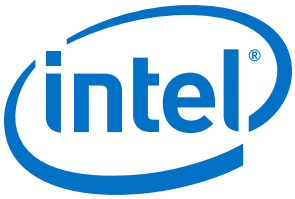


インテル® Arria®10 Avalon®-MM インターフェイスの PCI Express*デザイン例向けユーザーガイド

更新情報

フィードバック

UG-20040 | 2017.03.15

最新版をウェブからダウンロード: PDF | HTML

目次

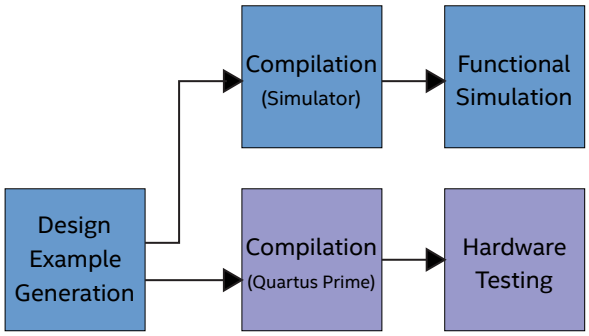
1 クイック・スタートガイド

インテル® Arria® 10 Hard IP の PCI Express* IP コアは、使用法の理解に役立つプログラミングされた I/O (PIO) のデザイン例を含んでいます。PIO 例は、ホスト・プロセッサーから対象デバイスにメモリーを転送します。低帯域幅のアプリケーションに適しています。デザイン例には、Avalon-ST から Avalon-MM へのブリッジが含まれています。このコンポーネントは、PCIe*のリンクで受信した TLP を、オンチップメモリーへの Avalon-MM リード および ライトコマンドに変換します。

このデザイン例は、Quartus® Prime ソフトウェアでシミュレーションおよび統合に必要なファイルを、自動で作成します。統合されたデザインは、Arria 10 GX FPGA 開発キットにダウンロードできます。デザイン例は広範囲に及ぶパラメーターをカバーします。しかしながら、自動的に生成したデザイン例は、PCIe IP コアのすべての可能なパラメーター設定をカバーしません。未サポートのパラメーター設定を選択した場合、生成できずエラーメッセージが表示されます。

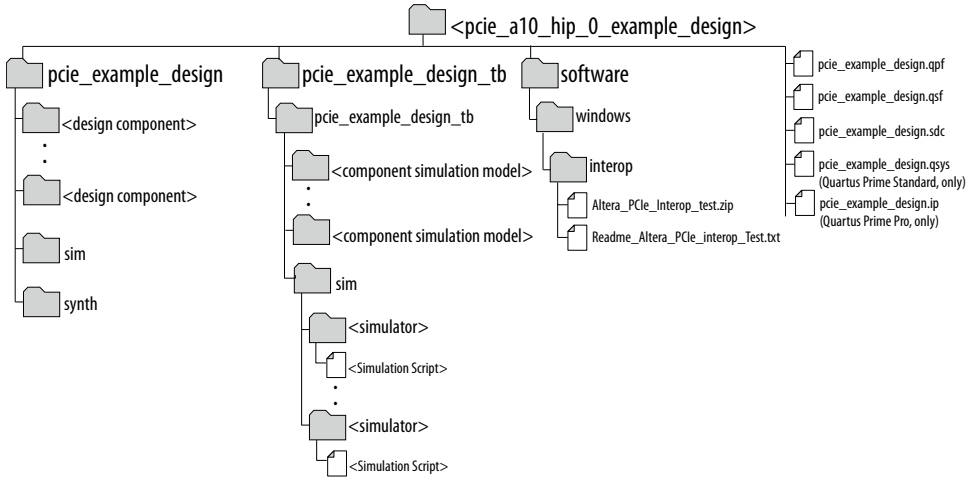
また、シミュレーションでの多くのスタティック・デザイン例は、`<install_dir>/ip/altera/altera_pcie/altera_pcie_a10_ed/example_design/a10` ディレクトリーにあるもののみ有効です。

