

この資料は英語版を翻訳したもので、内容に相違が生じる場合には原文を優先します。こちらの日本語版は参考用としてご利用ください。設計の際には、最新の英語版で内容をご確認ください。

このアプリケーション・ノートでは、Stratix® V デバイスで Interlaken PHY の IP インタフェースを使用してプロトコル固有の IP (Intellectual Property) コアの実装とシミュレーションを説明します。このアプリケーション・ノートでの参照デザイン・ファイルを使用して PHY の IP インタフェースおよび Stratix V デバイス・ファミリのデザイン・フローを評価します。

Quartus® ソフトウェア・バージョン 10.1 で始まる PHY の IP インタフェースは、自動的にフィジカル・コーディング・サブレイヤ (PCS) モジュールのための Stratix V デバイスの設定パラメータを設定して、ユーザー・コントロールのためにフィジカル・メディア・アタッチメント (PMA) モジュール内に少数のパラメータを残します。

PHY の IP デザインはモジュラーであり、標準インタフェースを使用します。すべての PHY の IP モジュールにはコントロール・レジスタおよびステータス・レジスタをアクセスするための Avalon® Memory-Mapped (Avalon-MM) 管理インタフェースおよびデータ転送用の MAC ブロックに接続するための Avalon Streaming (Avalon-ST) インタフェースが含まれています。ほとんどの PCS と PMA の関数は、FPGA のリソースを節約するためにハード・ロジックで実装されています。PCS および PMA ブロックは分離またはバイパスすることができません。キー・インタフェースは外部インタフェースからのシリアル・データ、内部 PLD インタフェースの平行・データ、および内部制御のインタフェースです。

