁に交換して，相手の受信バッファの残量を常に監視し合っています．

これらのほかに，データリンク層には物理層のリンク状態をトランザクション層に知らせたり，パワー・マネージメントの処理もあります．

表 1　データリンク層の提供するサービス

データ交換
● トランザクション層から受け取ったデータを物理層に渡す
● 物理層から受け取ったデータをトランザクション層に渡す
エラー検出と再送
● TLP シーケンス番号と LCRC の生成
● 再送に備えて TLP を保存しておくこと
● TLP と DLLP におけるデータ整合性チェック
● Ack/Nak プロトコル
● エラー検出時の報告
● タイムアウトと再送
フロー・コントロール・パケット通信
初期化とパワー・マネージメント
● リンク状態の追跡
● Active/Reset/Disconnected をトランザクション層に通知

Chapter 6

トランザクション層の概要からTLP送受信ブロックの設計

トランザクション層の概要

本章では，トランザクション層の設計を行います．トランザクション層の目的は，TLP（トランザクション層パケット）と呼ばれるパケットをやりとりして，メモリのリードやライト，イベント通知などのトランザクションを処理することです．

これらはユーザから見えるデータのやりとりですが，このほかにもトランザクション層にはフロー・コントロールや電源管理，QoS（Quality of Service）など，ユーザからは見えない地味な機能もあります．

1．トランザクション層の概要

図1に示すように，トランザクション層はPCI Expressのアーキテクチャの最上位に位置づけられています．トランザクション層は，ソフトウェア層（あるいはデバイス・コアと呼ばれる）から指令を受けてTLPを生成し，データリンク層を介して相手に送ります．つまり，

①デバイス・コアからの要求に従ってTLPを生成する
②何らかのリクエストTLPを受信したら，解読して，デバイス・コアに要求を伝える
③コンプリーションTLPを受信したら，ペイロードやステータスを取り出して，デバイス・コアに伝える
④（オプションで）エンド・ツー・エンドでのデータ整合性をチェックする

という動作を行います．いわば，従来のPCIでサポートされていたような，メモリやコンフィグレーション空間へのアクセスと互換性のある動作を提供します．

● トランザクションとは

PCI Expressでは，トランザクションという単位でデータの転送します．これは，リクエスタ（要求を出す側）とコンプリータ（完了させる側）との間で転送される一連のパケットのやりとりのことです．

PCI Expressでは，3種類のアドレス空間（メモリ空間，I/O空間，コンフィグレーション空間）とメッセージ空間が定義されています．そして，それぞれの空間に対してトランザクション（メモリ，I/O，コンフィグレーション，およびメッセージ）が定義されています（**表1**）．メモリやI/O，コンフィグレーション空間へのアクセスには，リードとライトが，メッセージにはベース・ライン[注1]というトランザクションが定義されています．

表1では，メモリ・ライトとメッセージ・トランザク

注1：ベース・ラインという用語には適切な訳が見つからないので，このまま用いた．

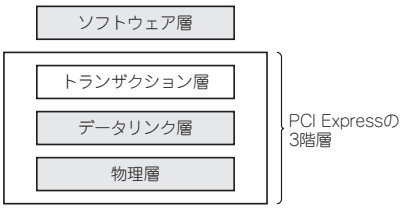


図1　トランザクション層の位置

表1　PCI Expressのトランザクションの種類

対象となるアドレス空間	トランザクションのタイプ		基本的な用途
メモリ	リード	NonPosted	メモリ空間へのデータ転送
	ライト	Posted	
I/O	リード	NonPosted	I/O空間へのデータ転送
	ライト	NonPosted	
コンフィグレーション	リード	NonPosted	デバイスのコンフィグレーションとセットアップ
	ライト	NonPosted	
メッセージ（ベンダ定義を含む）	ベース・ライン（ベンダ定義を含む）	Posted	イベント発生の通知と汎用的なメッセージ伝達

Chapter 7

PCIバスから大きく拡張された
プラグ&プレイを支えるレジスタ群

PCI Expressの コンフィグレーション空間の概要

1．PCI Expressの コンフィグレーション空間

PCI Expressにおけるコンフィグレーション空間の全体像を**図1**に示します．コンフィグレーション空間は全部で4,096バイトあり，下位256バイトは従来のPCIと互換性のある領域で**PCI（互換）コンフィグレーション空間**と呼ばれます．上位3,840バイトは，**PCI Express拡張コンフィグレーション空間**と呼ばれるPCI Express独自の領域です．

● PCIコンフィグレーション空間

先頭256バイトの領域へのアクセス方法ですが，PC/AT互換機の場合はI/OポートのCF8h/CFChを経由してアクセスします．

特に，PCIコンフィグレーション空間の先頭64バイトは，**コンフィグレーション・ヘッダ**と呼ばれ，従来のPCIと互換性のあるレジスタが並んでいます．

PCIコンフィグレーション空間の後ろの192バイトは**PCI Express機能構造体**などを格納するために，デバイス（やIPコア）の設計者が自由に使ってよい領域です．PCI Express機能構造体には，**PCI Express機能レジスタ**が含まれます．この領域はPCI Expressだけの特別なものではなく，PCI 2.2で定義されたものです．

