DMA機能付きPCIマスタ／ターゲットMegaCoreファンクション

1998年 7 月 ver.3 Data Sheet

機能

- pci_a は32ビット・ペリフェラル・コンポーネント・インタコネクト(PCI)マスタ／ターゲット・インタフェース機能を実現したMegaCore™ファンクション
- FLEX® 10Kアーキテクチャに最適化
- 下記による広範囲なハードウェア・テストを実施済み
  - HP E2925A PCIバス・エクセサイザ／アナライザ
  - FLEX 10K PC試作ボード
  - インテルの430および440チップセット、DEC社のPCI-to-PCIブリッジなどの標準的なPCIチップセットに対する検証を実施済み
- デザイン・サイクルを大幅に短縮
- FLEX 10K PC試作用ボードが付属
- ユーザ・シミュレーション用のテスト・ベクタを添付
- OpenCore™評価機能により、ライセンスの購入前にMAX+PLUS® IIを使用したデザインへのインスタンス化およびシミュレーションが可能
- FLEXデバイスの約1,000個のロジック・エレメント（LE）、EPF10K50デバイスが提供するリソースの約35%で構成可能
- PCIマスタ機能
  - メモリ・リード／ライト
  - バス・パーキング
  - アドレス・カウンタ・レジスタ、バイト・カウンタ・レジスタ、コントロール&ステータス・レジスタ、インタラプト・ステータス・レジスタを含む完全に集積化されたDMAエンジン
  - DMAターミナル・カウント、マスタ・アボート、ターゲット・アボート、ローカル側インタラプト機能を含むコンフィギュレーション可能なインタラプト・ソース
  - 64バイト（16ダブル・ワード、またはDWORD）のRAMバッファをFLEX 10Kのエンベデッド・アレイ・ブロック(EAB)で実現
  - ゼロ・ウェイト・ステートのPCIリードおよびライト・バースト動作
- PCIターゲット機能
  - タイプ-ゼロのコンフィギュレーション・スペース
  - パリティ・エラー検出機能
  - メモリ・リード／ライト、コンフィギュレーション・リード／ライト動作
  - ターゲット・リトライ、ディスコネクト
  - １Mバイトから2Gバイトのパラメータ化されたターゲット・メモリ・スペース
- コンフィギュレーション・レジスタ：
  - パラメータ化：デバイスID、ベンダID、クラス・コード、リビジョンID、ベース・アドレス・ゼロ、サブシステムID、サブシステム・ベンダID、
  - 非パラメータ化：コマンド、ステータス、ヘッダ・タイプ、レイテンシ・タイマ、インタラプト・ピン、インタラプト・ライン

イントロダクション

このデータシートでは、pci_a MegaCoreファンクションの動作を以下に示す各項目に沿って解説します。

更新記録
Version 2.0

pci_aファンクションのバージョン2.0のデータシートには、下記の内容が追加、補充されています。

- 追加サポート・デバイス
- ローカル・サイド・イニシエイティッドDMA
- パラメータ化されたベース・アドレス・レジスタ（BAR）
- 外部ターゲット・ライト転送中のバイト選択
- 外部ターゲット・ライト・トランズアクション実行時のl_holdnの使用
- DMAバイト・カウンタ・レジスタの拡張

追加サポート・デバイス

pci_aファンクションは下記のFLEX 10Kデバイスを含む多様なパッケージの幅広い製品をサポートしています。

- EPF10K30RC240
- EPF10K30RC208
- EPF10K30AQC240
- EPF10K30AQC208
- EPF10K40RC240
- EPF10K40RC208
- EPF10K50RC240
- EPF10K100ARC240
- EPF10K30BC356
- EPF10K50BC356
- EPF10K100ABC356

新製品のリリースにより、さらにサポートされるデバイスが追加される予定となっています。最新のデバイス・サポートの状況については、アルテラのウェッブ・サイト、http://www.altera.comで確認してください。

ローカル・サイド・イニシエイティッドDMA

このpci_aファンクションを使用してDMAバースト転送を実行するためには、DMAレジスタに適切な値を書き込み、転送のセットアップを行う必要があります。pci_aの以前のバージョンでは、ホストまたはPCIマスタがDMAレジスタに値を書き込む必要がありましたが、このpci_aのバージョン2.0では、ローカル側のデバイスからDMAリードおよびDMAライトのトランズアクションがダイレクトに行えるようになっています。詳細については、50ページの「ローカル側からのDMA転送のイニシャライズ」をご覧ください。

パラメータ化されたBAR

BAR0はパラメータ化されており、メモリの割り当てを最適な効率で実現できるようになっています。pci_aのバージョン1.3では、BAR0のアドレス・スペースが２つの512Kバイトのメモリ・スペースに分割された 1Mバイトの連続したアドレス・スペースとなっていましたが、このpci_aのバージョン2.0以降では、ユーザがBAR0のアドレス・スペースを1Mバイトから２Gバイトの範囲の連続した領域に設定できるようになっています。詳細は、24ページの「ベース・アドレス・レジスタ-0 (オフセット=10 Hex)」をご覧ください。

ターゲット・ライト転送中のバイト選択

ターゲット転送の実行時に、PCIのcben[3..0]のバス信号はバイト・イネーブル信号となり、どのバイトが有効なデータを転送しているかを示します。この場合、cben[3..0]バスのビット-３はバイト-３に適用され、ビット-０はバイト-０に適用されます。pci_aのバージョン2.0では、さらにローカル側のl_ben[3...0]のバス信号がcben[3..0]をバッファし、外部ターゲット・ライト・トランズアクションの実行時に、どのバイトがローカル側に対して有効なデータを転送しているかを示すようになっています。

外部ターゲット・ライト・トランズアクション実行時のl_holdnの使用

pci_aのバージョン1.3では、ローカル側のアプリケーションが２クロック以内にデータを供給するか、受け入れることが要求されていました。バージョン2.0では、低速のアプリケーションがl_holdnをアサートして、データの転送に必要な期間を延長できるようになっています。

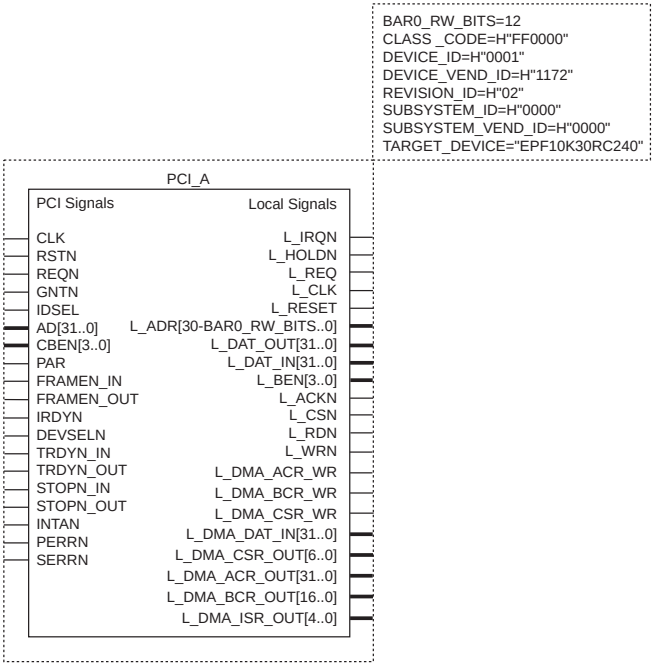
DMAバイト・カウンタ・レジスタの拡張

DMAバイト・カウンタ・レジスタは、16ビットから17ビットに拡張されています。この結果、マスタのDMAエンジンは、各DMAトランズアクションに対して128Kバイトまでのメモリ転送を行うことができます。

概要

pci_a MegaCoreファンクションは、32ビットのPCI周辺デバイスの集積化にタイムリなソリューションを提供します。また、要求されるPCI仕様に準拠していることがすでに検証済みとなっています。 pci_aはFLEX 10Kのデバイス・ファミリに最適化されているため、設計者はデザインに対する負担を軽減することができ、PCI周辺のカスタム・ロジックのデザインに注力することが可能です （注文コードは、PLSM-PCI/A）。図１はpci_aのシンボルを示したものです。

図1　pci_aのシンボル



PCI仕様準拠の概要

pci_aのファンクションは、PCI SIG（Special Interest Group）による***PCIローカル・バス仕様書 Revision 2.1***と***仕様準拠チェックリスト Revision 2.1***に準拠しています。また、このpci_aファンクションには幅広いハードウェアの検証テストが実施されています。PCIのテスト手順に従った検証が実施されており、正しい機能動作とPCI仕様への準拠が保証されています。これらのテストには、下記のハードウェアとソフトウェアが使用されています。

- アルテラのFLEX 10K PC試作用ボード
- BlueWater System社製、WinDK（Windows NT用）ソフトウェア・ドライバ
- HP社製、E2925A PCIバス・エクセサイザ／アナライザ

テストはPCIバスにデータをフルにロードした状態で実行されています。HP社のE2925A PCIバス・エクセサイザ／アナライザ、アルテラのPCI試作ボードに加えて、ホスト・ブリッジ、イーサネット・ネットワーク・アダプタ、ビデオ・カードなどのようなPCIバス・エージェントにより、このpci_aファンクションが大量のデータを取り扱うアプリケーションに使用された場合のテストも実施されています。これらの幅広いテストの実施によって、pci_aファンクションがもっとも厳しい条件でも完全に動作することが保証されています。

これらのテストにおいて、pci_aファンクションは、アルテラのPCI試作用ボードとの間でマスタおよびターゲットとして動作します。また、一般的なバーストおよびシングル・サイクルのトランズアクションにおいて、pci_aは多様なインタラプト・サイクルを実行して、複数の異常に対応したターミネーション動作を行います。データの正確性のチェックに加えて、HP社のE2925A PCIバス・エクセサイザ／アナライザを使用して、PCIバスにプロトコルの違反が発生していないことが確認されています。このようなテスト・プログラムを繰り返し実行することによって、EPF10K30デバイスで実現されたpci_aとホスト・メモリ間で65億バイト以上のデータが転送されることになります。これらのテストは複数のPCIプラットフォームで繰り返し実行され、多様なチップセットとの互換性が確保されています。表１は、このドキュメントの制作中に、pci_aに対するテストが実施されたハードウェア・プラットフォームを示したものです。

表１　pci_aに対する検証が実施されたハードウェア・プラットフォーム

プラットフォーム	チップセット	CPU スピード (MHz)	PCIバス・スピード (MHz)
Dell OptiPlex XM 5166	Intel 430 NX	166	33
Dell OptiPlex GX Pro	Intel 440FX PCISet (Bus 0)	200	33
	DEC21052-AB PCI-PCI bridge (Bus 1)	200	33
Dell OptiPlex GXL 5166	Intel 430 FX PCISet	166	33
U-tron (Pentium/MMX)	Intel 430 VX PCISet	166	33

pci_aファンクションは、上記のハードウェアによるテストの他に、表２に示す適用可能なPCIテスト・シナリオによっても検証されています。詳細については67ページの「PCI SIGテスト・ベンチ一覧」をご覧ください。

表２　pci_aに対して実施されたPCIバス・テスト　(1/2)

PCIテスト・シナリオ#	テスト・シナリオの概要	シミュレーション・ファイル名 *注(1)*
1.1	PCIバス・デバイス・スピード	pcicc101
1.2	PCIバス・シングル・データ・フェーズ・ターゲット・アボート・サイクル	pcicc102
1.3	PCIバス・シングル・データ・フェーズ・ターゲット・リトライ・サイクル	pcicc103
1.4	PCIバス・シングル・データ・フェーズ・ターゲット・ディスコネクト・サイクル	pcicc104
1.5	PCIバス・マルチデータ・フェーズ・ターゲット・アボート・サイクル	pcicc105
1.6	PCIバス・マルチデータ・フェーズ・ターゲット・リトライ・サイクル	pcicc106
1.7	PCIバス・マルチデータ・フェーズ・ターゲット・ディスコネクト・サイクル	pcicc107
1.8	PCIバス・マルチデータ・フェーズと trdyn サイクル	pcicc108
1.9	PCIバス・データ・パリティ・エラー・シングル・サイクル	pcicc109
1.10	PCIバス・データ・パリティ・エラー・マルチデータ・フェーズ・サイクル	pcicc110

表２　pci_aに対して実施されたPCIバス・テスト　(2/2)

PCIテスト・シナリオ#	テスト・シナリオの概要	シミュレーション・ファイル名 *注(1)*
1.11	PCIバス・マスタ・タイムアウト	pcicc111
1.13	PCIバス・マスタ・パーキング	pcicc113
1.14	PCIバス・マスタ・アービトレーション	pcicc114
2.5	ターゲットの予約コマンド（デュアル・アドレスを含む）の無視	pcicc205
2.6	コンフィギュレーション・サイクルのターゲット・レセプション	pcicc206
2.8	ターゲットにおけるアドレスとデータ・パリティ・エラーを含んだコンフィギュレーション・サイクルの受信	pcicc208
2.9	ターゲットのメモリ・サイクルの受信	pcicc209
2.10	ターゲットにおけるアドレスとデータ・パリティ・エラーを含んだメモリ・サイクルの受信	pcicc210
注(2)	DMAレジスタのプログラミングとバースト・リード転送	dma_rd
注(2)	DMAレジスタのプログラミングとバースト・ライト転送	dma_wr
注(2)	外部ターゲット・リード／ライト転送	trg_xrw

注：
(1)　ファイルの拡張子はシミュレータ・チャネル・ファイル(.scf)、ベクタ・ファイル(.vec)、VHDLファイルなど、使われるシミュレーション・ファイルの種類によって異なります。
(2)　これらのテストはPCI SIG *PCI ローカル・バス仕様書 Revision 2.1*では要求されていないため、テスト番号がありません。

PCIバス信号

pci_aファンクションで使用されているPCIバス信号を以下に示します。

- ***入力***－標準的な入力専用信号
- ***出力***－標準的な出力専用信号
- ***双方向***－トライ・ステートの入出力信号
- ***サステインド・トライ・ステート***－１度に１つのエージェント（PCIバス上で動作するデバイスやホストなど）によってドライブされる信号。サステインド・トライ・ステート・ピンをLowにドライブしているエージェントは、トライ・ステート状態にする前に１クロックの期間そのピンをHighにドライブしなければなりません。このとき、他のエージェントは、前のエージェントがリリースされた１クロック後でないと、サステインド・トライ・ステート信号をドライブすることはできません。
- ***オープン・ドレイン***－他のエージェントとワイヤードORされた信号。信号ソースとなるエージェントがオープン・ドレイン信号をアサートし、プルアップ・レジスタによりオープン・ドレイン信号がディアサートされます。プルアップ・レジスタがオープン・ドレイン信号をインアクティブにするまでには、２から３サイクルのPCIバス・クロックの期間が必要となります。

表3は、pci_aがPCIバスにインタフェースされるときのPCIバス信号を示したものです。ローカル側の信号の詳細については10ページの「ローカル側の信号」を参照してください。

表3　pci_aとPCIバスのPCIインタフェース信号 (1/2)

信号名	タイプ	極性	説　明
clk	入力	−	クロック信号。clkの入力はrstnとintan信号を除くすべてのPCIインタフェース信号の基準となる。
rstn	入力	Low	リセット信号。FLEX 10K PCIインタフェース回路を初期化する信号。 rstn入力は、PCIバスのclkエッジに対して非同期にアサートできる。この信号がアクティブになると、PCI出力信号はトライ・ステートとなり、serrnのようなオープン・ドレイン信号はフローティング状態になる。
gntn	入力	Low	グラント信号。この信号はマスタ・デバイスがPCIバスを制御していることを示す。すべてのマスタ・デバイスは、アービタと直接接続されている2本のアービトレーション・ライン（gntnとreqn）を持つ。
reqn	出力	Low	リクエスト信号。この出力は、マスタがトランズアクションを実行するためにPCIバスの制御を要求していることをアービタに対して示す。
ad[31..0]	トライ・ステート	−	アドレス／データ・バス信号。ad[31..0]はアドレスとデータが時分割されたバスとなっている。各バス・トランズアクションはアドレス・フェーズとそれに続く1つ以上のデータ・フェーズで構成される。データ・フェーズはirdynとtrdynがアサートされるときに発生する。
cben[3..0]	トライ・ステート マスタ：出力 ターゲット：入力	Low	コマンド／バイト・イネーブル信号。cben[3..0]はコマンドとバイト・イネーブルが時分割されたバスとなっている。アドレス・フェーズ期間にこのバスはコマンドを示し、データ・フェーズの期間にはバイト・イネーブルを示す。
par	トライ・ステート	−	パリティ信号。偶数パリティのトライ・ステート出力信号。ad[31..0]、cben[3..0]とpar信号の1の数が偶数になる。
framen *注(1)*	サステインド・トライ・ステート マスタ：出力 ターゲット：入力	Low	フレーム信号。動作中のバス・マスタがバス・オペレーションの開始と処理中を表示する信号。framenが最初にアサートされると、アドレスとコマンド信号がad[31..0]とcben[3..0]に与えられる。framen信号はデータ・オペレーションの期間にアサートされ、トランズアクションの終わりを示すときにディアサートされる。
irdyn	サステインド・トライ・ステート マスタ：出力 ターゲット：入力	Low	イニシエータ・レディ信号。バス・マスタからターゲットに出力される信号で、データ・トランズアクションが実行できることを示す。ライト・トランズアクションではirdynがad[31..0]に有効データがあることを示し、リード・トランズアクションではirdynはマスタがad[31..0]のデータを取り込み可能であることを示す。
devseln	サステインド・トライ・ステート マスタ：入力 ターゲット：出力	Low	デバイス・セレクト信号。ターゲットが自分自身のアドレスをデコードしたときにこのdevseln信号を出力する。

表3　pci_aとPCIバスのPCIインタフェース信号　(2/2)

信号名	タイプ	極性	説　明
trdyn *注(1)*	サステインド・トライ・ステート マスタ：入力 ターゲット：出力	Low	ターゲット・レディ信号。ターゲットがデータ・トランズアクションを実行できることを示す出力。リード動作では、trdynによって、ターゲットがad[31..0]にデータを出力していることが示され、ライト動作ではターゲットがad[31..0]のデータを取り込み可能な状態にあることが示される。
stopn *注(1)*	サステインド・トライ・ステート マスタ：入力 ターゲット：出力	Low	ストップ信号。ターゲット・デバイスがバス・マスタに対して処理中のトランズアクションの停止を要求する信号。
idsel	入力	High	デバイス・セレクトの初期化信号。コンフィギュレーション・リードやライト・オペレーションを行うときのチップ・セレクトとして使用される。
perrn	サステインド・トライ・ステート	Low	パリティ・エラー信号。データ・パリティ・エラーの発生を表示する。
serrn	オープン・ドレイン	Low	システム・エラー信号。システム・エラーとアドレス・パリティ・エラーを表示する。
intan	オープン・ドレイン	Low	インタラプト-A。ホストに対するアクティブLowのインタラプト信号で、インタラプトが必要なシングル・ファンクションのデバイスに使用される必要がある。

注：
(1)　PCIのセットアップ時間の仕様を満たすため、pci_aはframen、trdyn、stopnの信号を２つの片方向信号（入力と出力）に分けています。したがって、PCI信号のtrdynは入力のtrdyn_inとtrdyn_outに接続された形となっています。入力のtrdyn_inはFLEX 10Kの入力専用ピンに接続され、trdyn_outはFLEX 10KのI/Oピンに接続されます。

PCIバスとFLEX 10Kデバイスでは、IEEE Std. 1149.1のJTAG(Joint Test Action Group)バウンダリ・スキャン・テスト(BST)を実行することができます。JTAG BSTを実施する場合は、PCIバスのJTAGピンとFLEX 10KデバイスのJTAGピンを接続する必要があります。詳細については表４を参照してください。

表4　IEEE Std. 1149.1のJTAG信号

信号名	タイプ	極性	説　明
TCK	入力	High	テスト・クロック信号。TCKはテスト・モードおよびテスト・データの入出力に使用されるクロック。
TMS	入力	High	テスト・モード・セレクト信号。TMSはデバイス内のテスト・アクセス・ポート(TAP)コントローラを制御するために使われる信号。
TDI	入力	High	テスト・データ入力。TDIはデバイスへの命令とテスト・データをシフト・インするときに使用される。
TDO	出力	High	テスト・データ出力。TDOはデバイスへ命令とテスト・データをシフト・アウトするときに使用される。

ローカル側の信号

表５は、pci_aとローカル側の周辺デバイスをインタフェースするpci_aの信号をまとめたものです。

表５　pci_aのローカル側とのインタフェース信号　(1/3)

信号名	タイプ	極性	説　明
l_irqn	入力	Low	ローカル・サイド・インタラプト・リクエスト信号。ローカル側の周辺デバイスはPCIバス・インタラプト信号を送出したときにこのl_irqnをアサートする。ローカル側の周辺デバイスからDMA転送が要求されたとき、このl_irqn入力を使用してホスト側にサービスを要求することができる。
l_holdn	入力	Low	ローカル・ホールド信号。この信号がアサートされると、実行中のDMA転送が一時的に停止される。この信号がアクティブになっている期間は、pci_aとローカル側の周辺デバイスとの間でのデータ転送を行うことができない。ターゲット・トランズアクションの実行時にl_holdnをアサートして、外部ターゲット転送の期間を延長することができる。l_holdnがアサートされなかった場合、pci_aはl_csnがアサートされてから２番目のクロックでローカル側でのデータの送信または受信が可能と判断する。
l_req	入力	High	ローカルDMAリクエスト信号。DMAが有効なデータをロードした後で、ローカル側の周辺デバイスがl_reqをアサートし、pci_aにPCI DMAオペレーションが開始されることを示す。
l_dat_in[31..0]	入力	−	ローカル・データ・バス入力信号。pci_aがイニシエートしたDMAライト・トランズアクション（ローカル側のDMAリード・トランズアクションのこと）や、PCIバス・ターゲット・リード・トランズアクションの実行時に、ローカル側の周辺デバイスがこのl_dat_in[31..0]入力信号をアクティブにする。
l_dat_out[31..0]	出力	−	ローカル・データ・バス出力信号。pci_aがイニシエートしたDMAリード・トランズアクション（ローカル側DMAライト・トランズアクションのこと）や、PCIバス・ターゲット・ライト・トランズアクションの期間に、pci_aがこのl_dat_out[31..0]をドライブする。
l_ben[3..0]	出力	Low	ローカル・バイト・イネーブル信号。ターゲット・ライト転送時にpci_aがこのl_ben[3..0]をドライブして選択されたバイトを示す。
l_adr[30-BAR0_RW_BITS..0]	出力	−	ローカル・ターゲット・アドレス信号。l_adr[30-BAR0_RW_BITS..0]の出力は、ローカル側の周辺デバイスに対してターゲット・トランズアクションのアドレスを示す。
l_csn	出力	Low	ローカル・ターゲット・チップ・セレクト信号。この信号がアクティブになると、周辺デバイスにターゲット・トランズアクションの発生が通知される。l_acknとl_csnは同時にアサートされない。