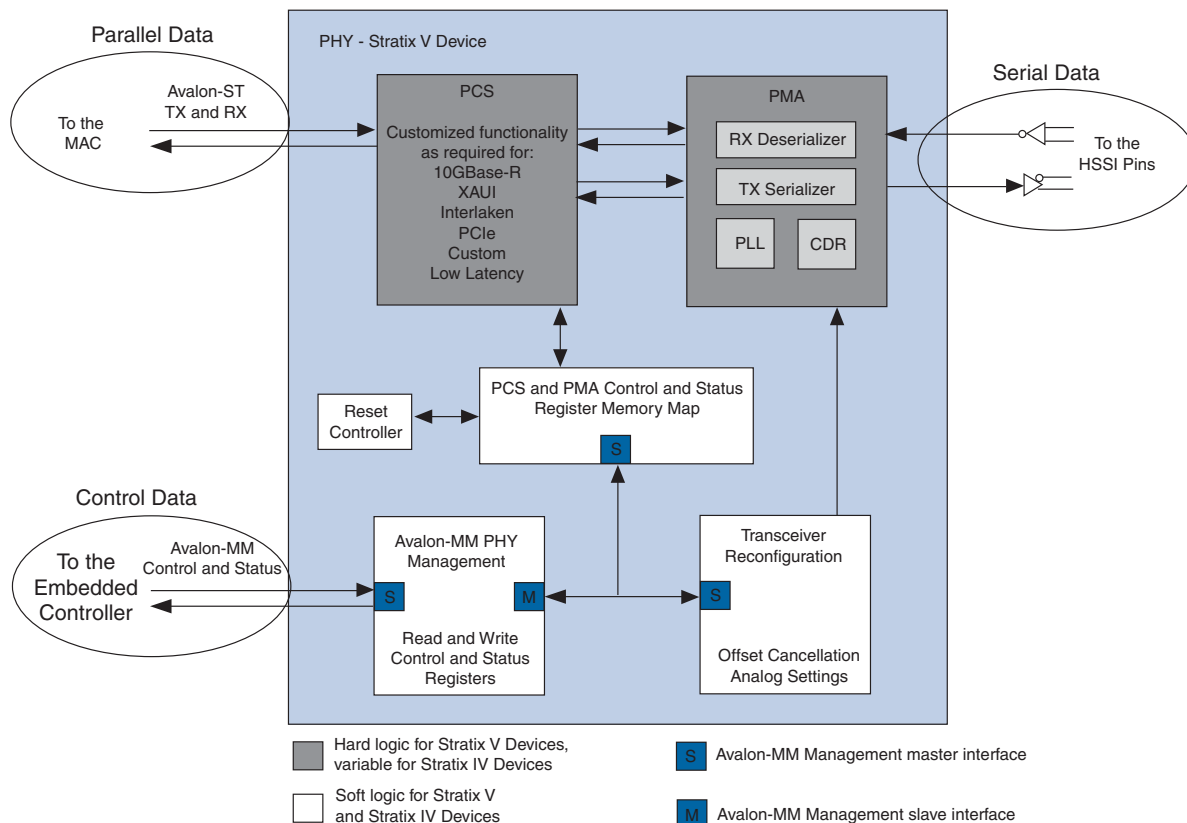
PHY の IP インタフェースは、次のものがあります。

- Avalon-ST TX/RX インタフェース — PCS の TX および RX パスのデータ・パケットから送信 (TX) および受信 (RX) のデータ・ストリームのための PLD 平行・インタフェース。
- Avalon-ST レディ・インタフェース — RX と TX のパスがアクティブおよびデータの受信または送信の準備ができていることを示す信号。
- シリアル RX および TX インタフェース — トランシーバの入力と出力のシリアル・データ。
- Avalon Memory-Mapped (Avalon-MM) 管理インタフェース — PHY の IP の内部レジスタと設定にアクセスするために間接的なアドレッシングです。PHY IP は `phy_mgmt_address [8:0]` ワード・アドレスとして予想しますが、しかし、バイト・アドレスは、マスタへの標準インタフェースであるため、Avalon マスタは、バイト・アドレス (`mgmt_address [9:0]`) を受け取り、ワード・アドレスに変換します。
- クロック・インタフェース — クロック・インタフェースは RX と TX のためにユーザーのクロック出力を含みます。他の入力には PLL (Phase-Locked Loop)、TX と RX クロック、および PHY の IP 管理クロック。



図 1 に、典型的な PHY の IP ブロックを示します。

図 1. PHY の IP インタフェース



PHY の IP による Interlaken インタフェースの実装

Interlaken インタフェース・プロトコルは、ネットワーク・アプリケーションのために狭くて高速のチップ間インタフェースのデザインが可能になります。XAUI および SPI 4.2 インタフェースに比べて Interlaken インタフェースの多くの利点としては、低い I/O 数、フロー制御、低オーバー・ヘッド・フレーミング、および広範な整合性チェックです。Interlaken はスケーラブルであり、SPI 4.2 構造に基づいて 10-、40-、および 100-Gbps のシステムをサポートします。Interlaken コンフィギュレーションでは、Stratix V デバイスはレーンごとに 3.125、5.0、6.25、6.375、または 10.3125 Gbps の送信データ・レートをサポートします。

Interlaken PHY の IP 例のリファレンス・デザイン・フロー

ここでは、Stratix V デバイスで Interlaken PHY の IP リファレンス・デザイン・フローの実行方法について説明します。

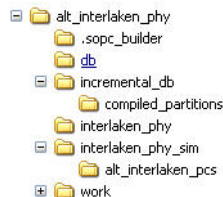
ハードウェアおよびソフトウェアの要件

Quartus II ソフトウェアのバージョン 10.1 またはそれ以上を使用してデザインを実装し、コンパイルします。現在ハードウェア・プラットフォームは存在しておらず、リファレンス・デザインのシミュレーションのみが完了できます。

次の PHY IP の作成は、MegaWizard™ Plug-In Manager を使用して PHY IP の生成を示します (PHY IP はリファレンス・デザインの一部として作成されている)。

