見本

第2章

PCIバスの基本から各信号の意味，タイミングまで

PCIバスの概要

滝　誠一

2.1 PCIバスの特徴

ここでは，PCIバスの特徴を簡単に説明し，PCIがどのようなバスであるのか，把握できるようにしてみたいと思います．

● PCIは完全同期型バス

PCIバスは，高速かつ安定したバスの動作を保証するために，ほとんどすべての信号線の動作タイミングをバス上のCLK信号を基準として統一的に規定しています．例外的にCLK信号と非同期に動作するのはリセット信号(RST#)とインタラプト信号(INTA#，INTB#，INTC#，INTD#)だけです．たとえばISAバスでは，場合場合ごとのタイミング・チャートとそれに伴うタイミング・パラメータだけで1冊の本ができてしまうほどの複雑な仕様となっています．しかし，PCIバスでは，三つのタイミング・チャートと16個のタイミング・パラメータだけですべての信号線の動作が統一的に規定されています．

なお信号名の最後に#が付く信号は，負論理信号を示します．

● クロックは可変

クロック信号CLKは最高周波数が33MHzまたは66MHzと規定されているだけです．CLK信号のための四つのタイミング・パラメータの条件が合っているかぎり，33MHzまたは66MHzより低いどのような周波数で動作させてもかまいません．動作の途中で周波数を変更してもかまいませんし，極端な場合は，クロックを止めてしまっても問題はありません．

システムの都合で33MHzまたは66MHzより低い周波数のバスを設計することができます．また，ハードウェアを動かす必要がないときには，クロック周波数を落としたり，クロックを止めることによって消費電力を削減するパワー・マネジメント機能としても使用できます．

● PCIはCMOS素子が前提

PCI上の信号線の電圧の定義には5V系と3.3V系の2種類あります．5V系では，TTLと互換性のある定義となっていますが，3.3V系では当然CMOSの特性を前提としています．

いずれの電圧定義においても，ドライバとレシーバの素子はCMOSであることを想定しています．もっとも注意を要するのは，ドライバのACスイッチング特性です．ドライバ単体のドライブ能力では必要な電圧変化を起こせないことを仮定し，信号線上の反射波の影響を加えても必要な電圧変化が得られるように，タイミング仕様などもできています．ドライバのDCドライブ能力も考慮に入れて，プルアップ抵抗による負荷などが重くなりすぎないようにする注意が必要です．

● ワイヤードORはなるべく排除

従来のバスでは，オープン・ドレイン(またはオープン・コレクタ)ドライバを使用したワイヤードORの信号線が多用されていました．しかし，この方式では，信号線をアクティブ(アサート＝“L”)の状態からインアクティブ(ディアサート＝“H”)の状態に戻すために，プルアップ抵抗に頼ることになり，状態変化に時間がかかるため，高速のバスには適しません．そこで，PCIバスでは，ほとんどの信号線について「サステインド・トライステート」(Sustained Tri-State，以下，s/t/sと略す)という方式が用いられています．

図1にs/t/sの動作を示します．

1) クロック①の立ち上がりエッジでは，ドライバAが信号線を“L”にドライブしています
2) クロック②の立ち上がりエッジで，ドライバAは信号線を“H”にドライブします

3) クロック③の立ち上がりエッジで，ドライバAはトライステート(ハイ・インピーダンス)に切り替えます
4) その後，クロック④の立ち上がりエッジまでは，プルアップ抵抗によって信号線の“H”状態が保たれます
5) ドライバBは，クロック③の時点で信号線が“H”であったことを確認し，そのひとつ後のクロック④から信号線のドライブを開始します．これより前にドライブを開始すると，ドライブAとBの出力どうしが衝突する可能性があります

オープン・ドレインを使用する信号線は，SERR#，IRQA#，IRQB#，IRQC#，IRQD#の5本だけです．これらの信号線では，同時に複数のドライバが信号線をドライブする可能性があり，s/t/sは使用できません．また，高速に状態をスイッチングする必要性もありません．

● PCIは完全32ビット・バス

16ビットのISAバスでは，8ビット・デバイスの存在を許していました．同様に，32ビットのEISAバスやマイクロチャネルでは，8ビット・デバイスや16ビット・デバイスが存在することを認めていました．これらのバスでは，バス幅の異なるデバイスの間で正しくデータ転送を行えるように，マザーボードの上に複雑なデータ・バス切り替え回路が存在していました．このバス切り替え回路の存在が，バスの動作タイミングを複雑にする要因のひとつでもありました．

