ISim ハードウェア協調シミュレーション : Virtex-5 エンベデッド イーサネット MAC を介したライブ イーサネット トラフィックの処理

UG819 (v13.3) 2011 年 11 月 11 日

XILINX®

本資料は英語版 (v.13.3) 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。
資料によっては英語版の更新に対応していないものがあります。
日本語版は参考用としてご使用の上、最新情報につきましては、必ず最新英語版をご参照ください。

この資料に関するフィードバックおよびリンクなどの問題につきましては、jpn_trans_feedback@xilinx.com までお知らせください。いただきましたご意見を参考に早急に対応させていただきます。なお、このメール アドレスへのお問い合わせは受け付けておりません。あらかじめご了承ください。

改訂履歴

次の表は、このマニュアルの改訂履歴を表示しています。

日付	バージョン	説明
2011/03/18	13.1	初版
2011/07/06	13.2	リリースに合わせて更新
2011/10/04	13.3	リリースに合わせて変更

目次

XILINX®

チュートリアル

このチュートリアルでは、ISim ハードウェア協調シミュレーションを使用してイーサネット デザインを実行するのに必要な手順を 5 つのセクションに分けて示します。手順は順番に実行してください。

このチュートリアルは、次のセクションから構成されています。

1. CORE Generator™ ツールで Virtex®-5 Embedded Tri-mode Ethernet MAC Wrapper を使用して、Virtex-5 イーサネット MAC サンプル デザインを生成します。
2. イーサネット MAC ラッパーとパケット プロセッサをバインドするテストベンチを作成します。
3. カスタム制約ファイルを作成して、サンプル デザインの ISim で制御するポートと外部 I/O にマップするポートを指定します。
4. ハードウェア協調シミュレーションを実行するサンプル デザインのテストベンチをコンパイルします。
5. ターゲット FPGA ボードをコンピューターに接続し、ISim シミュレーションを実行します。

概要

このチュートリアルでは、ISim ハードウェア協調シミュレーションを使用して、Virtex®-5 FPGA エンベデッド トライモード イーサネット MAC を介するライブ イーサネット トラフィックをキャプチャし、HDL (ハードウェア記述言語) テストベンチからキャプチャされたパケットを処理する方法を説明します。

イーサネットを使用する FPGA デザインを開発する際、イーサネット MAC (Machine Access Control)、マルチギガビット トランシーバー (MGT) (イーサネット MAC と PHY のインターフェイスに SGMII を使用する場合)、およびイーサネット PHY を含むデザイン全体を検証するのは困難です。従来は、ソフトウェアでデザイン全体をシミュレーションするか、ハードウェアでデザイン全体を実行していました。すべてをソフトウェアでシミュレーションする方法は、次の 2 つの点で便利です。

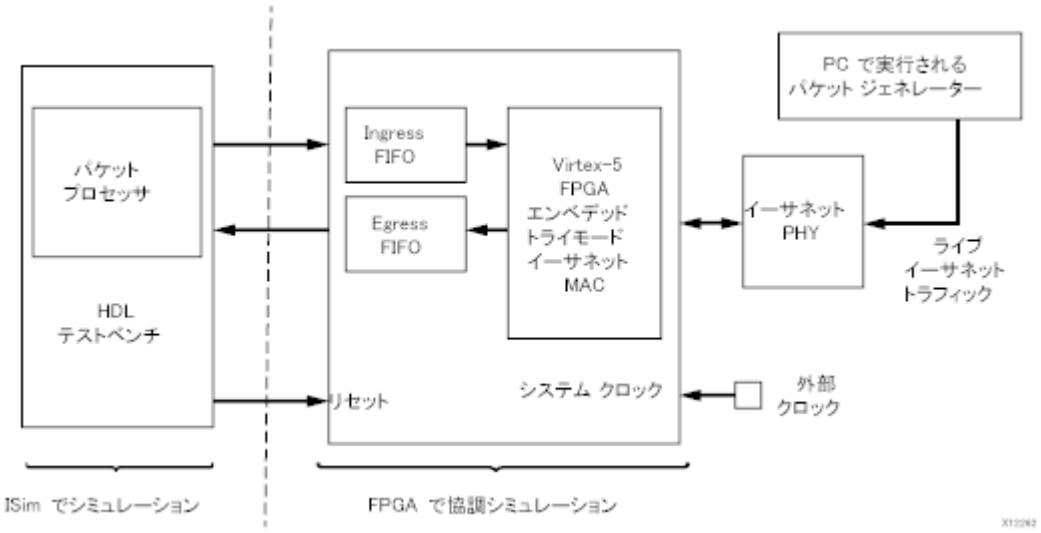
- デザインを完全に可視化できる
- テストベンチまたはデザインを変更して即再検証できる

イーサネット MAC、MGT、および PHY を網羅するテストベンチを作成し、現実的なシミュレーション速度を達成するのは困難なため、通常は MGT および PHY のシミュレーションで詳細事項をスキップしたり、MGT および PHY のシミュレーションを完全にスキップすることになります。

デザインをハードウェアで実行するとこれらの問題は回避されますが、デザインの可視性が低下し、ハードウェアでのテストベンチの設定および変更は複雑です。

ISim の協調シミュレーションでは、デザインの一部をハードウェアで実行し、残りの部分をソフトウェアでシミュレーションできます。たとえば、イーサネット MAC、MGT、および PHY をハードウェアで実行すると、これらを正確にモデル化でき、シミュレーションを高速に実行できます。テストベンチおよびデザインに含まれるパケット プロセッサなどの開発中のアプリケーション ロジックは、変更、検証、デバッグできるようにソフトウェアでシミュレーションする必要があります。次の図に、イーサネット パケット処理のデザインを分割して、ISim ハードウェア協調シミュレーションを利用する方法を示します。

ISim ハードウェア協調シミュレーション用にイーサネット デザインを分割する方法



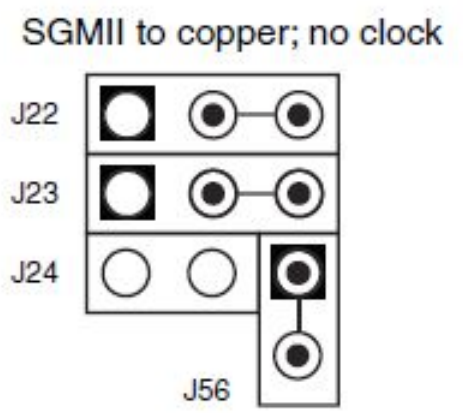
要件

このチュートリアルを実行するには、次のソフトウェアおよびハードウェアが必要です。

- ISE® Design Suite
- Virtex-5 ML506 評価キット
- デザイン ファイル：rdf0127_live_emac_tutorial.zip

注記：JTAG ケーブルを使用して ML506 をコンピューターに接続する方法は、『ML505/ML506/ML507 評価版プラットフォーム ユーザー ガイド』(UG347) を参照してください。このチュートリアルでは、イーサネット PHY とのインターフェイスに SGMII モードを使用します。次の図に示すように、ML506 ボードのジャンパー J22、J23、および J24 を変更し、SGMII をデフォルトの PHY インターフェイスとして選択してください。

ML506 の PHY インターフェイス モード ジャンパー



- J22：ピン 2 と 3 にジャンパーを配置
- J23：ピン 2 と 3 にジャンパーを配置
- J24：ジャンパーなし

チュートリアル ファイル

ファイル	説明
full_compile.bat	fuse コマンド ラインを使用してハードウェア協調シミュレーション用にデザインを完全にコンパイルする Windows バッチ ファイル
full_compile.sh	Fuse コマンド ラインを使用してハードウェア協調シミュレーション用にデザインを完全にコンパイルする Linux シェル スクリプト
incr_compile.bat	fuse コマンド ラインを使用してハードウェア協調シミュレーション用にテストベンチをインクリメンタルにコンパイルする Windows バッチ ファイル
incr_compile.sh	fuse コマンド ラインを使用してハードウェア協調シミュレーション用にテストベンチをインクリメンタルにコンパイルする Linux シェル スクリプト
ipcore_dir	Virtex-5 Embedded Tri-Mode Ethernet MAC Wrapper および FIFO コアを含む CORE Generator™ のディレクトリ
init.tcl	ISim でシミュレーションを初期化するためのカスタム シミュレーション コマンド ファイル
iseconfig	ISE コンフィギュレーション ファイルを含むディレクトリ

ファイル	説明
run_isim.bat	ISim シミュレーションを起動する Windows バッチ ファイル
run_isim.sh	ISim シミュレーションを起動する Linux シェル スクリプト
simple_arp.v	ARP (Address Response Protocol) 要求に応答するサンプル パケット プロセッサ
v5emac_hwcosim.ucf	ハードウェア協調シミュレーション用のカスタム制約ファイル。v5emac_top モジュールのどのポートを外部 I/O にマップするか、どのポートをテストベンチで制御するかを指定します。
v5emac_ml50x.gise	ISE プロジェクト ファイル
v5emac_ml50x.xise	このチュートリアル用の ISE プロジェクト
v5emac_tb.prj	PEEP 用のハードウェア協調シミュレーション ボード サポート ファイル
v5emac_tb.v	パケット プロセッサをインスタンシエートする最上位テストベンチ
v5emac_top.v	イーサネット MAC コア、ingress FIFO、egress FIFO、およびパケット カウント FIFO をインスタンシエートするラッパー
v5emac_tb.wcfg	カスタム波形コンフィギュレーション ファイル
_xmsgs	メッセージ ファイルを含むディレクトリ

注記： チュートリアルを開始する前に、すべてのデータ ファイルを作業ディレクトリにコピーします。

イーサネットの決定

複数のイーサネットがある場合は、イーサネット ポートを決定する必要があります。詳細は、「イーサネット ポートの決定」を参照してください。

チュートリアル

このチュートリアルでは、ISim ハードウェア協調シミュレーションを使用してイーサネット デザインを実行するのに必要な手順を 5 つのセクションに分けて示します。手順は順番に実行してください。

このチュートリアルは、次のセクションから構成されています。

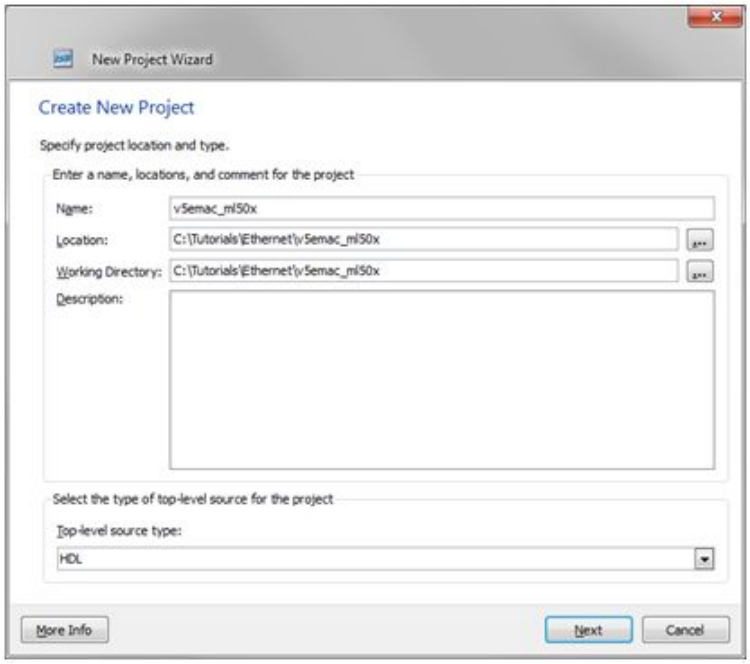
1. CORE Generator™ ツールで Virtex®-5 Embedded Tri-mode Ethernet MAC Wrapper を使用して、Virtex-5 イーサネット MAC サンプル デザインを生成します。
2. イーサネット MAC ラッパーとパケット プロセッサをバインドするテストベンチを作成します。
3. カスタム制約ファイルを作成して、サンプル デザインの ISim で制御するポートと外部 I/O にマップするポートを指定します。
4. ハードウェア協調シミュレーションを実行するサンプル デザインのテストベンチをコンパイルします。
5. ターゲット FPGA ボードをコンピューターに接続し、ISim シミュレーションを実行します。