図 2 に、リファレンス・デザインのディレクトリ構造を示します。

図 2. リファレンス・デザインのディレクトリ構造



Quartus II ソフトウェアのトップ・レベル作成

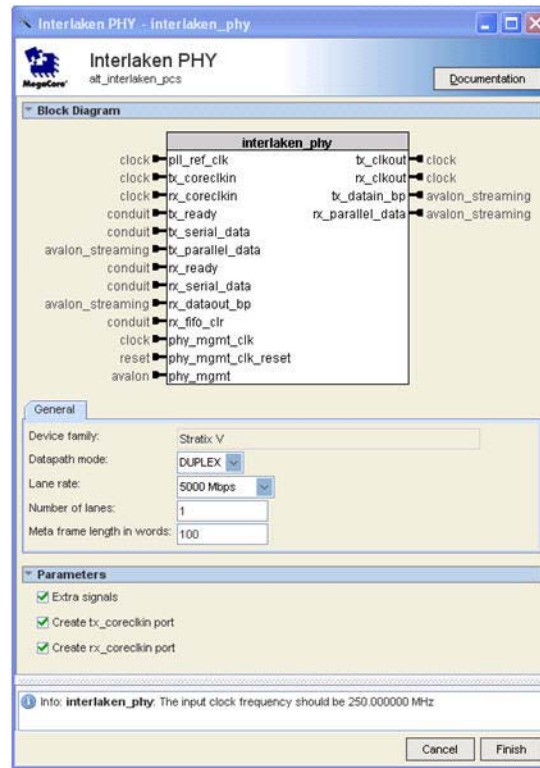
トランシーバに設定する多くの変数が存在する ALTGX メガファンクションを使用することを比較すると PHY の IP 実装では、トランシーバのプロトコル特有のインタフェースの簡単な作業になります。

新規 Quartus II のプロジェクト・ファイルおよびトップ・レベルのデザイン・フローを作成するには、以下のステップに従います。

1. トップ・レベルの Verilog ファイルに **top_interlaken_phy.v** の名前を付けます。
2. MegaWizard Plug-In Manager を実行して新しいカスタム・メガファンクションを作成します。
3. **Interface Plug-Ins** ドロップ・ダウン・メニューを選択して **Interlaken PHY v10.1** をクリックします。Verilog ファイルの **interlaken_ip** として出力を選択します。
4. **Interlaken PHY** ダイアログ・ボックスで、データ・パス・モード、レーン・レート、およびレーン数を選択できます。例えば、図 3 に示すように、一車線のための **5 Gbps** のデータ・レートで送信および受信するために、**duplex mode** を選択します。
5. 割り当てられているメタ・フレーム長を **100 word** に設定します。
6. パラメータ・インタフェース信号を選択します。余分な信号が **word_lock**、**sync_lock**、および **CRC32 error** であり、**rx_parrellel_data** バスの一部です。**tx_coreclk** 入力ポートは TX FIFO の書き込み側を駆動するために作成され、**rx_coreclk** 入力ポートは RX FIFO の書き込み側を駆動するために作成されます。
7. **Finish** をクリックして PHY IP を作成します。

図 3 に、Interlaken PHY の IP 設定を示します。

図 3. Interlaken PHY の IP



トップ・レベルの Verilog デザインでは、Avalon マスタ、擬似ランダム・バイナリ・シーケンス (PRBS) ジェネレータとチェッカー、および周波数チェッカーの合成可能なコードで構成されています。必要なすべてのデザイン・ファイルは、リファレンス・デザインのパッケージに含まれています。すべてのモジュールは、このリファレンス・デザインの一部として作成されます。ユーザーのデザインで使用する各 PHY の IP コアに Avalon マスタを適用します。トップ・レベル・デザインは次のモジュールをインスタンス化します。

- ***Interlaken_phy.v*** (MegaWizard Plug-In Manager で生成される。)
- ***Custom_avalon_master.v***
- ***Prbs_generator.v***
- ***Prbs_checker.v***
- ***Frequency_checker.v***

表 1 に、必要なステータスおよびエラー信号を示します。

表 1. ステータスおよびエラー信号

ステータスおよび エラー信号	説明
rx_sync_lock	4 つの同期ワードが検出されたときに、sync_lock ステータス信号を表示します。
rx_word_lock	最初のアラインメント・パターンが検出されたときに、word_lock ステータス信号を表示します。
rx_framing_error	フレーミング・エラーが検出されたときに、「1」に変更します。
rx_crc32_error	CRC32 エラーが検出されたときに、「1」に変更します。
rx_scrambler_error	スクランブラ・エラーが検出されたときに、「1」に変更します。
rx_sync_word_error	missing_sync エラーが検出されたときに、「1」に変更します。
txclkfreq_error	tx_clkout 周波数は範囲にない場合は、「1」に変更します。
rxclkfreq_error	rx_clkout 周波数は範囲にない場合は、「1」に変更します。
data_error	PRBS チェッカがエラーを検出するときに、「1」に変更します。
verifier_lock	ベリファイアがエラーを検出するときに、「1」に変更します。