しかし，PCIバスでは，すべてのデバイスが少なくとも32ビットのデータ・バス幅をもち，オプションとして64ビット・バス幅のデバイスが定義されています．32ビット・デバイスと64ビット・デバイスの間のデータ・バスの調整は当事者どうしで行うことになっており，システムはこの調整のための特別な回路をもっていません．

● アドレス・バスとデータ・バスは時分割

従来のバスでは，多くの場合，アドレス・バス用とデータ・バス用に別々の信号線のセットが用意されていましたが，PCIバスでは，アドレス・バスとデータ・バスに同じ32本の信号線を使用します．まず，アドレスでデータ転送の相手を指定し，つぎにデータの転送を行うと考えれば，アドレス線とデータ線が別々にある必然性は低いといえます．

さらに，PCIの高速性が意味をもってくるのは大量のデータをまとめて転送するときであるとの考えがあります．大量のデータを連続的に送る場合には，そのアドレスは固定であるか，連続的に変化するかのいずれかしか考えられません．したがって，アドレス情報を送るのは最初の1回だけで，あとはデータの転送のみを行う(バースト転送)とすれば，データ線とは別にアドレス線を用意しておくことはむだというものでしょう．

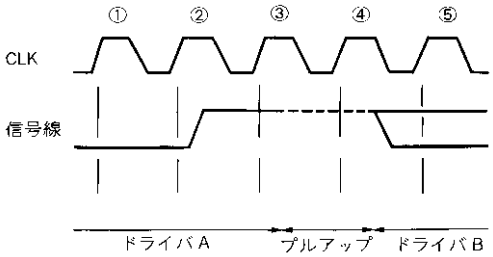


図1　サステインド・トライステートの動作

● 統一されたデバイスの初期化手段

PCIデバイスにはコンフィグレーション・レジスタと呼ばれるレジスタ群が実装されていなくてはなりません．この中には次のようなレジスタが含まれています．

1) デバイスの種類を示すIDレジスタ群
2) PCIバス上の動作のルールやシステムの特徴(たとえば，主記憶RAMに対するキャッシュ・メモリのライン・サイズ)を設定するためのレジスタ群
3) PCIの動作に関するデバイスのステータスを示すレジスタ群
4) デバイスが使用するメモリ・アドレスやI/Oアドレスなどのハードウェア・リソースを設定するためのレジスタ群
5) その他のデバイスに固有のレジスタ群

EISA，マイクロチャネル，PCMCIAでも同様な働きをするレジスタが定義されています．このレジスタ群にアクセスすることによって，システムの初期化を行うROM内蔵のプログラム，あるいは，システムのハードウェア構成を管理するコンフィグレーション・ユーティリティは，統一的な方法によって各PCIデバイスを初期化することができます．

一つの物理的なデバイス(マザーボード上のLSIまたはアドイン・ボード)の中に独立した複数のファンクションが実装されている場合(マルチファンクション・デバイス)には，各ファンクションごとに独立したコンフィグレーション・レジスタのセットを用意します．

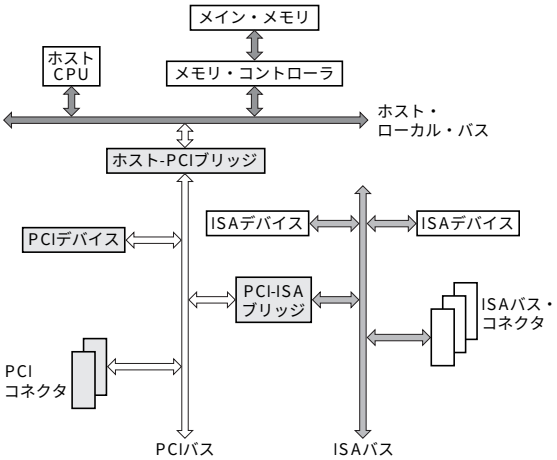


図2　PCIバスを搭載したシステムの構成例

コンフィグレーション・レジスタにアクセスするためのアドレス空間をコンフィグレーション空間と呼び，メモリ・アドレス，I/Oアドレス空間とは区別されています．また，バス上の動作としては，コンフィグレーション・レジスタに対するリード/ライトのためにメモリ・サイクル，I/Oサイクルとは別にコンフィグレーション・サイクルが定義されています．