手順 1：CORE Generator でのデザインの生成

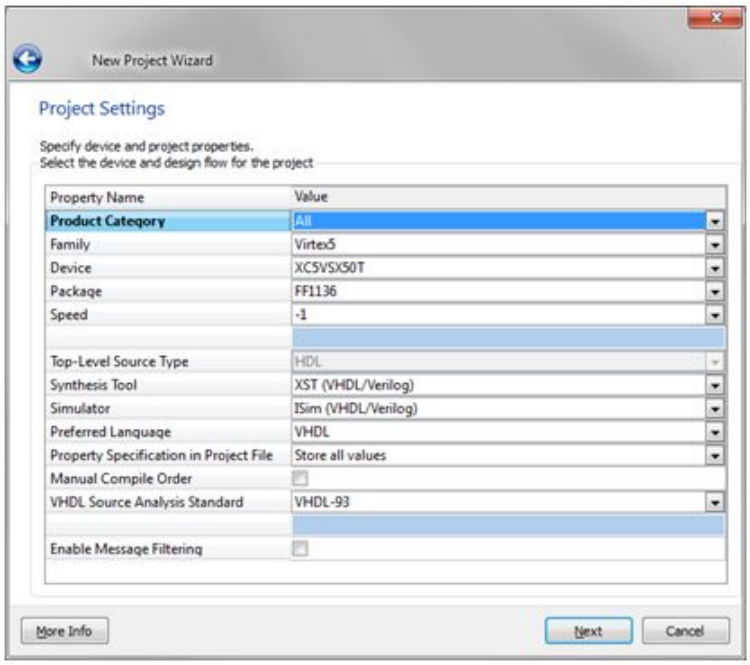
Virtex®-5 FPGA には、エンベデッド トライモード イーサネット MAC (EMAC) ブロックが含まれており、イーサネット PHY インターフェイスとの簡単で確実なインターフェイスを提供します。CORE Generator™ ツールの Virtex-5 Embedded Tri-Mode Ethernet MAC Wrapper を使用すると、イーサネット MAC ブロックを簡単にコンフィギュレーションできます。このチュートリアルでは、CORE Generator ツールで生成したサンプル デザインを使用し、Virtex-5 FPGA ML506 評価キットで動作する ISim ハードウェア協調シミュレーション テストベンチを作成します。

注記：このチュートリアルに含まれる図は、Virtex-5 Embedded Tri-Mode Ethernet MAC Wrapper v1.8 のものです。これ以外のバージョンでは、CORE Generator GUI が異なる場合があります。

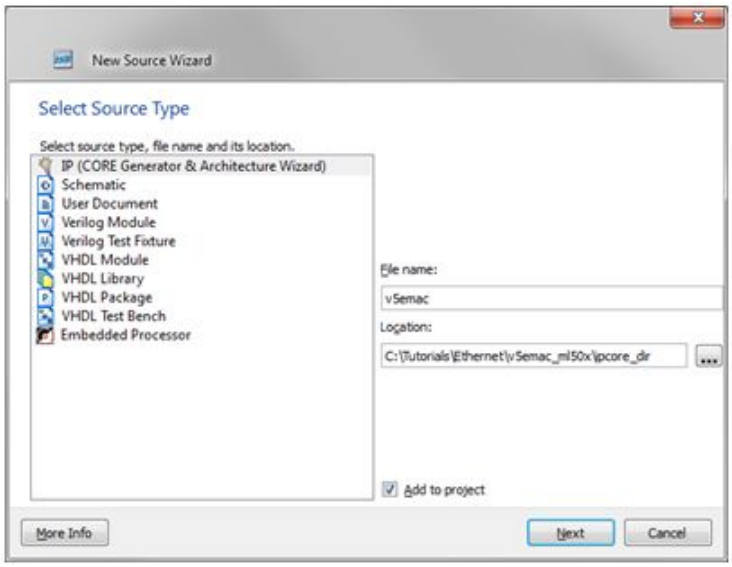
1. ISE® Project Navigator を起動します。
2. [File] → [New Project] をクリックし、New Project Wizard を開きます。プロジェクト名 (v5emac_ml50x) と保存ディレクトリを入力して [Next] をクリックします。



3. [Project Settings] ページで、[Family] に [Virtex5]、[Device] に [XC5VSX50T] (ML506 ボードの Virtex-5 デバイス)、[Package] に [FF1136]、[Speed] に [-1] を選択します。[Simulator] に [ISim]、[Preferred Language] に [VHDL] を選択します。[Next] をクリックして [Project Summary] ページで [Finish] をクリックし、プロジェクトの作成を完了します。

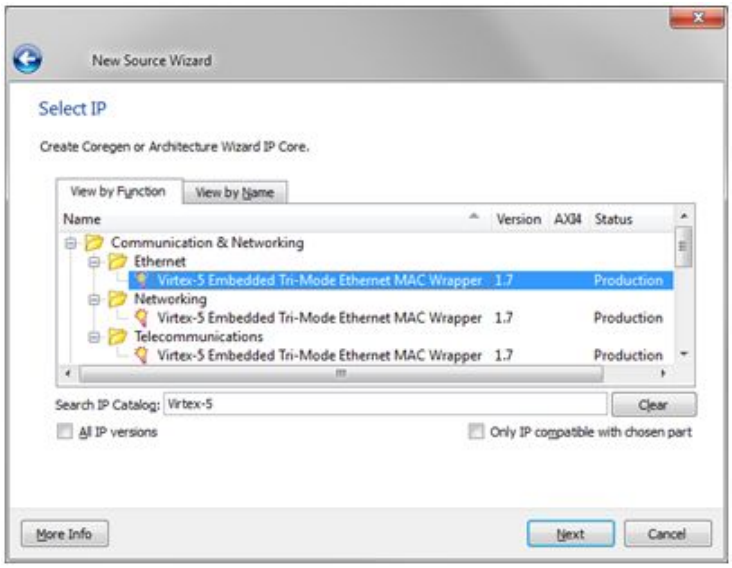


4. [Project] → [New Source] をクリックし、New Source Wizard を開きます。
5. [IP (CORE Generator & Architecture Wizard)] を選択し、[File name] に「v5emac」と入力して [Next] をクリックします。

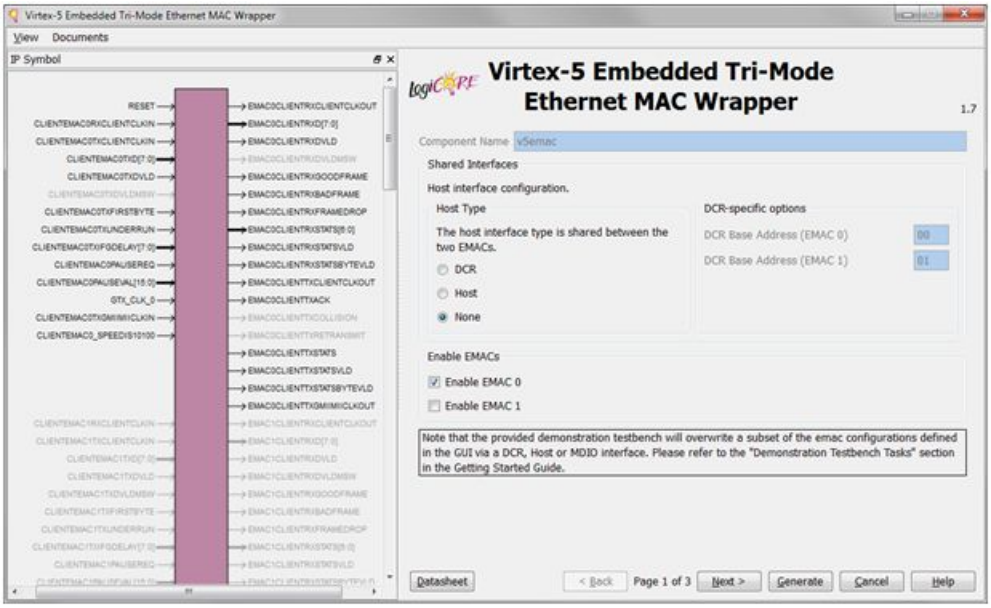


6. IP リストから次のいずれかのタブから [Virtex-5 Embedded Tri-Mode Ethernet MAC Wrapper] のバージョン 1.8 またはそれ以降を選択します。
   - [View By Function] → {Communications & Networking]
   - [View By Name] → [Embedded Tri-Mode Ethernet MAC Wrapper version 1.8]

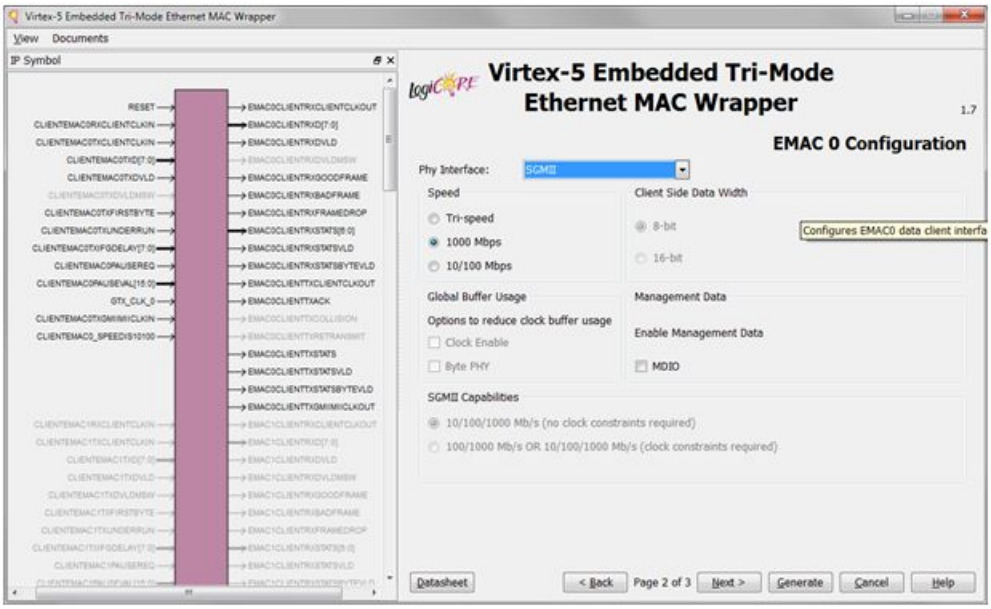
   [Next] をクリックし、次のダイアログ ボックスで [Finish] をクリックします。