カスタム Avalon マスタ

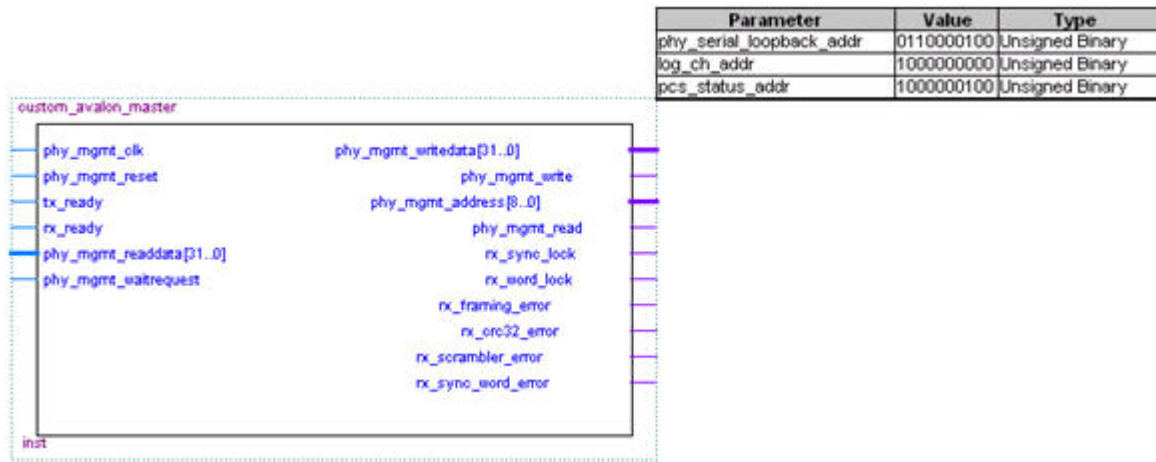
Quartus II ソフトウェア・バージョン 10.1 以前では、SOPC Builder または Qsys は、任意の Avalon インタフェースのインスタンスを作成することはできません。したがって、アルテラはアドレスのバイトを受け入れるようにおよび PHY IP にワード・アドレスを提供するために *custom_avalon_master.v* というカスタム Avalon マスタをデザインされます。

Avalon マスタは PHY IP にバイト・アドレスの mgmt_address [9:0] を受け入れ、ワード・アドレスの address mgmt_address [8:0] を提供します。カスタム Avalon マスタは、Interlaken インタフェースにデザインされましたが、それらはすべて同じような構造を持っているので、他の PHY IP に対して同じマスタを使用することができます。この例では、PMA の制御とステータス・レジスタ (CSR) を行使する内部シリアル・ループ・バック機能が有効になっています。

カスタム Avalon マスタは、FPGA ロジック・コアと PHY の IP 間のインタフェースの中心です。このモジュールを使用して他の PHY の IP プロトコルに適応し、変更できます。各 PHY の IP プロトコル・タイプは、内部レジスタの異なるセットを持っていて、Avalon マスタを介して読み取りや書き込みすることができます。PHY の IP とロジック・コア間の入出力のデータ・バス幅は 32 ビットです。