機能構造体や新機能レジスタの役割は，本章の最後で説明します．

● PCI Express拡張コンフィグレーション空間

コンフィグレーション空間のアドレス100h～FFFhの3,840バイトは，PCI Express拡張コンフィグレーション空間[注1]と呼ばれ，**PCI Express機能拡張レジスタ**を格納

注1：機能レジスタや機能構造体，拡張機能などと似たような名前が多く非常にややこしいが，「拡張」と付いたものは100h番以降のアクセスしにくい領域にあると覚えておけばよい．

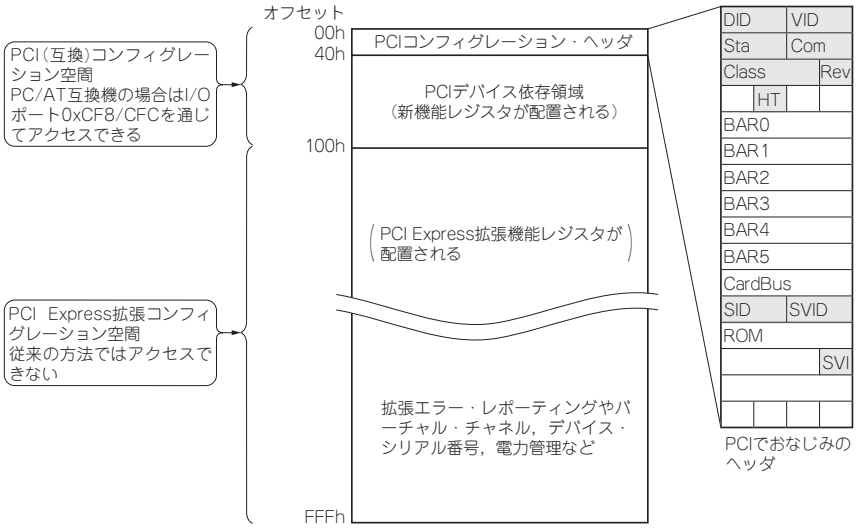


図1
コンフィグレーション空間の全体像
従来のPCIコンフィグレーション空間のほかに広大な領域が用意された．

Chapter 8

コンフィグレーション空間とメモリ空間を実装して
メモリ・アクセスをテストする

最も基本的なPCI Expressデバイスの設計

基礎知識編で説明してきた，トランザクション層とユーザ回路との間をつなぐ部分を，コア接続層と呼ぶことにします．

コア接続層は，**図1**のような構造をしています．この中には，コンフィグレーション・レジスタやメモリ・アクセスのヒット条件を判定する回路が入ります．しかし，コア接続層の具体的な動作は，PCI Expressの規格では定義されていないので，ここから先はオリジナルの方法で実装を考えることになります．本章では，このコア接続層の設計方法について解説します．

1．コンフィグレーション空間の実装

● コンフィグレーション・リードの実装

第7章で説明したように，PCI Expressのエンドポイントには，さまざまなコンフィグレーション・レジスタがあります．最初に，VHDLでこれらのレジスタを宣言するところから始めましょう（**リスト1**）．