図 -1: デザイン例での開発手順



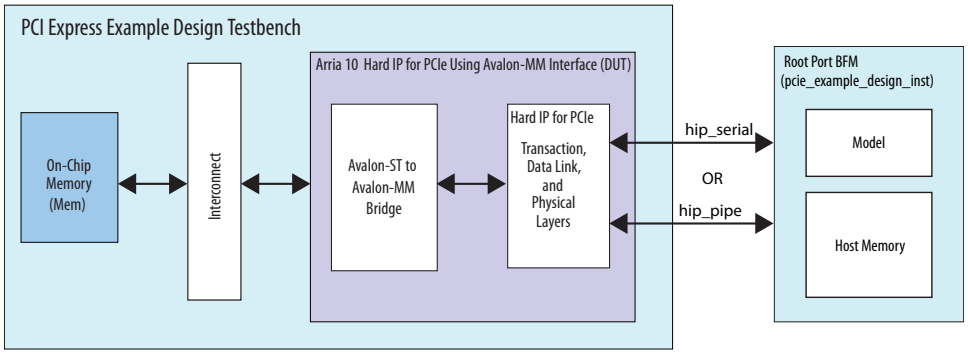
1.1 ディレクトリー構造

図 -2: 生成したデザイン例のディレクトリー構造



1.2 Avalon-MM エンドポイントでのデザイン構成

図 -3: Qsys PIO デザイン例のシミュレーション・テストベンチのブロック図



1.3 デザインの生成

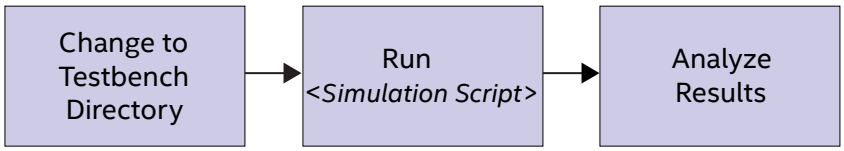
1. Qsys を起動します。
   Open System ダイアログボックスが表示されます。
2. **New** をクリックし、デザインで Quartus Prime プロジェクト名とカスタム IP バリエーション名を指定します。次に、**Create** をクリックします。
3. IP Catalog で、**Arria 10 Hard IP for PCI Express** を検索し、選択します。Parameter editor が表示されます。
4. **IP Settings** タブで、IP バリエーションのパラメーターを指定します。
5. Connections パネルで、次の接続を行います。

a. `coreclkout_hip` を `refclk` に接続
b. `rxm_bar0` を `refclk` に接続

6. デフォルトでインスタンス化された `clock_in` と `reset_in` コンポーネントを削除します。
7. **Example Design** タブでは、IP バリエーションで **PIO** デザインが使用できます。
8. **Example Design Files** の場合、**Simulation** と **Synthesis** オプションを選択します。
9. **Generated HDL Format** の場合、**Verilog** のみが使用できます。
10. **Target Development Kit** の場合、**Arria 10 FPGA Development Kit** オプションを選択します。
11. **Generate Example Design** をクリックします。ソフトウェアは、**Arria 10 FPGA Development Kit** でシミュレーションとハードウェア・テストの実行に必要なすべてのファイルを生成します。

1.4 デザインのシミュレーション

図 -4: 手順



1. テストベンチ・シミュレーション・ディレクトリーを変更します。
2. 選択のシミュレーターでシミュレーション・スクリプトを実行します。下の表を参照してください。
3. 結果を解析します。

表 1. シミュレーション実行手順

シミュレーター	作業ディレクトリー	説明
ModelSim*	*<example_design>*/pcie_example_design_tb/pcie_example_design_tb/sim/mentor/	a. `do msim_setup.tcl` b. `ld_debug` c. `run -all` d. シミュレーションが正常に終了すると、「Simulation stopped due to successful completion!」というメッセージが表示されます。
VCS*	<example_design>/pcie_example_design_tb/pcie_example_design_tb/sim/synopsys/vcs	a. sh vcs_setup.sh USER_DEFINED_SIM_OPTIONS="" b. シミュレーションが正常に終了すると、「Simulation stopped due to successful completion!」というメッセージが表示されます。
Cadence*	*<example_design>*/pcie_example_design_tb/pcie_example_design_tb/sim/cadence	a. sh ncsim_setup.sh USER_DEFINED_SIM_OPTIONS="" b. シミュレーションが正常に終了すると、「Simulation stopped due to successful completion!」というメッセージが表示されます。