コンフィグレーション空間の定義によれば，一つのPCIバスに接続可能なデバイスの数は最大32個，一つのデバイスの中に実装できるファンクションの数は最大8個，そして，一つのファンクションに割り当てられるコンフィグレーション空間のサイズは256バイト(64ダブル・ワード)となります．

● ブリッジによるバスの拡張

理論上，デバイスは最大で32個とのことですが，実際には電気的な負荷や安定性の観点から，10個程度が限界となります．それでは，もっと多くのデバイスを接続したい場合にはどうするのでしょうか．その場合には，システム内に複数のPCIバスを用意し，バス間をPCI-PCIブリッジ回路で接続します．

PCIアーキテクチャでは，ブリッジ回路によって複数のバスを接続することを最初から考慮しています．具体的には次のようなブリッジ回路が考えられます．

1) PCIバスをCPUのローカル・バスに接続するホスト-PCIブリッジ
2) PCIバスを拡張するためのPCI-PCIブリッジ
3) PCIバスとそのほかの拡張バス(ISA，EISA，マイクロチャネル，PCMCIA，CardBusなど)を接続するためのブリッジ

図2にPCIバスを搭載したシステムの構成例を示します．一般的なPC/AT互換機はホスト-PCIブリッジおよびISAバス・ブリッジが搭載されています．

● 複数バスの同時動作

ブリッジ回路を経由してのデータ転送は，ときとしてPCIバスの高速性をそこなうおそれがあります．しかし，ブリッジによって隔てられた二つのバスが，同時に別々のデータ転送を実行できるとすれば，ブリッジの欠点を捕えるばかりでなく，より高いシステム・スループットを実現する可能性もあります．

さまざまなケースが考えられるでしょうが，既に実現されている簡単な例として，ポスティッド・ライト(Posted Write)と呼ばれる手法があります．

例えば，CPUが大量のデータを主記憶RAMから読み出し，それをVRAMに書き込む場合を考えてみましょう．VRAMにデータを書き終わるまでCPUのライト・サイクルは終了しないものとすると，この間，CPUはほかの処理ができなくなってしまいます．このようなときに，ブリッジの中にバッファを用意しておき，このバッファにデータを書き込んだ時点でCPUのライト・サイクルを終了させてしまうのがポスティッド・ライトです．ブリッジがバッファの中のデータをVRAMに向けて転送している間，CPUもホスト・ローカル・バスも自由ですから，この間にCPUは新しいデータをメモリからリードすることが可能になります．また，ブリッジは，後続のデータがバッファにたまっていることを検知できれば，前述のバースト転送が簡単に実行できるようになります．

● イニシエータ/ターゲット/エージェント

一般にコンピュータのバス上でデータ転送を行う場合には，データの送り手と受け手の間に，「マスタ」と「スレーブ」の関係があります．「マスタ」とはアドレス線をドライブしてデータ転送の相手を指定し，メモリ・リードやI/Oライトなどのアクセス要求を送るデバイスです．反対に「スレーブ」はアドレスやアクセス要求を受け取り，それをデコードしてアクセス要求に応答するデバイスです．

PCIバスにおいては，「マスタ」と「スレーブ」に該当する言葉として，「イニシエータ」および「ターゲット」という用語が多く使われます．「スレーブ」という言葉はほとんど使われず，つねに「ターゲット」という言葉が使用されます．「マスタ」と「イニシエータ」はどちらも使いますが，一応の使い分けがあるようです．ある特定のデータ転送動作について論じると