トランザクション層は，ルート・コンプレックスからコンフィグレーション・リード・リクエストを受信すると，受け取ったアドレスをrcvd_addr_opバスに出力し，

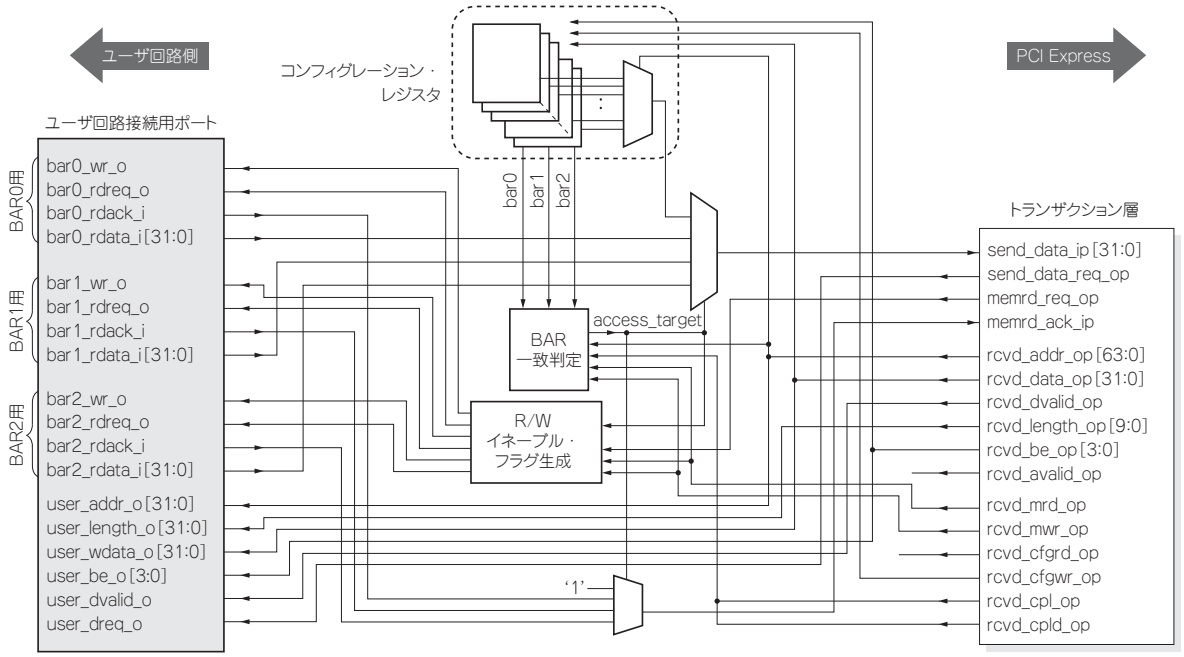


図1　今回作成するコア接続層の概略構成図
トランザクション層とユーザ回路をつなぐのが主な役割．複雑なロジックはなく，セレクタやアドレスの一致判定回路などがある．

Chapter 9

Windows上からコンフィグレーション空間やメモリ空間にアクセスする

汎用PCI Expressデバイス・ドライバの作成

● PCIデバッグ・ライブラリの問題点

第8章の実験では，開発中のハードウェアにアクセスするために，参考文献(4)に記載された汎用PCIデバッグ・ライブラリを使用してきました．このライブラリは，ユーザ・モードのアプリケーションから，

```
_MemReadLong( 0x12345678);
```

や，

```
_IoWriteChar( 0x378, 0x0f);
```

のようなユーザ向け関数を呼ぶだけで，ハードウェアへの直接アクセスができてしまうという大変素晴らしいものでした．しかも，自分で作ったハードウェアだけではなく，任意のアドレスに対してもアクセスできてしまうという危険な魅力も持ち合わせていました．

汎用PCIデバッグ・ライブラリは，デバイス・ドライバ(pcidbgnt.sys)ならびに，それを利用するためのDLL(pcidebug.dll)から構成されています．SYSファイルはDLLの起動時に自動的に組み込まれるため，ユーザ・アプリケーションからDLLをリンクするだけでよいという使い勝手のよさがありました．

しかし，このライブラリを使ってPCI Expressにアクセスしようとすると，以下の問題点があることがわかってきました．

- PCI Expressに対するバースト転送を発行できない
- NT形式のデバイス・ドライバなので，今後も使えるかどうかわからない
- コンフィグレーション・リード/ライトを発行したときに，関係のないレジスタまでアクセスされてしまう

そこで，バースト転送の発行を最大の目的として，WDM(Windows Driver Model)形式の汎用デバイス・ドライバを開発することにします．

なお，デバイス・ドライバの開発について詳しく説明することはできないので，ある程度の開発経験がある読者を対象に話を進めます．ドライバの開発を行ったことがないという方は，参考文献(4)や(12)をご一読ください．後者は薄めの書籍ですが，DDKを用いたビルド方法からハードウェアを直叩きする方法まで，短くまとめて解説されています．これほどまでにエッセンスが凝縮された本は，ほかに見たことがありません．

それに対して，USBを使ったデバイス・ドライバの解説本や解説記事はたくさんありますが，初めて学ぶのには不向きです．なぜなら，USBドライバはハードウェアに直接アクセスしないため，PCI Expressとはだいぶ違うからです．

1．WDM形式の汎用ドライバを作る

● WDM形式とNT形式の違い

Windowsのデバイス・ドライバには，古いNT形式(SYS形式とも呼ぶ)と，Windows 2000以降で使われるようになったWDMやWDFといった形式があります．これらの違いを一言でいえば，ドライバがカーネル・モードの特権を利用して勝手にハードウェアにアクセスできていた時代から，ドライバがOSの管理下におかれる階層の一部分となったということです(**図1**)．

なぜ，自分が作ったハードウェアを自由に触らせてくれないのでしょうか．組み込みCPUを触っているエンジニアなら，きっとハードウェアを直接アクセスしたくて仕方がないでしょう．しかし，考えてみてください．ハードウェアを直接アクセスしたところで，せいぜいI/Oポートとメモリ・アクセスしかできませんから，USBやネットワークの先にあるファイル・システムを扱うようなドライバまで全部自分で作るのは，とても現実的ではありません．