図 -5: 正常な Avalon-ST PIO シミュレーション・テストベンチからの部分的なトランスクリプト

```
# INFO:           60504 ns            New Link Speed: 8.0GT/s
# INFO:           60576 ns   RP PCI Express Link Control Register (0040):
# INFO:           60576 ns       Common Clock Config: System Reference Clock Used
# INFO:           61640 ns   RP PCI Express Link Capabilities Register (01606483):
# INFO:           61640 ns        Maximum Link Width: x8
# INFO:           61640 ns      Supported Link Speed: 8.0GT/s or 5.0GT/s or 2.5GT/s
# INFO:           61640 ns                 L0s Entry: Supported
# INFO:           61640 ns                 L1  Entry: Not Supported
# INFO:           61640 ns          L0s Exit Latency: 2 us to 4 us
# INFO:           61640 ns          L1  Exit Latency: Less Than 1 us
# INFO:           61640 ns               Port Number: 01
# INFO:           61768 ns   RP PCI Express Device Control Register (5010):
# INFO:           61768 ns   Error Reporting Enables: 0
# INFO:           61768 ns          Relaxed Ordering: Enabled
# INFO:           61768 ns               Max Payload: 128 Bytes
# INFO:           61768 ns              Extended Tag: Disabled
# INFO:           61768 ns          Max Read Request: 4KBytes
# INFO:           61768 ns   RP PCI Express Device Status Register (0000):
# INFO:           62096 ns Configuring Bus 000, Device 000, Function 00
# INFO:           62096 ns   RP Read Only Configuration Registers:
# INFO:           62096 ns                 Vendor ID: 1172
# INFO:           62096 ns                 Device ID: E001
# INFO:           62096 ns               Revision ID: 01
# INFO:           62096 ns                Class Code: FF0000
# INFO:           62096 ns             Interrupt Pin: INTA# used
# INFO:           62784 ns BAR Address Assignments:
# INFO:           62784 ns BAR    Size       Assigned Address  Type
# INFO:           62784 ns BAR0   Disabled
# INFO:           62784 ns BAR1   Disabled
# INFO:           62784 ns ExpROM Disabled
# INFO:           66680 ns Completed configuration of Endpoint BARs.
# INFO:           67728 ns TASK:downstream_loop
# INFO:           68584 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# INFO:           69448 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# INFO:           70296 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# INFO:           71160 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# INFO:           72008 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# INFO:           72864 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# INFO:           73720 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# INFO:           74568 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# INFO:           75432 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# INFO:           76280 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
# SUCCESS: Simulation stopped due to successful completion!
```