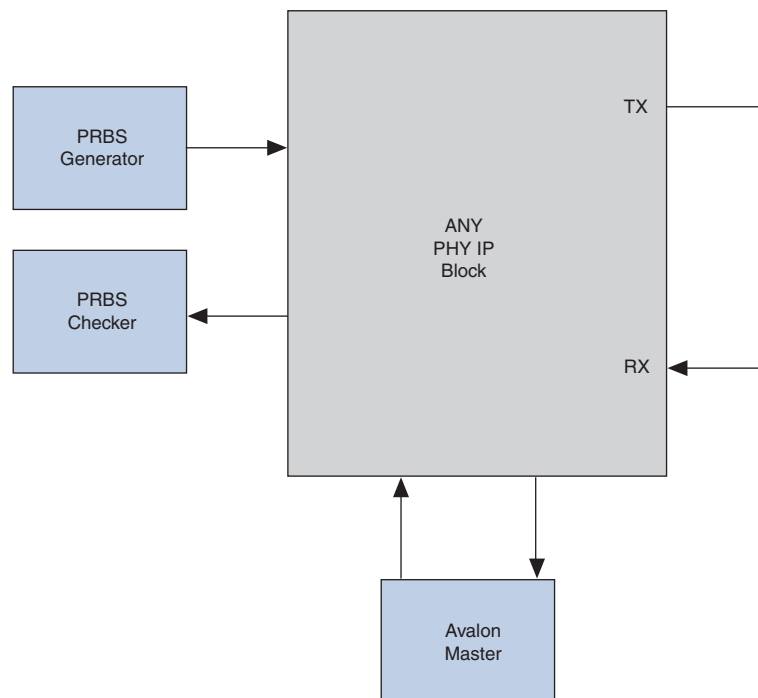
図 4 に、生成されたシンボルの custom_avalon_master 信号フローを示します。

図 4. カスタム Avalon マスタの信号フロー (custom_avalon_master)



このリファレンス・デザインは、Interlaken プロトコルのためであっても、ユーザーが他の PHY の IP プロトコルの主要なビルディング・ブロックの一部を再利用することができます。図 5 に示すように、リファレンス・デザインで再利用できるコードは PRBS ジェネレータ、PRBS チェッカ、および最も重要なコードの Avalon マスタ・モジュールです。

図 5. PHY の IP テストのキー・モジュール



PRBS ジェネレータとチェッカー

64 ビットの PRB23 データ・ジェネレータおよびチェック・モジュールは ループバック・テストに使用されます。

トップ・レベルのテストベンチ

ここでは、RTL (Register Transfer Level) のシミュレーション・テストベンチおよびリファレンス・デザインのシミュレーション結果について説明します。


RTL シミュレーション・テストベンチはソフトウェア・レベルで RTL デザインの検証を提供します。リファレンス・デザインは PRBS をテスト・パターン・ソースとして使用して、TX および RX ファンクション間のループ・バック環境のためにシンプルなテストベンチを提供します。テストベンチはステータスとエラー信号をチェックして必要な出力情報を表示します。

テスト対象のデバイス (DUT) は、PHY の IP です。クロック・ジェネレータはトランシーバ基準クロック (pll_ref_clk) およびデータ・パス・クロック (phy_mgmt_clk) を含む PHY の IP クロック・インタフェース用のテストベンチに作成されています。

ModelSim Example Script を実行

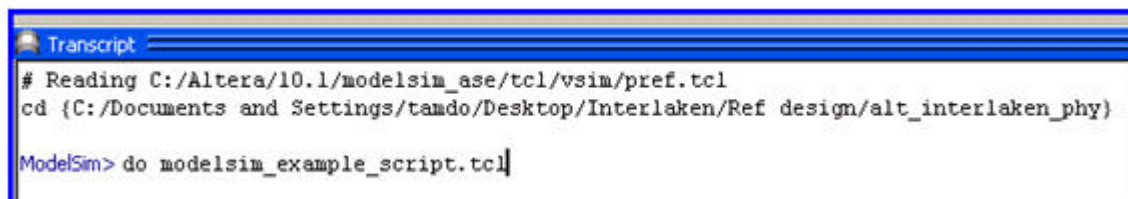
ModelSim® Altera ソフトウェアでは、リファレンス・デザインのシミュレーションを実行するには、次の手順を実行します。

1. ModelSim Starter Edition v 6.6c 以降を起動します。
2. File メニューで、**Change Directory** をクリックして、インストールされたリファレンス・デザインの場所を変更します。

 **modelsim_example_script.tcl** ファイルはディレクトリ・フォルダを配置されていることを確認します。

3. ModelSim コマンド・プロンプトで、[図 6](#) に示すように、do modelsim_example_script.tcl を入力してシミュレーションを実行します。

図 6. ModelSim Script



```
Transcript
# Reading C:/Altera/10.1/modelsim_ase/tcl/vsim/pref.tcl
cd {C:/Documents and Settings/tamdo/Desktop/Interlaken/Ref design/alt_interlaken_phy}
ModelSim> do modelsim_example_script.tcl
```