そのため，いろいろなシステムで共通して使われる低レ

Chapter 10

リード／ライトともにバースト転送で転送レートを上げるには

DMAの実装とWindowsにおけるメモリ管理

1．DMAについての基礎知識

● DMAとは何か

DMA（Direct Memory Access）とは，周辺装置がメモリ・バスを操作してメイン・メモリに直接アクセスすることをいいます（**図1**）．DMAには，大量のデータを転送する高速なものと，非同期の低速データを扱うものがあります．ここでの目的は，アドイン・カード上のデータを高速に読み出すことなので，高速な方のDMAを扱います．

また，PCI Expressのルート・コンプレックス（チップ・セットに内蔵されている）は，メイン・メモリへのアクセスも担うので，厳密に言えばDMAを行うのはアドイン・カードではなくルート・コンプレックスになります．PCI Expressのエンドポイントは TLP（トランザクション層パケット）を発行しているだけであって，CPUに対してDMAを実行しているわけではありません．

しかし，一般的には，アドイン・カード上に置かれたDMAコントローラから要求を発行してメイン・メモリとアドイン・カードとの間で直接データ転送を行うことをDMAと呼んでいます．PCIのカードでは，バス・マスタがバースト転送を発生させてメイン・メモリにアクセスすることをDMAと呼んでいましたが，それのPCI Express版と考えればよいでしょう．

● バス・マスタDMAとシステムDMA

Windowsには，バス・マスタDMAとシステムDMAという概念があります．システムDMAというのは，マザーボード上に搭載されている共有のDMAコントローラを使って行うものです．これは本書では扱いません．

バス・マスタDMAというのは，アドイン・カード上に置かれたDMAコントローラがメイン・メモリと通信するものです．DMAコントローラはアドイン・カードの設計者が設計するもので，その仕様はカードごとに異なります．本書では，こちらのDMAを扱います．

● DMAのやり方

PCI ExpressにおけるDMAは，ハードウェア的には難しいものではありません．エンドポイントの側からメモリ・リード・リクエストやメモリ・ライト・リクエストを発行して，パソコンのメイン・メモリ上のデータを読み出したり，データを書き込んだりするだけのことです．したがって，メイン・メモリ上の物理アドレスと長さを正しく指定したTLPを発行すれば，自由にパソコンの中のメモリを読み書きできてしまうはずです．

それでは，ここで使う物理アドレスと長さはどうやって調べればよいのでしょうか？ ここが，DMAを行うためのデバイス・ドライバのややこしいところであり，Windowsの提供するしくみの素晴らしいところでもあります．

本章では，このしくみについて詳しく見ていきます．

2．ハードウェアの実装

● TLP送信ステート・マシンの改良

DMAのハードウェアは驚くほど簡単です．第6章で説

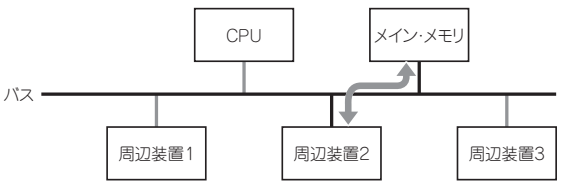


図1　DMAとは

DMAは周辺装置がバスを操作してメイン・メモリに直接アクセスすることをいう．DMAの実行中は，CPUがバスを開放する．この図は簡略化しているが，実際のパソコンのバスはもっと複雑である．

Chapter 11

非同期で発生するイベントをシステムに通知する方法

PCI Expressの割り込み処理の実装

1．割り込みのハードウェア

● PCIの割り込み処理

レガシなPCIには，INTA# ～ INTD#という4本の**割り込み信号線**（以下，INTxと略）があり，アドイン・カードはこの信号線のどれか一つを“L”レベルにすることによって，システムに割り込みをかけられるようになっていました．

この信号は，マザーボード上に並んだPCIのスロットで**共有**されていて，順番にひねったようにつながっています（**図1**）．そして，各信号がワイヤードORでつながっていて，複数のカードで同一の割り込みラインを共有することも可能でした．例えば，**図1**の例では，スロット1のINTA#とスロット2のINTB#は，同一の線を使っています．それぞれのアドイン・カードがINTA#とINTB#を使うなら，これらの割り込みは共有になります．

また，INTxはエッジではなく，レベル・トリガです．つまり，INTxが“L”レベルの状態にある限り，システムに割り込みがかかり続けます．

レガシPCIは，**レベル・トリガ**と**ワイヤードOR**によって，複数の割り込みが共有できていました．

注1：Message Signaled Interrupt．PCI 3.0でオプションとして規定された方式．

● PCI Expressの割り込み発生方法

PCI Expressの割り込みには，従来のレガシPCIで使われていたINTA# ～ INTD#をエミュレートする方法と，**MSI**[注1]または**MSI-X**を用いる方法があります．レガシなINTxをエミュレートする方法は，下位互換性のために残されています．INTxを用いる方法は簡単であり，システムのブート時からすぐに使えるというメリットがあります．しかし，複数の割り込み要因が共有されるというデメリットもあります．

PCI Expressでは，INTx割り込みを発生させるデバイスは，MSIまたはMSI-Xか，もしくはその両方をサポートすることが必須となりました．デバイスが両方の割り込み方法をサポートしていても，同時に使われるのはどちらか一方です．