1.5 ハードウェアでのテストとデザインの統合

図 -6: 手順

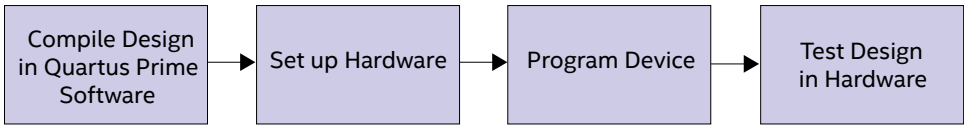
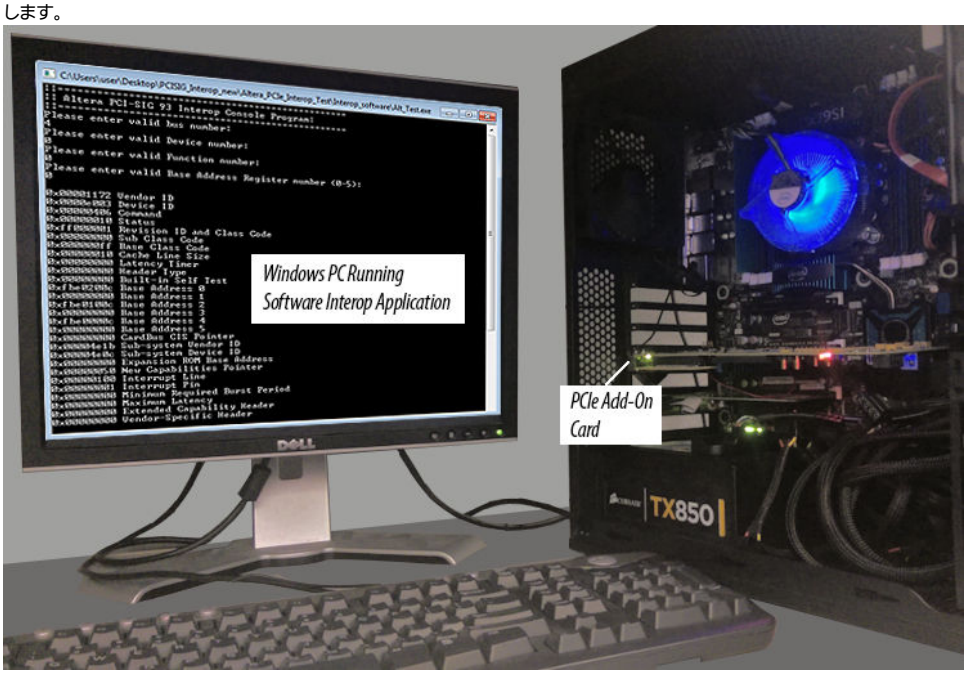


図 -7: **Arria 10 GX FPGA 開発キットでの PCI Express デザイン例のテスト用ソフトウェア・アプリケーション**

Windows PC 上で動作するソフトウェア・アプリケーションは、すべての PCI Express デザイン例で同じハードウェア・テストを実行します。



Arria 10 GX FPGA 開発キットで PCI Express デザイン例をテストするためのソフトウェア・アプリケーションは、32 ビットと 64 ビット Windows プラットフォームの両方で使用可能です。このプログラムは次のタスクを実行します。

1. Configuration Space、レーンレート、およびレーン幅を印刷します。
2. 指定された BAR にオフセット 0x00000000 で 0x00000000 を書き込み、メモリーを初期化して読み込みます。
3. 指定された BAR のオフセット 0x00000000 に 0xABCD1234 を書き込みます。それを読み込んで比較します。

正常に終了すると、テストプログラムは「PASSED」のメッセージを表示します。

次の手順に従って、Quartus Prime ソフトウェアでデザイン例をコンパイルします。

1. Quartus Prime ソフトウェアを起動し、<*example_design*>`pcie_example_design.qpf` を開けます。
2. **Processing** > menu で、**Start Compilation** を選択します。

   デザイン例とデザイン・コンポーネントのタイミング制約は、コンパイル時に自動的にロードされます。

ハードウェアでデザイン例をテストするには、次の手順を実行します。

1. <*example_design*>`/software/windows/interop` ディレクトリーで、`Altera_PCIe_Interop_Test.zip` を解凍します。

注意: ハードウェア・テストの実行の指示について、同じディレクトリー内の `readme_Altera_PCIe_interop_Test.txt` ファイルを参照することも可能です。

2. **altera_pcie_win_driver.inf** で、Windows ホストマシーンに PCIe 用の インテル FPGA Windows デモドライバーをインストールします。

   注意: コンポーネント GUI で指定されたデフォルト Vender ID または Device ID を変更した場合は、**altera_pcie_win_driver.inf** でもこれらを変更する必要があります。

   a. <*example_design*> ディレクトリーで、Quartus Prime ソフトウェアとコンパイルするデザインを起動します (**Processing** > **Start Compilation**)。

   b. 開発ボードをホスト・コンピューターに接続します。

   c. 生成した `.sof` ファイルで、開発ボードに FPGA をコンフィグレーションします (**Tools** > **Programmer**)。

   d. Windows デバイス・マネージャーを開き、ハードウェアの変更をスキャンします。

   e. 不明な PCI デバイスとしてリストされている インテル FPGA を選択し、**Windows_driver** ディレクトリーの適切な 32 ビットまたは 64 ビット・ドライバー (**altera_pice_win_driver.inf**) を指します。

   f. ドライバーが正常に読み込まれた後、Windows デバイス・マネージャーに新しいデバイス名の **Altera PCI API Device** が表示されます。