.tcl スクリプト・ファイルは ModelSim で Interlaken PHY のインスタンスのコンパイルおよびシミュレーションの例です。必要なすべての Verilog HDL または VHDL ライブラリ・ファイルは、リファレンス・デザインのプロジェクトに含まれ、どちらかの言語を選択できます。新たな要件に応じてこのスクリプト・ファイルを変更して、デザインをコンパイルするおよびシミュレートするために使用できます。

トップ・レベルのテストベンチ・ファイルの `top_interlaken_phy_tb.v` を使用してシミュレーションを完了できます。このファイルはテストおよびクロックのジェネレータで、トップ・レベル PHY の IP プロジェクトをインスタンス化します。テストベンチはステータスおよびエラー信号をチェックし、すべてのエラー情報を表示します。

シミュレーション結果

成功したシミュレーションは、以下のキー・ステータス信号を持っています。

- この例では、TX PCS は 56661 ps で準備が出来上がっている。
- この例では、RX PCS は 156651 ps で準備が出来上がっている。
- 7055961 ps での rx_sync_lock。
- 7102623 ps での rx_word_lock。
- 7740000 ps でのベリファイア・ロック。

図 7 に、正常なシミュレーションの終了時に ModelSim シミュレーションの出力を示します。

図 7. 正常な ModelSim でのシミュレーション結果

```
# Info: reference_clock_frequency = 250.000000 Mhz
# Info: output_clock_frequency = 800 ps
# Info: phase_shift = 200 ps
# Info: duty_cycle = 50
# Info: sim_additional_refclk_cycles_to_lock = 0
# Info: output_clock_high_period = 400.000000
# Info: output_clock_low_period = 400.000000
# Info: Instantiating FRBS Module with parameters:FRBS=      23 DATAWIDTH=      64
# Info: Instantiating FRBS Module with parameters:FRBS=      23 DATAWIDTH=      64
# Info: RX PCS is not ready
# Info: TX PCS is not ready
# Info: TX PCS is ready at          56661
# Info: RX PCS is ready at          156651
#
# Info: rx_sync_lock = phy_mgmt_readddata[25] is detected, 4 sync words are detected at          7055961
# Info: rx_word_lock = phy_mgmt_readddata[24] is detected, first alignment pattern is detected at          7102623
#
# Info: verifier lock is detected at          7740000
#
** Note: $finish : ./top_interlaken_phy_tb.v(271)
# Time: 20020 ns Iteration: 0 Instance: /top_interlaken_phy_tb
# 1
# Break in Module top_interlaken_phy_tb at ./top_interlaken_phy_tb.v line 271
# Simulation Breakpoint: 1
# Break in Module top_interlaken_phy_tb at ./top_interlaken_phy_tb.v line 271
# MACRO ./modelsim_example_script.tcl PAUSED at line 235
VSIM(paused)>
```

信号を見る場合に、ModelSim Objects ウィンドウから選択します。(図 8)。

Wave ウィンドウ (図 9) で異なる信号を表示するには、以下のステップに従います。

1. Wave ウィンドウ (図 8) で表示する信号をハイライトします。
2. Add メニューで、**To wave** および **Signals in design** を選択します (図 9)。
3. VSIM コマンド・プロンプトで、run 1000 を入力して、Wave ウィンドウを開始します。