● INTxエミュレーションによる割り込み発生

PCI ExpressにはINTxの物理的な制御線はないので，**仮想ワイヤ**というしくみでそれをエミュレートします．これは，エンドポイントとルート・コンプレックスはINTxの状態を覚えておいて，**メッセージ・パケット**を使って変化するタイミングを知らせるというものです（**図2**）．

INTxを“L”レベルに下げたい場合には**Assert_INTx**メッセージを送信し，“H”レベルに戻したい場合は**Deassert_INTx**メッセージを送信します．このメッセージ・パケットは，**図3**に示す構造をしています．

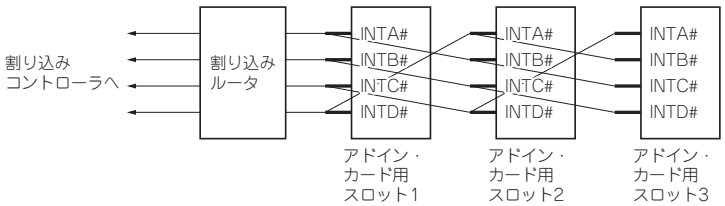


図1
レガシPCIにおける割り込みのルーティング
割り込みルータは，PCIデバイスの割り込み線をシステムの割り込み線に割り当てて接続する．

Chapter 12

1チップでPCI Expressデバイスを実現できる
ArriaGXデバイスの使い方

ArriaGX内蔵トランシーバの使い方とPCI Expressコアの移植

以前は，トランシーバ内蔵のFPGAはとても高価（10万円前後）で，小さいパッケージのものがなかったため10層くらいの基板を作らなければならず，敷居が高いものでした．しかし，現在ではギガビット・トランシーバが内蔵されたミドル・クラスのFPGAが安価に発売されて，入手も容易です．また，消費電流も減って基板の設計も楽になったように思います．

そこで本章では，これまでに作ってきたコアを米国Altera社のミドル・レンジFPGAであるArriaGXに移植し，内蔵トランシーバを使って動作させることにします．

1．ArriaGXと評価ボード

ArriaGXは，Altera社のミドル・レンジFPGAのファミリの一つで，ギガビット・トランシーバを内蔵しています．

CQ出版社から発売されているArriaGX評価キット（**写真1**）は，ArriaGXのEP1AGX20CF484C6Nを搭載したFPGA評価ボードで，PCIとPCI Expressの両方のカード・エッジを持っています．FPGAの型番のEP1AGX20Cはデバイスの規模を，F484はパッケージの形状（484ピンBGA）を表しています．

EP1AGX20Cには，ALM（アダプティブ・ロジック・モジュール）が8,632個，内蔵メモリが1,229,184ビット，18×18の乗算器が40個入っています．ALMというのはこのFPGAのロジック・リソースで，8入力の組み合わせ回路と加算器，2個のレジスタから構成されています．

8,632個のALMは，ロジック・エレメント数に換算すると21,580個相当になります[注1]．

ArriaGXのトランシーバは，Stratix II GXのものをベースに構築されており，600Mbps～3.125Gbpsの速度で通信ができます．このトランシーバは，PCI Express，ギガビットEthernet，XAUI，SDI，Serial RapidIOなど各種プロトコルの物理層へ適合するようにコンフィグレーションできます．

EP1AGX20Cには，このようなトランシーバが4チャネル入っています．この評価ボードでは，トランシーバの四つのチャネルがPCI Expressのカード・エッジに出ていて，最大で4レーンの構成ができるようになっています．

2．内蔵ギガビット・トランシーバ

EP1AGX20Cに内蔵されているトランシーバの特徴を，

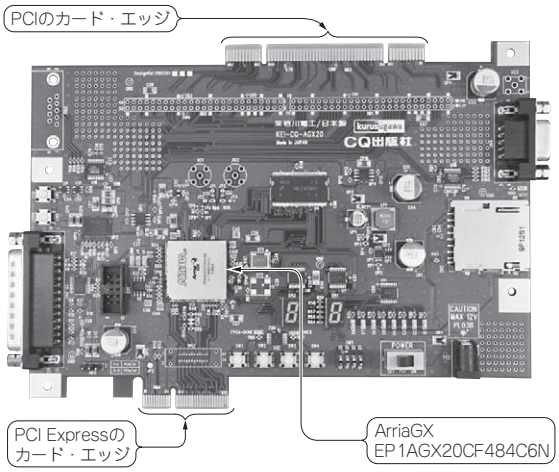


写真1　ArriaGX評価キット（CQ出版社）

注1：ロジック・エレメントで言われても直感的にわかりにくいと思うので怒られそうだが，米国Xilinx社のSpartanシリーズと比べるなら，XC3S1400Aのちょっと上くらいか，XC3S4000の下くらい．Spartan-6と比べるならLX25よりも若干大きくLX45よりも小さいくらいだろうか．あえて言うなら240万ゲートくらいだと筆者は勝手に思っている．