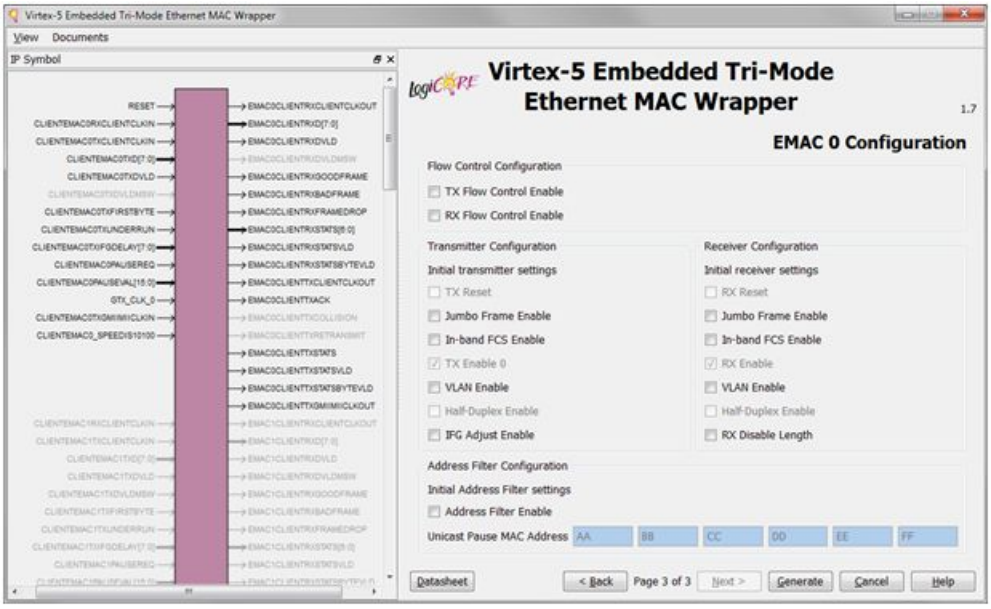
7. Virtex-5 Embedded Tri-Mode Ethernet MAC Wrapper コアの GUI が開いたら、[Host Type] で [None] をオンにします。[Enable EMAC 0] をオンにし、[Enable EMAC 1] をオフにします。[Next] をクリックします。



8. [Phy Interface] を [SGMII] に設定し、[Speed] で [1000 Mbps] をオンにします。[MDIO] チェック ボックスはオフのままにします。[Next] をクリックします。



9. [Flow Control Configuration]、[Transmitter Configuration]、[Receiver Configuration]、および [Address Filter Configuration] は、デフォルト設定のままににします。[Generate] をクリックしてコアを生成します。



注記 : Ethernet MAC Wrapper コアが生成されたら、サンプル デザインの pcore_dir/v5emac/example_design に生成されている LocalLink サブモジュール v5emac_locallink を使用します。

このチュートリアルではサンプル デザイン v5emac_example_design 全体は使用せず、デザインを次のように 2 分割します。

- plugin_ism テストベンチを使用してロックステップでシミュレーションされるデザイン部分
- ハードウェア協調シミュレーションを使用して FPGA でフリーランニングさせるデザイン部分

パケット処理モジュールは ISim でシミュレーションされ、デバッグの際にデザインを完全に可視化できるようにし、変更をすばやく適用できるようにします。イーサネット MAC および MGT は FPGA 上でフリーランニングさせ、ML506 ボード上の外部イーサネット PHY チップとインターフェイスさせます。これにより、イーサネット MAC で実際のイーサネット接続のイーサネット パケットを送受信できます。

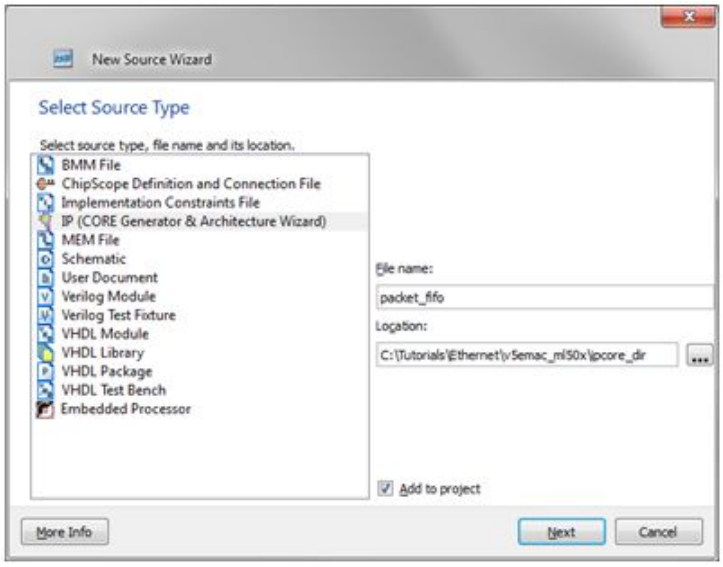
10. [Project] → [Add Source] をクリックします。`ipcore_dir/v5emac/example_design` ディレクトリに移動します。

    ISE プロジェクトファイルに次の HDL ファイル (`v5_emac_example_design.vhd` を除くすべての HDL ファイル) を追加します。

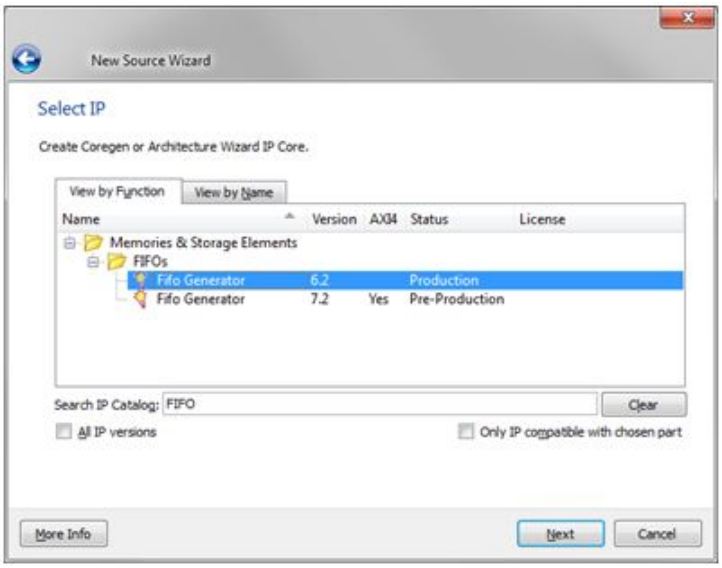
```
v5emac.v
v5emac_block.v
v5emac_locallink.v
client/address_swap_module_8.v
client/fifo/eth_fifo_8.v
client/fifo/rx_client_fifo_8.v
client/fifo/tx_client_fifo_8.v
physical/gtp_dual_1000X.v
physical/rocketio_wrapper_gtp.v
physical/rocketio_wrapper_gtp_tile.v
physical/rx_elastic_buffer.v
```

    LocalLink インターフェイスでは、LocalLink FIFO がクロック ドメイン切り替わり部分の非同期バッファーとして機能するので、デザインを LocalLink インターフェイスで分割できます。ただし、サンプル デザインの LocalLink FIFO は大きさが十分でなく、ISim で生成されたエミュレート クロックではうまく機能しないので、LocalLink FIFO を補足するため、ingress パケットおよび egress パケットを取り込む非同期 FIFO のペアを追加します。

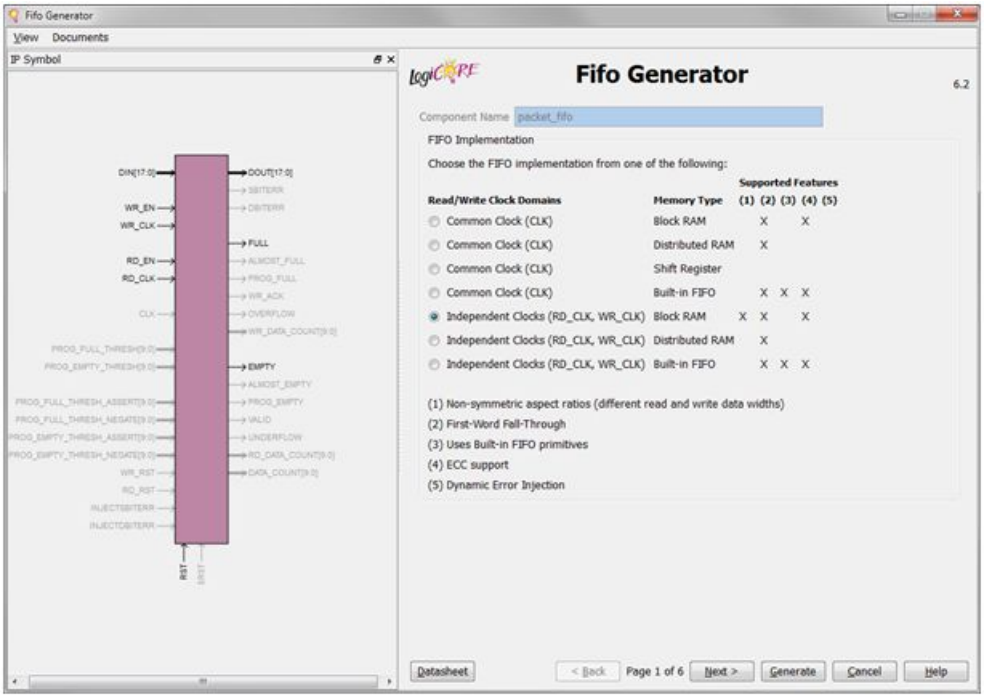
11. [Project] → [New Source] をクリックし、New Source Wizard を開きます。[IP (CORE Generator & Architecture Wizard)] を選択し、[File name] に「packet_fifo」と入力します。[Next] をクリックします。



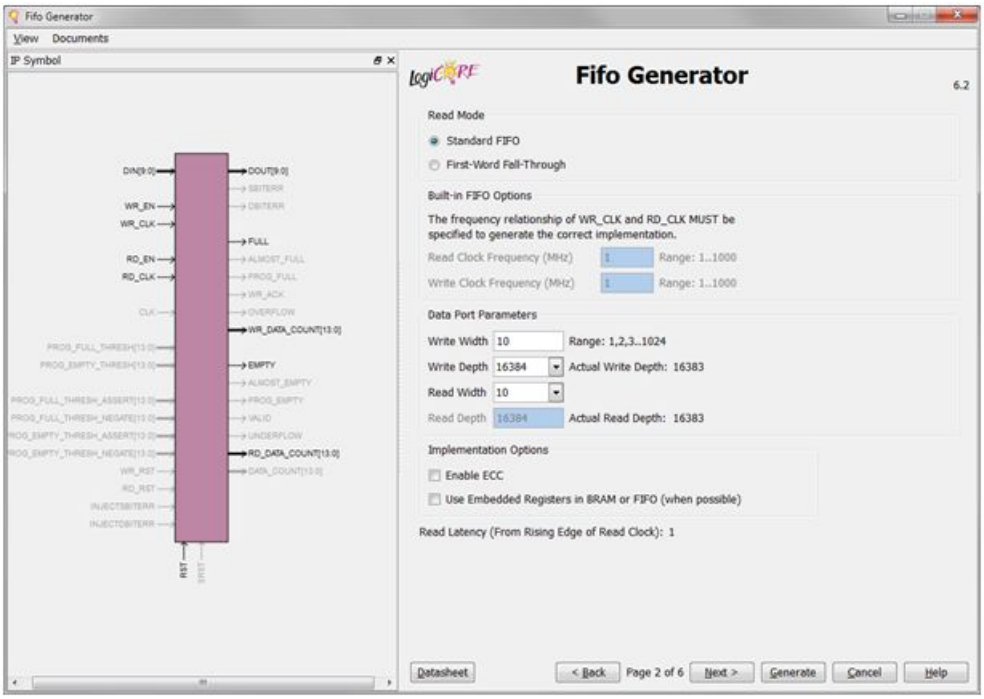
12. IP リストから [Fifo Generator] のバージョン 8.3 を選択します。[Next] をクリックし、次のダイアログ ボックスで [Finish] をクリックします。