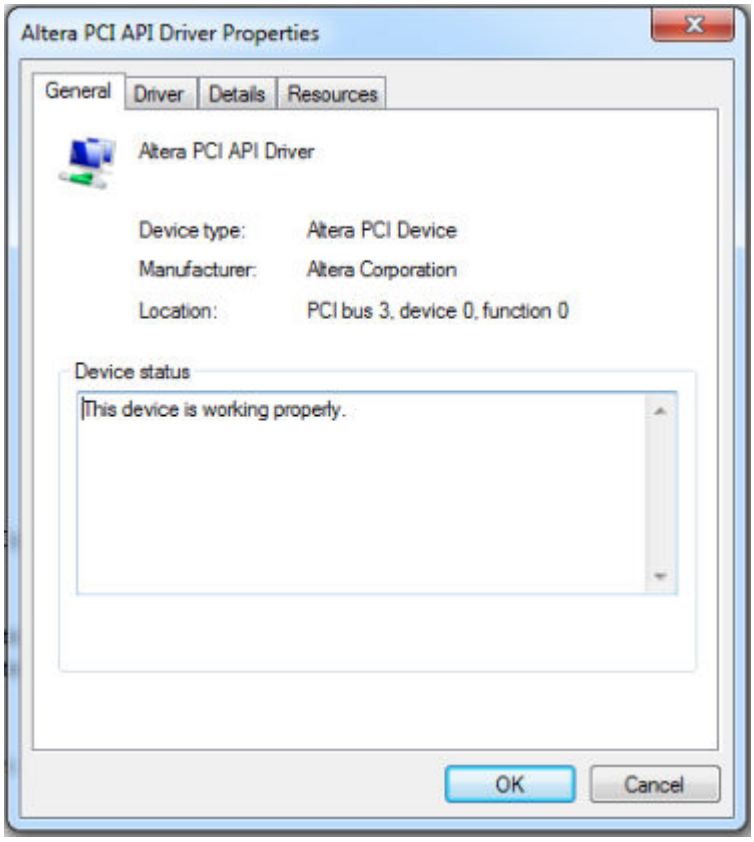
   g. Windows デバイス・マネージャーのリストにある **Altera PCI API Device** で、Bus、Device、および Function Number を決定します。

      i. デバイスの下の **Altera PCI API Driver** タブを展開します。

      ii. **Altera PCI API Device** で右クリックし、**Properties** を選択します。

      iii. Bus、Device、およびデバイスの Function Number に注意してください。次の図で一例を示します。

図 -8:　新しい PCIe デバイスでの Bus、Device、および Function Number の決定



3. *<example_desing>*`/software/windows/interop/Altera_PCIe_Interop_Test/Interop_software` ディレクトリーで、`Alt_Test.exe` をクリックします。
4. プロンプトが表示されたら、Bus、Device、および Function Number を入力し、IP コアのパラメーター化の際に指定した BAR 番号 (0-5) を選択します。

   注意: ハードウェア設定での Bus、Device、および Function Number は、異なる場合があります。
5. テストが正常に終了すると、「PASSED」のメッセージが表示されます。

関連情報

Arria 10 GX FPGA Development Kit

2 デザイン例の説明

2.1 デザイン階層と一致する SignalTap II Debug File の作成

Arria 10 デバイスでは、Quartus Prime スタンダード・エディション・ソフトウェアは、`build_stp.tcl` と `<ip_core_name>.xml` の2つのファイルを生成します。これらのファイルで、デザイン階層に一致しているプローブポイントを含んだ SignalTap II ファイルの生成できます。

Quartus Prime ソフトウェアはこれらのファイルを `<IP core directory>/synth/debug/stp/` ディレクトリーに保存します。