Chapter 13

デバイス内蔵高速トランシーバ機能を使って
1チップでPCI Expressデバイスを実現する

Spartan-6LXT内蔵トランシーバの使い方とPCI Expressコアの移植

最近は，低価格帯のFPGAにも，ギガビット・トランシーバが内蔵されるようになりました．米国Xilinx社製のSpartan-6にも，Spartan-6LXTからはトランシーバとと PCI Expressハードウェア・コアが内蔵されています．

そこで本章では，Spartan-6LXTに内蔵されたギガビット・トランシーバの使い方について説明します．

1. Spartan-6と内蔵ギガビット・トランシーバ

Spartan-6はXilinx社の低価格FPGAで，Spartan-6LXとSpartan-6LXTの二つのシリーズがあります．LXはロジックのみで構成されていますが，LXTはギガビット・トランシーバとIntegrated EndPoint Blockを内蔵しています．EndPoint Blockというのは，いわゆるハードウェア・マクロで，ユーザのロジック・リソースを使わずにPCI Express Base Specification 1.1に準拠したエンドポイントを作れます．Spartan-6のラインナップを**表1**に示します．

● ギガビット・トランシーバの概要

Spartan-6のトランシーバの概要を**表2**に示します．最大3.125Gbpsの速度で，SATA，Aurora，ギガビットEthernet，PCI Express，OBASI，CPRI，EPON，GPON，DisplayPort，XAUIなどのプロトコルに対応しています．ギガビット・トランシーバはさまざまなプロトコルをサポートしていますが，PCI Express用に設定した場合は，FPGA側のインターフェースへはPIPEに準拠した信号が出てきます．

内蔵ギガビット・トランシーバはGTPと呼ばれ，二つのチャネルがセットになったGTP_DUALという単位で構成され，プリミティブの名前はGTPA1_DUALです[注1]．

表1　Spartan-6のラインナップ

- FPGAの規模が大きくてもパッケージが小さいとGTPトランシーバのすべてが使えるわけではない
- エンドポイントはGen1で，×1レーン構成．PCI Express Base Specification 1.1に準拠

デバイス	ロジック・セル	DSP48A1スライス	ブロックRAMブロック	最大GTP数	PCIeエンドポイント・ブロック数
XC6SLX4	3840	8	12	0	0
XC6SLX9	9152	16	32	0	0
XC6SLX16	14579	32	32	0	0
XC6SLX25	24051	38	52	0	0
XC6SLX45	43661	58	116	0	0
XC6SLX75	74637	132	172	0	0
XC6SLX100	101261	180	268	0	0
XC6SLX150	147443	180	268	0	0
XC6SLX25T	24051	38	52	2	1
XC6SLX45T	43661	58	116	4	1
XC6SLX75T	74637	132	172	8	1
XC6SLX100T	101261	180	268	8	1
XC6SLX150T	147443	180	268	8	1

注1：「GTP」が何の略語かは不明．Virtex-II Proのころは Rocket IOと呼ばれていたが，Virtex-5では Rocket IO GTPになり，Spartan-6ではただのGTPと呼ぶようになった．

表2　Spartan-6 GTPの特徴

速度	614Mbps～810Mbps，1.22Gbps～1.62Gbps，2.45Gbps～3.125Gbpsの三つの範囲
対応プロトコル	SATA，Aurora，ギガビットEthernet，PCI Express，OBASI，CPRI，EPON，GPON，DisplayPort，XAUIなど
主要PCSコンポーネント	8B/10Bエンコーダ，カンマ・アラインメント，チャネル・ボンディング，クロック・コレクションなど
OOB（規格外）信号のサポート	PCI Express用のビーコン，SATA用のCOM信号など
そのほかの特徴	レシーバ・アイ・スキャン，ダイナミック・リコンフィギャラブル，ループバック 疑似乱数パターンによるシグナル・インテグリティの検査，受信/送信バッファのバイパスなど

Chapter 14

デバイス内蔵 PCI Express 対応ハード・マクロを
使いこなして PCI Express デバイスを実現する

Spartan-6LXT 内蔵エンドポイント・ブロックを使った PCI Express の設計

1. 内蔵エンドポイント・ブロックの概要

Spartan-6 LXTに内蔵されているエンドポイント・ブロックは，以下のような特徴を持ちます．

- ハードウェア・マクロで実装されていて，無償で使える
- PCI Express Base Specification v1.1 に準拠
- ×1レーン
- リンク速度は2.5Gビット/秒
- ユーザ・インターフェースは62.5MHz/32ビット幅

すなわち，FPGA内にハードウェア・マクロとして構成されたエンドポイント･ブロックとギガビット･トランシーバを持つことで，貴重なユーザ・ロジックを使用せず，追加のライセンスや費用なしに，FPGA単体でPCI Expressに対応したデバイスを作れるようになるというわけです．