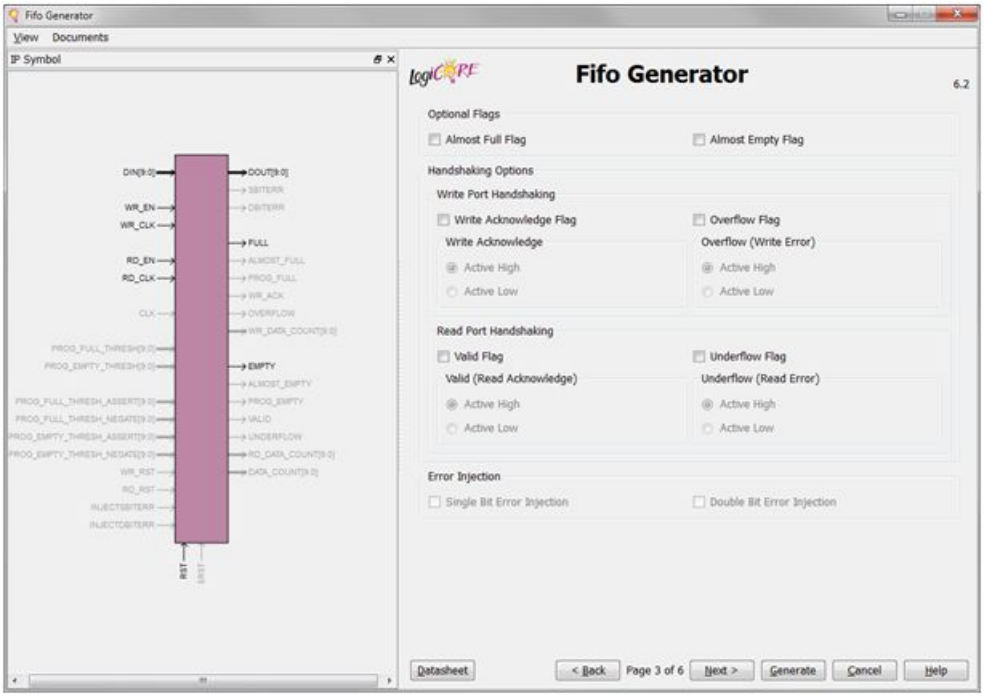
13. FIFO Generator コアの GUI が開いたら、[FIFO implementation] で [Independent Clocks (RD_CLK, WR_CLK) Block RAM] をオンにします。[Next] をクリックします。



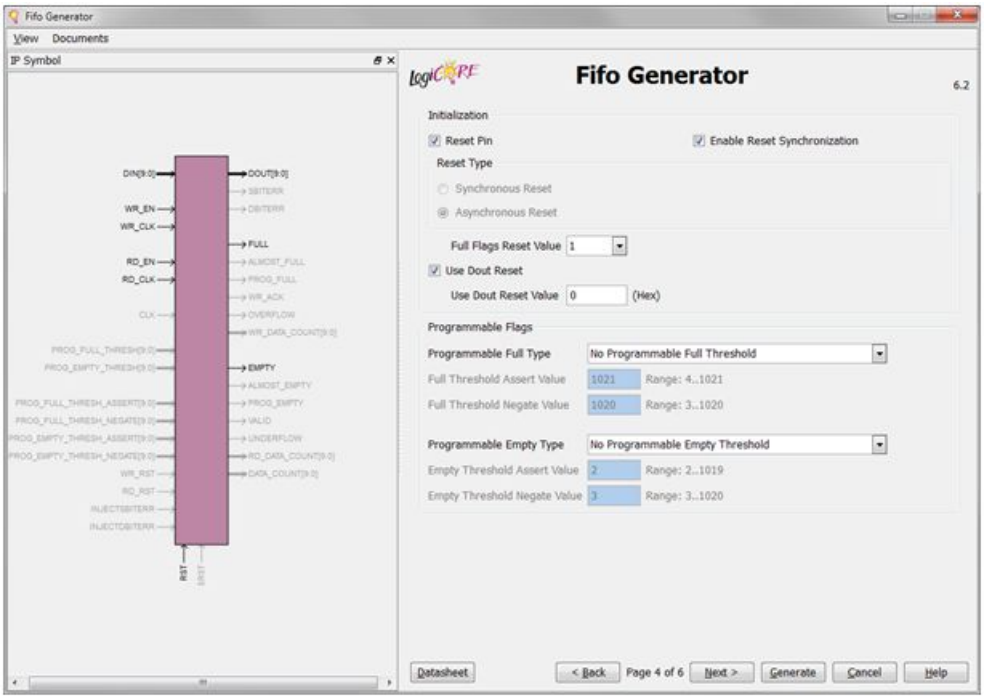
14. [Read Mode] で [First-Word Fall-Through] をオンにします。[Write Width] および [Read Width] を [10] に、[Write Depth] を [16384] に設定します。[Next] をクリックします。



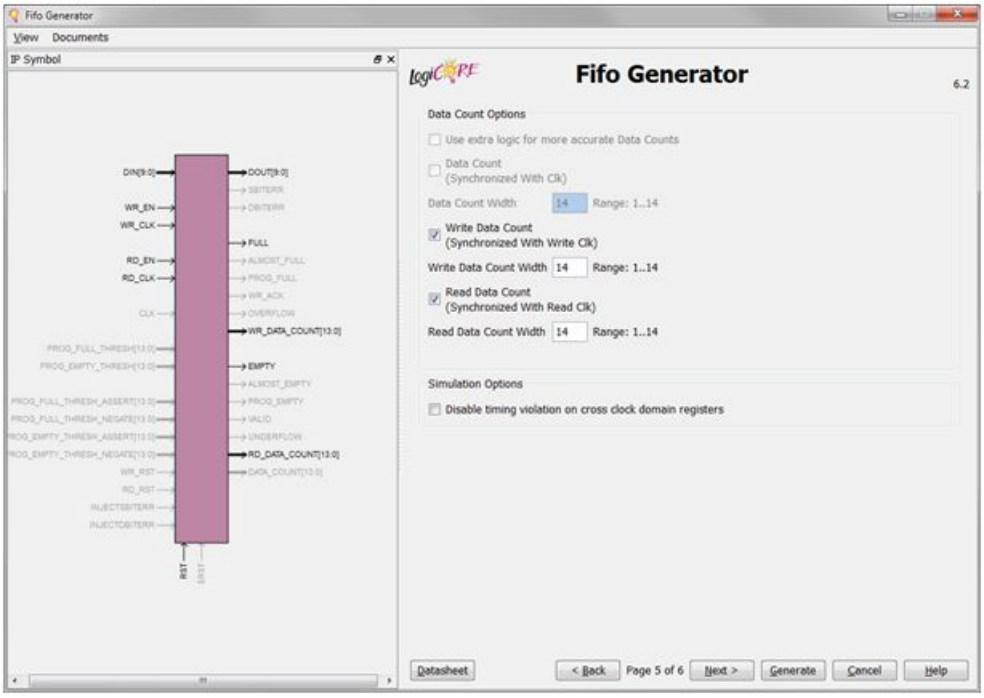
15. [Optional Flags]、[Handshaking Options]、および [Error Injection] はデフォルト設定のままにします。[Next] をクリックします。



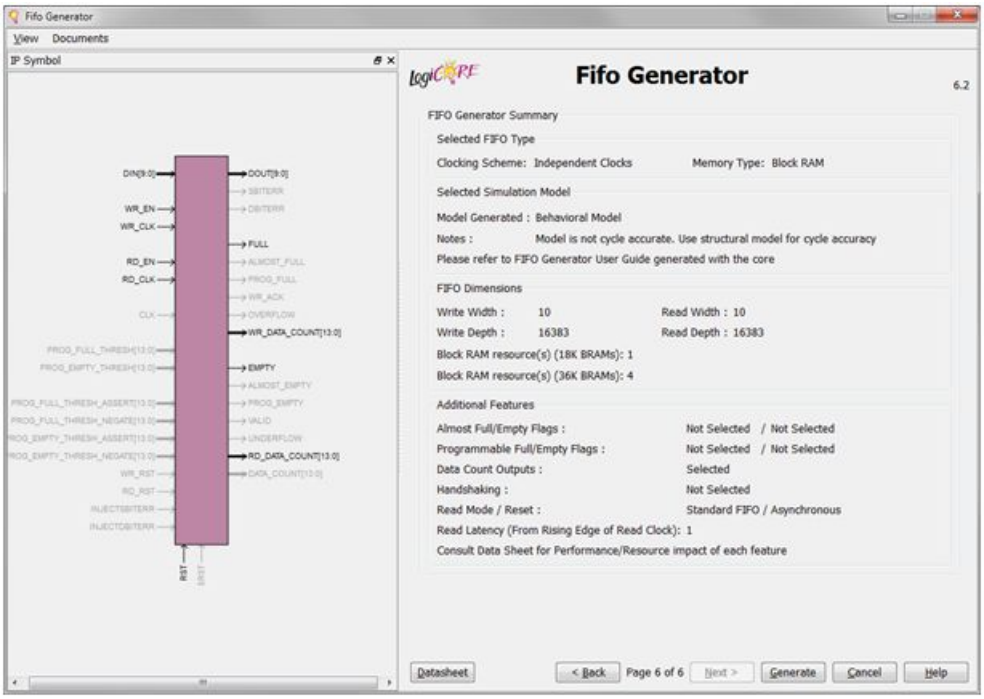
16. [Initialization] および [Programmable Flags] はデフォルト設定のままにします。[Next] をクリックします。



17. [Write Data Count] および [Read Data Count] をオンにし、[Write Data Count Width] および [Read Data Count Width] を [5] に設定します。[Next] をクリックします。

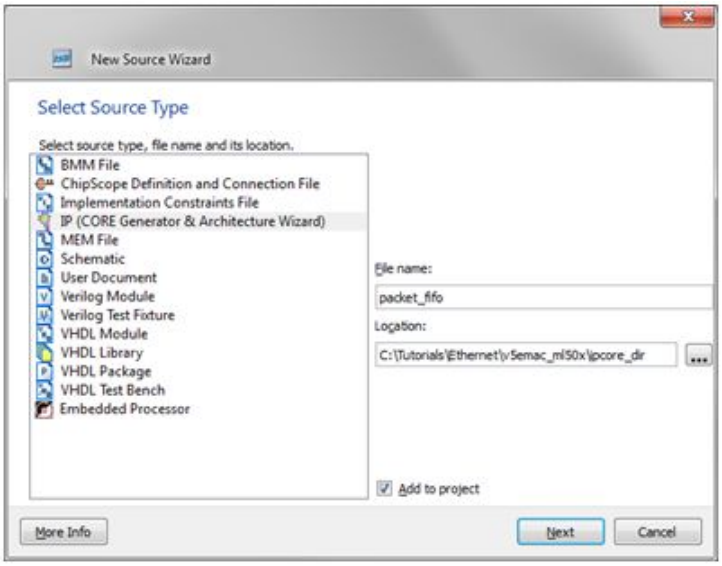


18. [Generate] をクリックして FIFO コアを生成します。

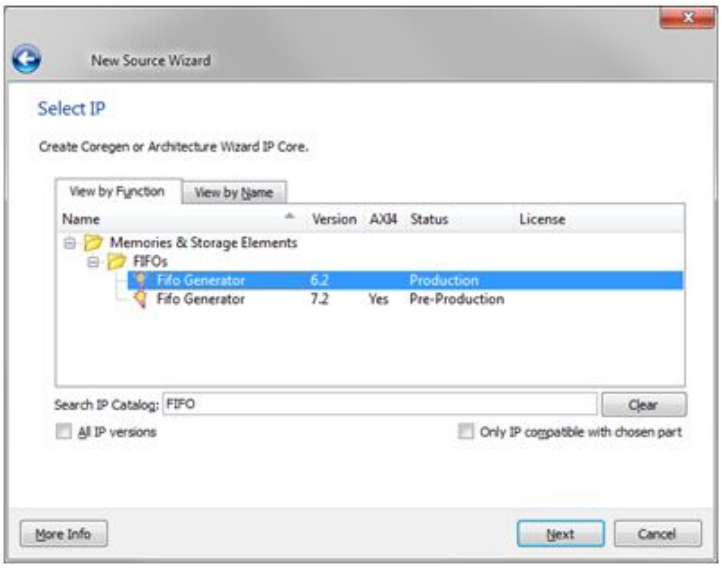


注記： このチュートリアルでは、egress FIFO に完全なパケットが少なくとも 1 つある場合にのみ TX LocalLink First In First Out (FIFO) にパケットが送信されるようにします。この目的のため、egress FIFO のパケット数をモニターする別の FIFO を追加します。