表5　pci_aのローカル側とのインタフェース信号　(2/3)

信号名	タイプ	極性	説　明
l_rdn	出力	Low	リード信号。pci_aはl_rdnをアサートして、ローカル側の周辺デバイスに対してリード・アクセスであることを通知する。pci_aは周辺デバイスのターゲット・レジスタからのリード動作や、PCI DMAライト・トランズアクションを行うときにl_rdnを出力する。ターゲット・リードの動作では、pci_aがl_csnとl_rdn信号をアサートし、DMAライト動作では、l_acknとl_rdn信号をアサートする。
l_wrn	出力	Low	ライト信号。pci_aはローカル側の周辺デバイスにライト・アクセス動作を通知するときに、このl_wrnをアサートする。pci_aは周辺デバイスのターゲット・レジスタへのライト動作や、PCI DMAリード・トランズアクションを行うときにl_wrnを使用する。ローカル側へのライト動作では、pci_aがローカル側にターゲット・アクセスを行うためのl_csnとl_wrn信号をアサートするか、あるいはDMAリード・アクセスを行うためのl_acknとl_wrn信号をアサートする。
l_ackn	出力	Low	ローカルDMAアクノレッジ信号。l_acknがLowレベルになると、ローカル側の周辺デバイスに対してDMAリードまたはライトのトランズアクションを許可されていることが通知される。この場合、周辺デバイスはpci_aを通してPCIバスへ、またはPCIバスからのデータ転送が行えるようになる。
l_clk	出力	−	ローカルPCIクロック信号。l_clkはPCIバスのクロックをバッファしたもので、周辺デバイスのすべてのコントロール・ロジックをpci_aに同期化させるために使われる。
l_reset	出力	High	ローカル・リセット信号。ローカル側の周辺デバイスをリセットするときに、pci_aがこのl_resetをアサートする。このl_reset信号はPCIマスタ・リセットの期間にアクティブとなり、その後はl_rstビット（DMAコントロール・ステータス・レジスタの2番目のビット）の状態に従う。
l_dma_acr_wr	入力	High	ローカルDMAアドレス・カウンタ・レジスタ・ライト信号。ローカル側がDMAアドレス・カウンタ・レジスタに対するライト・アクセスを行うときにこの信号をアサートする。この信号をHighレベルにすると、l_dma_in[31..0]バス上のデータがdma_acrに書き込まれる。
l_dma_bcr_wr	入力	High	ローカルDMAバイト・カウンタ・レジスタ・ライト信号。ローカル側がDMAバイト・カウンタ・レジスタに対するライト・アクセスを行うときにこの信号をアサートする。この信号をHighレベルにすると、l_dma_dat_in[31..0]バス上のデータがdma_bcrに書き込まれる。

表5　pci_aのローカル側とのインタフェース信号　(3/3)

信号名	タイプ	極性	説　明
l_dma_csr_wr	入力	High	ローカルDMAコントロール・ステータス・レジスタ・ライト信号。ローカル側がDMAコントロール／ステータス・レジスタに対するライト・アクセスを行うときにこの信号をアサートする。この信号をHighレベルにすると、l_dma_dat_in[31..0]バス上のデータがdma_csrに書き込まれる。
l_dma_dat_in[31..0]	入力	−	ローカルDMAデータ入力。DMAライト信号のいずれかが（l_dma_acr_wr、l_dma_bcr_wr、またはl_dma_csr_wr）アサートされたとき、対応するDMAレジスタに書き込まれるデータが供給される。
l_dma_csr_out[6..0]	出力	−	ローカルDMAコントロール・ステータス・レジスタ出力信号。DMAコントロール／ステータス・レジスタからのダイレクト出力。
l_dma_acr_out[31..0]	出力	−	ローカルDMAアドレス・カウンタ・レジスタ出力。DMAアドレス・カウンタ・レジスタからのダイレクト出力。
l_dma_bcr_out[16..0]	出力	−	ローカルDMAバイト・カウンタ・レジスタ出力。DMAバイト・カウンタ・レジスタからのダイレクト出力。
l_dma_isr_out[4..0]	出力	−	ローカルDMAインタラプト・ステータス・レジスタ出力。DMAインタラプト・ステータス・レジスタからのダイレクト出力。

ファンクション・プロトタイプ

pci_aのアルテラ・ハードウェア記述言語（AHDL）のファンクション・プロトタイプを以下に示します。

```
FUNCTION pci_a (clk, framen_in, gntn, idsel,
   l_dat_in[31..0], l_holdn, l_irqn, l_req, rstn,
   stopn_in, trdyn_in, l_dma_acr_wr, l_dma_bcr_wr,
   l-dma_csr_wr, l_dma_dat_in[31..0])

   WITH (SUBSYSTEM_ID, SUBSYSTEM_VEND_ID, DEVICE_ID,
   DEVICE_VEND_ID, CLASS_CODE, REVISION_ID, BAR0_RW_BITS,
   TARGET_DEVICE)

   RETURNS (framen_out, l_ackn,
   l_adr[30-BAR0_RW_BITS..0], l_clk, l_csn,
   l_dat_out[31..0], l_ rdn, l_reset, l_wrn, stopn_out,
   trdyn_out, ad[31..0], cben[3..0], devseln, intan,
   irdyn, par, perrn, reqn, serrn; l_dma_csr_out[6..0],
   l_dma_acr_out[31..0], l_dma_bcr[16..0],
   l_dma_isr_out[4..0], l_ben[3..0]);
```

パラメータ

BAR0_RW_BITSとTARGET_DEVICEを除くpci_aのパラメータは、リード・オンリとなっているPCIバス・コンフィギュレーション・レジスタに設定されます。これらのレジスタは、デバイス・アイデンティフィケーション（ID）レジスタと呼ばれています。デバイスIDレジスタの詳細については、18ページの「コンフィギュレーション・レジスタ」を参照してください。

BAR0_RW_BITSのパラメータはBAR0に対してインスタンス化されるリード／ライトのビット数をコントロールします。PCIの仕様によると、このBAR0に対してインスタンス化されるリード／ライトのビット数はBAR0によってリザーブされているメモリ・アドレスの範囲をコントロールするようになっています。BAR0_RW_BITSのパラメータの値は、1から12の範囲となっている必要があります。TARGET_DEVICEのパラメータを設定することによって、特定のデバイスやパッケージに対してデザインがもっとも最適化されるようになり、ターゲット・デバイスによってタイミングの規格が確実に守られるようになります。最新のサポート・デバイスとパッケージについては、pci_aに添付されているreadme.htmのファイルで確認してください。表6はpci_aファンクションの各パラメータを解説したものです。

表6　パラメータ

パラメータ名	フォーマット	デフォルト値	説　明
BAR0_RW_BITS	10進	12	BARアドレス・スペースのサイズ
TARGET_DEVICE	記号	"EPF10K30RC240"	デバイスの選択
CLASS_CODE	24ビットHex	H"FF0000"	クラス・コード・レジスタ
DEVICE_ID	16ビットHex	H"0001"	デバイスIDレジスタ
DEVICE_VEND_ID	16ビットHex	H"1172"	デバイス・ベンダIDレジスタ
REVISION_ID	8ビットHex	H"02"	リビジョンIDレジスタ
SUBSYSTEM_ID	16ビットHex	H"0000"	サブシステムIDレジスタ
SUBSYSTEM_VEND_ID	16ビットHex	H"0000"	サブシステム・ベンダIDレジスタ

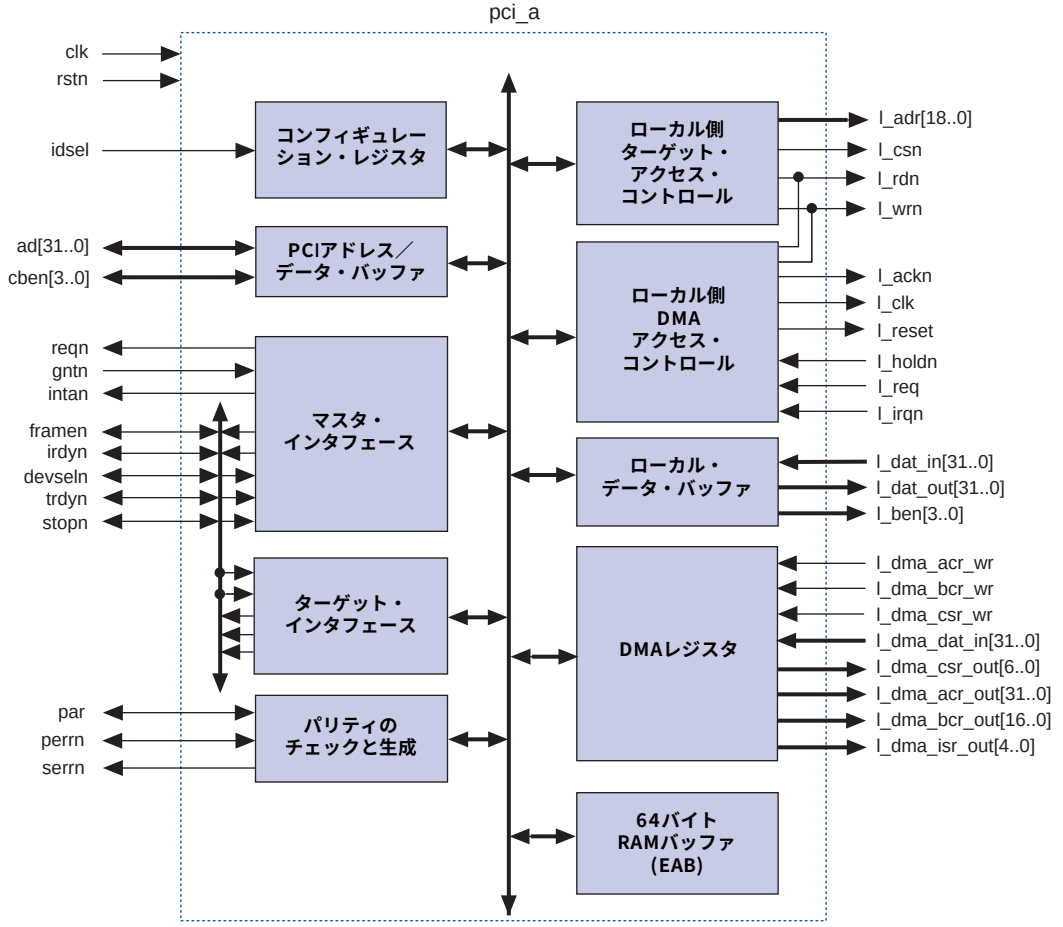
機能説明

pci_aは３つの主要な機能ブロックによって構成されています。

- あらかじめ定義された64バイトのPCIバス・コンフィギュレーション・レジスタ・スペースとマスタ・コントロール・ロジック。
- ターゲット・デコードとレジスタ・リード／ライト信号を含むPCIバス・ターゲット・インタフェース・コントロール・ロジック。
- 64バイト（16 DWORD）のRAMバッファ、および４個のレジスタを内蔵したエンベデッドDMAコントロール・エンジンと、マスタ／ターゲット・アクセス用のPCIバス・アービトレーションとリード／ライト・コントロール機能を含むローカル側とのインタフェースとDMAコントロール・ロジック。

図２はpci_aのブロック図を示したものです。

図２　pci_aのブロック図



サステインド・トライ・ステート信号の動作

PCIの仕様では、絶えず異なるバス・エージェントによってサンプルされるが、１度に１つのエージェント（信号源）によってドライブされている信号をサステインド・トライ・ステート信号として定義しています。例えば、framenは複数の異なるPCIのバス・ターゲットによって絶えずサンプルされていますが（トランズアクションの開始を検出するため）、１度に１つのPCIバス・マスタによってドライブされます。

PCIの仕様では、サステインド・トライ・ステート信号をトライ・ステート状態にする前に１クロックの期間にこれらの信号をインアクティブにドライブする必要があると規定されています。また、マスタ・デバイスがリード動作時にアドレスをアサートした後でad[31..0]をリリースするのと同じように、サステインド・トライ・ステート信号をリリースする場合にも、他のデバイスがその信号をドライブする前に、１つのフル・クロック・サイクルでトライ・ステート状態にする必要があると規定されています。

PCIの仕様では、他のバス・エージェントがトライ・ステートになっているサステインド・トライ・ステート信号をドライブできるようにするためのクロック・サイクルを、ターン・アラウンド・サイクルとして定義しています。このターン・アラウンド・サイクルはバス上における信号の衝突を避けるために使用されます。

マスタ・デバイス信号と信号のアサート

図３は、pci_aとPCIバスをインタフェースするPCI準拠のマスタ・デバイス信号を示したものです。各信号は機能的にグループ化され、信号の方向はPCIバス上でpci_aがマスタとして動作することを想定して図示されています。

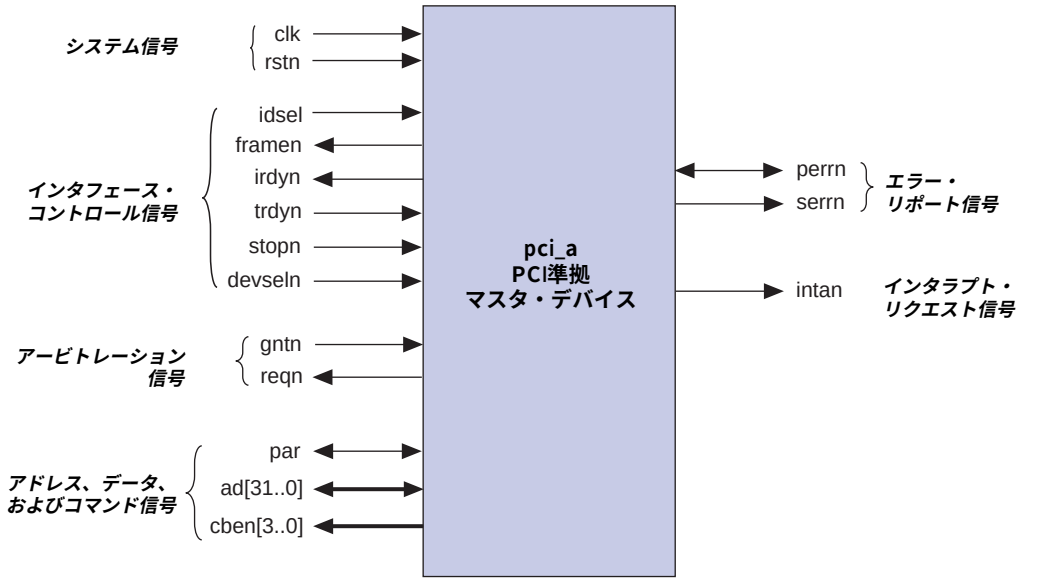
pci_aのマスタ・シーケンスは、PCIバス上でマスタになることを要求するreqnのアサートから開始されます。アービタ（通常はPCIのホスト・ブリッジ）からgntnが受信され、バス・アイドルが検出された後で、pci_aはframenをアサートして、ad[31..0]にPCIアドレスを、またcben[3..0]にバス・コマンドを１クロック・サイクルの期間ドライブして、アドレス・フェーズを開始します。

pci_a がバス上にデータを送出できるようになると、pci_aはirdynをアサートします。このとき、pci_aのマスタ・ロジックは、ターゲット・デバイスによってドライブされている制御信号をモニタします（トランズアクションのアドレス・フェーズ期間では、PCIバス上に現れるアドレスとコマンド信号をデコードすることによって、ターゲット・デバイスが決定されます）。ターゲット・デバイスは以下に示すいずれかの状態を表示するため、devseln、trdyn、stopnの各制御信号をドライブします。

- データ・トランズアクションがデコードされ、許可された状態。
- ターゲット・デバイスでデータ・オペレーションの準備が整っている状態（trdynとirdynの双方がアクティブの場合は、送信デバイスから受信デバイスへDWORDのデータがクロックに同期して送出される）。

- マスタ・デバイスが処理中のトランズアクションを停止しなければならない状態。

図3　pci_aマスタ・デバイス信号



ターゲット・デバイス信号と信号のアサート

図4は、pci_aとPCIバスをインタフェースするPCI準拠のターゲット・デバイスの信号を示したものです。各信号は機能別にグループ化されており、信号の方向はPCIバス上でpci_aがターゲットとして動作する場合を想定したものとなっています。

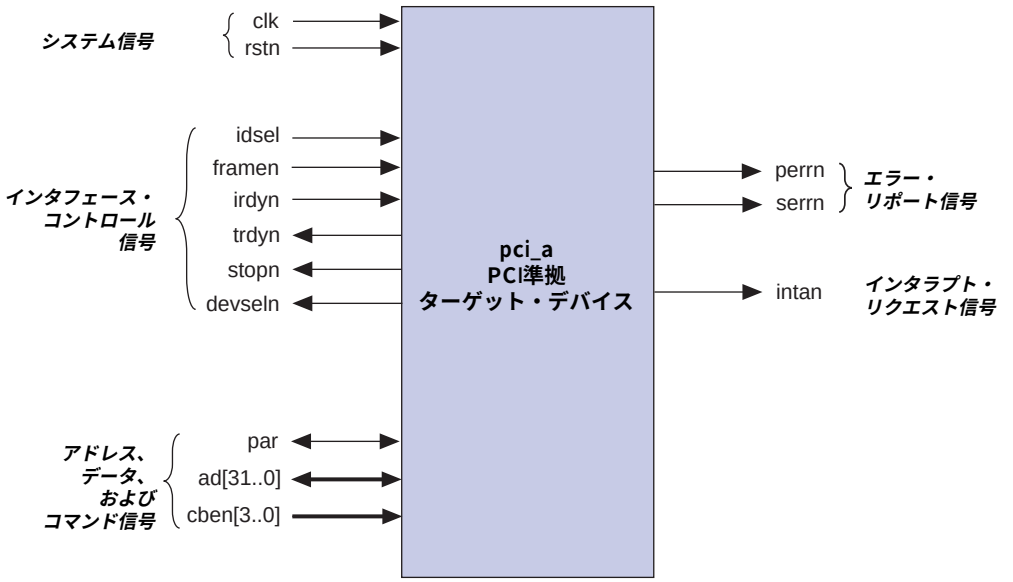
pci_aのターゲット・シーケンスは、マスタ・デバイスがframenをアサートし、PCIバス上にターゲットのアドレスとコマンド信号をドライブすることによって開始されます。ターゲット・デバイスはPCIバス上のアドレスをデコードし、devselnをアサートすることによって、マスタに対してトランズアクションが許可されていることを通知します。次に、マスタはirdynをアサートして、以下に示す状態にあることをターゲット・デバイスに通知します。

- リード・オペレーションの場合は、マスタ・デバイスがデータ・トランズアクションを完了する状態。
- ライト・オペレーションの場合は、有効なデータがad[31..0]にある状態。

15ページの「マスタ・デバイス信号と信号のアサート」で述べられたように、 pci_aが選択されたターゲット・デバイスとして動作する場合は、devseln、trdyn、stopnの各制御信号がpci_aによってドライブされます。

pci_aがターゲット・デバイスとして動作する場合は、シングル・サイクルのアクセスだけがサポートされます。したがって、pci_aはstopnとtrdynを同時にアクティブにドライブします。そして、irdyn信号がアクティブになったことが検知されると、送信デバイスから受信デバイスにデータ・ワードがクロックに同期して送出されます。

図4　pci_aのターゲット・デバイス信号



パリティ信号の動作

すべてのバス・サイクルにはパリティが含まれています。ad[31..0]バス上での転送を行う各デバイスは、マスタ・デバイスの出力するアドレスに含まれるpar信号をドライブしなければなりません。また、PCIバス上のパリティは偶数となっており、ad[31..0]、cben[3..0]、parの1の総数が偶数となる必要があります。パリティ・チェックを必要としないシステムもありますが、エージェントのPCIコマンド・レジスタにより、パリティ・チェックをイネーブルにすることが可能です。アドレスのパリティ・エラーはserrnに出力され、データのパリティ・エラーはperrnに出力されます。parのビットはad[31..0]より１クロック・サイクル遅れて表示され、パリティ・エラー信号はparのビットに対してさらに１クロック・サイクル遅れて出力されます。したがって、パリティ・エラー信号はアドレスやデータに対して２クロック・サイクル遅れて出力されることになります。

PCIバス・コマンド

表7はpci_aでサポートされているPCIバス・コマンドを示したものです。

表7　サポートされているPCIバス・コマンド

cben[3..0]の値	バス・コマンド・サイクル	ターゲット・サポート	マスタ・サポート
0110	メモリ・リード	√	√
0111	メモリ・ライト	√	√
1010	コンフィギュレーション・リード	√	
1011	コンフィギュレーション・ライト	√	

pci_aはメモリ・リード／ライトやコンフィギュレーション・リード／ライトのコマンドをサポートしています。pci_aがマスタ・デバイスとして動作するときは、pci_aが標準的なメモリ・リード／ライト動作を実行します。また、ターゲットとして動作するときは、pci_aが標準的なメモリ・リードおよびライト・トランズアクションに応答します。さらに、pci_aはコンフィギュレーション・リードおよびライトの動作にも応答します。

コンフィギュレーション・レジスタ

PCIバスの各論理デバイスには、コンフィギュレーション・レジスタを実現するためにリザーブされている64ワードのコンフィギュレーションDWORDのブロックが含まれています。最初の16ワード分のDWORDのフォーマットには、PCI SIGの***PCI仕様準拠チェックリスト Revision 2.1***により、タイプ1とタイプ0の2つのヘッダ・タイプが定義されています。ヘッダ・タイプ-1はPCI-PCIブリッジ用であり、ヘッダ・タイプ-0はpci_aを含む他のすべてのデバイスで使用されます。

表8は、定義済みの64バイト分のコンフィギュレーション・レジスタ・スペースを示したものです。この範囲内にあるレジスタはデバイスを確定したり、PCIバスの制御、PCIバス・ステータスを表示するときに使用されます。塗りつぶされている部分は、pci_aがサポートしているレジスタです。