Quartus Prime ソフトウェアでデザインを合成します。

1. **View** > **Utility Windows** > **Tcl Console** をクリックし、Tcl コンソールを開きます。
2. Tcl コンソールで、次のコマンドを実行します。
   `source <IP core directory>/synth/debug/stp/build_stp.tcl`
3. 次のコマンドを入力し、STP ファイルを生成します。
   `main -stp_file <output stp file name>.stp -xml_file <input xml_file name>.xml -mode build`
4. プロジェクトにこの SignalTap II ファイル (**.stp**) を追加するために、**Project** > **Add/Remove Files in Project** を選択します。次に、デザインをコンパイルします。
5. **Tools** > **Programmer** をクリックし、FPGA をプログラムします。
6. **Quartus Prime** > **Tools** > **SignalTap II Logic Analyzer** をクリックし、SignalTap II Logic Analyzer を開始します。

   ソフトウェア生成スクリプトは、`<output stp file name>.stp` で SignalTap II のアクイジション・クロックを割り当てない可能性があります。その結果、Quartus Prime ソフトウェアは `auto_stp_external_clock` というクロックピンを自動的に作成します。適切なクロック信号を各 STP インスタンスの SignalTap II サンプリング・クロックとして手動で置き換える必要がある場合があります。
7. デザインを再コンパイルします。
8. **Run Analysis** をクリックし、IP コアの状態を監視します。

   デザインでの使用不可を表す赤色の信号または SignalTap II インターフェイスが見られる場合があります。たいていの場合、これらの信号やインターフェイスを支障なく無視できます。これらは、ソフトウェアが幅の広いバスを生成し、デザインに含まないインスタンスが存在するために見られます。

2.2 Arria 10 開発キット･コンジット･インターフェイス

Arria 10 開発キット･コンジット･インターフェイス信号は、デザインを Arria 10 FPGA 開発キットに接続できるオプション信号です。このインターフェイスは、コンポーネント GUI の **Configuration、Debug、および Extension Options** タブで **Enable Arria 10 FPGA Development Kit connection** を選択し、イネーブルします。`devkit_status` 出力ポートは 、デバッグに役立つ信号を含みます。

表 2.

信号名	入力 / 出力	説明
`devkit_status[255:0]`	出力	`Devkit_status[255:0]` バスは次の信号状態から構成されています。 • `devkit_status[1:0]: current_speed` • `devkit_status[2]: derr_cor_ext_rcv` • `devkit_status[3]: derr_cor_ext_rpl` • `devkit_status[4]: derr_err` • `devkit_status[5]: rx_par_err` • `devkit_status[7:6]: tx_par_err` • `devkit_status[8]: cfg_par_err` • `devkit_status[9]: dlup` • `devkit_status[10]: dlup_exit` • `devkit_status[11]: ev128ns` • `devkit_status[12]: ev1us` • `devkit_status[13]: hotrst_exit` • `devkit_status[17:14]: int_status[3:0]` • `devkit_status[18]: l2_exit` • `devkit_status[22:19]: lane_act[3:0]` • `devkit_status[27:23]: ltssmstate[4:0]` • `devkit_status[35:28]: ko_cpl_spc_header[7:0]` • `devkit_status[47:36]: ko_cpl_spc_data[11:0]` • `devkit_status[48]: rxfc_cplbuf_ovf` • `devkit_status[49]: reset_status` • `devkit_status[255:50]: Reserved`
`devkit_ctrl[255:0]`	入力	`devkit_ctrl[255:0]`バスは次の信号状態から構成されています。オプションでこれらのピンをバイパス適合テストなどの PCI-SIG 準拠テスト用のオンボードスイッチに接続できます。 • `devkit_ctrl[0]: test_in[0] is typically set to 1'b0` • `devkit_ctrl[4:1]: test_in[4:1] is typically set to 4'b0100` • `devkit_ctrl[6:5]: test_in[6:5] is typically set to 2'b01` • `devkit_ctrl[31:7]: test_in[31:7] is typically set to 25'h3` • `devkit_ctrl[63:32]: is typically set to 32'b0` • `devkit_ctrl[255:64]: is typically set to 192'b0`

A 改訂履歴

表 3. 改訂履歴

日付	バージョン	変更内容
2017 年 3 月 15 日	16.1.1	商標を「インテル」へ変更。
2016 年 10 月 31 日	16.1	初版