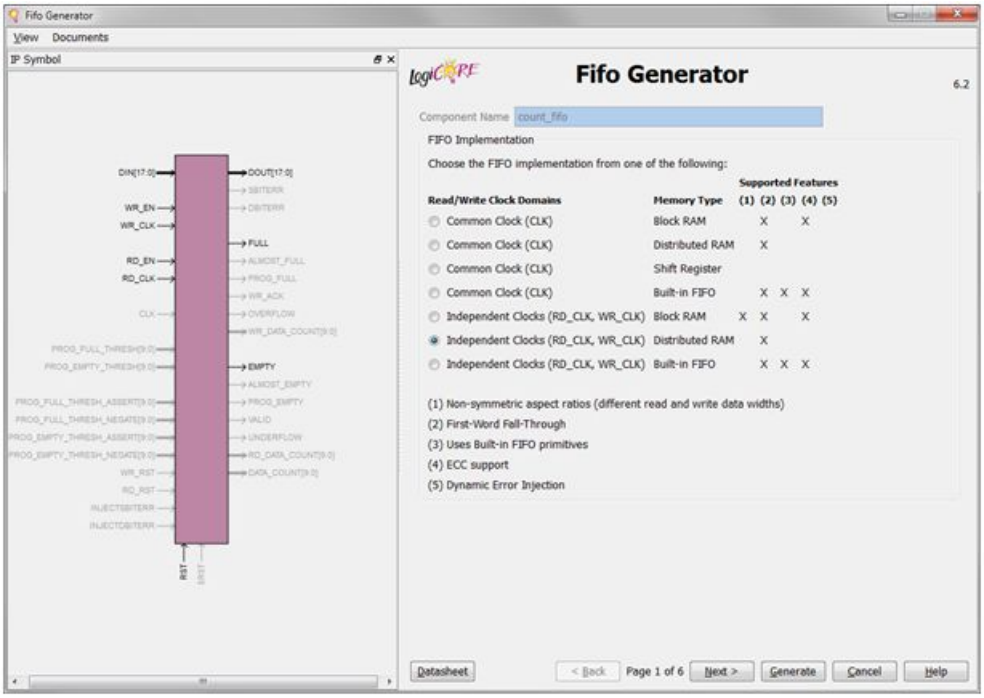
19. [Project] → [New Source] をクリックし、New Source Wizard を開きます。[IP (CORE Generator & Architecture Wizard)] を選択し、[File name] に「count_fifo」と入力します。[Next] をクリックします。



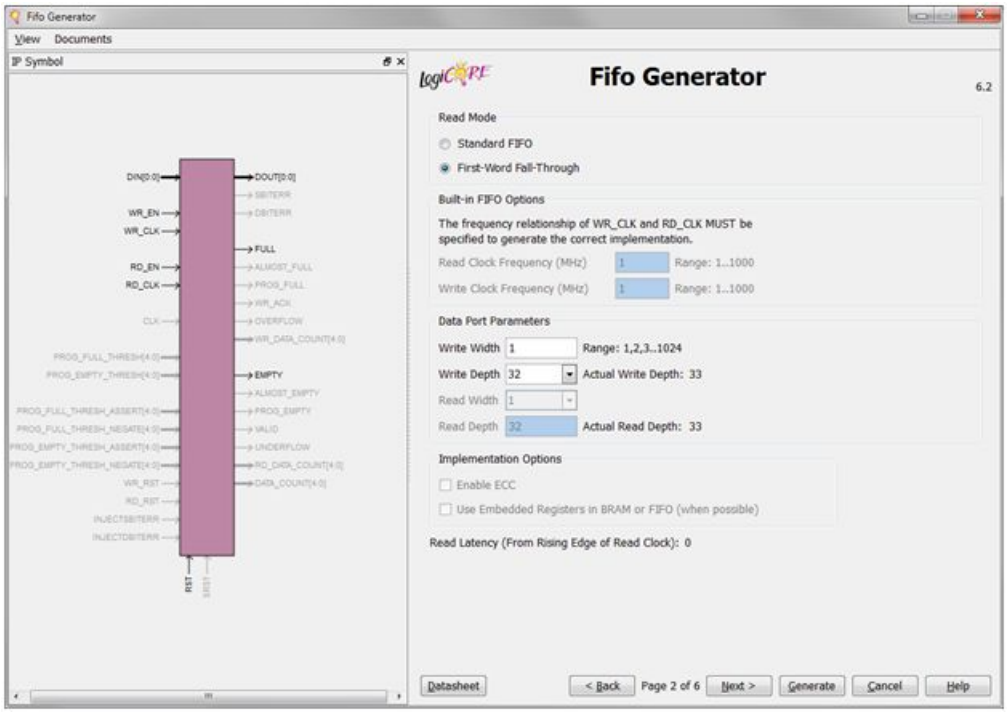
20. IP リストから [Fifo Generator] のバージョン 8.3 を選択します。[Next] をクリックし、次のダイアログ ボックスで [Finish] をクリックします。



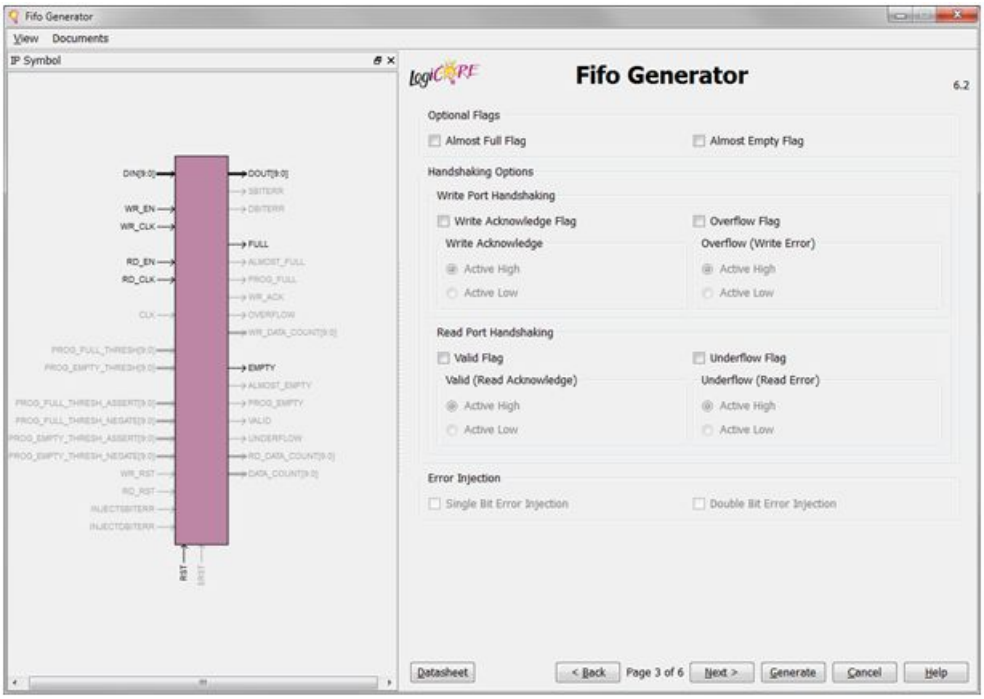
21. FIFO Generator コアの GUI が開いたら、[FIFO implementation] で [Independent Clocks (RD_CLK, WR_CLK) Distributed RAM] をオンにします。[Next] をクリックします。



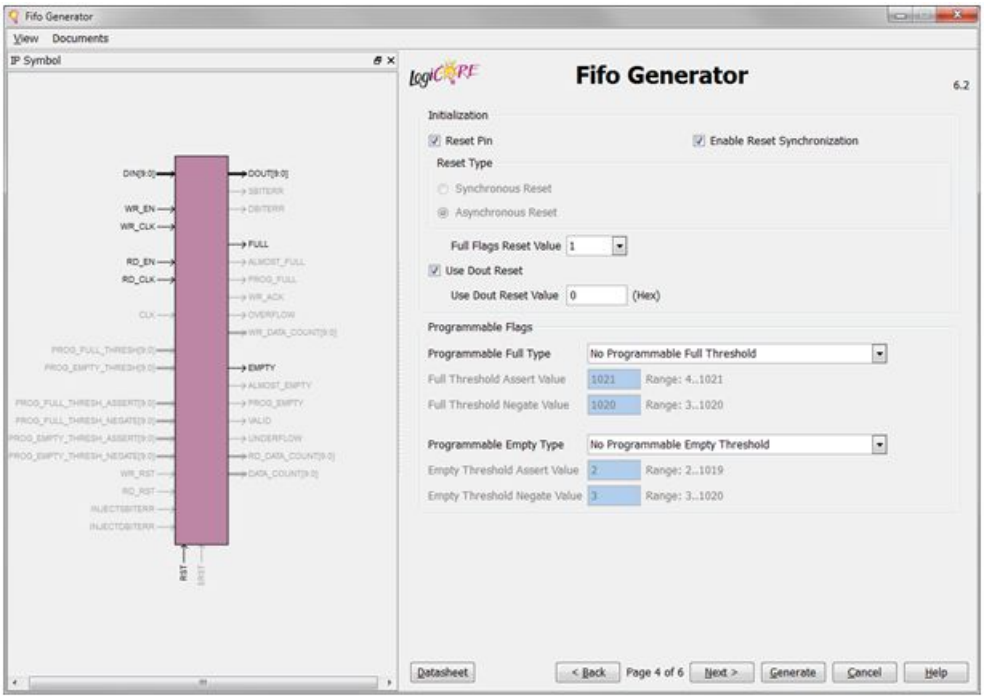
22. [Read Mode] で [First-Word Fall-Through] をオンにします。[Write Width] および [Read Width] を [1] に、[Write Depth] を [32] に設定します。[Next] をクリックします。