表８　PCIバスのコンフィギュレーション・レジスタ

アドレス	バイト			
	3	2	1	0
00H	デバイスID		ベンダID	
04H	ステータス・レジスタ		コマンド・レジスタ	
08H	クラス・コード			リビジョンID
0CH	BIST	ヘッダ・タイプ	レイテンシ・タイマ	ｷｬｯｼｭ･ﾗｲﾝ･ｻｲｽﾞ
10H	ベース・アドレス・レジスタ0			
14H	ベース・アドレス・レジスタ1			
18H	ベース・アドレス・レジスタ2			
1CH	ベース・アドレス・レジスタ3			
20H	ベース・アドレス・レジスタ4			
24H	ベース・アドレス・レジスタ5			
28H	カード・バスCISポインタ			
2CH	サブシステムID		サブシステム・ベンダID	
30H	拡張ROMベース・アドレス・レジスタ			
34H	予約			
38H	予約			
3CH	最大レイテンシ	最小グラント	インタラプト・ピン	インタラプト・ライン

表９は、pci_aがサポートしているコンフィギュレーション・レジスタのアドレス・マップを示したものです。リード／ライトは、他のPCIバスのエージェントから見た動作状態で示されています。pci_aをMAX+PLUS IIでデザイン内にインスタンス化するときにパラメータをセットすることによって、複数のリード・オンリ・レジスタを設定することができます。例えば、MAX+PLUS IIでDEVICE_IDパラメータを書き換えることによって、デバイスIDレジスタのデフォルト値を変更することができます。PCIバスがリセットされるとき、指定したデフォルト値がレジスタのステートに応じて定義されます。

表９　pci_aがサポートするコンフィギュレーション・レジスタのアドレス・マップ　(1/2)

アドレス・オフセット (16進)	予約範囲 (16進)	使用バイト／予約	リード／ライト	ニーモニック	レジスタ名
00	00-01	2/2	リード	ven_id	ベンダID
02	02-03	2/2	リード	dev_id	デバイスID
04	04-05	2/2	リード／ライト	comd	コマンド
06	06-07	2/2	リード／ライト	status	ステータス
08	08-08	1/1	リード	rev_id	リビジョンID

表9　pci_aがサポートするコンフィギュレーション・レジスタのアドレス・マップ　(2/2)					
アドレス・オフセット（16進）	予約範囲（16進）	使用バイト／予約	リード／ライト	ニーモニック	レジスタ名
09	09-0B	3/3	リード	class	クラス・コード
0D	0D-0D	1/1	リード／ライト	lat_tmr	レイテンシ・タイマ
0E	0E-0E	1/1	リード	header	ヘッダ・タイプ
10	10-13	4/4	リード／ライト	bar0	ベース・アドレス・レジスタ0
2C	2C-2D	2/2	リード	sub_ven_id	サブシステム・ベンダID
2E	2E-2F	2/2	リード	sub_id	サブシステムID
3C	3C-3C	1/1	リード／ライト	int_ln	インタラプト・ライン
3D	3D-3D	1/1	リード	int_pin	インタラプト・ピン
3E	3E-3E	1/1	リード	min_gnt	最小グラント
3F	3F-3F	1/1	リード	max_lat	最大レイテンシ

ベンダIDレジスタ（オフセット = 00 Hex）

ベンダIDレジスタはデバイスの製造メーカ（pci_aはアルテラ）を定義するための16ビットのリード・オンリ・レジスタです。このレジスタの値はPCI SIGにより指定されています。このレジスタのデフォルト値はアルテラのベンダIDで、1172（hex）となっています。DEVICE_VENDパラメータを設定することで（表6を参照してください）、ベンダIDレジスタの値をPCI SIGが指定した別のベンダID値に変更することも可能です。表10を参照してください。

表10　ベンダIDレジスタのフォーマット			
データ・ビット	ニーモニック	リード／ライト	定　義
15..0	ven_id	リード	PCIベンダID

デバイスIDレジスタ（オフセット = 02 Hex）

デバイスIDレジスタはデバイスのタイプを定義するための16ビットのリード・オンリのレジスタです。このレジスタの値は製造メーカによって決定されています。（pci_aのデバイスIDレジスタの値はアルテラが指定）デバイスIDレジスタのデフォルト値は0001（hex）ですが、DEVICE_IDパラメータの設定により、このデバイスIDレジスタの値を変更することも可能です（13ページの表6を参照してください）。

コマンド・レジスタ（オフセット = 04 Hex）

コマンド・レジスタは、pci_aのPCIバス・アクセスに対する応答やアクセスをコントロールするための16ビットのリード／ライト・レジスタです。表11を参照してください。

表11　コマンド・レジスタのフォーマット

データ・ビット	ニーモニック	リード／ライト	定　義
0	未使用	–	–
1	mem_ena	リード／ライト	メモリ・アクセス・イネーブル。このビットがHighになると、pci_aがイネーブルとなり、pci_aがターゲットとしてPCIバス・メモリ・アクセスに応答できるようになる。DMAレジスタはメモリ・ターゲット・アクセスによりセットされるため、pci_aにDMA転送を実行させる場合は、初期化動作の中でmem_enaのビットをセットする必要がある。
2	mstr_ena	リード／ライト	マスタ・イネーブル。このビットがHighになると、pci_aがPCIバスのマスタの権利を獲得できるようになる。pci_aにDMA転送を実行させるためには、初期化動作の一部としてmstr_enaのビットをセットする必要がある。
5..3	未使用	–	–
6	perr_ena	リード／ライト	パリティ・エラー・イネーブル。このビットがHighになると、pci_aがperrn出力からパリティ・エラーを通知できるようになる。
7	未使用	–	–
8	serr_ena	–	システム・エラー・イネーブル。このビットがHighになると、pci_aはserrn出力からアドレス・パリティ・エラーを通知できるようになる。ただし、システム・エラー信号を通知するときには、perr_enaビットもHighにする必要がある。
15..9	未使用	–	–

ステータス・レジスタ（オフセット = 06 Hex）

ステータス・レジスタはバス関連イベントのステータスを表示するための16ビット・レジスタです。ステータス・レジスタへのリード・トランズアクションは標準的な動作ですが、ライト・トランズアクションは通常のライト・トランズアクションと異なり、ステータス・レジスタのビットをクリアすることはできますが、セットすることはできません。ステータス・レジスタのビットは、このビットにロジック-1を書き込むことによってクリアされます。4000（hex）をステータス・レジスタに書き込むことによって14ビット目がクリアされ、残りのビットは元の値を保持します。ステータス・レジスタのデフォルト値は0400（hex）です。表12を参照してください。

表12　ステータス・レジスタのフォーマット

データ・ビット	ニーモニック	リード／ライト	定　義
7..0	未使用	–	–
8	dat_par_rep	リード／ライト	データ・パリティをリポートするビット。このビットがHighになると、pci_aがマスタ・デバイスとしてリード・トランズアクション中にperrnをアサートしているか、または、ライト・トランズアクション中に、ターゲット・デバイスによってperrnがアサートされていることが示される。perr_enaのビット（コマンド・レジスタの６ビット目）がHighになるときだけ、このビットもHighになる。
10..9	devsel_tim	リード	デバイス・セレクト・タイミング。このビットはdevselnn出力を通じてpci_aのターゲット・アクセス・タイミングを表示する。pci_aは低速ターゲット・デバイスとして設計されている。
11	未使用	–	–
12	tar_abrt	リード／ライト	ターゲット・アボート。このビットがHighになると、処理中のターゲット・デバイスのトランズアクションが終了させられたことが示される。
13	mstr_abrt	リード／ライト	マスタ・アボート。このビットがHighになると、処理中のマスタ・デバイスのトランズアクションが終了させられたことが示される。
14	serr_set	リード／ライト	システム・エラーを通知するビット。このビットがHighになると、pci_aがserrn出力をアクティブにドライブして、アドレス・フェーズ・パリティ・エラーが発生したことが示される。
15	det_par_err	リード／ライト	パリティ・エラー検出。このビットがHighになると、pci_aがアドレスかデータ・パリティ・エラーのいずれかを検出したことが示される。perr_enaにより、パリティ・エラーの通知がディセーブルされた場合でも、pci_aはこのdet_par_errビットをセットする。

リビジョンIDレジスタ（オフセット = 08 Hex）

リビジョンIDレジスタはデバイスのリビジョンを示す8ビットのリード・オンリ・レジスタです。このレジスタの値は製造者によって指定されます（pci_aはアルテラが指定）。したがって、pci_aのリビジョン番号がリビジョンIDレジスタのデフォルト値として設定されます。表13を参照してください。また、REVISION_IDパラメータを設定することにより、リビジョンIDレジスタのデフォルト値を変更することも可能です（表6を参照してください）。

表13　リビジョンIDレジスタのフォーマット

データ・ビット	ニーモニック	リード／ライト	定　義
7..0	rev_id	リード	PCIリビジョンID

クラス・コード・レジスタ（オフセット = 09 Hex）

クラス・コード・レジスタは24ビットのリード・オンリ・レジスタで、ベース・クラス、サブ・クラス、プログラミング・インタフェースの3つのサブ・レジスタに分割されています。この詳細については***PCIローカル・バス仕様書 Revision 2.1***を参照してください。表14はこのレジスタのフォーマットを示したものです。クラス・コード・レジスタのデフォルト値はFF0000（hex）ですが、CLASS_CODEパラメータを設定することにより、この値を変更することもできます（表6を参照してください）。

表14　クラス・コード・レジスタのフォーマット

データ・ビット	ニーモニック	リード／ライト	定　義
23..0	class	リード	クラス・コード

レイテンシ・タイマ・レジスタ（オフセット = 0D Hex）

レイテンシ・タイマ・レジスタは8ビットのレジスタで、このうち0、1、2の各ビットがGNDになっています。このレジスタには、pci_aがPCIバスでマスタの状態を保持するために必要とするPCIバス・クロック・サイクルの最大時間が設定されます。トランズアクションの開始後、pci_aはクロックの立ち上がりエッジごとにこのレイテンシ・タイマをディクリメントします。レイテンシ・タイマ・レジスタのデフォルト値は00（hex）です。表15を参照してください。

表15　レイテンシ・タイマ・レジスタのフォーマット

データ・ビット	ニーモニック	リード／ライト	定　義
2..0	lat_tmr	リード	レイテンシ・タイマ・レジスタ
7..3	lat_tmr	リード／ライト	レイテンシ・タイマ・レジスタ

ヘッダ・タイプ・レジスタ（オフセット = 0E Hex）

ヘッダ・タイプ・レジスタは、pci_aをシングル・ファンクション・デバイスとして定義するときに使用される８ビットのリード・オンリ・レジスタです。ヘッダ・タイプ・レジスタのデフォルト値は00（hex）です。表16を参照してください。

表16　ヘッダ・タイプ・レジスタのフォーマット

データ・ビット	ニーモニック	リード／ライト	定　義
7..0	header	リード	PCI ヘッダ・タイプ

ベース・アドレス・レジスタ-0（オフセット = 10 Hex）

BAR0_RW_BITSのパラメータの設定値に応じて、ベース・アドレス・レジスタ-0（BAR0）が12ビットから１ビットの範囲のレジスタで構成されます。このBAR0_RW_BITSのパラメータの値はpci_aファンクションをインスタンス化するときに設定することができ、これによってpci_aのターゲット・スペースが決定されます。このプロセスは***PCIローカル・バス仕様書 Revision 2.1***にしたがって行われます。このPCIの仕様には、リード／ライト・レジスタとして構成されるビット数が、BARによってリザーブされるメモリ・アドレス・スペースの容量を定義すると述べられています。電源投入時にソフトウェアにより、BARへすべて1を書き込み、その後に各値を読み出すことで、１個のデバイスに要求されるアドレス・スペースを決定することが可能です。要求されるアドレス・スペースを決定するとき、pci_aはすべての下位ビットを0に戻します。要求されるアドレス・スペースの容量は、一般的にBAR0_RW_BITSのパラメータの値の関数となります。BAR0_RW_BITS = *n*の場合、リザーブされるアドレス・スペースは、$2^{(32-n)}$となります。例えば、BAR0_RW_BITS = 4の場合は、リザーブされるアドレス・スペースが$2^{(32-4)}$、すなわち256Mバイトとなります。表17を参照してください。

表17　ベース・アドレス・レジスタのフォーマット　(1/2)

データ・ビット	ニーモニック	リード／ライト	定　義
0	mem_ind	リード	メモリ・インディケータ。このビットはレジスタがI/Oデコーダになっているか、あるいはメモリ・アドレス・デコーダになっているかを示す。pci_aではこのビットがGNDになっており、メモリ・アドレス・デコーダとなっていることを示す。
2..1	mem_type	リード	メモリ・タイプ。これらのビットはpci_aのメモリ・アドレス・スペースに実現されるメモリのタイプを表示する。pci_aでは、これらのビットがGNDになっており、メモリ・ブロックを32ビットのアドレス・スペースの任意の位置に配置できることを示す。

表17　ベース・アドレス・レジスタのフォーマット　(2/2)			
データ・ビット	ニーモニック	リード／ライト	定　義
3	pre_fetch	リード	メモリ・プリフェッチ可能。BAR0によって定義されたメモリ・ブロックに対して、ホスト・ブリッジによるプリフェッチが可能かどうかを表示するビット。pci_aでは、アドレス・スペースのプリフェッチは不可能になっており、このビットからはLowがリードされる。
31-BAR0_RW_BITS	未使用	–	–
31..(32-BAR0_RW_BITS)	bar0	リード／ライト	ベース・アドレス・レジスタ-0

サブシステム・ベンダIDレジスタ（オフセット = 2C Hex）

サブシステム・ベンダIDレジスタは16ビットのリード・オンリ・レジスタであり、異なるベンダによって設計されたカード上に同じ機能のデバイスが実装されている場合に、そのアドイン・カードを識別するために使用されます。このレジスタの値はPCI SIGによって指定されています。表18を参照してください。サブシステム・ベンダIDレジスタのデフォルト値は0000（hex）ですが、SUBSYSTEM_VEND_IDパラメータの設定により、デフォルト値を変更することが可能です（表6を併せて参照してください）。

表18　サブシステム・ベンダIDレジスタのフォーマット			
データ・ビット	ニーモニック	リード／ライト	定　義
15..0	sub_vend_id	リード	PCIサブシステム／ベンダID

サブシステムIDレジスタ（オフセット = 2E Hex）

サブシステムIDレジスタはサブシステムを識別するためのもので、このレジスタの値はサブシステム・ベンダ（設計者）により定義されます。表19を参照してください。サブシステムIDレジスタのデフォルト値は0000（hex）ですが、SUBSYSTEM_IDパラメータの設定により、デフォルト値を変更することが可能です（表6を併せて参照してください）。

表19　サブシステムIDレジスタのフォーマット			
データ・ビット	ニーモニック	リード／ライト	定　義
15..0	sub_id	リード	PCIサブシステムID

インタラプト・ライン・レジスタ（オフセット = 3C Hex）

インタラプト・ライン・レジスタは8ビットのレジスタによって構成されており、intan出力が接続されるシステム・インタラプト・リクエスト・ライン（システム・インタラプト・コントローラ）を表示します。インタラプト・ライン・レジスタの内容は電源の立ち上げ時にシステム・ソフトウェア

によって書き込まれ、そのデフォルト値はFF（hex）です。表20を参照してください。

表20　インタラプト・ライン・レジスタのフォーマット			
データ・ビット	ニーモニック	リード／ライト	定　義
7..0	int_ln	リード／ライト	インタラプト・ライン・レジスタ

インタラプト・ピン・レジスタ（オフセット = 3D Hex）

インタラプト・ピン・レジスタは8ビットのリード・オンリ・レジスタで構成され、ここにpci_aのPCIバスのインタラプト・リクエスト・ラインとなるintanが定義されます。インタラプト・ピン・レジスタのデフォルト値は01（hex）です。表21を参照してください。

表21　インタラプト・ピン・レジスタのフォーマット			
データ・ビット	ニーモニック	リード／ライト	定　義
7..0	int_pin	リード	インタラプト・ピン・レジスタ

最小グラント・レジスタ（オフセット = 3E Hex）

最小グラント・レジスタは8ビットのリード・オンリ・レジスタで構成され、ここにpci_aがPCIバス上でマスタ状態を保持する時間が定義されます。このレジスタにセットされる時間は要求されるバースト期間を表し、その値は250ns単位でインクリメントされます。pci_aは4µsecの期間を必要とします。最小グラント・レジスタのデフォルト値は10（hex）です。表22を参照してください。

表22　最小グラント・レジスタのフォーマット			
データ・ビット	ニーモニック	リード／ライト	定　義
7..0	min_gnt	リード	最小グランド・レジスタ

最大レイテンシ・レジスタ（オフセット = 3F Hex）

最大レイテンシ・レジスタは8ビットのリード・オンリ・レジスタで、ここにpci_aがPCIバスにアクセスするときの周波数が定義されます。pci_aは最大レイテンシに対して特に強い要求を持っていないため、最大レイテンシ・レジスタの値は00（hex）に設定されます。表23を参照してください。

表23　最大レイテンシ・レジスタのフォーマット			
データ・ビット	ニーモニック	リード／ライト	定　義
7..0	max_lat	リード	最大レイテンシ・レジスタ

PCIバス・トランズアクション

このセクションでは、pci_aのPCIバス・トランズアクションについて解説します。このセクションにある図を参照する場合は、以下のことが考慮される必要があります。

- PCIバスに対するpci_aのすべてのDMAアクセスは、4バイトまたは32ビット転送です。このため、すべてのバイト・イネーブルがデータ転送期間中にアクティブになります。pci_aが外部ターゲット・ライト・アクセスを実行するときの転送バイト幅は選択可能です。
- 図5から図16では、pci_aによってドライブされていないPCIバス信号がトライ・ステート状態になっていますが、サステインド・トライ・ステート信号がPCIバス上のエージェントによってドライブされていないときは、これらの信号がプルアップ・レジスタの存在により、実際にはHighレベルとなっています。

pci_aがPCIバスをアクセスする場合、3種類のトランズアクションが存在します。

- ターゲット
- コンフィギュレーション
- マスタ

ターゲット・トランズアクション

すべてのターゲット転送を開始させるためのイベントのシーケンスは、全く同一です。マスタがPCIバス上でマスタの権利を獲得した後、ターゲット・リードまたはライトの動作が開始されます。マスタ・デバイスはframenをアサートし、ad[31..0]バス上にアドレスを、またcben[3..0]にコマンドを送出します。pci_aは、framenがアサートされたときに、最初のクロック・エッジでアドレスとコマンド信号をラッチし、アドレスのデコードを開始します。

ターゲット・リード・トランズアクション

pci_aは2種類のターゲット・リード・トランズアクションをサポートしています。

- ***内部ターゲット・リード***—内部DMAレジスタからのターゲット・リード・トランズアクション
- ***外部ターゲット・リード***—ローカル側のターゲット・メモリ・スペースからのターゲット・リード・トランズアクション

この2つのターゲット・リード・トランズアクションのイベント・シーケンスは同一ですが、タイミングは異なります。（詳細は29ページの「外部ターゲット・リード・トランズアクション」を参照してください。）ローカル側のターゲット・メモリ・スペースからのターゲット・リード・トランズアクションでは、 pci_aがローカル側から供給されるデータを待つ必要があるため、より長い時間が必要になります。

内部ターゲット・リード・トランズアクション

アドレス・フェーズ（クロック-4）の終了直後に、マスタはframenをディアサート、irdynをアサートして、以下のことを表示します。

- トランズアクションがシングル・データ・フェーズであること
- pci_aがad[31..0]バスに送出したデータを、マスタ・デバイスがリードできる状態にあること

pci_aがアドレスをラッチした後、マスタ・デバイスがクロック-5でad[31..0]バスをトライ・ステート状態にします。pci_aはクロック-6の始まりでad[31..0]バスをドライブすることができます。マスタがバースト・アクセスを実行する場合は、framenとirdynの双方が継続してアサートされます。ただし、pci_aはターゲット・バーストをサポートしていないため、stopnをアサートしてマスタにディスコネクトを通知します。この場合、マスタは１クロック・サイクルの期間、framenをディアサート、irdynをアサートして、トランズアクションを終了させます。

図５では、pci_aがクロック-7でdevselnをアサートし、トランズアクションを許可したことをマスタに通知しています。devselnはマスタ・デバイスによりクロック-8の立ち上がりエッジでサンプルされ、これはPCIの仕様で定義される低速デコードになります。図５はpci_aの内部ターゲット・リード・トランズアクションのタイミングを示したものです。

図５　内部ターゲット・リード・トランズアクション

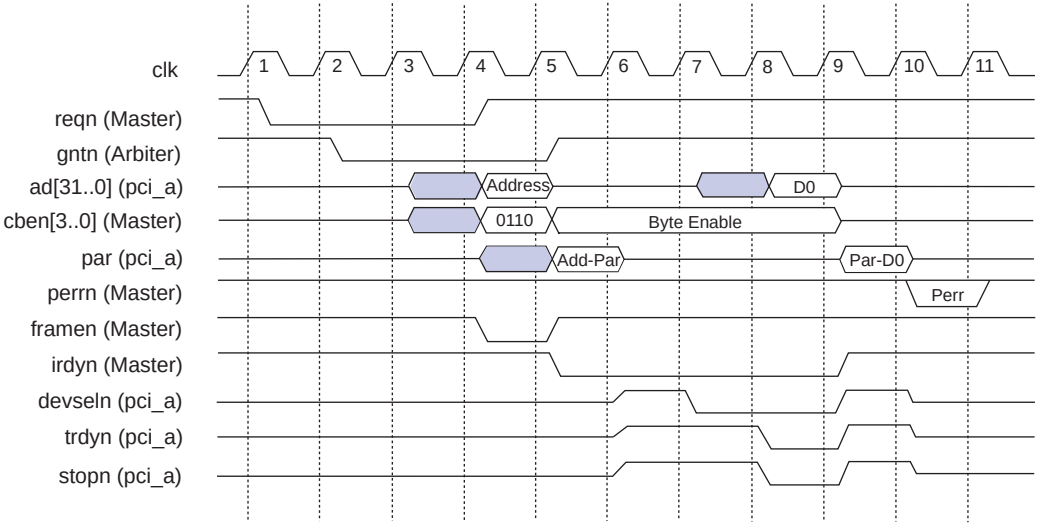


図５で示したように、pci_aはクロック-8でtrdynとstopnをアサートすることでad[31..0]バス上に有効データがあることを示し、ディスコネクトを要求します。データはクロック-8で転送され、irdynとtrdynがアクティブの場合、これらがマスタ・デバイスによりクロック-9の立ち上がりエッジでラッチされます。 PCI仕様書では、バースト転送をサポートしていないターゲットがバースト転送を要求された場合に、 最初のデータ・フェーズの期間にディスコネクトを発行しなければならないと規定されています。このPCIの仕様に準拠させるため、pci_aは常にstopnとtrdynを同時にアサートします。

マスタはクロック-5でアドレス・パリティのparをアクティブにし、pci_aはクロック-9でデータ・パリティであるparをアクティブにします。ターゲット・リード・トランズアクションでは、マスタ・デバイスがperrnをドライブしてデータ・パリティ・エラーを表示します。