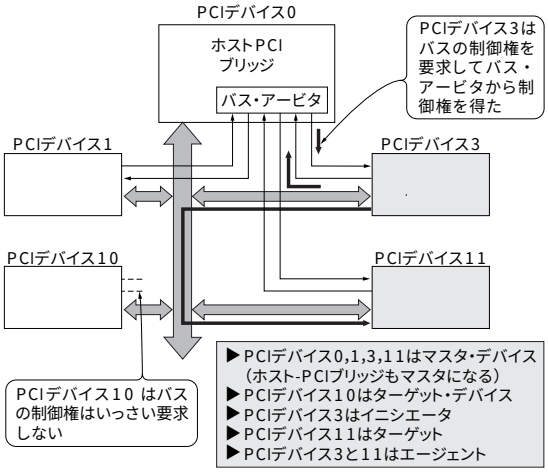


図3　デバイスとエージェントの関係

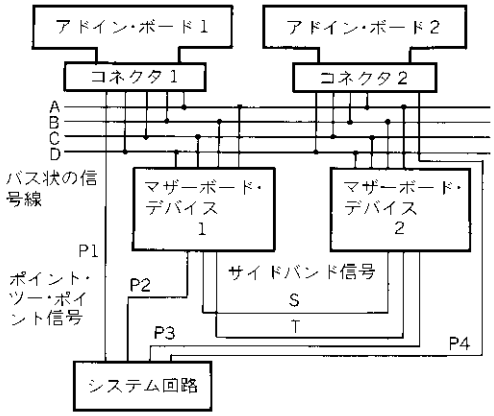


図4　ポイント・ツー・ポイント信号とサイドバンド信号

きには，その転送を制御しているデバイスのことを「イニシエータ」と呼び，「イニシエータ」として動作しうるデバイスのことを「マスタ・デバイス」と呼ぶのが普通です．

SCSIでも「マスタ」，「イニシエータ」という言葉を使いますが，少し意味合いは異なっています．

また，「マスタ」あるいは「イニシエータ」としてPCIバス上のデータ転送に参加するデバイスのことを総称して，そのPCIバスの「エージェント」と呼びます(**図3**)．

● DMAの代わりにバス・マスタ

バスを経由したブロック・データ転送は，I/Oデバイスとメイン・メモリとの間で行われることが多いでしょう．この場合，データ転送の方法には大きく3通りの方法があります．

1) CPUがプログラムによってデータ転送を実行する
2) DMAによるデータ転送
3) I/Oデバイスがバス・マスタとしてバスを制御し，直接メイン・メモリにアクセスする

1)の方法は状況によっては，簡単でかつ効果的な方法です．PCIバスでは，2)のDMA転送は定義されていません．CPUに負担をかけずにデータ転送を行うためには，3)のバス・マスタによる転送が必要となります．

● ポイント・ツー・ポイント信号とサイドバンド信号

バスというと，信号線の結線は一般に**図4**の信号線AからDのようなものを想像するでしょう．もちろんPCIバスにおいてもほとんどの信号線は，これと同様のいわゆるバス状結線となっています．しかし，いくつかの例外があります．

その一つが，**図4**の中にあるP1からP4のような，ポイント・ツー・ポイント信号線と呼ばれるものです．各PCIデバイス(またはコネクタ)の同じ名前のピンは，互いに接続されるのではなく，それぞれ別々にシステム回路に接続されています．例としては，INTA#からINTD#のインタラプト要求線やREQ#，GNT#線があります．

もう一つ，サイドバンド信号というものが定義されています．これは，マザーボード上のデバイスは，必要があれば，PCIの仕様書に定義されている以外の特殊な信号線を用意して，独自のバス制御を行ってもよいということです．典型的な例としては，PCIバスの使用権を調停するためのアービトレーション・プロトコルがあります．アドイン・バス・マスタ・ボードは仕様書で規定しているREQ#，GNT#を用いたプロトコルでしかアービトレーションに参加できません．しかし，バス・ブリッジなどのマザーボード上のデバイスはサイドバンド信号を使った裏取引によって特権的にアービトレーションに参加することが認められているのです．

2.2 アドイン・ボードとコネクタ

● アドイン・ボードと形状とスロット

写真1にPCIアドイン・ボード(拡張ボード)の外観を，**図5**にアドイン・ボードの寸法を示します．PCIバスには，データ・バス幅が32ビットのものと，64ビットのものが存在します．アドイン・ボードのカードエッジ部分は，下位の32ビット分はどちらも共通

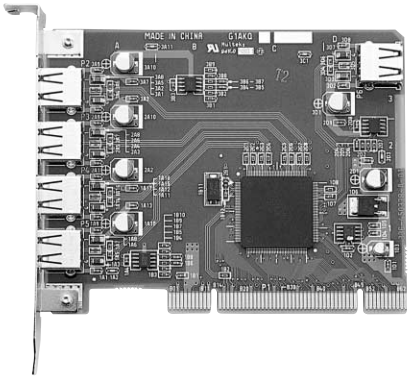


写真1　PCIアドイン・ボードの例(32ビット)