内蔵エンドポイント・ブロックは，以下のことをハードウェア的に処理してくれます．

- 物理層およびデータ・リンク層レベルの処理
- コンフィグレーション空間の管理とコンフィグレーション・レジスタの実装
- BAR0 ～ BAR5（ベース・アドレス・レジスタ）のアドレス空間にヒットしたかどうかの判断
- 割り込みの発生
- 電源管理（ASPM）の処理

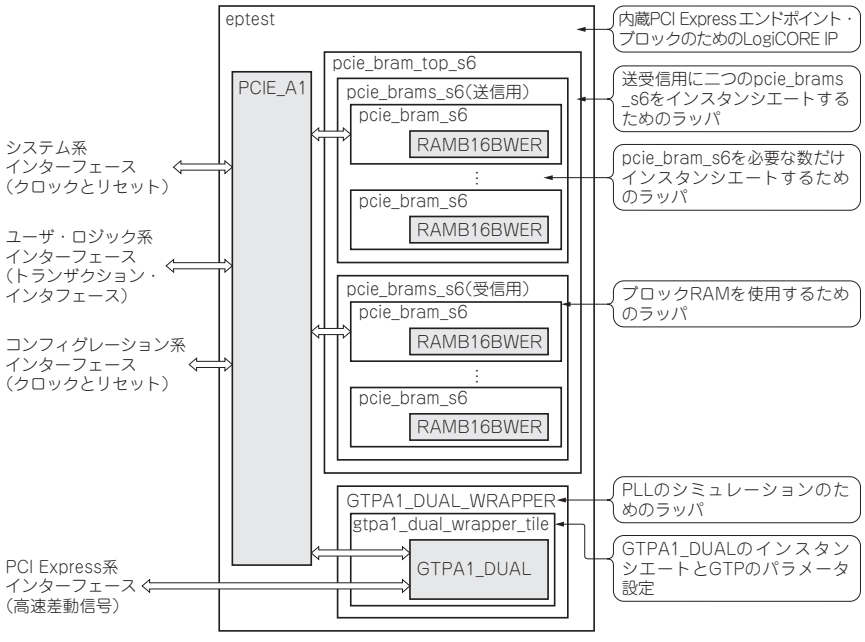


図1
内蔵エンドポイント・ブロックと周辺回路の構成
コアの本体であるPCIE_A1のほかに，GTPA_DUALとRAMB16BWER（ブロックRAM）から構成されている．プリミティブを何重にもラッパしているが，各階層は複雑なロジックを実装しているわけではない．基本的に，VHDLの条件コンパイルを行ったり，不要なポートを削除したり，固定値を与えているにすぎない．つまり，すべての機能はユーザ・ロジックを使わずに実現されている．

Chapter 15

物理的寸法から信号線の引き回し，
基板のインピーダンス評価まで

PCI Expressアドイン・カードの設計

1. アドイン・カードの物理的寸法

● アドイン・カードの最大サイズ

アドイン・カードの物理的寸法は，規格書『PCI Express Card Electromechanical Specification Revision 2.0』（以後，CEMと略す）に記載されています．

この規格では，アドイン・カードの高さについて「標準（Standard）」と「ロー・プロファイル（Low Profile）」という二つの基準があり，それぞれの最大の高さが示されています．また，アドイン・カードの長さにも「フル・レングス（Full Length）」と「ハーフ・レングス（Half Length）」という二つの基準があり，それぞれの最大長が示されています（**表1**，**図1**）．

例えば，×1のアドイン・カードを作る場合，長さについてはハーフまたはフルにするかで最大167mmか312mmという選択肢があります．しかし，実際には312mmの長さのアドイン・カードを納められる筐体は少ないので，規格書では241.30mmよりも短くすることが推奨されています．また，×1のアドイン・カードであっても，消費電流

表1　アドイン・カードの最大サイズ（単位はmm）

		高さ（最大）	長さ（最大）
×1 ×4 ×8	標準高さ ハーフ・レングス	111.15	167.65
×1 ×4 ×8 ×16	標準高さ フル・レングス	111.15	312.00
	ロー・プロファイル	68.90	167.65