クロック-9でデータはすでにサンプルされているため、pci_aがad[31..0]バスをリリースし、マスタがcben[3..0]をリリースします。devseln、trdyn、stopnの各信号はクロック-9でHighレベルにドライブされ、１クロック後にpci_aによってリリースされます。このようにして、サステインド・トライ・ステート信号をリリースする前に１クロック・サイクルの期間、Highレベルにドライブするという要求が満足されています。

外部ターゲット・リード・トランズアクション

外部ターゲット・リード・トランズアクションのイベント・シーケンスは、内部ターゲット・リード・トランズアクションと同様です。ただし、ローカル側へのDMAアクセスは他のローカル側へのアクセスよりも重要な動作として取り扱われるため、外部ターゲット・リード・トランズアクションはDMAがアイドル・ステートになっている状態のときだけに実行されます。また、DMAがアイドル・ステートでないときにpci_aが外部ターゲット・リード・トランズアクションを受信した場合は、pci_aがリトライ信号を発行します。

pci_aは、ローカル側がデータを供給できるようになるまで待つ必要があるため、ローカル側のターゲット・メモリ・スペースからのターゲット・リード・トランズアクション（外部ターゲット・リード）には、より長い時間が必要になります。ローカル側のロジックがl_csnとl_rdnがアサートされた１クロック後にデータを供給できない場合は、l_holdnをLowにしてデータの転送を停止させることが可能です。この場合、 データがl_dat_in[31..0]のバス上に提供されるまで、 l_holdn信号がLowレベルにドライブされることになります。

PCIの仕様では、ターゲット・トランズアクションの最初のデータ・フェーズは16クロック・サイクル以内に完了する必要があると規定されています。したがって、l_holdnを長い期間にわたってアサートした場合でも、ローカル・デバイスがこのPCI仕様に違反しないようになっている必要があります。

図６は、pci_aの外部ターゲット・リード・トランズアクションのタイミングを示したものです。

図６　外部ターゲット・リード・トランズアクション

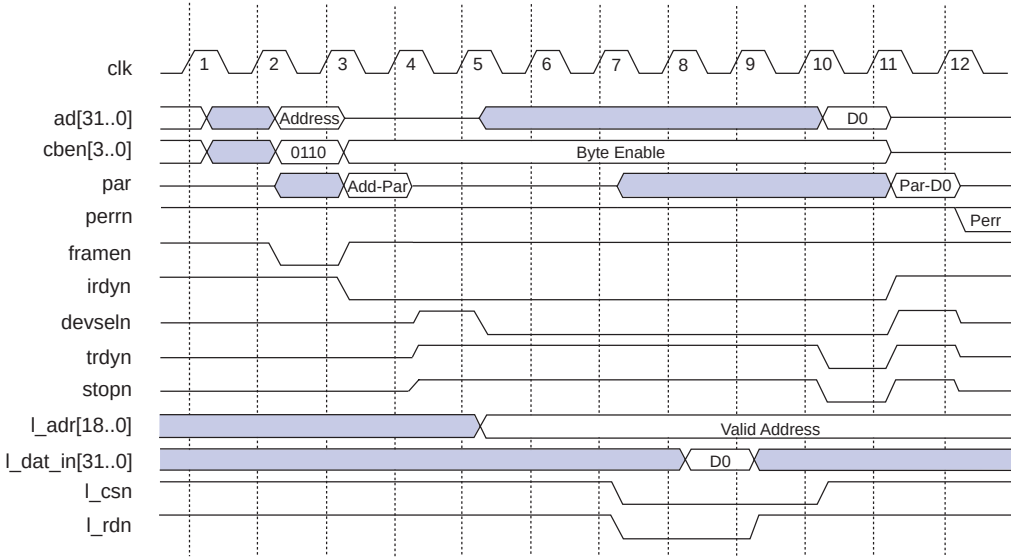
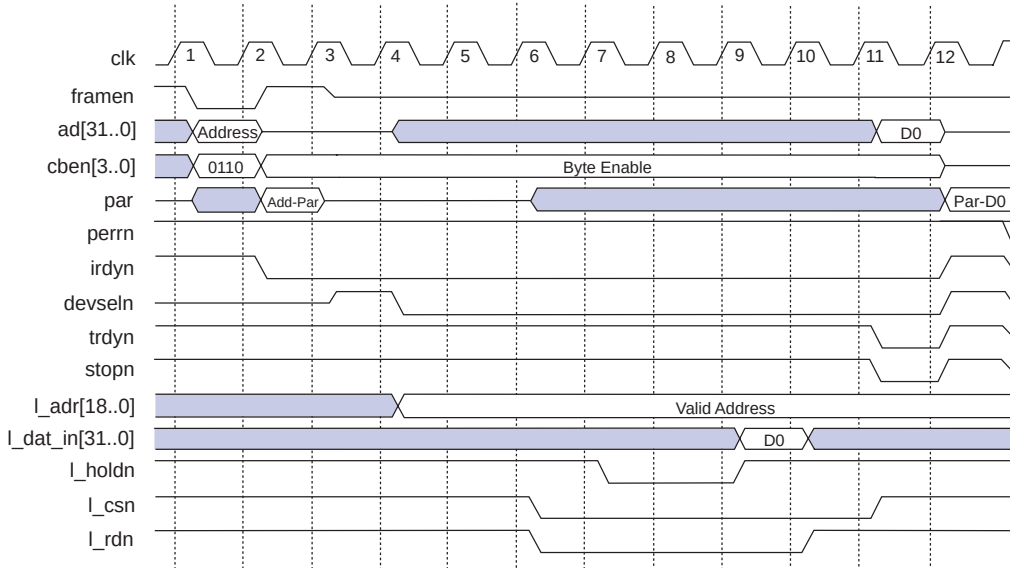


図７は、l_holdnをアサートして、追加のウェイト・ステートをローカル側に挿入した場合の外部ターゲット・リード・トランズアクションを示したものです。ここで、l_csnとl_rdnがアサートされたときにローカル側がすぐにデータを供給できないため、ローカル側のロジックがクロック-８でl_holdnを２クロック・サイクルの期間アサートしています。そして、ローカル側はクロック-10でデータをl_dat_in[31..0]のバス上に送出し、l_holdnをディアサートしています。pci_aはクロック-11の立ち上がりエッジでデータをラッチし、l_rdnをディアサートしています。l_csnはその１クロック後にディアサートされます。pci_aはローカル側からのデータをラッチした１クロック後（クロック-12）にデータをPCIバス上にドライブします。l_holdnはレジスタを通るようになっているため、ローカル側はl_holdnをドライブする場合に要求されるt_{SU}のタイミング（MAX+PLUS IIのタイミング・アナライザから提供される）を守る必要があります。

レイテンシの超過を防ぐため、PCIの仕様では、framenがアサートされてから、PCIのターゲット・デバイスが16クロック以内に最初のデータ・トランズアクションを完了することが規定されています（ローカル側のロジックもこのPCIの仕様に適合している必要があります）。このため、l_holdnを10クロック・サイクル以上にわたってアクティブにすることはできません。

図7　l_holdnをアサートしたときの外部ターゲット・リード・トランズアクション



ターゲット・ライト・トランズアクション

pci_aは2種類のターゲット・ライト・トランズアクションをサポートしています。

- ***内部ターゲット・ライト***—内部DMAレジスタへのターゲット・ライト
- ***外部ターゲット・ライト***—ローカル側のターゲット・メモリ・スペースへのターゲット・ライト

この2つのターゲット・ライト・トランズアクションのイベント・シーケンスは同一ですが、タイミングは異なります。

内部ターゲット・ライト・トランズアクション

アドレス・フェーズの直後、マスタはframenをディアサート、irdynをアサートして以下のことを表示します。

- トランズアクションがシングル・データ・フェーズであること
- マスタ・デバイスがターゲット・デバイスで受信するデータを、ad[31..0]バスにライトできる状態にあること

マスタ・デバイス側でデータ・フェーズを開始する準備ができていない場合は、irdynが遅延し、irdynがアクティブになるクロック・サイクルまでframenがディアサートされません。マスタがバースト転送を実行する場合は、framenとirdynの双方が継続してアサートされます。ただし、pci_aはターゲット・バースト転送をサポートしていないため、stopnをアサートし

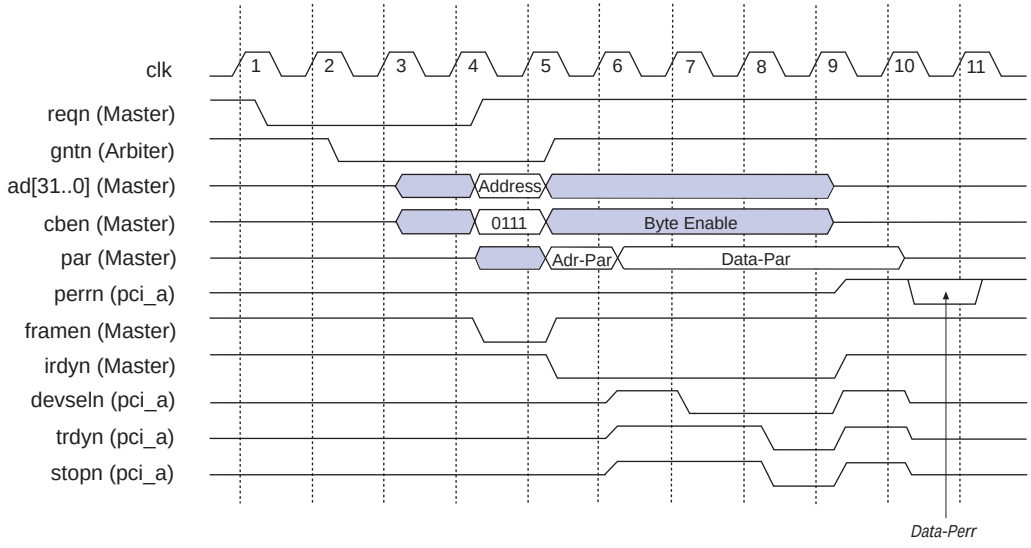
てマスタに対してディスコネクトを通知します。マスタは1クロック・サイクルの期間、framenをディアサート、irdynをアサートして、トランズアクションを終了させます。

図8は内部ターゲット・ライト・トランズクションの標準的なタイミング波形を示したものです。アドレス・フェーズはクロック-4で開始され、データ・フェーズはクロック-5で開始されています。pci_aがクロック-8でdevselnをアサートして、トランズアクションを要求しています。クロック-9の立ち上がりエッジで、irdynとtrdynの双方がアサートされているため、データがマスタ・デバイスからpci_aに転送されます。この図では、pci_aはtrdynをアサートするのと同時にstopnもアサートして、データをさらに受信できないことを通知しています。pci_aは常にstopnとtrdynを同時にアサートして、各ターゲット・トランズアクションに1回のみのデータ・フェーズが発生するようにしています。

マスタ・デバイスはアドレス・ビットのパリティをクロック-5でアクティブにドライブし、クロック-6でデータ・ビットのパリティをアクティブにします。パリティ・エラーが発生した場合は、pci_aが1クロック後にperrnをドライブします。

クロック-9では、データがすでにサンプリングされた状態となっているため、pci_aがad[31..0]とcben[3..0]バスをリリースします。マスタ・デバイスはその1クロック後にparをリリースします。pci_aはdevseln、trdyn、stopnをクロック-9でHighにドライブし、1クロック後にこれらの信号をリリースします。

図8　内部ターゲット・ライト・トランズアクション

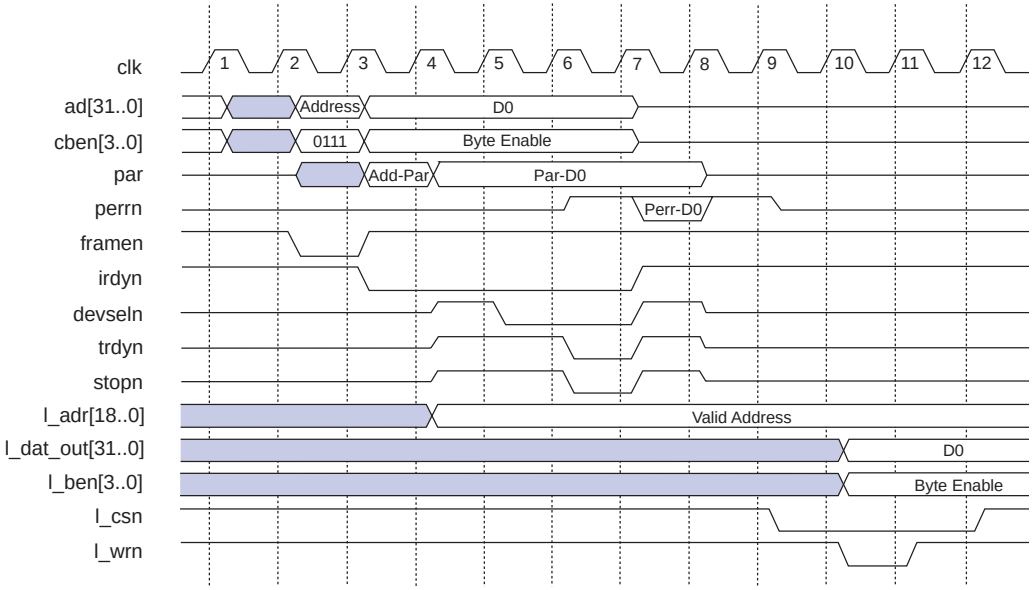


外部ターゲット・ライト・トランズアクション

外部ターゲット・ライト・トランズアクションのイベント・シーケンスは、内部ターゲット・ライト・トランズアクションと同一ですが、タイミングは異なります。

外部ターゲット・ライト・トランズアクションを高速で完了できるようにするため、pci_aにはシングル・アドレスとシングル・データ・ホールディング・レジスタが提供されています。外部ターゲット・ライト・アクセスが行われているとき、pci_aはアドレスとデータを内部ホールディング・レジスタにストアし、PCIバスへの転送を終了させます。そして、pci_aはl_csn信号をアサートしてローカル側にデータ転送がペンディングになっていることを通知します。 そして、その１クロック後（クロック-10）に l_wrnがアサートされ、データがl_dat_out[31..0]のバス上にドライブされて、バイト・イネーブルがl_ben[3..0]バスにドライブされます。図９は外部ターゲット・ライト・トランズアクションのタイミングを示したものです。

図９　外部ターゲット・ライト・トランズアクション



外部ターゲット・リード・トランズアクションの場合と同じように、ローカル側のロジックがl_dat_out[31..0]バスから32ビットのデータを受信できない場合は、l_holdnをアサートしてデータの転送を遅延させることができます。34ページの図10は、l_holdnをアサートすることによって、ローカル側でデータ転送が可能になるまで時間を延長させた場合の外部ターゲット・ライト・トランズアクションのタイミング波形を示しています。

pci_aが、l_csnをLowにドライブすると、その1クロック後にl_wrnがLowにドライブされます。ローカル側のロジックはライト・データを受信できないため、クロック-10でl_holdnをアサートします。

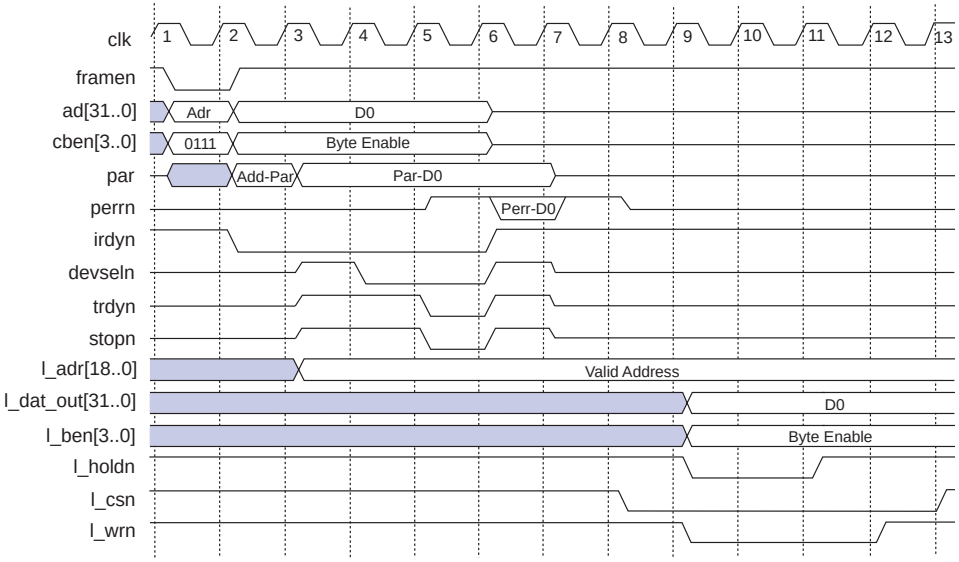
クロック-8でl_csnがアサートされていて、l_rdnがアサートされていない状態になっていれば、ローカル側はローカル・ターゲット・データ転送がライト・サイクルであることを検出することができます。

pci_aは、l_holdnのアサートを検出できるため、l_holdnがディアサートされるまで、l_dat_out[31..0]バス上のdata0（D0）、l_csn、l_wrnを継続してドライブします。ローカル側のアプリケーションはデータ・サイクルの期間を延長させるため、クロック-10までl_holdnをアサートしておく必要があります。

ローカル側のロジックはクロック-13でデータをラッチしています。l_wrnはl_holdnがディアサートされた1クロック後までアサートされ、l_wrnがディアサートされた1クロック後にl_csnがディアサートされています。

pci_aファンクションはデータがローカル側に提供される前にPCIバス上のデータ転送を完了させます。外部ターゲット・ライト・トランズアクションの実行時に、PCIバスの性能に影響を与えることなく、l_holdnを複数の長いクロック・サイクルにわたってアクティブにすることができます。ただし、l_holdnはできるだけ迅速にディアサートしておくことが適切な処理方法となります。そうでないと、pci_aが有効なデータを保持しているときにPCIのエージェントがpci_aを再度アクセスした場合に、pci_aがリトライを発行することになります。

図10　l_holdnを使用した外部ターゲット・ライト・トランズアクション



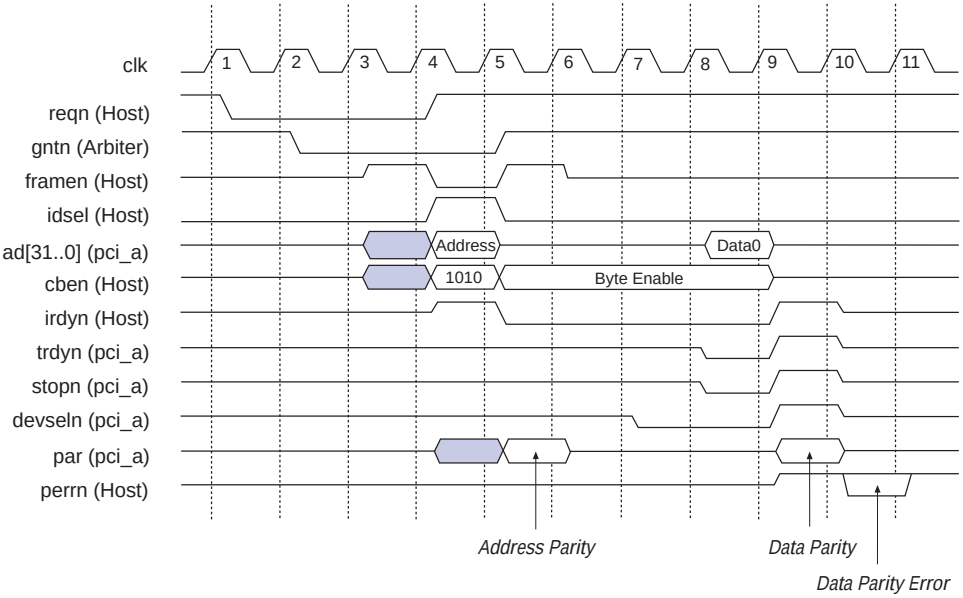
コンフィギュレーション・トランズアクション

コンフィギュレーション・トランズアクションは、ホスト-PCI間のブリッジやPCI-to-PCIのブリッジによって実行されます。コンフィギュレーション・トランズアクションのアドレス・フェーズでは、PCIブリッジがアクセス要求をしているPCIバスのエージェントのidsel信号をドライブします。PCIバスのエージェントがコンフィギュレーション・コマンドをデコードし、idselがHighレベルになっていることを検出した場合、そのエージェントはdevselnをアサートしてコンフィギュレーション・アクセスを要求します。

PCIコンフィギュレーション・リード・トランズアクション

図11は、pci_aコンフィギュレーション・リード・トランズアクションのタイミングを示したものです。このプロトコルは、コンフィギュレーション・トランズアクションのアドレス・フェーズの期間にidsel信号がアクティブになる点を除いて、27ページの「ターゲット・リード・トランズアクション」のプロトコルと同様になります。

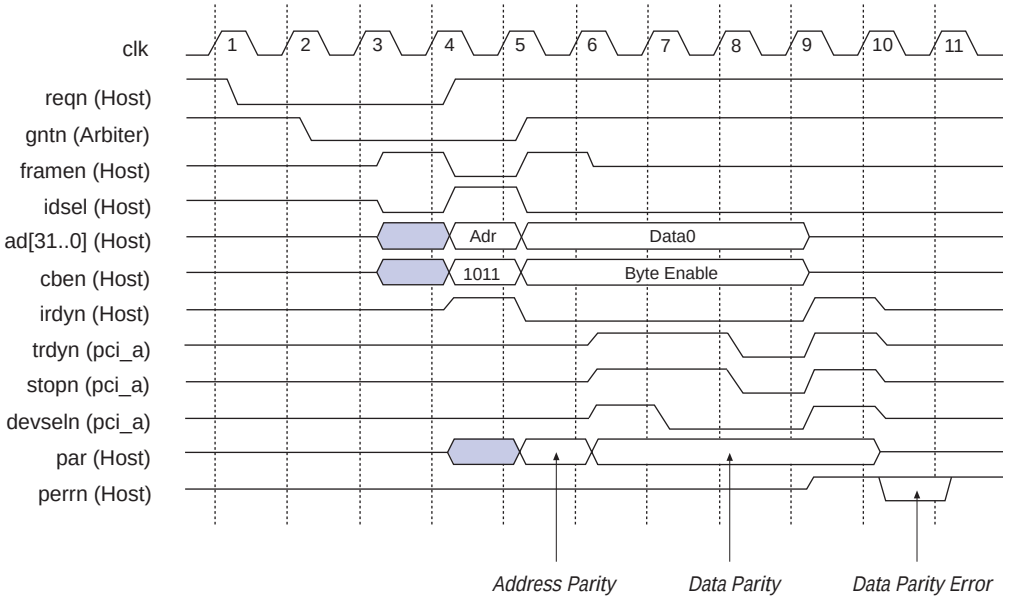
図11　コンフィギュレーション・リード・トランズアクション



PCIコンフィギュレーション・ライト・トランズアクション

図12はpci_aコンフィギュレーション・ライト・トランズアクションのタイミングを示したものです。このプロトコルは、コンフィギュレーション・トランズアクションのアドレス・フェーズの期間に idsel信号がアクティブになる点を除いて、31ページの「ターゲット・ライト・トランズアクション」のプロトコルと同様になります。