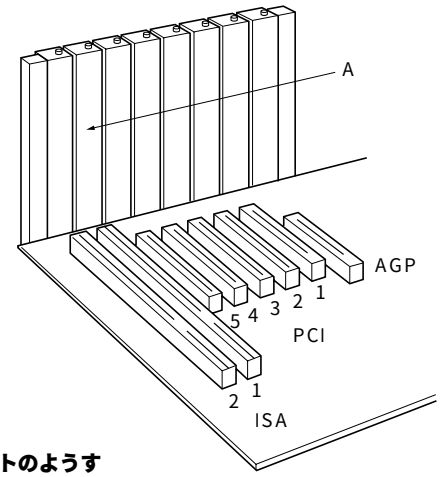


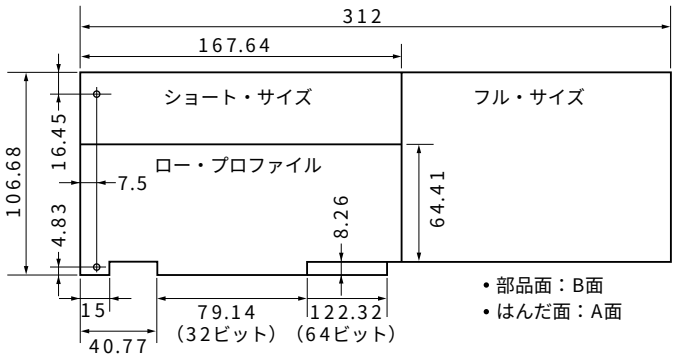
図6
共用スロットのようす

のピン配置になっています．

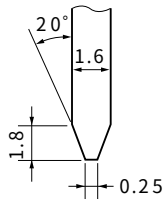
またボードの形状としては，フル・サイズとショート・サイズ，そして高さの低いロー・プロファイルの寸法規定があります．

PCIアドイン・ボードとISAバスのアドイン・ボードとの違いの一つは，エッジ・コネクタにあります．ISAボードのコンタクトは100ミル(2.54mm)ピッチです．しかし，PCIボードのほうは50ミル(1.27mm)ピッチのコンタクトを採用しています．

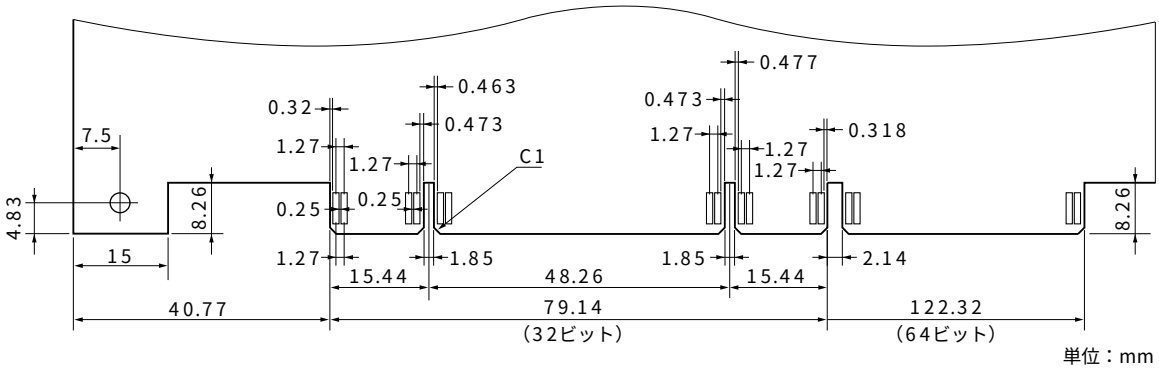
もう一つの違いは，部品面とはんだ面の関係がISAボードの場合と逆になっていることです．これは，共用スロット(Shared Slot)と呼ばれるものを実現するためです．**図6**はシステムの共用スロットの図です．共用スロット用の窓がAで，このスロットにISAボー



(**a**) 基板寸法



(**c**) カードエッジ・コネクタ断面



(**b**) カードエッジ・コネクタ部分(ユニバーサル・キー)