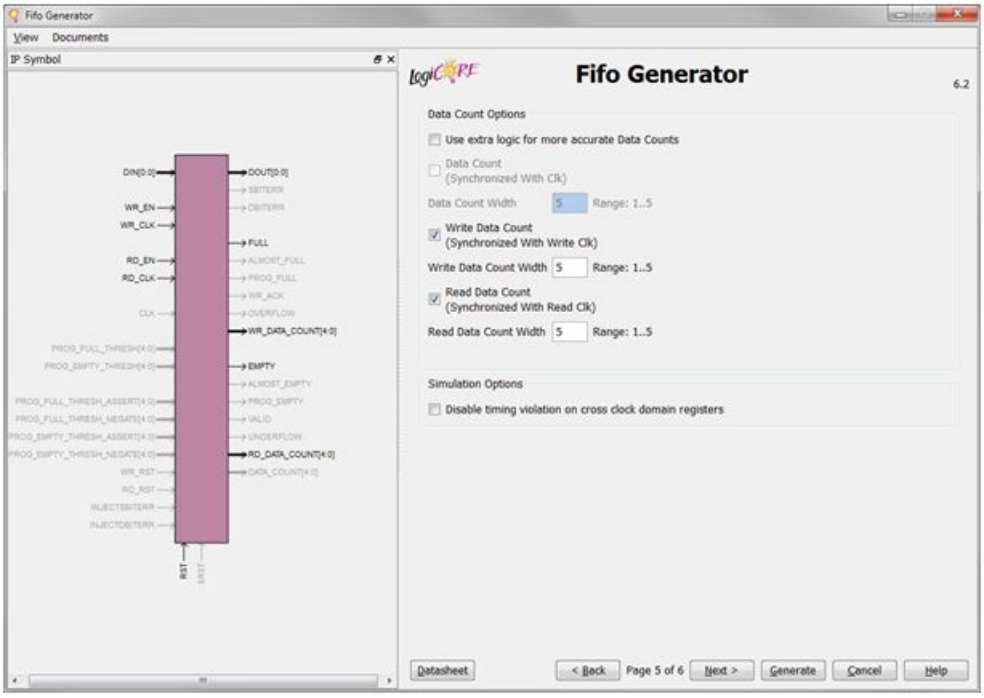
23. [Optional Flags]、[Handshaking Options]、および [Error Injection] はデフォルト設定のままにします。[Next] をクリックします。



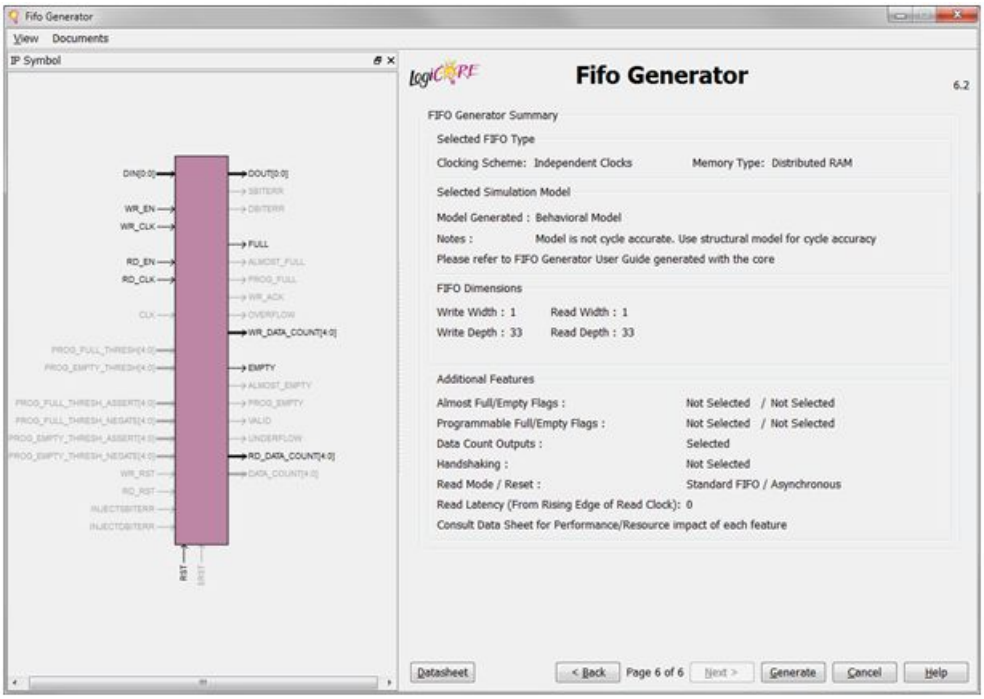
24. [Initialization] および [Programmable Flags] はデフォルト設定のままにします。[Next] をクリックします。



25. [Write Data Count] および [Read Data Count] をオンにし、[Write Data Count Width] および [Read Data Count Width] を [5] に設定します。[Next] をクリックします。



26. [Generate] をクリックして FIFO コアを生成します。



次に、v5_emac_locallink モジュール、ingress FIFO、egress FIFO、および egress パケット カウント FIFO をインスタンシエートする最上位モジュール `v5emac_top` を追加します。このチュートリアルに含まれる完成した `v5emac_top.vhd` を使用できます。

27. [Project] → [Add Source] をクリックし、`v5emac_top.vhd` を追加します。

手順 2 : テストベンチの作成

`v5emac_top` インスタンスをパケット プロセッサにバインドする VHDL テストベンチ モジュール `v5emac_tb.v` を追加します。このチュートリアルに含まれる完成した `v5emac_tb.v` を使用できます。

[Project] → [Add Source] をクリックします。このチュートリアルに含まれる `v5emac_tb.v` および `simple_arp.v` を追加します。

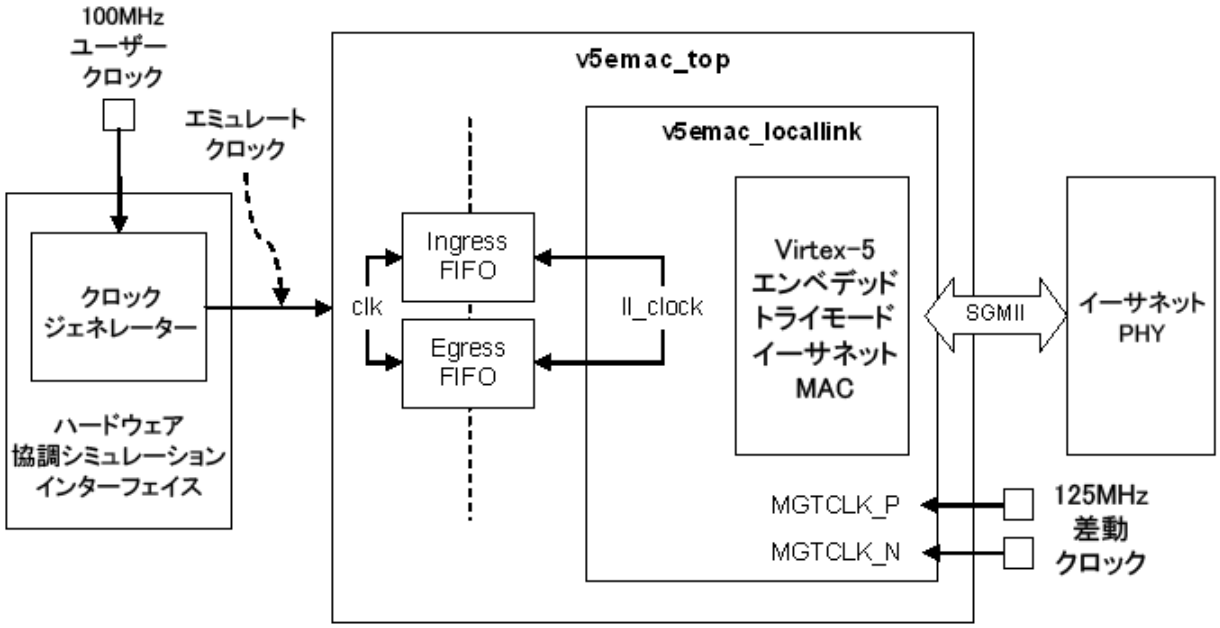
手順 3 : カスタム制約ファイルの作成

デザインをロックステップ部分とフリーランニング部分に分割

このチュートリアルで重要なのは、デザインを次の 2 つの部分に分割するということです。

- Virtex®-5 エンベデッド イーサネット MAC を介して外部イーサネット PHY とインターフェイスするフリーランニング部分。外部 I/O とクロックを接続し、イーサネット インターフェイスに必要なフル クロック スピードで動作します。
- ISim を使用して HDL テストベンチで駆動するロックステップ部分。ISim に同期化されており、ハードウェア協調シミュレーション インターフェイスからスティミュラスおよびクロック イベントを受信します。そのため、動作速度はかなり遅くなります。

次の図に、ハードウェア協調シミュレーションでのイーサネット デザインへのクロック供給方法を示します。ハードウェア協調シミュレーション インターフェイスは、コンパイル中に自動的に挿入されます。ハードウェア協調シミュレーション インターフェイスは、ML506 ボード上の 100MHz のユーザー クロックに基づいてエミュレート クロックを生成します。エミュレート クロックは、テストベンチの clk 信号のクロック イベントに対応し、ハードウェアで動作している v5emac_top の clk ポートを駆動します。イーサネット MAC の MGT クロックは、ML506 ボード上の 125MHz 差動クロックから供給されます。ingress FIFO および egress FIFOは、エミュレート クロック clk と LocalLink インターフェイス クロック ll_clock の間でクロック ドメインが切り替わる部分の非同期パケット バッファーとなります。



ポートを外部 I/O およびクロックに割り当て

カスタム制約ファイル (ザイリンクスの UCF フォーマット) を使用して、ハードウェア協調シミュレーションでどのポートを FPGA IOB にマップするか、HDL テストベンチでどのポートを制御するかを指定できます。ISim コンパイラにより、UCF ファイルに含まれる LOC 制約が検索されます。

- LOC 制約が設定されているポートは、対応する FPGA IOB (入出力バッファー) にマップされます。
- LOC 制約が設定されていないポートは、ハードウェア協調シミュレーション インターフェイスにマップされ、HDL テストベンチでアクセスできます。

デザインのフリーランニング部分とロックステップ部分への分割は、クロック ポートのマップ方法により決定されます。

- LOC 制約を使用してクロック ポートを FPGA IOB にマップすると、このクロックで駆動されるロジックはフリーランニング部分に含まれます。
- クロック ポートに LOC 制約が設定されていない場合、テストベンチで対応するクロック イベントが発生したときに、ハードウェア協調シミュレーション インターフェイスによりこのポートの値がトグルされます。このクロックで駆動されるロジックは、ロックステップ部分に含まれます。

フリーランニング部分とロックステップ部分は異なるクロックを使用して異なる速度で動作するので、この 2 つの部分の間でのクロックドメインの切り替わりをデザインで適切に処理する必要があります。ISim ハードウェア協調シミュレーションのコンパイルではデザインの内部は変更されないので、2 つの部分の間の速度差と同期化をデザインで適切に処理できることを前提としています。

次の表に、v5emac_top モジュールの外部 I/O にマップされるポートと、テストベンチで制御されるポートを示します。

v5emac_top モジュール ポートのマップ

外部 I/O にマップされるポート	テストベンチで制御されるポート
CTXP_0 TXN_0 RXP_0 RXN_0 MGTCLK_P MGTCLK_N PHY_RST_N	clk reset resetdone ingress_sof_n ingress_eof_n ingress_data ingress_rd_count ingress_re ingress_empty egress_sof_n egress_eof_n egress_data egress_wr_count egress_we egress_full count_we_o count_wr_count