図12　コンフィギュレーション・ライト・トランズアクション



マスタ・トランズアクション

pci_aのマスタ・トランズアクションは、DMAエンジンによってコントロールされます。pci_aのマスタ・トランズアクションは、ユーザがDMAレジスタに適切な値をロードした後に開始されます（DMAレジスタ・ローディングに関する詳細は、55ページの「一般的なプログラミング・ガイドライン」を参照してください）。pci_aはローカル側がDMA動作の開始可能を表示するl_reqがアサートされるのを待ちます。

DMAリード（PCIからローカル側へ）のトランズアクションでは、pci_aがすぐにreqnをアサートして、PCIバスのマスタの権利を獲得します。アービタがgntnをアサートした後、pci_aがframenをアサートし、ad[31..0]にアドレスを、cben[3..0]バスにコマンドをドライブして、アドレス・フェーズを開始します。

DMAライト(ローカル側からPCIへ)のトランズアクションでは、pci_aがローカル側から最初の16 DWORDを読み出し、これを内部のRAMバッファにストアします。 この時点で、DMAはregnをアサートしてPCIバスのマスタの権利を獲得します。アービタがgntnをアサートした後で、pci_aはアドレス・フェーズを開始します。

マスタ・リード・トランズアクション

pci_aは以下の２種類のマスタ・リード・トランズアクションをサポートしています。

- シングル・サイクル・マスタ・リード
- マスタ・バースト・リード

シングル・サイクル・マスタ・リード・トランズアクション

マスタ・リード・トランズアクションでは、データがPCI側からローカル側へ転送されます。pci_aがPCIバスのマスタの権利を獲得した場合、pci_aがframenをアサートすることによって、マスタ・リード・トランズアクションの開始が表示されます。

マスタ・リード・トランズアクションが開始された後、framenがアクティブとなってアドレス・デコードが開始されると、ターゲット・デバイスはクロック・エッジでアドレスとコマンドをラッチします。pci_aはクロック-5までデータをリードできる状態にならないため、クロック-5までframenがディアサートされず、irdynはアサートされません。

選択されたターゲット・デバイスは、クロック-3でdevselnをアサートします。devselnはクロック-4の立ち上がりエッジでpci_aによってサンプリングされます。これは、高速デコード・ターゲット・デバイスの動作ということになります。

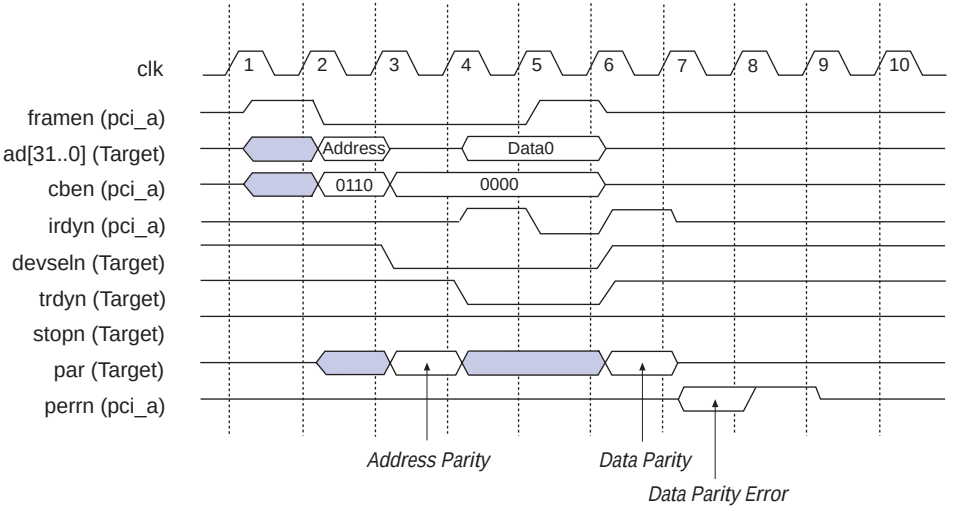
ターゲット・デバイスはデータを送出できる状態になっていることを示すため、クロック-４でtrdynのアサートとad[31..0]へのデータのドライブを同時に行います。データ・フェーズはirdynとtrdynが共にアクティブのときにクロック-５で開始され、クロック-６の立ち上がりエッジでpci_aにデータがラッチされ、データ・フェーズが終了します。

pci_aはクロック-３でparをアクティブにし、アドレスとコマンド・ビットのパリティを表示します。選択されたターゲットは、クロック-６でparをアクティブにし、データとバイト・イネーブル・ビットのパリティを表示します。

pci_aはクロック-３でad[31..0]をリリースし、cben[3..0]バスをクロック-６で、par信号をクロック-４で、それぞれリリースします。

図13は、pci_aマスタ・リード・トランズアクションのタイミングを示したものです。この図は、pci_aがすでにPCIバス上のマスタの権利を獲得している状態を想定したものとなっています。

図13　シングル・サイクル・マスタ・リード・トランズアクション



マスタ・バースト・リード・トランズアクション

マスタ・バースト・リード・トランズアクションのアドレス・フェーズ用プロトコルは、37ページの「シングル・サイクル・マスタ・リード・トランズアクション」と同一です。ただし、このプロトコルはアドレス・フェーズ後に実行される追加のリード・トランズアクションに応じて変化します。

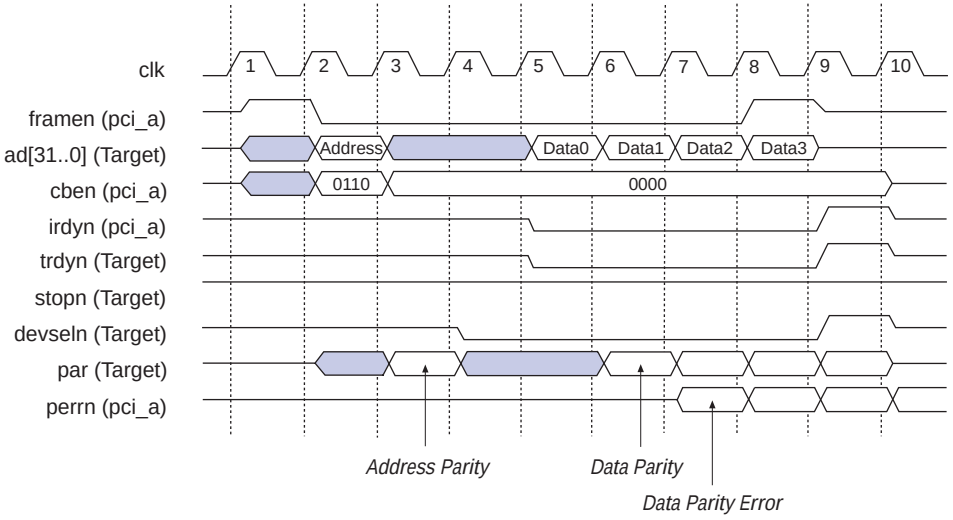
マスタ・バースト・リード・トランズアクションが開始された後、選択されたターゲット・デバイスはクロック-３でdevselnをアサートし、pci_aはクロック-５の立ち上がりエッジでdevselnをサンプリングします。この例は、高速デコード・ターゲットの場合を想定しています。次に、ターゲット・デバイスはクロック-４でtrdynをドライブし、ad[31..0]バスをアクティブにすることによって、pci_aにデータの送信が可能であることを通知します。

pci_aはクロック-３でparをアクティブにし、アドレスとコマンド・ビットのパリティを表示します。クロック-６で、ターゲット・デバイスはparをアクティブにドライブし、最初のデータ・フェーズ（データ-0）のパリティを表示します。同様に、ターゲット・デバイスは、クロック-７、-８、-９でもparをアクティブにし、２、３、４番目のデータ・フェーズのパリティを表示します。

図14は、４つの連続したクロック・サイクルでデータ・フェーズが発生する16バイトのデータ・トランズアクションを示したものです。データ・フェーズはクロック-5で開始され、クロック-8で終了しており、pci_aがframenをリリースするクロック-８が最終のデータ・フェーズの開始であることが表示されます。

データがリードされた状態になっているため、クロック-9でpci_aがirdyn をリリースするときに、ターゲット・デバイスがdevseln、trdyn、ad[31..0] バスを同時にリリースします。

図14　マスタ・バースト・リード・トランズアクション



マスタ・ライト・トランズアクション

pci_aは以下の2種類のマスタ・ライト・トランズアクションをサポートしています。

- シングル・サイクル・マスタ・ライト
- マスタ・バースト・ライト

シングル・サイクル・マスタ・ライト・トランズアクション

マスタ・ライト・トランズアクションでは、データがローカル側からPCI側へ転送されます。pci_aがPCIバスのマスタの権利を獲得した場合、pci_aはframenをアサートして、マスタ・デバイスのライト・トランズアクションの開始を表示します。

マスタ・デバイス・ライト・トランズアクションが開始された後、framenがアクティブとなりアドレス・デコードが開始されるクロック・エッジで、ターゲット・デバイスがアドレスとコマンドをラッチします。pci_aのマスタ・デバイス・ライト・トランズアクションからのデータはクロック-5まで確定しないため、クロック-5までの期間にframenはディアサートされず、irdynはアサートされません。

選択されたターゲット・デバイスは、クロック-４でdevselnをアサートし、クロック-5でpci_aにサンプリングされます。これは、中速のデコード・ターゲット・デバイスの動作ということになります。

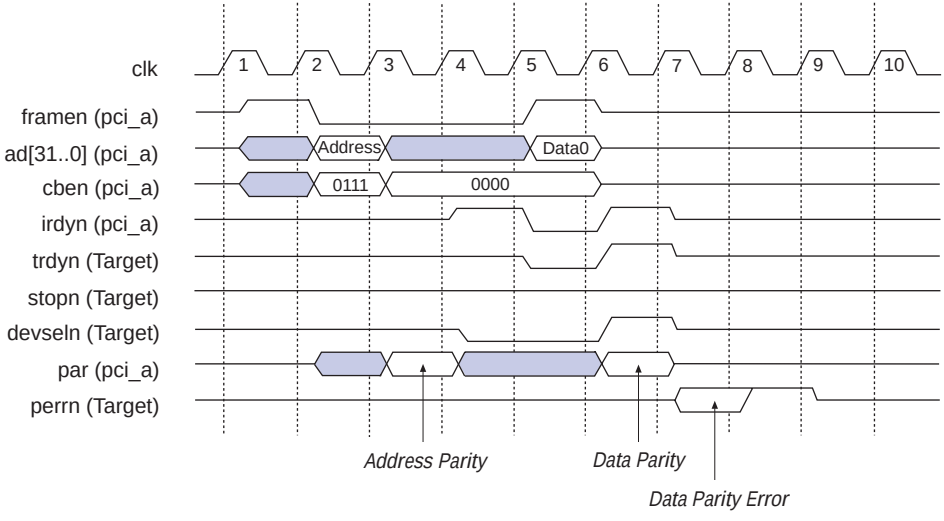
ターゲット・デバイスはクロック-5でtrdynをアクティブにドライブし、データを受信できる状態になったことを表示します。次に、pci_aはクロック-５の立ち上がりでad[31..0]バスにデータをドライブし、同時にirdynをアサートします。irdynとtrdynが共にアクティブであれば、クロック-５でデータ・フェーズが開始されます。そして、クロック-６の立ち上がりエッジで、データが選択されたターゲット・デバイスにラッチされ、データ・フェーズが終了します。

pci_aはクロック-３でparをアクティブにし、アドレスとコマンド・ビットのパリティを表示します。また、クロック-６でparをアクティブにドライブし、データとバイト・イネーブル・ビットのパリティを表示します。

データ・フェーズが終了しているため、pci_aはクロック-６でad[31..0]バスとcben[3..0]をリリースします。その１クロック後に、pci_aはparをリリースし、ターゲット・デバイスがdevselnとtrdynをリリースします。また、サステインド・トライ・ステート信号はリリースされる前の１クロックの期間にHighにドライブされる必要があるため、pci_aがirdynをクロック-７でリリースする前に、クロック-６でirdynをHighレベルにドライブしています。

図15は、pci_aマスタ・ライト・トランズアクションのタイミングを示したものです。この図は、pci_aがPCIバス上のマスタの権利を獲得している状態を想定したものとなっています。

図15　シングル・サイクル・マスタ・ライト・トランズアクション



マスタ・バースト・ライト・トランズアクション

マスタ・バースト・ライト・トランズアクションにおけるアドレス・フェーズからデータ・フェーズ-１までのプロトコルは、39ページに述べた「シングル・サイクル・マスタ・ライト・トランズアクション」と同一です。ただし、データ・フェーズ-２以降のプロトコルは、実行される追加のライト・トランズアクションに応じて変化します。

マスタ・バースト・ライト・トランズアクションが開始された後、選択されたターゲット・デバイスがクロック-４でdevselnをアサートし、pci_aがクロック-５の立ち上がりエッジでdevselnをサンプリングします。この例は中速デコードのターゲットを想定したものとなっています。次に、ターゲット・デバイスはクロック-５でtrdynをドライブし、マスタ・デバイスに対してデータの受信が可能になっていることを通知します。

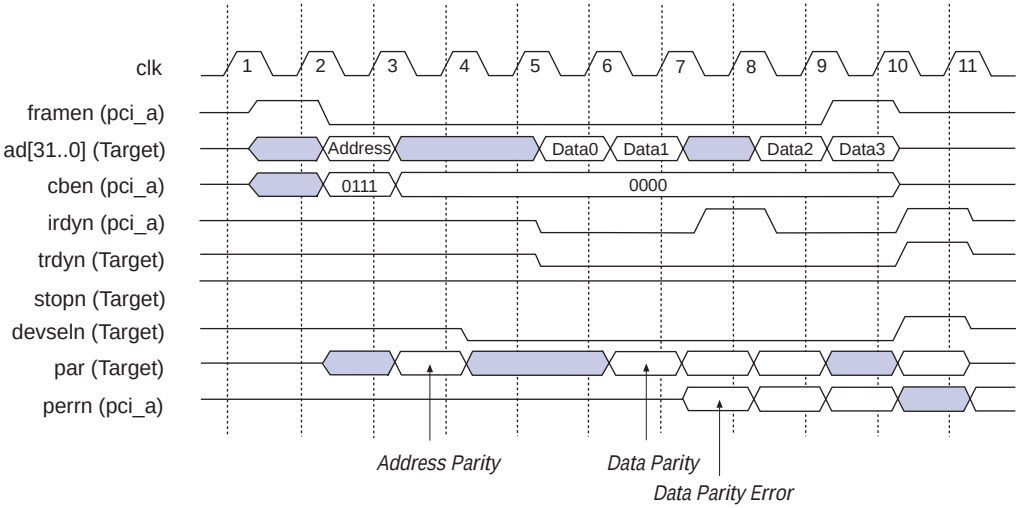
図16は、マスタ・バースト・ライト・トランズアクションの例を示したものです。この例では、irdynとtrdynが共にアクティブのとき、クロック-５、-６、-７、 -９でデータ・フェーズが発生しています。

pci_aの内部データ・パスのパイプラインにおけるデータの同期化を確実に行うため、クロック-８でマスタ・バースト・ライト・トランズアクションにウェイト・ステートが挿入されています。バースト・ライト・トランズアクションで、ターゲット・デバイスがウェイト・ステートを挿入しない場合は、pci_aがバースト・トランズアクション全体に対してウェイト・ステートを１つだけ挿入します。ただし、ターゲットがバースト・ライト・トランズアクション中にウェイト・ステートを挿入した場合でも、pci_aは追加のウェイト・ステートを挿入します。pci_aがクロック-９でframenのディアサートとirdynのアサートを同時に行うと、最終データが転送されます。

pci_aはクロック-３でpar信号をアクティブにドライブし、アドレス・ビットのパリティを表示し、クロック-６でデータ・ビットのパリティを表示します。

図16は、16バイト・データ転送を行った場合を想定したpci_aのバースト・ライト・トランズアクションのタイミングを示したものです。

図16　マスタ・バースト・ライト・トランズアクション



DMA動作

このセクションでは、DMAエンジンの詳細を以下のサブ・セクションごとに解説します。

- ターゲット・アドレス・スペース
- 内部ターゲット・レジスタのメモリ・マップ
- DMAレジスタ
- DMAトランズアクション
- ローカル側からのDMA転送のイニシャライズ
- 一般的なホスト・プログラミング・ガイドライン

ターゲット・アドレス・スペース

pci_aのメモリ・マップ・ターゲット・レジスタ（内部および外部）の内容は、PCIバスを通してBAR0メモリ・スペースにリードまたはライトされます。BAR0のメモリ・スペースに対するアクセスは、32ビットの転送となります。表24は、pci_aのメモリ・スペース・アドレス・マップを示したものです。pci_aのBAR0のアドレス・スペースは、２つの同じサイズの領域（下位および上位）に分割された1Mバイトから２Gバイトの連続したアドレスとなっています。それぞれの領域はBAR0にリザーブされているトータルなアドレス・スペースの半分を使用します。下位の領域（内部ターゲット・アドレス・スペース）にはpci_a のDMAコントロール・レジスタが含まれ、上位の領域（外部ターゲット・アドレス・スペース）には、ユーザ定義用のメモリ・スペースが含まれます。

表24　メモリ・スペース・アドレス・マップ

メモリ・スペース	ブロック・サイズ (DWORDs)	アドレス・オフセット *注(1)*	使用ワード	リード／ライト	概　要
BAR0	予約された領域の半分	00000h-7FFFFh	４バイト	リード／ライト	DMAレジスタ
BAR0	予約された領域の半分	80000h-FFFFFh	全部	リード／ライト	512Kバイトから2Gバイトまでの範囲のユーザ定義によるメモリ・スペース

注：
(1)　これらの値は、BAR0_RW_BITSパラメータを12に設定した場合のものとなっています。

内部ターゲット・レジスタのメモリ・マップ

pci_aの内部ターゲット・アドレス・スペースは、 DMAコントロール／ステータス・レジスタ、DMAアドレス・カウンタ・レジスタ、DMAバイト・カウンタ・レジスタ、インタラプト・ステータス・レジスタを含むDMAレジスタとして使用されます。表25はpci_a のDMAレジスタのメモリ・マップを示したものです。

表25　内部ターゲット・レジスタのメモリ・マップ

予約範囲 *注(1)*	使用バイト／予約バイト数	リード／ライト	ニーモニック	デフォルト・ステート (Hex)	レジスタ名
00000h-00003h	8/32	リード／ライト	dma_csr	00000000	DMAコントロール／ステータス
00004h-00007h	32/32	リード／ライト	dma_acr	00000000	DMAアドレス・カウンタ
00008h-0000Bh	17/32	リード／ライト	dma_bcr	00000000	DMAバイト・カウンタ
0000Ch-0000Fh	8/32	リード	dma_isr	00000000	DMAインタラプト・ステータス

注：
(1)　これらの値は、BAR0_RW_BITSパラメータを12に設定した場合のものとなっています。

DMAレジスタ

このセクションではDMAレジスタについて解説します。PCIバスがリセットされたときの、各ストレージ・エレメントのステートが、デフォルト・ステートとして定義されています。pci_aには以下に示すDMAレジスタが内蔵されています。

- コントロール／ステータス
- アドレス・カウンタ
- バイト・カウンタ
- インタラプト・ステータス

コントロール／ステータス・レジスタ（オフセット = 00000 Hex）

DMAコントロール／ステータス・レジスタ(dma_csr)は、pci_aのDMAエンジンの構成、pci_a のDMA動作のコントロール、処理中のデータ転送のステータスを表示するためのものです。表26を参照してください。

表26　DMAコントロール／ステータス・レジスタのフォーマット　(1/2)

データ・ビット	ニーモニック	リード／ライト	定　義
0	int_ena	リード／ライト	PCIインタラプト・イネーブル。err_pendまたはdma_tcビットがdma_isrからHighレベルにドライブされるか、l_irqn信号がアクティブのとき、このビットがintan出力をイネーブルにする。
1	flush	ライト	フラッシュ・バッファ。この値がHighに設定されると、dma_tcとad_loaded（インタラプト・ステータス・レジスタの3ビット目と4ビット目）がリセットされ、内部のEABのRAMキュー内のすべてのバイトが無効になる。flushビットは自分自身でリセットするため、常に0がリードされる。 dma_onビットがセットされるDMA転送の実行中には、flushビットをセットすることはできない。
2	l_rst	リード／ライト	ローカル・リセット。このビットは、ローカル側のアドオン・ロジックに対するソフトウェア・リセットとして用意されている（10ページの“ローカル側の信号”を参照）。pci_aのl_reset出力は、l_rstのビットがHighである限りアクティブとなる（l_reset出力はPCIバス・リセットでもアクティブになる）。
3	write	リード／ライト	メモリ・リード／ライト。pci_a DMA転送の方向を決定するビット。このビットがHighであれば、データはPCIバス（PCIバス・ライト）からローカル・メモリに転送される。Lowであれば、データはローカル・デバイスからPCIバス（PCIバス・リード）に転送される。
4	dma_ena	リード／ライト	DMAイネーブル。このビットがHighであれば、PCIバス動作がインタラプトなどによるペンディングで停止されない限り、pci_aはローカル側(l_req)からのDMAリクエストに応答することができる。
5	tci_dis	リード／ライト	転送完了インタラプト・ディセーブル。このビットがHighになると、tci_disがPCIバス・インタラプトの発生からdma_tc（DMAインタラプト・ステータス・レジスタの3ビット目）をディセーブルにする。

表26　DMAコントロール／ステータス・レジスタのフォーマット　(2/2)			
データ・ビット	ニーモニック	リード／ライト	定　義
6	dma_on	リード	DMAオン。このビットがHighになると、pci_aがローカル側から要求されたときに、PCIバス上のマスタの権利を要求することができる状態（reqn）となっていることを示す。アドレスがロードされると（ad_loaded）、dma_onビットはHighとなり、DMAがイネーブルになって、ペンディング・エラーがないことが表示される。 DMA転送のシーケンスは、このdma_onビットがセットされて開始される。通常の条件（DMAがイネーブルにされ、エラー・ペンディングがない状態）では、DMAアドレス・カウンタ・レジスタに対するライト・トランズアクションが発生したときに、dma_onビットがセットされる。このdma_onのビットはローカル側からのライト・トランズアクション、またはターゲット・アクセスが発生したときにセットされる。
31..7	未使用	–	–