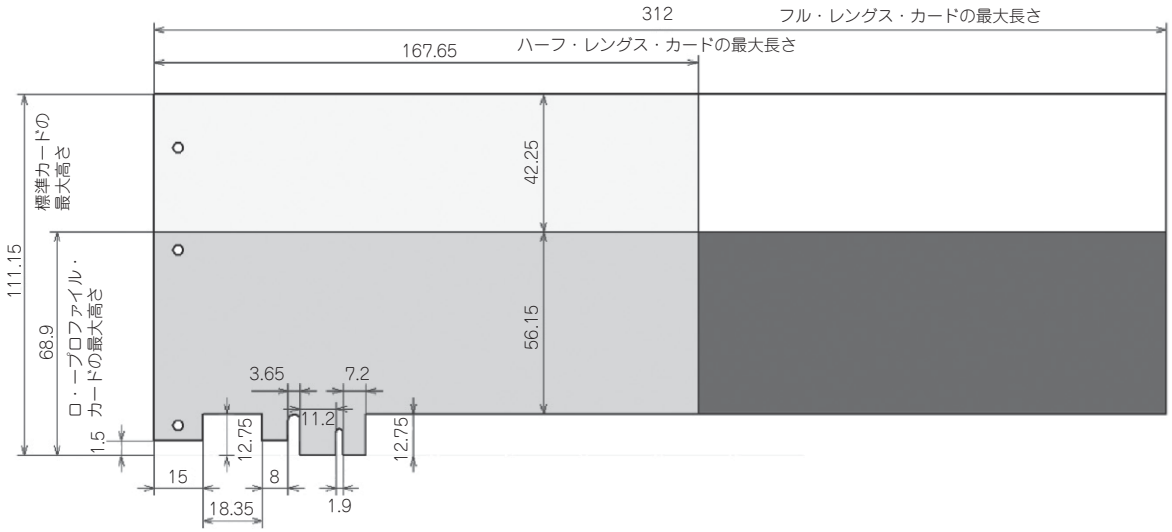


図1　PCI Expressアドイン・カードの最大サイズ（単位はmm）

Chapter 16

送出レベルやディエンファシスを変えると
信号波形の何が変わるのか

PCI Expressアドイン・カードの信号波形を観測

最後の本章では，設計したPCI Expressアドイン・カードを実際に動かし，その送受信波形を測定してみます．

1．測定環境と差動信号について

● 測定環境

北八王子にあるアジレント・テクノロジー（株）本社に伺って，Infiniium DSA91304Aという高速オシロスコープ使わせていただきました．そのときの測定機材のようすを**写真1**に示します．

設計したターゲット基板をPCI Expressコンプライアンス・ベースボードに挿して，ベースボード上のTx_+とTx_-と書かれたコネクタから送信信号を取り出し，2本の同軸ケーブルでオシロスコープに接続します［**写真2**（**a**）］．

正しい測定を行うためには，このように2本のシングル

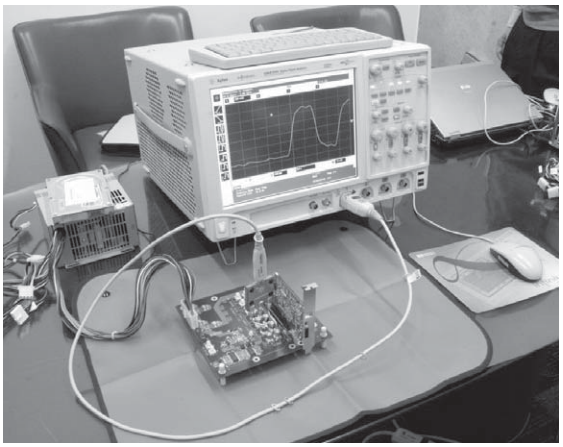
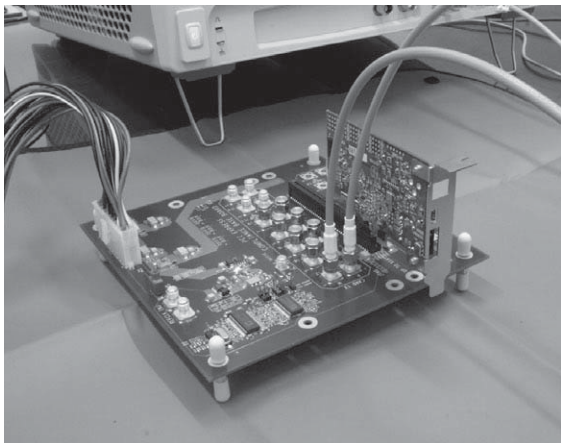
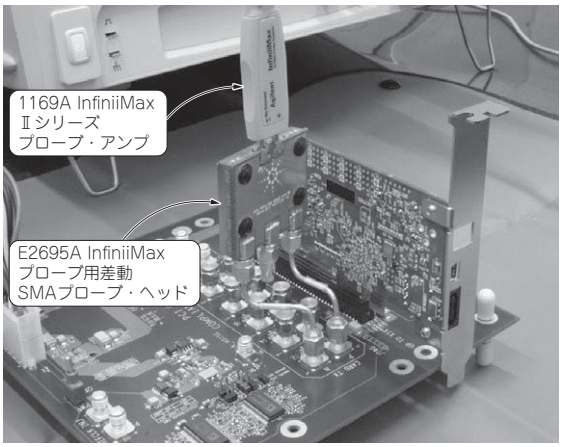


写真1　Infiniium DSA91304Aデジタル・シグナル・アナライザを用いた測定の様子
コンプライアンス・ベースボード上にターゲット・ボードが装着されている．



（**a**）2本の同軸ケーブルでTx_+とTx_-を測定



（**b**）差動プローブも装着可能

写真2　測定の様子