Virtex-5 Embedded Tri-Mode Ethernet MAC Wrapper で提供されるサンプル デザインには、UCF ファイル `ipcore_dir/v5emac/example_design/v5emac_example_design.ucf` が含まれています。このチュートリアルでは、これをテンプレートとして使用して、ハードウェア協調シミュレーション用のカスタム制約ファイルを作成します。

1. `ipcore_dir/v5emac/example_design/v5emac_example_design.ucf` を `v5_emac_top.v` が含まれる ISE® プロジェクト ディレクトリにコピーします。コピーしたファイルの名前を `v5emac_hwcosim.ucf` に変更します。
2. `v5emac_hwcosim.ucf` ファイルを次のように ML506 ボード用に変更します。
   a. エンベデッド イーサネット MAC の AREA_GROUP 制約をコメントアウトします。

```
#INST v5_emac_ll/* AREA_GROUP = AG_v5_emac ;
#AREA_GROUP "AG_v5_emac" RANGE = CLOCKREGION_X1Y2,CLOCKREGION_X1Y3 ;
```

   MGTCLK_P の LOC 制約を P4 に、MGTCLK_N の LOC 制約を P3 に変更します (値が異なる場合)。

```
INST "MGTCLK_N" LOC = "P3";
INST "MGTCLK_P" LOC= "P4";
```

   b. GTP プリミティブの LOC 制約を GTP_DUAL_X0Y2 から GTP_DUAL_X0Y3 に変更します。

```
INST "*GTP_DUAL_1000X_inst?GTP_1000X?tile0_rocketio_wrapper_i?gtp_
dual_i" LOC = "GTP_DUAL_X0Y3";
```

c. イーサネット MAC プリミティブの EMAC0 の自動ネゴシエーションをデフォルトでイネーブルにします。

```
INST "*?v5_emac" EMAC0_PHYINITAUTONEG_ENABLE = TRUE;
```

d. TXN_0、TXP_0、RXN_0、RXP_0、および PHY_RST_N に、ML506 ボードのピン割り当てに一致する LOC 制約を追加します。

```
INST "TXN_0" LOC = "N2";
INST "TXP_0" LOC = "M2";
INST "RXN_0" LOC = "P1";
INST "RXP_0" LOC = "N1";
INST "PHY_RST_N" LOC = "J14";
```

3. 次に、ISim ハードウェア協調シミュレーションの要件に応じて v5emac_hwcosim.ucf ファイルを変更します。

a. 次の制約で階層パスの最初にワイルドカード文字「*」を追加します。これは、v5emac_locallink がハードウェア協調シミュレーション用にコンパイルされる際にサブモジュールとしてラッパーに含まれるからです。

```
NET "*clk125" TNM_NET = "clk_gtp";
```

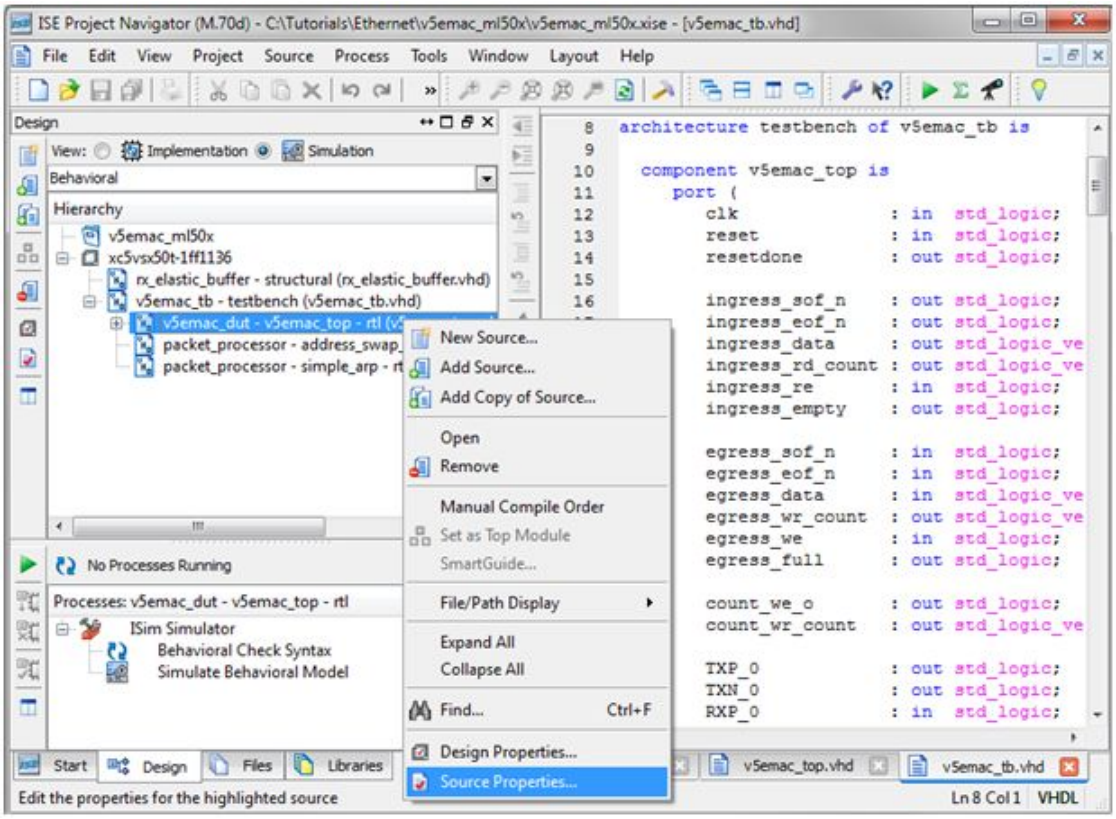
b. resetdone 信号は ISim テストベンチにより監視されるので、TIG (タイミング無視) 制約を設定してタイミング エラーが発生しないようにします。

```
NET "*resetdone" TIG;
```

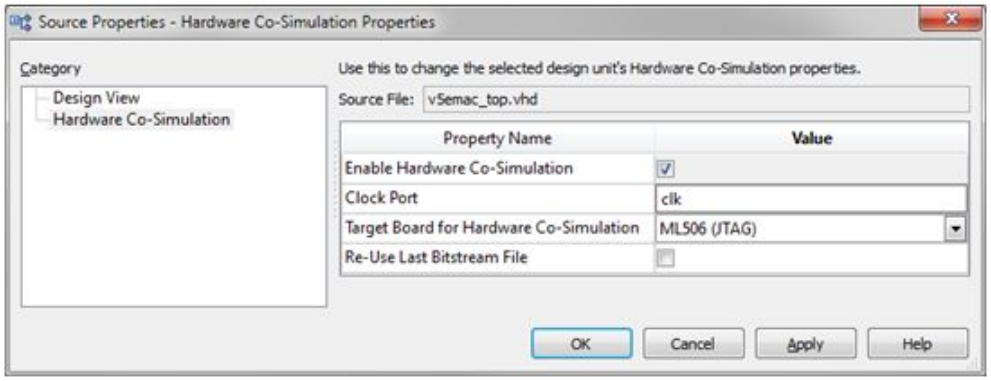
手順 4：ハードウェア協調シミュレーション用のデザインのコンパイル

テストベンチとカスタム制約ファイルを作成したら、ISim コンパイラを使用して、デザインをハードウェア協調シミュレーション用にコンパイルします。これは、Project Navigator でデザインの選択したインスタンスでハードウェア協調シミュレーションをイネーブルにすると実行できます。選択したインスタンスとそれに含まれるサブモジュールは、ISim シミュレーション時にハードウェアで協調シミュレーションされます。その他のモジュールは、ソフトウェアでシミュレーションされます。

1. Project Navigator の [View] ペインで [Simulation] をオンにします。[Hierarchy] ペインで [v5emac_dut - v5emac_top] インスタンスを右クリックし、[Source Properties] をクリックします。



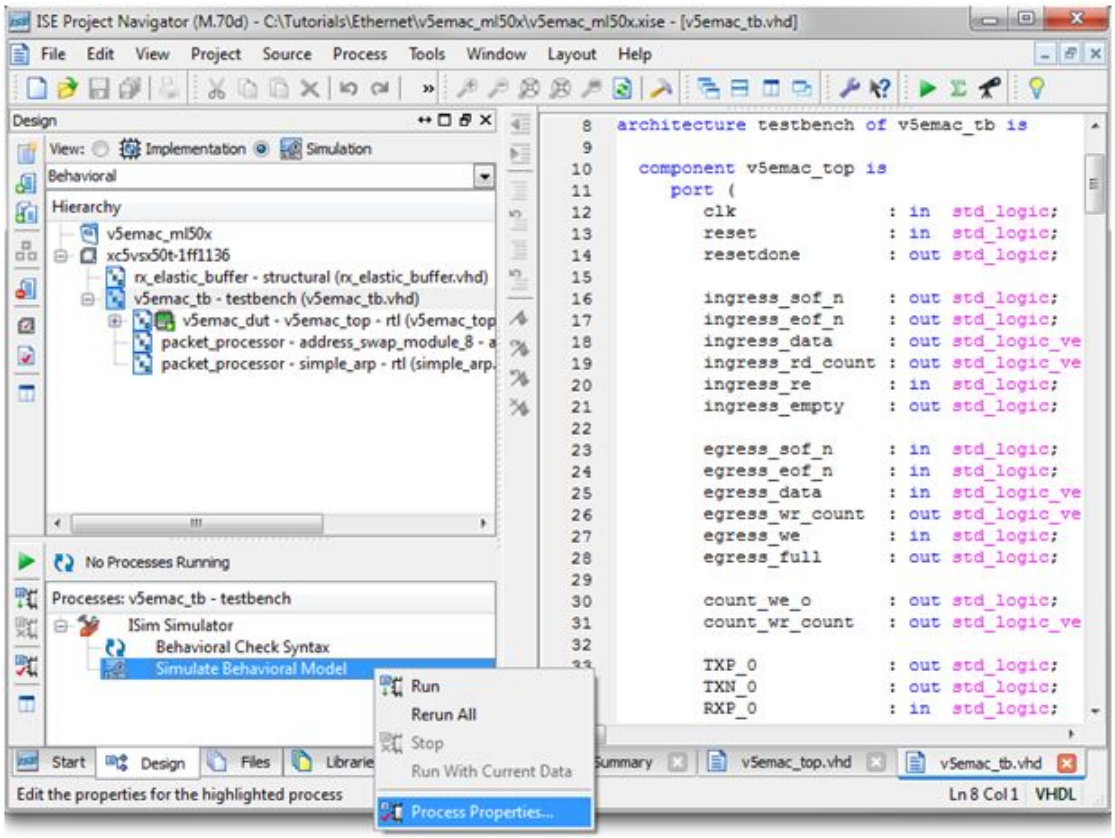
2. [Category] で [Hardware Co-Simulation] を選択します。[Enable Hardware Co-Simulation] をオンにします。[Clock Port] を [clk] に設定します。[Target Board for Hardware Co-Simulation] を [ML506 (JTAG)] に設定します。



注記： ハードウェア協調シミュレーションをイネーブルにしたインスタンスには、 アイコンが付きます。

ハードウェア協調シミュレーション用に選択されたインスタンスがその後の実行で変更されない場合は、[Enable Incremental Implementation] をオンにすると、ハードウェア協調シミュレーション用の合成、インプリメンテーション、ビットストリーム生成がスキップされます。このオプションを使用すると、テストベンチまたはソフトウェアでシミュレーションする部分をすばやく変更し、再シミュレーションできます。

3. [Hierarchy] ペインで [v5emac_tb] をクリックします。[Processes] ペインで [Simulate Behavioral Model] を右クリックして [Process Properties] をクリックします。