アドレス・カウンタ・レジスタ（オフセット = 00004 Hex）

DMAアドレス・カウンタ・レジスタ(dma_acr)は32ビット・レジスタで、30ビットのカウンタ（31から2ビット目まで）とGNDに接続された2ビット（1から0ビット目まで）によって構成されています。このdma_acrには処理されているメモリ転送のPCIバス・アドレスが含まれ、PCIバス上のデータ転送ごとにカウンタがインクリメントされます。pci_aにより開始されたPCIバスのメモリ転送は、DWORDで指定された境界で開始される必要があります。メモリ転送中にl_dma_acr_out[]ポートを介してdma_acrをリードすることによって、処理の状況をモニタすることができます。表27を参照してください。

表27　DMAアドレス・カウンタ・レジスタのフォーマット			
データ・ビット	名　称	リード／ライト	定　義
1..0	dma_acr	リード	GNDに接続されるビット
31..2	dma_acr	リード／ライト	30ビット・カウンタ

バイト・カウンタ・レジスタ（オフセット = 00008 Hex）

DMAバイト・カウンタ・レジスタ(dma_bcr)は17ビット・レジスタで、15ビット・カウンタ（16ビットから2ビット目まで）とGNDに接続されている2ビット（1と0ビット目）によって構成されています。このdma_bcrはpci_aにより開始されたメモリ転送用のバイト・カウントを保持し、PCIバス上での各データ転送後、4バイトごとにディクリメントされます。pci_aによって開始されたPCIバス上のメモリ転送は、DWORDで指定されたバイト数となっている必要があります。dma_bcrはメモリ転送中にリード可能となっており、l_dma_bcr_out[] のポートを介して処理状況のモニタに利用することができます。表28を参照してください。

表28　DMAバイト・カウンタ・レジスタのフォーマット			
データ・ビット	名　称	リード／ライト	定　義
1..0	byte_cntr	リード	GNDに接続されるビット
16..2	byte_cntr	リード／ライト	15ビット・カウンタ
31..17	未使用	–	–

インタラプト・ステータス・レジスタ（オフセット = 0000C Hex）

DMAインタラプト・ステータス・レジスタ（dma_isr）は、インタラプトを処理するデバイスにすべてのインタラプト・ソース・ステータス信号を供給します。表29を参照してください。

表29　DMAインタラプト・ステータス・レジスタのフォーマット			
データ・ビット	ニーモニック	リード／ライト	定　義
0	int_pend	リード	pci_aはインタラプトがペンディングになっていることを表示するとき、このint_pendを自動的にアサートする。pci_aからのインタラプト信号には、err_pend、dma_tc、int_irqの3つの場合がある。
1	err_pend	リード	このビットがHighになっていると、pci_aによって開始されたPCIバス・メモリ転送でエラーが発生したことが表示され、またインタラプトを処理するデバイスがPCIコンフィギュレーション・ステータス・レジスタをリードし、適切なビットをクリアしなければならないことが表示される。PCIステータス・レジスタ・ビット（mstr_abrt、 tar_abrt、 det_par_err）のいずれかが、err_pendをアサートする。44ページの「コントロール／ステータス・レジスタ（オフセット = 00000 Hex）」を参照。
2	int_irq	リード	このビットがHighであれば、ローカル側がインタラプトをリクエストしており、l_irqnがアサートされていることを示す。
3	dma_tc	リード	このビットがHighであれば、pci_aによって開始されたDMA転送が完了したことが示される。pci_aによってこのdma_tcビットがセットされると、インタラプトがint_enaビット（dma_csrのビット-0）によってイネーブルされていてtci_disビット（dma_csrのビット-5）によってディセーブルされない限り、intan出力にインタラプトが発生する。このビットは以下の3つの方法のいずれかでリセットされる。dma_isrへのリード・トランズアクション； flushビット（dma_csrのビット-1）をセットするdma_csrへのライト・トランズアクション；ローカル側からのdma_acrへのライト。
4	ad_loaded	リード	このビットがHighになれば、dma_acrを経由してアドレスがロードされたことが表示される。このビットは、以下の3つの方法のいずれかでクリアされる。DMAオペレーションが終了してdma_tcビットがセットされたとき；flushビットがセットされたとき；rstn入力がPCIバスからアサートされたとき。ad_loadedビットはdma_acrレジスタのdma_onビットをセットするため、これがDMA動作を開始させるトリガとなる。dma_acrへのライト動作が実行されるとき、pci_aによってこのビットが自動的にセットされる。このため、dma_acrは、DMAの動作がDMAレジスタにロードされる最後にライトされる必要がある。
31..5	未使用	–	–

DMAトランズアクション

pci_aは、マスタ・デバイスとしてシステム・メモリ（通常はホスト・ブリッジを経由）またはバースト・ターゲット・データ転送機能を持っている他のPCIバス・エージェントに対するDMAリードおよびライト・トランズアクションを実行します。

メモリからローカル側へのDMAリード・トランズアクションは、次の２つの異なる転送で構成されています。

- PCIバスからRAMバッファへのPCIバス・バースト・リード
- DWORDと同数のローカル側への転送

pci_aから実行されるすべてのDMAリード・トランズアクションには、メモリ・リード・コマンドが使用されます。

同様に、pci_aからのDMAライト・トランズアクションは、次の２つの異なる転送で構成されています。

- ローカル側からRAMバッファへの１から16 DWORDの転送
- RAMバッファからPCIエージェントへのPCIバースト・ライト

pci_aから実行されるすべてのDMA（PCIバス）ライト・トランズアクションには、メモリ・ライト・コマンドが使用されます。

PCIバスのDMAリード・トランズアクションと信号のシーケンス

PCIバス内部のDMAリード・トランズアクションでは、データがシステム・メモリからローカル側のバッファに転送されます。すなわち、PCIバスのDMAリードは以下の動作で構成されます。

- マスタ・デバイスであるpci_aが、PCIのエージェントからデータをリードしてpci_aのRAMバッファに転送する。
- pci_aのRAMバッファのデータをローカル側の周辺デバイスにライトする。

PCIバスDMAリード・トランズアクションにおける信号のシーケンスを以下に示します。

1. ホストはDMAレジスタに適切な値をライトし、DMAリード転送をセットアップします。dma_onのビットがセットされたときに、実際のDMA転送のシーケンスが開始されます。通常の条件（DMAがイネーブルになっていて、エラーがペンディングされていない状態）では、DMAアドレス・カウンタ・レジスタに対するライト・トランズアクションが発生したときに、dma_onのビットがセットされます。

2. ローカル側の周辺デバイスがl_reqをアサートし、DMA転送を要求します。

3. pci_aはPCIバスのマスタの権利を獲得する前に、reqnをアサートして、gntnがアクティブになるのを待ちます。

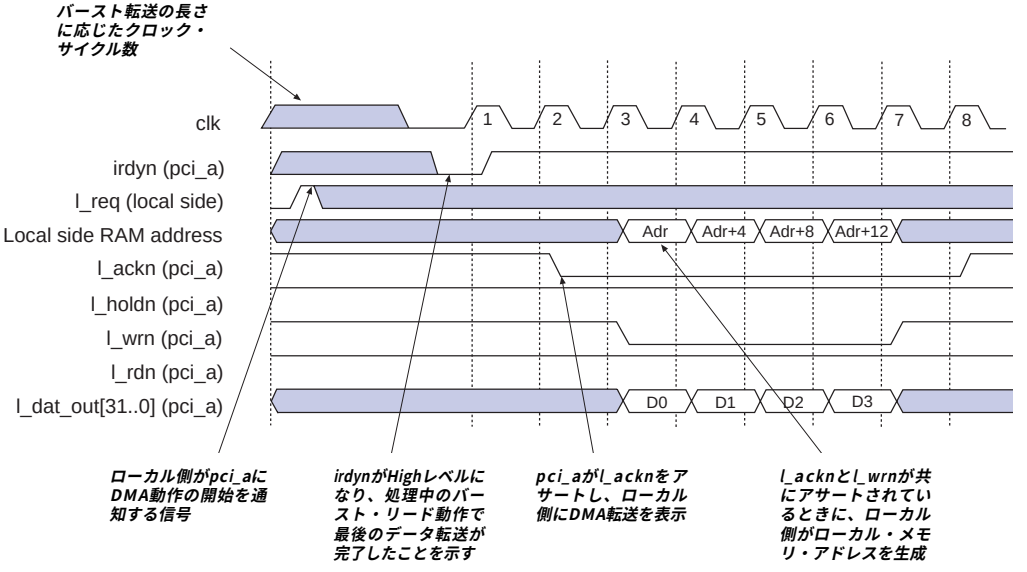
4. pci_aはPCIバス・システム・メモリから16 DWORDまでをリードし、pci_aのRAMバッファにそのデータをロードします。

5. PCI転送が終了した場合は、pci_aがローカル側の周辺デバイスに対してl_acknとl_wrnをアサートして、16 DWORDまでの転送を行います。pci_aはデータがライトされるローカル側アドレス・ロケーションを持っていないため、ローカル側のDMA転送中にローカル側でアドレスを生成しなければなりません。図17では、pci_a側からアドレスが生成されていません。

6. pci_aはl_dat_out[31..0]上にpci_aのRAMバッファからのデータをライトします。最後のデータ・ワードがライトされると、pci_aはl_acknとl_wrnをディセーブルします。

7. dma_bcrのカウントが終了すると（規定されたデータ・バイト数が転送されたとき）、pci_aがdma_isrレジスタのdma_tcのビットをセットし、intanをアサートし、インタラプトがイネーブルとなって、tci_dis=0の状態となります。それ以外の場合は、dma_bcrが無効になるか、またはDMAエラーが発生するまで、ステップ-2から-5までの処理が繰り返されます。図17を参照してください。

図17　PCIバスDMAリード・トランズアクション



PCIバスDMAライト・トランズアクションと信号のシーケンス

PCIバス内部のDMAライト・トランズアクションでは、データがローカル側からシステム・メモリに転送されます。PCIバス上のDMAライトは、以下の動作で構成されます。

- ローカル側からpci_aのRAMバッファへデータを転送する
- マスタであるpci_aが、pci_aのRAMバッファからデータをPCIバス・エージェントにライトする

PCIバスDMAライト・トランズアクションの信号シーケンスを以下に示します。

1. ホストまたはローカル側がDMAレジスタに適切な値をライトして、DMAライト転送をセットアップします。dma_onビットがセットされたときに、実際のDMA転送シーケンスが開始されます。通常の条件（DMAがイネーブルになり、エラーがペンディングされていない状態）では、DMAアドレス・カウンタ・レジスタに対するライト・トランズアクションが発生したときに、dma_onビットがセットされます。

2. ローカル側周辺デバイスがl_reqをアサートし、DMA転送を要求します。

3. pci_aがl_acknとl_rdnをアサートし、DMAリクエストに応答します。そして、ローカル側の周辺デバイスから16 DWORDまでのデータをラッチします。

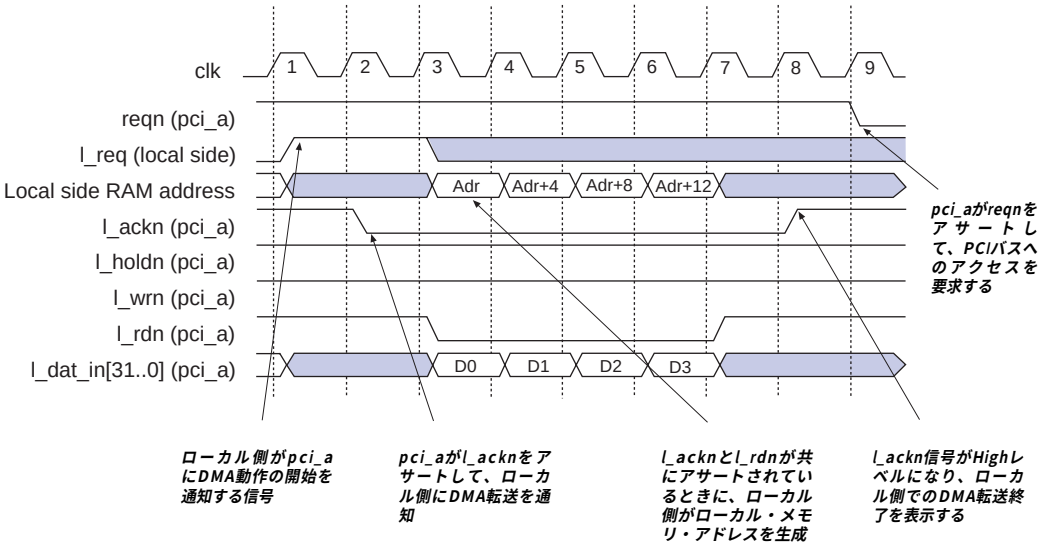
4. pci_aはpci_a RAMバッファのl_dat_in[31..0]からデータをリードします。 pci_aは、DMA転送の最後のDWORDがリードされたとき、あるいはRAMバッファがフル状態の場合に、l_acknとl_rdnをディセーブルします。

5. pci_aはPCIバスのマスタの権利を獲得する前に、reqnをアサートして、gntnがアクティブになるのを待ちます。

6. pci_aはRAMバッファからPCIバス・ターゲット・デバイスへの16 DWORDまでの転送を行います。

7. dma_bcrのカウントが終了すると（規定されたデータ・バイト数が転送されたとき）、pci_aがdma_isrレジスタのdma_tcのビットをセットし、intanをアサートし、インタラプトがイネーブルとなって、tci_dis＝0の状態となります。 それ以外の場合はdma_bcrが無効になるか、またはDMAエラーが発生するまで、ステップ-2から-5までの処理が繰り返されます。図18を参照してください。

図18　PCIバスDMAライト・トランズアクション



ローカル側からのDMA転送のイニシャライズ

pci_aファンクションのバージョン2.0では、ホストとローカル側の双方がDMAリード・トランズアクションを実行できるようになっています。このセクションでは、ローカル側がマスタ転送を開始させるときにDMAレジスタをどのようにセットアップするかを解説します。ホストがDMAを開始させる場合の方法については、55ページの「一般的なホスト・プログラミング・ガイドライン」をご覧ください。

pci_aのDMAエンジンは64バイトのRAMバッファと４個プログラマブル・レジスタを持っており、pci_aがPCIバスのマスタの権利を獲得するときのコントロール・チャネルとなっています。

コンフィギュレーション・スペース・レジスタが適切に設定された後、ホストまたはローカル側のロジックがpci_aのDMAレジスタに書き込みを行うことで、バーストDMA転送が開始されます。このセクションは以下の２つの項目に分割されています。

- DMAリード・トランズアクションに対するpci_aのイニシャライズ
- DMAライト・トランズアクションに対するpci_aのイニシャライズ

DMAリード・トランズアクションに対するpci_aのイニシャライズ

DMAリード・サイクルをイニシャライズするときは、ローカル側のロジックが、dma_csr、dma_bcr、およびdma_acrの各レジスタに対してシーケンシャルに書き込みを行います。ローカル側のロジックによるdma_acrに対する書き込みが行われた後、dma_isrレジスタのad_loadedのビットがセットされます。 DMAがイネーブルとなっていて（dma_csrの４ビット目）、ペンディングになっているエラーがなければ（dma_isrの１ビット目）、ad_loadedのビットがdma_csrレジスタのdma_onのビットをセットします。このdma_onのビットがセットされると、pci_aはPCIバスのマスタの権利を要求し、DMAリード・トランズアクションを開始する前に、ローカル側のデバイスがl_reqをアサートするのを待ちます。この場合、dma_bcrとdma_csrのレジスタに適切な値が設定された後、最後にdma_acrへの書き込みが行われているかをチェックすることが重要です。表30を参照してください。

表30　DMAリード動作に対するpci_aのイニシャライズ

アドレス（16進）	レジスタ名	データ（16進）	定　義
BAR0: 0.0000	dma_csr	0000.0031	dma_csrの値がインタラプトとDMAエンジンをイネーブルにし、DMAターミナル・カウント・インタラプトをディセーブルする。
BAR0: 0.0008	dma_bcr	00084	このレジスタに書き込まれた値がDMA転送のデータ・サイズ（バイト数）を示す。この値はDWORDの倍数となっている必要がある。
BAR0: 0.0004	dma_acr	00400000	PCIバスの転送が開始される位置を示すアドレス。このアドレスは各データ転送の完了後に自動的にアップデートされる。

52ページの図19はローカル側のDMAリード・トランズアクションのタイミング波形を示したものです。この例では、ローカル側のロジックが00400000（hex）のアドレスから開始されるシステム・メモリから33 DWORD（132バイト）のリードを要求しています。図19では、以下の信号のシーケンスが示されています。

1. ローカル側のロジックがクロック-１でl_reqをアサートし、転送が開始できることを示します。l_reqのアサートは、ローカル側がDMA転送を開始できる状態になるまで遅延させることができます。

2. クロック-2 で、ローカル側のロジックがl_dma_dat_in[31..0] のバス上にデータを供給すると共に、l_dma_csr_wrをアサートします。16進の31という値はDMAコントロール・アンド・ステータス・レジスタのビット-0、-4および-5がセットされ、DMAとインタラプトがイネーブルにされ、DMAターミナル・カウント・インタラプトがディセーブルにされていることを示しています。この場合、ビット-3がセットされておらず、DMAリード転送であることが示されています。

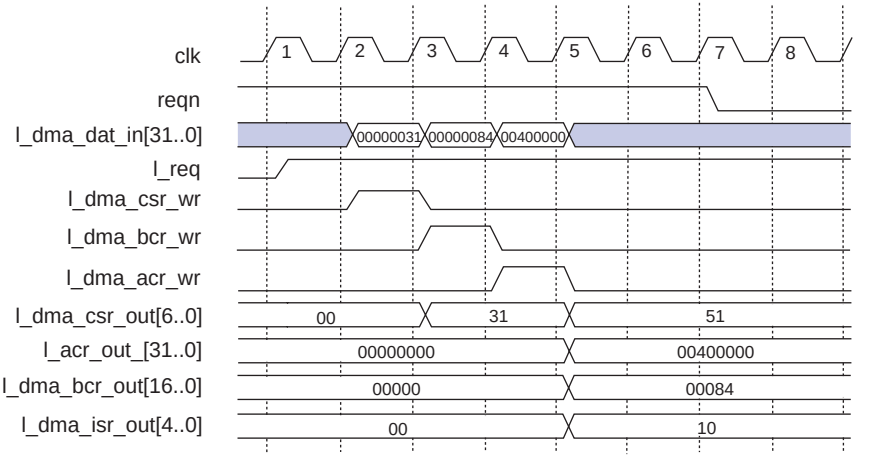
3. クロック-3で、ローカル側のロジックがl_dma_dat_in[31..0]バスにdma_bcrレジスタに対するデータを供給すると共に、l_dma_bcr_wrをアサートします。16進の84の値は10進の132バイトを示しており、pci_aが33 DWORDのリード動作を行うことを示しています。l_dma_csr_out[6..0]の値はクロック-2で書き込まれる値に変更されるため、dma_csrレジスタに対する書き込み動作はクロック-3で有効となります。

4. ローカル側のロジックはl_dma_dat_in[31..0]バスにdma_acrレジスタに対するデータを供給すると共に、l_dma_acr_wrをアサートします。このトランズアクションでは、dma_acrレジスタに00400000（hex）の値が書き込まれています。pci_aは00400000（hex）のアドレスからのリードを行うとします。

5. クロック-5では、dma_bcrとdma_acrのレジスタに対する書き込みが有効になります。図19では、l_dma_bcr_out[16..0]とl_dma_acr_out[31..0]のバス上の値が変化していることが示されています。また、図19では、l_dma_isr_out[4..0]とl_dma_csr_out[6..0]のバス上の値も変化していることが示されています。これは、ad_loadedとdma_onのビットがセットされていたためです。

6. l_reqはすでにアサートされているため、pci_aはreqn信号をクロック-7でアサートしてPCIバスのマスタの権利を獲得しようとします。図19を参照してください。

図19　ローカル側が開始したDMAリード・トランズアクション



DMAライト・トランズアクションに対するpci_aのイニシャライズ

ローカル側のロジックからバースト・ライト・トランズアクションを行う場合のDMAレジスタの設定方法は、DMAリード・トランズアクションの場合と同じです。ローカル側のロジックは、dma_csr、dma_bcrおよびdma_acrの各レジスタに対するシーケンシャルな書き込み動作を行います。ローカル側のロジックによるdma_csr、dma_bcr、dma_acrの各レジスタに対する書き込みが行われた後、ad_loadedのビット（ dma_isrレジスタの４ビット目）がセットされます。ad_loadedのビットはdma_onのビット（dma_csrの４ビット目）をセットしてDMA動作の開始をトリガします。これによって、pci_aはl_acknをアサートし、ローカル側から16 DWORDのデータを読み込んでDMAライト動作を開始します。この場合、dma_bcrとdma_csrのレジスタに適切な値が設定された後で、最後にdma_acrへの書き込みが行われているかをチェックすることが重要です。表31を参照してください。

表31　DMAライト動作に対するpci_aのイニシャライズ

アドレス（16進）	レジスタ名	データ（16進）	定　義
BAR0: 0.0000	dma_csr	**0000.0039**	dma_csr**の値がインタラプトをイネーブルにし、DMA動作がライト・トランズアクションであることを示すと共に、DMAエンジンをイネーブルにし、DMAターミナル・カウント・インタラプトをディセーブルする。**
BAR0: 0.0008	dma_bcr	**00084**	**このレジスタに書き込まれた値がDMA転送のデータ・サイズ（バイト数）を示す。この値はDWORD（４バイト）の倍数となっている必要がある。**
BAR0: 0.0004	dma_acr	**00400000**	**PCIバスの転送が開始される位置を示すアドレス。このアドレスは各データ転送の完了後に自動的にアップデートされる。**

54ページの図20はローカル側のDMAレジスタ・ライト・トランズアクションのタイミング波形を示したものです。このタイミング図は下記の信号シーケンスを示しています。

1. ローカル側のロジックがクロック-１でl_reqをアサートし、DMA転送の開始が可能であることを示します。l_reqのアサートは、ローカル側がDMA転送を開始できる状態になるまで遅延させることができます。

2. クロック-２ で、ローカル側のロジックがl_dma_dat_in[31..0]のバス上にデータを供給すると共に、l_dma_csr_wrをアサートします。dma_csrレジスタには16進の39という値が書き込まれていますが、この値はインタラプトをイネーブルにし、 DMAターミナル・カウント・インタラプトをディセーブルし、DMAエンジンをイネーブルにしてライト・サイクルを要求しています。