図5　PCIアドイン・ボードの寸法

ドを設置する場合には，1のISAコネクタに差し込みます．また，同じスロットにPCIボードを設置する場合には，5のPCIコネクタに差し込みます．

● コネクタのピン配置

すでに説明したように，PCIアドイン・ボードのカードエッジ部分は，下位の32ビット分はどちらも共通のピン配置になっていて，64ビットに拡張された部分はスペーサ分の隙間をあけてコネクタが並ぶ形になります．

もう一つPCIバスには，信号線の電圧として5V TTL互換の5V系システムと，3.3V CMOSの3.3V系システムの2種類あります．電気的にこの二つは区別する必要があります．ちなみに最新のPCI Rev.3.0では5V系システムの規定は廃止されましたが，市場では依然として5V系システムも数多く存在しているため，ここでは5V系システムについても解説します．

表1にPCIバスのピン配置を示します．5V系システムと3.3V系システムの大きな違いは，(I/O)と書かれたピンに，5V系システムでは5V，3.3V系システムでは3.3Vの電源が供給されている点です．また，5V系システムでは50番と51番ピンに，3.3V系システムでは12番，13番ピンにキー・ウェイがあります．キー・ウェイとはスロットにボードを差し込めないようにする出っ張り部分のことで，5V系システムを想定して設計されたアドイン・ボードを，3.3V系システムのマザーボードに差し込めないようにするためのものです．

ただしアドイン・ボードの設計によっては，5Vトレラントな PCIデバイスを採用するなどして，5V系システムでも3.3V系システムでも，どちらのマザーボード上でも動作するカードを実現することもできます．このようなアドイン・カードの場合，3.3Vキーと5Vキーの両方を空けておきます(**図7**)．

図5に示したアドイン・ボードのカードエッジ・コネクタ部分は，3.3V/5V系両対応のため，3.3Vキーと5Vキーの両方が空いているカードということになります．しかも64ビット対応のため，64ビット・スペーサの分の隙間も空いています．

以上をまとめると，次のようになります．

1) コネクタは32ビットの基本部分と64ビットの拡張部分に分けられる
2) コネクタには5V系システム用と3.3V系システム用の2種類がある
3) アドイン・ボードには5V系システム用，3.3V系システム用とユニバーサル用の3種類がある

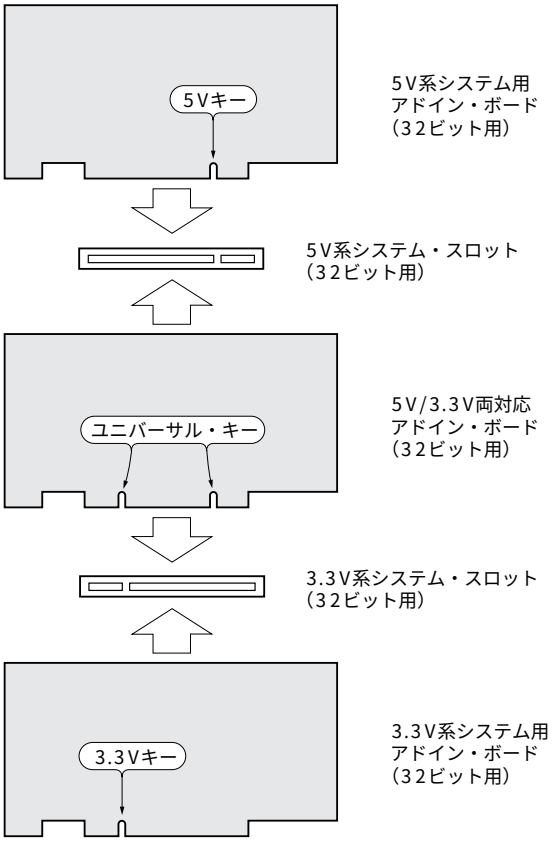


図7　コネクタ・キーによる5Vモード/3.3Vボード共用の仕組み

なお，クロック周波数66MHzは，3.3V系システムの場合にのみ規定されています．よってM66ENという信号ピンも，3.3V系システムにしか定義されていません．

2.3 PCIバスの信号線

● 信号線の分類

PCIの信号線は，その機能・目的，必須かオプションか，信号の方向やドライバの種類によって分類することができます．これらの分類を**表2**にまとめておきます．

● システム信号

▶CLK

RST#とインタラプト線を除くすべてのPCI信号は，CLK信号線に同期して動作します．

▶RST#

電源投入時，およびシステムのリセットのときにアサート(assert；アクティブにする)されます．RST#が