図 8. ModelSim の Objects ウィンドウ

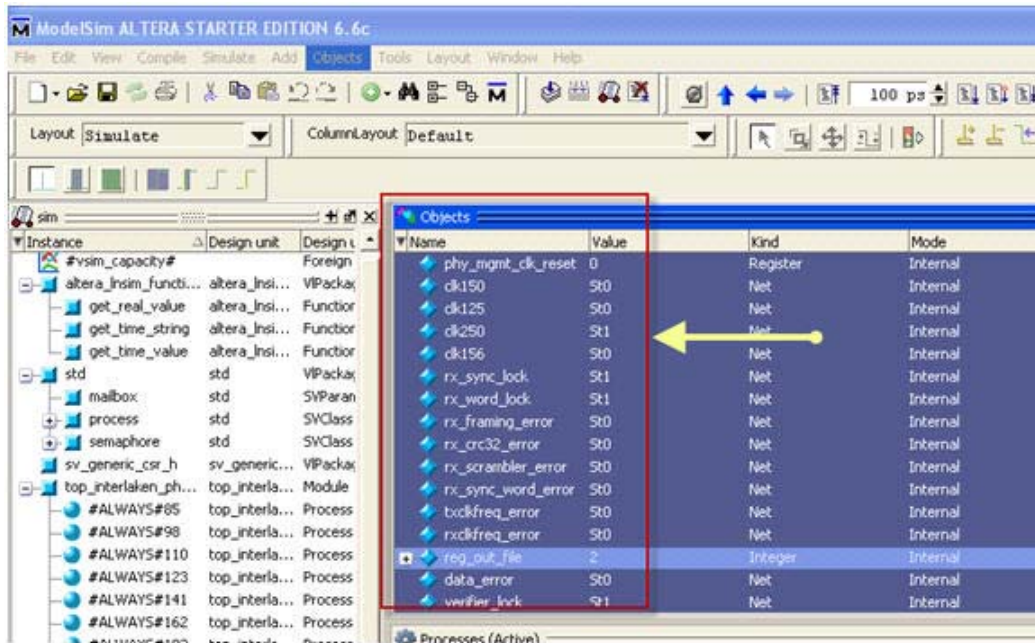
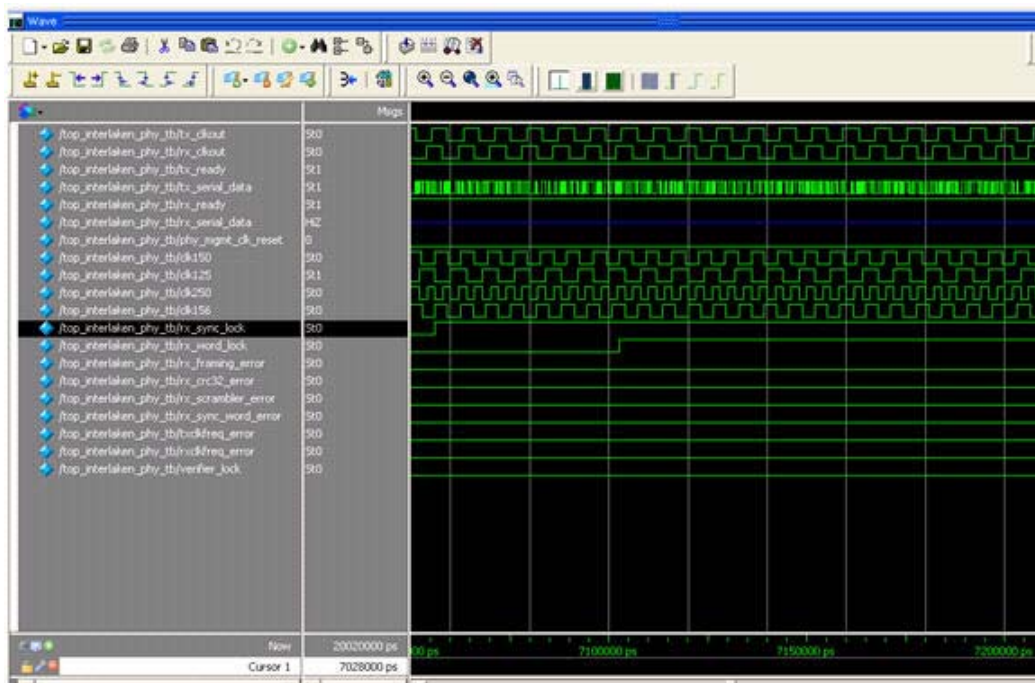


図 9. Wave ウィンドウ



改訂履歴

表 2 に、このアプリケーション・ノートの改訂履歴を示します。

表 2. 改訂履歴

日付	バージョン	変更内容
2010 年 12 月	1.0	初版。