3. クロック-３で、ローカル側のロジックがl_dma_dat_in[31..0]バスに

データを供給すると共に、l_dma_bcr_wrをアサートします。この信号シーケンスでは、16進の84の値（132バイト）がdma_bcrレジスタに書き込まれています。l_dma_csr_out[6..0]の値はクロック-2で書き込まれる値に変更されるため、dma_csrレジスタに対する書き込み動作はクロック-3で有効となります。

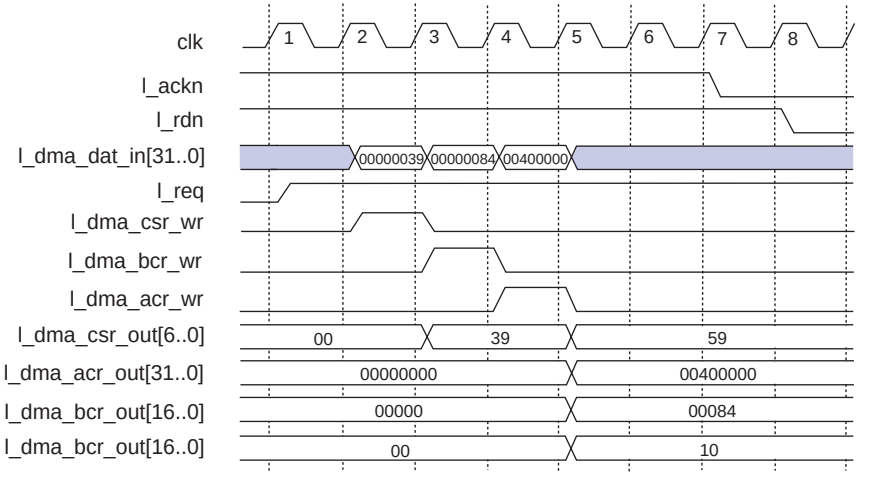
4. クロック-4で、ローカル側のロジックはl_dma_dat_in[31..0]バスにデータを供給すると共に、l_dma_acr_wrをアサートします。この信号シーケンスでは、dma_acrレジスタに00400000（hex）の値が書き込まれています。pci_aは00400000（hex）のアドレスへのPCIライト動作を開始します。

5. クロック-5では、dma_bcrとdma_acrのレジスタに対する書き込みが有効になります。図20では、l_dma_bcr_out[16..0]とl_dma_acr_out[31..0]のバス上の値が変化していることが示されています。また、図20では、l_dma_isr_out[4..0]とl_dma_csr_out[6..0]のバス上の値も変化していることが示されており、ad_loadedとdma_onのビットがセットされています。

6. pci_aはl_acknをアサートし、ローカル側からデータを受け取れる状態になっていることを示します。

7. クロック-9の立ち上がりエッジで、ローカル側のロジックがl_dat_in[31..0]バス上にバッファへ転送するデータの供給を開始します。

図20　ローカル側が開始したDMAライト・トランズアクション



一般的なホスト・プログラミング・ガイドライン

DMA転送はホストおよびローカル側のロジックからも可能です。このセクションでは、DMAがホスト側によってコントロールされる場合の一般的なプログラミング・ガイドラインを次の４つのタスクに分類して解説します。

- pci_aのイニシャライズ
- DMA動作
- インタラプト・サービス・オペレーション
- エラー・ビットのクリア

pci_aのイニシャライズ

pci_aのイニシャライズは下記の方法で行えます。

1. pci_aがサポートしているPCIバス・コンフィギュレーション・レジスタをコンフィギュレーションする。
2. dma_csrレジスタをコンフィギュレーションする。表32を参照してください。

表32　pci_aのイニシャライズ

ステップ	アドレス（16進）	レジスタ名	データ（16進）	定　義
1	04	PCI バス・コマンド／ステータス・レジスタ	0000.0146	PCIバス・コマンド・レジスタがこの値になると、メモリ転送、マスタ・オペレーション、データ・パリティ・エラー発生時におけるperrnのアサート、アドレス・パリティ・エラー発生時におけるserrnのアサートがそれぞれイネーブルになる。
2	BAR0: 0.0000		0000.0011	dma_csrがこの値になると、DMAエンジンとインタラプトの双方がイネーブルになる。

DMA動作

DMA動作を開始させる場合は、下記の手順で行います。

1. dma_bcrをロードする（データの次のブロックに対するバイト・カウントが現在のブロックと同じ場合、このステップはオプションとして取り扱われます）。
2. dma_acrをロードする（43ページの「内部ターゲット・レジスタのメモリ・マップ」を参照）。
3. ローカル側の周辺デバイスをコンフィギュレーションする。このステップでは、ローカル側に要求されるアドレス生成のプロセスをセットアップし、ローカル側でl_reqをアサートできるようにします。ただし、イ

ンテリジェントなPCIのエージェント（マイクロプロセッサ等）がローカル側で動作する場合は、このステップが不要になります。 表33を参照してください。

4. この時点で、バイト・カウントが完了するため、pci_aファンクションがインタラプト・コントローラに対してPCIインタラプト(intan)を生成する。

表33　DMA動作

ステップ	アドレス（16進）	レジスタ名	データ（16進）	定　　義
1	BAR0: 0.0008	dma_bcr	ユーザ定義	DMA転送のデータ量（バイト数）。
2	BAR0: 0.0004	dma_acr	ユーザ定義	転送が開始されるPCIバス・アドレス。このアドレスは、データ転送毎に自動的にアップデートされる。
3	BAR0: 8.0000	外部ターゲット・レジスタ	ユーザ定義	このステップには、ローカル・アドレス生成のセットアップや、ローカル側からのl_reqのアサートが含まれる。

インタラプト・サービス・オペレーション

インタラプト・サービス・オペレーションは下記の手順で実行します。

1. dma_isrをリードする。

   a. dma_tcのビットがHighでerr_pendビットがLowになっていれば、DMA動作が正常に行われ、pci_aは次のDMA転送が可能な状態となり、55ページの「DMA動作」のステップ-１へ戻ることができます。

   b. err_pendのビットがHighになっていれば、DMA動作がエラーによって停止したことになり、57ページの「エラー・ビットのクリア」に記述されているステップ-２の処理を行う必要があります。ただし、ここで処理を継続する前にエラー・ビットをクリアする必要があります。表34を参照してください。

表34　インタラプト・サービス・ルーチン

ステップ	アドレス（16進）	レジスタ名	データ（16進）	定　　義
1	BAR0: 0.000C	dma_isr	ユーザ定義	このdma_isrレジスタの値が、DMA動作の状態と動作が終了した原因を表示する。

エラー・ビットのクリア

エラー・ビットのクリアは下記の手順で実行します。

1. dma_isrをリードし、err_pendビットがアクティブになっていれば、ステップ-2へ移行する。

2. flushビットをアサートしてad_loadedのビット（dma_isrの４ビット目）をクリアし、dma_csrをコンフィギュレーションする。

3. PCIバス・コンフィギュレーション・ステータス・レジスタをリードし、エラーがアサートされているかどうかを確認する（15、12または13ビット目）。

4. pci_aがサポートしているPCIステータス・レジスタをコンフィギュレーションし、対応するエラー・ビットのフィールドにロジック-１をライトする。ステータス・レジスタのビットに-１をライトすることで、そのビットをクリアすることができます。ステータス・レジスタをリードし、同じ値をライトすることによって、エラー状態をクリアすることができます。

アプリケーション

pci_aは、アドイン・カードのアプリケーションに最適なファンクションとなっています。図21は、pci_aとローカル側のインテリジェント・ホストとの一般的な接続例を示したものです。この例では、ローカル側に対するアクセスに、ターゲットとDMAコントロール・ブロックが必要です。ローカル側のデータ・バスは、l_holdn出力によってコントロールされる双方向バスとなっています。ホストは、ローカル・バスがアクセス中の場合に、l_holdnをアサートします。また、PCIバス・アドレスはローカル側のアドレスと異なることがあるため、DMAアクセスの実行時にホストがローカル側のアドレスを生成する必要があります。

図21　シェアド・メモリ・バスを持つローカル側のインテリジェント・ホストとのインタフェース

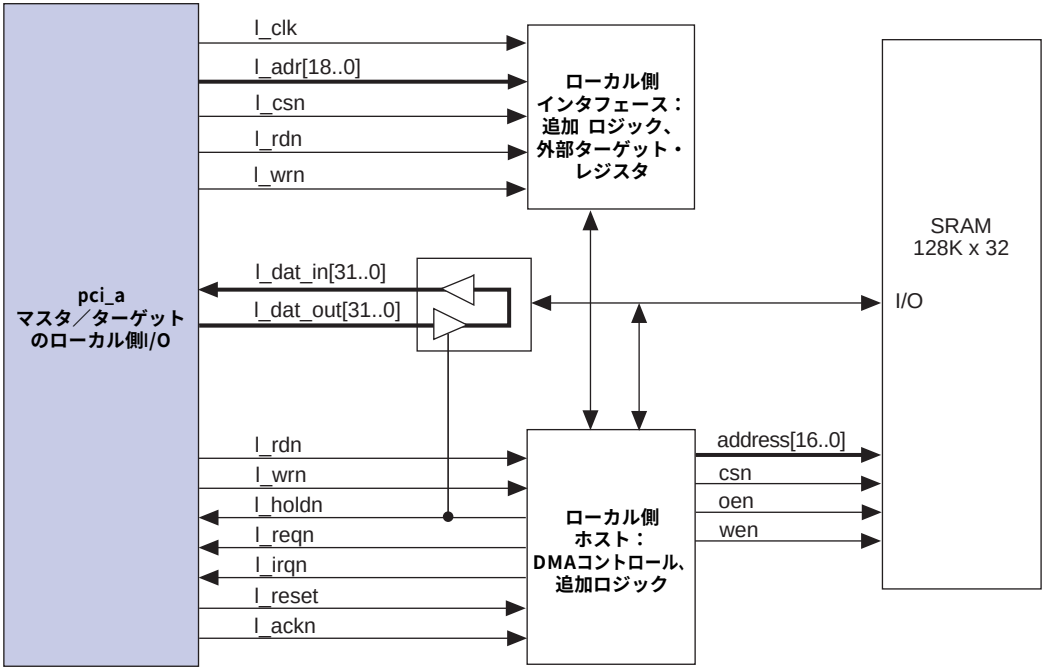
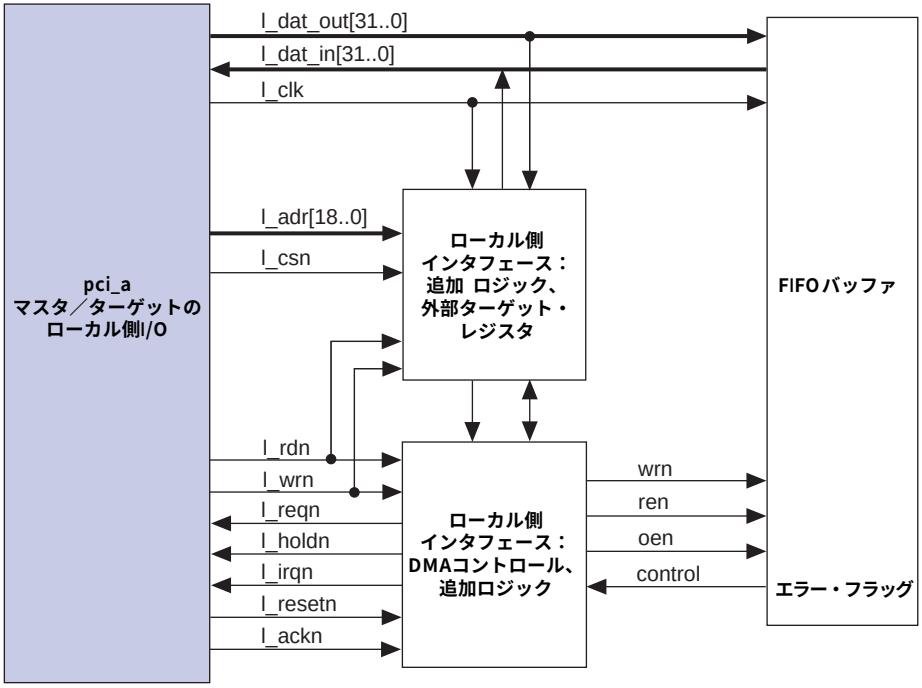


図22はpci_aとFIFOバッファとの一般的な接続例を示したものです。この例では、ローカル側へのアクセスに、ターゲットとDMAコントロール・ブロックが必要となっています。

この図のローカル側はDMAアクセス時に必要なアドレス信号やコントロール信号を生成する機能を持っていないため、DMAコントロール・ブロックがターゲット・アクセスを通じて得られるPCIバスからのコンフィギュレーション・データやコントロール・データを受け入れるようにしておくことができます。図22には、２つのコントロール・ブロック間で伝送される双方向信号による処理プロセスが示されています。

図22　FIFOバッファとのローカル側インタフェース



PCI SIG プロトコル・チェックリスト

表35から表42は、PCI SIGプロトコルに要求される項目を*PCIの仕様準拠チェックリスト Revision 2.1*に基いてリストしたものです。この表の中でYesの欄にチェック・マークがあれば、pci_aがこの要求を満足していることを示しています。アルテラのFLEX 10Kに実現されるpci_aファンクションに該当しないチェック項目はリストされていません。また、この表で"ー"となっている項目はPCI SIGの要求には該当していないことを示しています。

表35　コンポーネント・コンフィギュレーション

CO#	要求内容	Yes	No
1	各PCIリソースは、セクション-6.1で定義されているように、64バイトの定義済みのヘッダと192バイトのデバイス定義領域を持った256バイトのテンプレートをベースにしたコンフィギュレーション・スペースを持っているか？	√	
2	デバイス内のすべてのファンクションは、ヘッダ内でベンダID、デバイスID、コマンド、ステータス、ヘッダ・タイプ、クラス・コード・フィールドをサポートしているか？	√	
3	コンフィギュレーション・スペースは、常時アクセス可能になっているか？	√	
4	リザーブされているレジスタまたはリード・オンリのビットへのライトが正常に完了したか、およびそのデータは放棄されるか？	√	
5	リザーブされているレジスタや未使用のレジスタやビットへのデータ・リードが問題なく完了し、0が返されるか？	√	
6	ベンダIDがPCI SIGによって割り当てられているか？	√	
7	ヘッダ・タイプ・フィールドは、有効なエンコーディングが可能か？	√	
8	マルチ・バイト・トランズアクションで適切なレジスタがアクセスされるか？ レジスタはリトル・エンディアンになっているか？	√	
9	すべてのリード・オンリ・レジスタの値は、決められた範囲内となっているか？例えば、インタラプト・ピン・レジスタは0から４までの値をとらなければならない。	√	
10	クラス・コードはappendix Dで規定されたものに準拠しているか？	√	
11	コンフィギュレーション・レジスタの定義済みヘッダ部分は、バイト、ワード、DWORD単位でアクセス可能か？	√	
12	デバイスはマルチ・ファンクション・デバイスか？		√
13	デバイスがマルチ・ファンクション・タイプの場合、シングル・ファンクション・タイプに実現されたファンクションへのコンフィギュレーション・スペース・アクセスは無視されるか？		√

表36　コンポーネント・コンフィギュレーション・スペース一覧　(1/2)

ロケーション	名称	必須／オプション	N/A	サポート
00h-01h	ベンダID	必須		√
02h-03h	デバイスID	必須		√
04h-05h	コマンド	必須		√

表36　コンポーネント・コンフィギュレーション・スペース一覧　(2/2)

ロケーション	名称	必須／オプション	N/A	サポート
06h-07h	ステータス	必須		√
08h	リビジョンID	必須		√
09h-0Bh	クラス・コード	必須		√
0Ch	キャッシュ・ライン・サイズ	メモリ・ライトおよびインバリデート・サイクルを生成できるマスタ・デバイス／ファンクションに必須	√	
0Dh	レイテンシ・タイマ	２つ以上のデータ・フェーズを継続するマスタ・デバイス／ファンクションに必須		√
0Eh	ヘッダ・タイプ	デバイスがマルチ・ファンクションの場合は、7ビット目を１にセットする		√
0F	BIST	オプション	√	
10h-13h	BAR0	オプション		√
14h-27h	BAR1-BAR5	オプション	√	
28h-2Bh	カードバスCISポインタ	オプション	√	
2Ch-2Dh	サブシステム・ベンダID	オプション		√
2Eh-2Fh	サブシステムID	オプション		√
30h-33h	拡張ROMベース・アドレス	拡張ROMを備えているデバイス／ファンクションに必須	√	
34h-3Bh	予約			
3Ch	インタラプト・ライン	インタラプト・ピンを使用するデバイス／ファンクションに必須		√
3Dh	インタラプト・ピン	インタラプト・ピンを使用するデバイス／ファンクションに必須		√
3Eh	Min_Gnt	オプション		√
3Fh	Max_Lat	オプション	√	

表37　デバイス・コントロール一覧

ロケーション	必須／オプション	Yes	No
DC1	コマンド・レジスタに0000hがロードされた場合、コンフィギュレーション・アクセスを除いて、デバイス／ファンクションがPCIバスから論理的に切り離されるか？（ブート・コード・パス内のデバイスが排除されるか？）	√	
DC2	PCIのrstnがアサートされた後、デバイス／ファンクションがディセーブルされるか？（ブート・コード内のデバイスが排除されるか？）	√	

表38　コマンド・レジスター覧

Bit	名称	必須／オプション	N/A	ターゲット	マスタ
0	I/Oスペース	I/Oスペースにマッピングされたレジスタを内蔵しているデバイス／ファンクションに必須	√		
1	メモリ・スペース	メモリ・スペース・アクセスに応答するデバイス／ファンクションに必須		√	
2	バス・マスタ	必須			√
3	スペシャル・サイクル	スペシャル・サイクルに応答できるデバイス／ファンクションに必須	√		
4	メモリ・ライトおよびインバリデート	メモリ・ライトおよびインバリデート・サイクルを生成するデバイス／ファンクションに必須	√		
5	VGAパレット・スヌープ	VGAパレットを参照するVGAやグラフィック・デバイス／ファンクションに必須	√		
6	パリティ・エラー・レスポンス	必須			√
7	ウェイト・サイクル・コントロール	オプション	√		
8	serrnイネーブル	serrnピンを持つデバイス／ファンクションに必須			√
9	高速バック・ツー・バック・イネーブル	異なるターゲット間で高速のバック・ツー・バック・サイクルをサポートするマスタ・デバイス／ファンクションに必須	√		
10..15	予約				

表39　デバイス・ステータス

DS#	要求内容	Yes	No
1	ステータス・レジスタ内のすべてのリード／ライト・ビットは0にリセットされるか？	√	
2	リード／ライト・ビットは、デバイス／ファンクションによって排他的に１にセットされるか？	√	
3	PCIのrstn信号がアサートされたとき、リード／ライト・ビットは0にリセットされるか？	√	
4	このビットに１をライトすることにより、リード／ライト・ビットは0にリセットされるか？	√	

表40　ステータス・レジスター覧　(1/2)

Bit	名称	必須／オプション	N/A	ターゲット	マスタ
4..0	予約	必須			
5	66MHz対応	66MHz動作を保証するデバイスに必須	√		
6	UDFサポート	オプション	√		

表40　ステータス・レジスター覧　(2/2)

Bit	名称	必須／オプション	N/A	ターゲット	マスタ
7	高速バック・ツー・バック対応	オプション	√		
8	データ・パリティの検知	必須			√
10..9	DEVSELタイミング	必須		√	
11	ターゲット・アボートの通知	ターゲット・アボート対応のデバイス／ファンクションに必須	√		
12	ターゲット・アボートの受信	必須			√
13	マスタ・アボートの受信	必須			√
14	システム・エラーの通知	serrnのアサートが可能なデバイス／ファンクションに必須			√
15	パリティ・エラーの検知	セクション 3.7.2 で免除されていない限り必須			√

表41　コンポーネント・マスタ・チェックリスト　(1/3)

MP#	要求内容	Yes	No
1	すべてのサステインド・トライ・ステート信号が、トライ・ステートになる前に1クロックの期間、Highレベルにドライブされる。（セクション2.1）	√	
2	メモリ・ライト・インバリデート・サイクルのデータ・フェーズ中に、IUT (Interface Under Test)はすべてのバイト・イネーブルを常にアサートする。（セクション3.1.1）	√	
3	メモリ・ライト・インバリデート・サイクルにおいて、IUTはリニア・バースト・オーダリングを常に使用する。（セクション3.1.1）	—	
4	ライト・トランズアクションの期間にデータが有効の場合、IUTがirdynをドライブする。（セクション3.2.1）	√	
5	irdynとtrdynが共に同じクロックの立ち上がりエッジでアサートされるときのみ、IUTがデータを転送する。（セクション3.2.1）	√	
6	IUTがirdynをアサートした場合、処理中のデータ・フェーズが終了するまでframenの状態が保持される。（セクション3.2.1）	√	
7	IUTがirdynをアサートした場合、処理中のデータ・フェーズが終了するまでirdynの状態が保持される。（セクション3.2.1）	√	
8	IUTはリザーブされているバースト・オーダリング（ad[1..0] = "01"）を使用しない。（セクション3.2.2）	√	
9	IUTはリザーブされているバースト・オーダリング（ad[1..0] = "11"）を使用しない。（セクション3.2.2）	√	

表41　コンポーネント・マスタ・チェックリスト　(2/3)

MP#	要求内容	Yes	No
10	idselがアサートされず、ad[1..0]が"00"でない場合、IUTはコンフィギュレーション・コマンドを無視する。（セクション3.2.2）	√	
11	各アドレス、データ・フェーズ中、IUTのアドレス線は安定した値にドライブされる。（セクション3.2.4）	√	
12	IUTのcben[3..0]出力バッファは、データ・フェーズの最初のクロックからトランズアクションの最後までイネーブル状態を保持する。（セクション3.3.1）	√	
13	データ・フェーズの期間中、IUTのcben[3..0]には有効なバイト・イネーブル情報が含まれている。（セクション3.3.1）	√	
14	irdynがアサートされないかぎり、IUTはframenをディアサートしない。（セクション3.3.3.1）	√	
15	framenがディアサートされた後、最低１クロックの間、IUTはirdynをディアサートしない。（セクション3.3.3.1）	√	
16	IUTがframenをディアサートした場合、同一トランズアクション中でframenを再びアサートしない。（セクション3.3.3.1）	√	
17	ターゲットがdevselnをアサートした場合、IUTはマスタ・アボートで終了しない。	√	
18	framenのアサートが最初にサンプルされた後、５クロックまではIUTがマスタ・アボートを通知しない。（セクション3.3.3.1）	√	
19	リトライによって終了した場合、IUTは確実にオリジナル処理のためのアクセスを繰り返す。（セクション3.3.3.2.2）	√	
20	IUTはgntnがアサートされない限り、サイクルを開始しない。（セクション3.4.1）	√	
21	バスがアイドル状態でframenがネゲートされているとき、gntnのネゲーション後、IUTは１クロック以内にcben [3..0]とad [31..0]をトライ・ステート状態にする。（セクション3.4.3）	√	
22	バスがアイドル状態のとき、gntnのアサーション後、IUTは８クロック以内にcben[3..0]とad[31..0]をドライブする。（セクション3.4.3）	√	
23	全てのデータ・フェーズにおいて、IUTは８クロック以内にirdynをアサートする。（セクション3.5.2）	√	
24	IUTはリード・トランズアクションでロック・オペレーションを開始する。（セクション3.6）	—	
25	ターゲット・アボートやマスタ・アボートによりアクセスが終了した場合は、IUTがLOCK#をリリースする。（セクション3.6）	—	
26	連続したロック・オペレーション中、IUTは最小の１アイドル・サイクルの期間にLOCK#をディアサートする。（セクション3.6）	—	
27	IUTはコンフィギュレーション・サイクルに、リニア・バースト・オーダリングを常時使用する。（セクション3.7.4）	√	
28	IUTは常に、cben[3..0]とad[31..0]がドライブされた１クロック以内にparをドライブする。（セクション3.8.1）	√	

表41　コンポーネント・マスタ・チェックリスト　(3/3)			
MP#	要求内容	Yes	No
29	IUTはad[31..0]とcben[3..0]、parの1の総数が偶数になるように、parをドライブする。（セクション3.8.1）	√	
30	パリティ・エラーが検知されたデータの２クロック後に、IUTは常時、perrn（イネーブルされているとき）をアクティブにドライブする。（セクション3.8.2.1）	√	
31	パリティ・エラーが検知されたデータ・フェーズに対して、IUTは最低１クロックの期間、perrn（イネーブルされているとき）をドライブする。（セクション3.8.2.1）	√	
32	IUTはDUALコマンドに続くサイクルで、アサートされているframenを常に保持する。（セクション3.10.1）	—	
33	アドレスの上位32ビットがゼロの時、IUTはDUALサイクルを生成しない。（セクション3.10.1）	—	

表42　コンポーネント・ターゲット・チェックリスト　(1/2)			
TP#	要求内容	Yes	No
1	すべてのサステインド・トライ・ステート信号は、トライ・ステートになる前に１クロックの期間、Highレベルにドライブされる。（セクション2.1）	√	
2	IUTはサイクルを要求してデータ・フェーズが終了するまで、perrnをリポートしない。（セクション2.2.5）	√	
3	IUTは予約されたコマンドを使用して他のコマンド動作を行わない。（セクション3.1.1）	—	
4	32ビット・アドレッサブルIUTは、DUALコマンドを予約コマンドとして取り扱う。（セクション3.1.1）	—	
5	IUTがtrdynをアサートした場合、データ・フェーズが終了するまでtrdynは変化しない。（セクション3.2.1）	√	
6	IUTがtrdynをアサートした場合、データ・フェーズが終了するまでdevselnは変化しない。（セクション3.2.1）	√	
7	IUTがtrdynをアサートした場合、データ・フェーズが終了するまでstopnは変化しない。（セクション3.2.1）	√	
8	IUTがstopnをアサートした場合、データ・フェーズが終了するまでstopnは変化しない。（セクション3.2.1）	√	
9	IUTがstopnをアサートした場合、データ・フェーズが終了するまでtrdynは変化しない。（セクション3.2.1）	√	
10	IUTがstopnをアサートした場合、データ・フェーズが終了するまでdevselnは変化しない。（セクション3.2.1）	√	
11	irdynとtrdynが共に同じクロックの立ち上がりエッジでアサートされるときのみ、IUTはデータを転送する。（セクション3.2.1）	√	
12	リード・サイクルでデータが有効なとき、IUTはtrdynをアサートする。（セクション3.2.1）	√	

表42　コンポーネント・ターゲット・チェックリスト　(2/2)			
TP#	要求内容	Yes	No
13	バイト・イネーブルにより定義された全てのI/Oアクセスを終了できない場合は、IUTがターゲット・アボートを通知する。（セクション3.2.2）	—	
14	IUTは予約エンコーディングに応答しない。（セクション3.2.2）	√	
15	idselがアサートされず、ad[31..0] = "00"でないとき、IUTはコンフィギュレーション・コマンドを無視する。（セクション3.2.2）	√	
16	予約されたバースト・モードが検知されたとき、IUTは最初のデータ・フェーズの後で常にディスコネクトを行う。（セクション3.2.2）	—	
17	各アドレスとデータ・フェーズ中、IUTのad[31..0]線は安定した値にドライブされる。（セクション3.2.4）	√	
18	IUTのcben[3..0]出力バッファは、データ・フェーズの最初のクロックからトランズアクションの最後までイネーブル状態を保持する。（セクション3.3.1）	√	
19	ターンアラウンド・サイクル中のリード時に、IUTはtrdynをアサートしない。（セクション3.3.1）	√	
20	IUTは最後のデータ・フェーズに続くクロックで、trdyn、stopn、devselnをディアサートする。（セクション3.3.3.2）	√	
21	バーストがリソースのバウンダリを超える場合は、IUTがディスコネクトを発行する。（セクション3.3.3.2）	—	
22	IUTは、framenがディアサートされた直後のサイクルでstopnをディアサートする。（セクション3.3.3.2.1）	√	
23	IUTがstopnをアサートしていた場合、IUTはframenがネゲートされるまでstopnをディアサートしない。（セクション3.3.3.2.1）	√	
24	IUTはターゲット・アボートを通知する前にtrdynをディアサートする。（セクション3.3.3.2.1）	—	
25	IUTはstopnをディアサートせず、トランズアクションを継続する。（セクション3.3.3.2.1）	√	
26	IUTは16クロック・サイクル以内に、最初のデータ・フェーズを完了する。（セクション3.5.1.1）	√	
27	IUTは常に最小16バイトのロックを行う。（セクション3.6）	—	
28	IUTはどの応答よりも先にdevselnを発行する。（セクション3.7.1）	√	
29	IUTによってdevselnがアサートされた場合、ターゲット・アボートの通知を除き、IUTは最後のデータ・フェーズが完了するまで、devselnをディアサートしない。（セクション3.7.1）	√	
30	IUTはスペシャル・サイクルには応答しない。（セクション3.7.2）	√	
31	IUTはcben[3..0]とad[31..0]がドライブされている１クロック以内にparをドライブする。（セクション3.8.1）	√	
32	IUTは常にad[31..0]とcben[3..0]、parの１の総数が偶数になるように、parをドライブする。（セクション3.8.1）	√	

PCI SIG テスト・ベンチ一覧

表43から表60までに示す項目は、***PCI仕様準拠チェックリスト Revision 2.1***によるPCI SIGテスト・ベンチ・シナリオに対応したものとなっています。Yes欄のチェック・マークは、pci_aが仕様に準拠していることを表しています。リストに載っていないアルテラのFLEX 10Kに実現されるpci_aファンクションに該当しない項目は、リストされていません。

表43　テスト・シナリオ：1.1 PCIデバイス・スピード（devselで表される）のテスト

#	要求内容	Yes	No
1	高速メモリ・スレーブに対するライト後のデータ転送。	√	
2	高速メモリ・スレーブからのリード後のデータ転送。	√	
3	中速メモリ・スレーブに対するライト後のデータ転送。	√	
4	中速メモリ・スレーブからのリード後のデータ転送。	√	
5	低速メモリ・スレーブに対するライト後のデータ転送。	√	
6	低速メモリ・スレーブからのリード後のデータ転送。	√	
7	サブトラクティブ・メモリ・スレーブに対するライト後のデータ転送。	√	
8	サブトラクティブ・メモリ・スレーブからのリード後のデータ転送。	√	
9	サブトラクティブ・メモリ・スレーブより遅いメモリ・スレーブに対するライト後のマスタ・アボート・ビットのセット。	√	
10	サブトラクティブ・メモリ・スレーブより遅いメモリ・スレーブからのリード後のマスタ・アボート・ビットのセット。	√	

表44　テスト・シナリオ：1.2 PCIバス・ターゲット・アボート・サイクル　(1/2)

#	要求内容	Yes	No
1	高速メモリ・スレーブに対するライト後のターゲット・アボート・ビットのセット。	√	
2	IUTはライト・トランズアクションを繰り返さない。	√	
3	高速メモリ・スレーブからのリード後のIUTターゲット・アボート・ビットのセット。	√	
4	IUTはリード・トランズアクションを繰り返さない。	√	
5	中速メモリ・スレーブに対するライト後のターゲット・アボート・ビットのセット。	√	
6	IUTはライト・トランズアクションを繰り返さない。	√	
7	中速メモリ・スレーブからのリード後のIUTターゲット・アボート・ビットのセット。	√	
8	IUTはリード・トランズアクションを繰り返さない。	√	
9	低速メモリ・スレーブに対するライト後のターゲット・アボート・ビットのセット。	√	
10	IUTはライト・トランズアクションを繰り返さない。	√	
11	低速メモリ・スレーブからのリード後のIUTターゲット・アボート・ビットのセット。	√	
12	IUTはリード・トランズアクションを繰り返さない。	√	
13	サブトラクティブ・メモリ・スレーブに対するライト後のターゲット・アボート・ビットのセット。	√	

表44　テスト・シナリオ：1.2 PCIバス・ターゲット・アボート・サイクル　(2/2)			
#	要求内容	Yes	No
14	IUTはライト・トランズアクションを繰り返さない。	√	
15	サブトラクティブ・メモリ・スレーブからのリード後のIUTターゲット・アボート・ビットのセット。	√	
16	IUTはリード・トランズアクションを繰り返さない。	√	

表45　テスト・シナリオ：1.3 PCIバス・ターゲット・リトライ・サイクル			
#	要求内容	Yes	No
1	高速メモリ・スレーブに対するライト後のデータ転送。	√	
2	高速メモリ・スレーブからのリード後のデータ転送。	√	
3	中速メモリ・スレーブに対するライト後のデータ転送。	√	
4	中速メモリ・スレーブからのリード後のデータ転送。	√	
5	低速メモリ・スレーブに対するライト後のデータ転送。	√	
6	低速メモリ・スレーブからのリード後のデータ転送。	√	
7	サブトラクティブ・メモリ・スレーブに対するライト後のデータ転送。	√	
8	サブトラクティブ・メモリ・スレーブからのリード後のデータ転送。	√	

表46　テスト・シナリオ：1.4 PCIバス・シグナル・データ・フェーズ・リトライ・サイクル			
#	要求内容	Yes	No
1	高速メモリ・スレーブに対するライト後のデータ転送。	√	
2	高速メモリ・スレーブからのリード後のデータ転送。	√	
3	中速メモリ・スレーブに対するライト後のデータ転送。	√	
4	中速メモリ・スレーブからのリード後のデータ転送。	√	
5	低速メモリ・スレーブに対するライト後のデータ転送。	√	
6	低速メモリ・スレーブからのリード後のデータ転送。	√	
7	サブトラクティブ・メモリ・スレーブに対するライト後のデータ転送。	√	
8	サブトラクティブ・メモリ・スレーブからのリード後のデータ転送。	√	

表47　テスト・シナリオ：1.5 PCIバス・シングル・データ・フェーズ・ディスコネクト・サイクル　(1/2)			
#	要求内容	Yes	No
1	高速メモリ・スレーブに対するライト後のターゲット・アボート・ビットのセット。	√	
2	IUTはライト・トランズアクションを繰り返さない。	√	
3	高速メモリ・スレーブからのリード後のIUTターゲット・アボート・ビットのセット。	√	

表47　テスト・シナリオ：1.5 PCIバス・シングル・データ・フェーズ・ディスコネクト・サイクル (2/2)			
#	要求内容	Yes	No
4	IUTはリード・トランズアクションを繰り返さない。	√	
5	中速メモリ・スレーブに対するライト後のターゲット・アボート・ビットのセット。	√	
6	IUTはライト・トランズアクションを繰り返さない。	√	
7	中速メモリ・スレーブからのリード後のIUTターゲット・アボート・ビットのセット。	√	
8	IUTはリード・トランズアクションを繰り返さない。	√	
9	低速メモリ・スレーブに対するライト後のターゲット・アボート・ビットのセット。	√	
10	IUTはライト・トランズアクションを繰り返さない。	√	
11	低速メモリ・スレーブからのリード後のIUTターゲット・アボート・ビットのセット。	√	
12	IUTはリード・トランズアクションを繰り返さない。	√	
13	サブトラクティブ・メモリ・スレーブに対するライト後のターゲット・アボート・ビットのセット。	√	
14	IUTはライト・トランズアクションを繰り返さない。	√	
15	サブトラクティブ・メモリ・スレーブからのリード後のIUTターゲット・アボート・ビットのセット。	√	
16	IUTはリード・トランズアクションを繰り返さない。	√	

表48　テスト・シナリオ：1.6 PCIバス・マルチ・データ・フェーズ・リトライ・サイクル			
#	要求内容	Yes	No
1	高速メモリ・スレーブに対するライト後のデータ転送。	√	
2	高速メモリ・スレーブからのリード後のデータ転送。	√	
3	中速メモリ・スレーブに対するライト後のデータ転送。	√	
4	中速メモリ・スレーブからのリード後のデータ転送。	√	
5	低速メモリ・スレーブに対するライト後のデータ転送。	√	
6	低速メモリ・スレーブからのリード後のデータ転送。	√	
7	サブトラクティブ・メモリ・スレーブに対するライト後のデータ転送。	√	
8	サブトラクティブ・メモリ・スレーブからのリード後のデータ転送。	√	

表49　テスト・シナリオ：1.7 PCIバス・マルチ・データ・フェーズ・ディスコネクト・サイクル (1/2)			
#	要求内容	Yes	No
1	高速メモリ・スレーブに対するライト後のデータ転送。	√	
2	高速メモリ・スレーブからのリード後のデータ転送。	√	
3	中速メモリ・スレーブに対するライト後のデータ転送。	√	
4	中速メモリ・スレーブからのリード後のデータ転送。	√	

表49　テスト・シナリオ：1.7 PCIバス・マルチ・データ・フェーズ・ディスコネクト・サイクル　(2/2)			
#	要求内容	Yes	No
5	低速メモリ・スレーブに対するライト後のデータ転送。	√	
6	低速メモリ・スレーブからのリード後のデータ転送。	√	
7	サブトラクティブ・メモリ・スレーブに対するライト後のデータ転送。	√	
8	サブトラクティブ・メモリ・スレーブからのリード後のデータ転送。	√	

表50　テスト・シナリオ：1.8 PCIバス・マルチ・データ・フェーズおよびtrdynサイクル　(1/2)			
#	要求内容	Yes	No
1	trdynが２つ目のクロックの立ち上がりでリリースされ、framen後の３つ目の立ち上がりクロックでアサートされたとき、データがプライマリ・ターゲットに対してライトされたことを確認する。	√	
2	trdynが２つ目のクロックの立ち上がりでリリースされ、framen後の３つ目の立ち上がりクロックでアサートされたとき、データがプライマリ・ターゲットからリードされたことを確認する。	√	
3	trdynが３つ目のクロックの立ち上がりでリリースされ、framen後の４つ目の立ち上がりクロックでアサートされたとき、データがプライマリ・ターゲットにライトされたことを確認する。	√	
4	trdynが３つ目のクロックの立ち上がりでリリースされ、framen後の４つ目の立ち上がりクロックでアサートされたとき、データがプライマリ・ターゲットからリードされたことを確認する。	√	
5	trdynが３つ目のクロックの立ち上がりでリリースされ、framen後の５つ目の立ち上がりクロックでアサートされたとき、データがプライマリ・ターゲットにライトされたことを確認する。	√	
6	trdynが３つ目のクロックの立ち上がりでリリースされ、framen後の５つ目の立ち上がりクロックでアサートされたとき、データがプライマリ・ターゲットからリードされたことを確認する。	√	
7	trdynが４つ目のクロックの立ち上がりでリリースされ、framen後の６つ目の立ち上がりクロックでアサートされたとき、データがプライマリ・ターゲットにライトされたことを確認する。	√	
8	trdynが４つ目のクロックの立ち上がりでリリースされ、framen後の６つ目の立ち上がりクロックでアサートされたとき、データがプライマリ・ターゲットからリードされたことを確認する。	√	
9	trdynが１クロック・サイクルの期間リリースされ、framenの後、１クロック・サイクルの期間アサートされたとき、データがプライマリ・ターゲットにライトされたことを確認する。	√	
10	trdynが１クロック・サイクルの期間リリースされ、framenの後、１クロック・サイクルの期間アサートされたとき、データがプライマリ・ターゲットからリードされたことを確認する。	√	

表50　テスト・シナリオ：1.8 PCIバス・マルチ・データ・フェーズおよびtrdynサイクル　(2/2)			
#	要求内容	Yes	No
11	trdynが２クロック・サイクルの期間リリースされ、framenの後、２クロック・サイクルの期間アサートされたとき、データがプライマリ・ターゲットにライトされたことを確認する。	√	
12	trdynが２クロック・サイクルの期間リリースされ、framenの後、２クロック・サイクルの期間アサートされたとき、データがプライマリ・ターゲットからリードされたことを確認する。	√	

表51　テスト・シナリオ：1.9 PCIバス・データ・パリティ・エラー・シングル・サイクル			
#	要求内容	Yes	No
1	プライマリ・ターゲットがIUTメモリ・ライトでperrnをアサートしたとき、IUTがデータ・パリティ・エラー検知ビットをセットしたことを確認する。	√	
2	IUTメモリ・リードの最初のデータ・フェーズ（奇数パリティ）後に、perrnが２クロックの期間にわたってアクティブになることを確認する。	√	
3	奇数パリティがIUTメモリ・リードで検知されたとき、IUTによってパリティ・エラー検知ビットがセットされることを確認する。	√	

表52　テスト・シナリオ：1.10 PCIバス・データ・パリティ・エラー・マルチ・データ・フェーズ・サイクル			
#	要求内容	Yes	No
1	プライマリ・ターゲットがIUTマルチ・データ・フェーズ・メモリ・ライトでperrnをアサートしたとき、IUTによってパリティ・エラー検知ビットがセットされることを確認する。	√	
2	IUTのマルチ・データ・フェーズ・メモリ・リードで最初のデータ・フェーズで（奇数パリティが検知された場合）perrnが２クロック期間にわたってアクティブになることを確認する。	√	
3	パリティが奇数のとき、IUTがパリティ・エラー検知ビットをセットすることを確認する。	√	

表53　テスト・シナリオ：1.11 PCIバス・マスタ・タイムアウト			
#	要求内容	Yes	No
1	４個のデータ・フェーズが終了する前にメモリ・ライト・トランズアクションが終了する。	√	
2	４個のデータ・フェーズが終了する前にメモリ・リード・トランズアクションが終了する。	√	

表54　テスト・シナリオ：1.13 PCIバス・マスタ・パーキング			
#	要求内容	Yes	No
1	IUTはgntnの8PCIクロック以内にad[31..0]を安定した値でドライブする。	√	
2	IUTはgntnの8PCIクロック以内にcben[3..0]を安定した値でドライブする。	√	
3	IUTはad[31..0]をドライブした1クロック後に、parをドライブする。	√	
4	IUTはgntnがリリースされたとき、ad[31..0]、cben[3..0]、parをそれぞれトライ・ステートにする。	√	

表55　テスト・シナリオ：1.14 PCIバス・マスタ・アービトレーション			
#	要求内容	Yes	No
1	gntnのディアサートとframenのアサートが同時に行われたとき、IUTはトランズアクションを終了する。	√	

表56　テスト・シナリオ：2.5 ターゲットの予約コマンド無視			
#	要求内容	Yes	No
1	IUTは予約コマンドに応答しない。	√	
2	イニシエータが、各転送処理でマスタ・アボートを検知する。	√	
3	IUTは64ビット・サイクル（デュアル・アドレス）に応答しない。	√	

表57　テスト・シナリオ：2.6 ターゲットのコンフィギュレーション・サイクル受信			
#	要求内容	Yes	No
1	IUTはすべてのタイプ-0のコンフィギュレーション・サイクル、リード／ライト・サイクルに対して、適切に応答する。	√	
2	IUTはidselがインアクティブのとき、タイプ-0のコンフィギュレーション・サイクルに応答しない。	√	

表58　テスト・シナリオ：2.8 アドレス／データ・パリティ・エラー時における、ターゲットのコンフィギュレーション・サイクル受信			
#	要求内容	Yes	No
1	コンフィギュレーション・リード／ライト・サイクル中、IUTはserrnを経由してアドレス・パリティ・エラーをリポートする。	√	
2	コンフィギュレーション・ライト・サイクル中、IUTはperrnを通じてデータ・パリティ・エラーをリポートする。	√	

表59　テスト・シナリオ：2.9 ターゲットのメモリ・サイクル受信			
#	要求内容	Yes	No
1	IUTはシングル・メモリ・リード／ライト・サイクルを正常に終了させる。	√	

表60　テスト・シナリオ：2.10 アドレス／データ・パリティ・エラー時におけるターゲットによるメモリ・サイクル受信			
#	要求内容	Yes	No
1	すべてのメモリ・リード／ライト・サイクルの期間中、IUTはserrnを経由してアドレス・パリティ・エラーをリポートする。	√	
2	すべてのメモリ・ライト・サイクルの期間中、IUTはperrnを通じてデータ・パリティ・エラーをリポートする。	√	

参考資料

このpci_aファンクションに関連した参考資料は以下の通りです。

- PCI Special Interest Group. ***PCI Local Bus Specification. Revision 2.1.*** Portland, Oregon: PCI Special Interest Group, June 1995.
- PCI Special Interest Group. ***PCI Compliance Checklist. Revision 2.1.*** Portland, Oregon: PCI Special Interest Group, June 1995.
- Altera Corporation. ***1996 Data Book.*** San Jose, California: Altera Corporation, June 1996.
- Institute of Electrical and Electronics Engineers, Inc. ***IEEE Standard VHDL Language Reference Manual*** (ANSI/IEEE Std 1076-1993). New York: Institute of Electrical and Electronics Engineers, Inc., June 1994.

日本アルテラ株式会社

〒163-0436
東京都新宿区西新宿2-1-1
新宿三井ビル私書箱261号
TEL. 03-3340-9480 FAX. 03-3340-9487
http://www.altera.com/japan/

本社 Altera Corporation

101 Innovation Drive,
San Jose, CA 95134
TEL : (408) 544-7000
http://www.altera.com