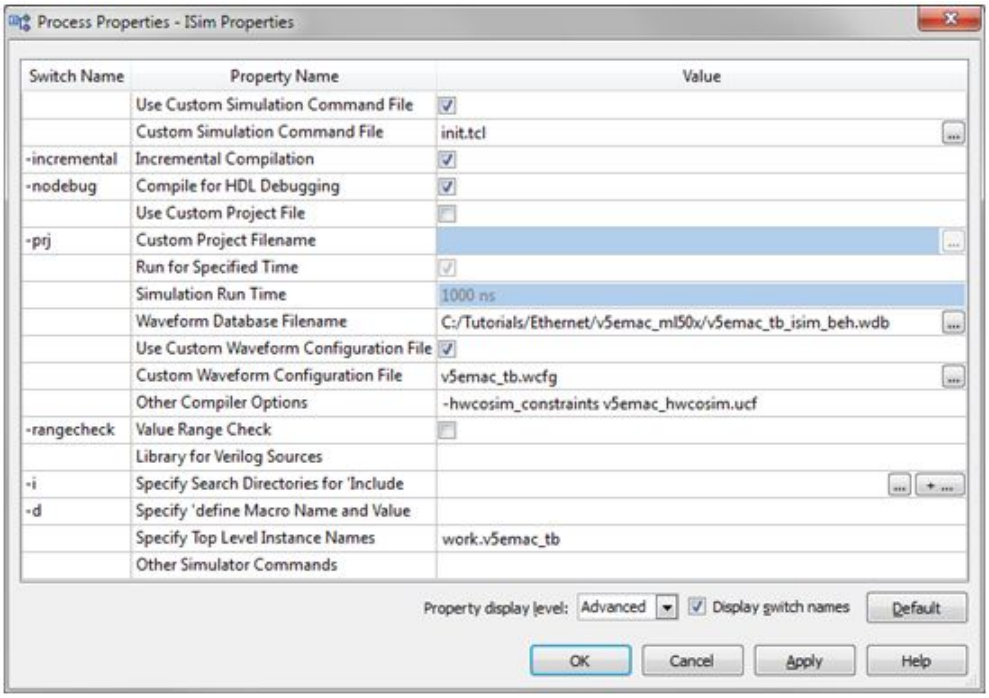
4. [Process Properties] ダイアログ ボックスで [Property display level] を [Advanced] に変更します。[Simulate Behavioral Model] プロセスの次のプロパティを設定します。

   - [Use Custom Simulation Command File] : オン
   - [Custom Simulation Command File] : init.tcl
   - このオプションは、[Use Custom Waveform Configuration File] がオンのときのみ設定できます。
   - [Other Compiler Options] : -hwcosim_constraints v5emac_hwcosim.ucf

   ISim シミュレーションを開始すると、init.tcl スクリプトが実行されます。このスクリプトはシミュレーションを 50ns 実行してデザインの初期リセットを実行します。

   `v5emac_tb.wcfg` ファイルは、このチュートリアル用にカスタマイズされた波形コンフィギュレーション ビューを提供します。

注記： ハードウェア協調シミュレーション用のカスタム制約ファイルは、-hwcosim_constraints オプションで指定します。このオプションは [Other Compiler Options] で指定します。



5. v5emac_tb インスタンスの [Simulate Behavioral Model] プロセスをダブルクリックしてシミュレーションを実行します。

コマンド ラインからのコンパイル

ISim コンパイラを fuse コマンド ライン ツールを使用して起動できます。完全なソフトウェア シミュレーション フローと同様に、プロジェクト ファイル、デザインの最上位モジュール、およびリンクするライブラリやライブラリ検索パスなどの引数を指定して、fuse を実行します。ハードウェア協調シミュレーション用にデザインをコンパイルするには、次に示す引数を指定する必要があります。

```
fuse -prj <project file> <top level modules>
     -hwcosim_instance <instance>
     -hwcosim_clock <clock>
     -hwcosim_board <board>
     -hwcosim_constraints <constraint file>
     -hwcosim_incremental [0|1]
```

- `-hwcosim_instance` : ハードウェアで協調シミュレーションするためにインスタンスの完全階層パスを指定します。
- `-hwcosim_clock` : インスタンスのクロック入力のポート名を指定します。
  - これはロックステップ部分のクロックで、テストベンチで制御されます。
  - 複数のクロックを使用するデザインでは、このオプションで最高速のクロックを指定し、ISim でシミュレーションが最適化されるようにします。その他のクロック ポートは、通常のデータ ポートとして処理されます。
- `-hwcosim_board` : 協調シミュレーションに使用するハードウェア ボードを指定します。デフォルトでは、次の Virtex®-5 ボードがサポートされています。
  - ml501-jtag :ザイリンクス ML501 評価プラットフォーム
  - ml505-jtag :ザイリンクス ML505 評価プラットフォーム
  - ml506-jtag :ザイリンクス ML506 評価プラットフォーム
  - ml507-jtag :ザイリンクス ML507 評価プラットフォーム
  - ml510-jtag :ザイリンクス ML510 評価プラットフォーム
  - xupv5-jtag :ザイリンクス XUPV5-LX110T 評価プラットフォーム
- `-hwcosim_constraints`（オプション）: ハードウェア協調シミュレーション用にインスタンスをインプリメントするための追加制約を含むカスタム制約ファイルを指定します。この制約ファイルでは、インスタンスのどのポートを外部 I/O またはクロックにマップするかも指定します。
- `-hwcosim_incremental`（オプション）: fuse で前回生成されたハードウェア協調シミュレーション ビットストリームを再利用し、インプリメンテーション フローをスキップするよう指定します。

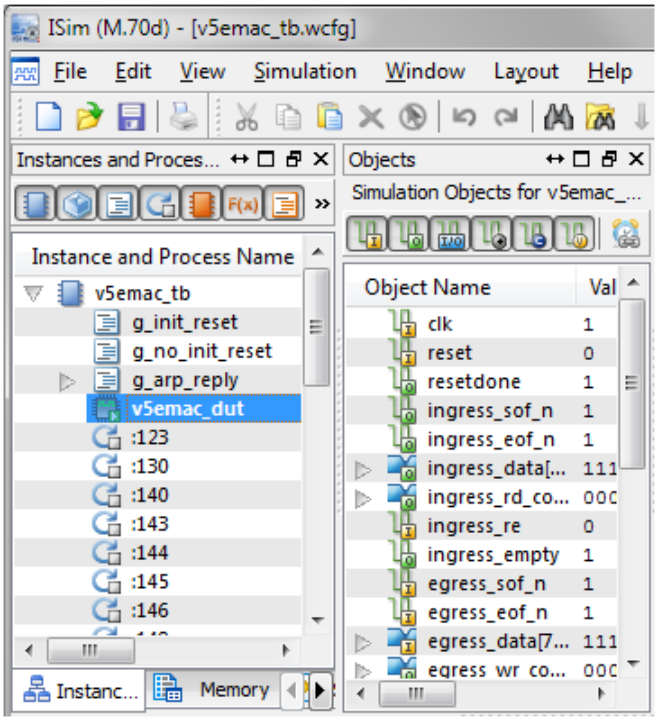
たとえば、このチュートリアルの EMAC デザインをコンパイルするには、次のようにコマンド ラインに入力して Fuse を実行できます。

```
fuse -prj v5emac_tb.prj v5emac_tb
     -o v5emac_tb.exe
     -hwcosim_instance /v5emac_tb/v5emac_dut
     -hwcosim_clock clk
     -hwcosim_board ml506-jtag
     -hwcosim_constraints v5emac_hwcosim.ucf
```

手順 5：ISim ハードウェア協調シミュレーションの実行

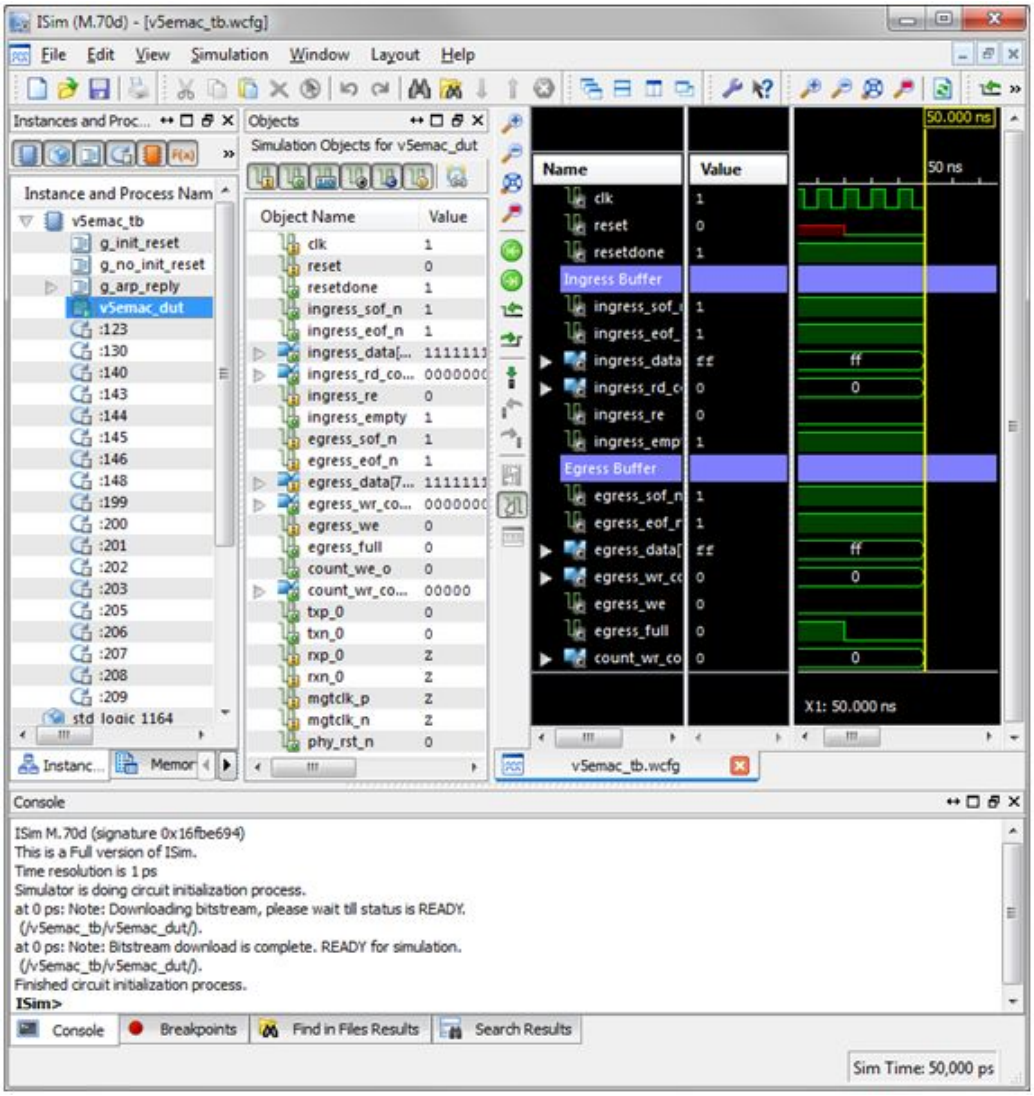
ISim コンパイラで生成されるシミュレーション実行ファイルは、完全なソフトウェア シミュレーションおよびハードウェア協調シミュレーション フローの両方で同様に使用できます。コンパイルが終了すると、Project Navigator によりシミュレーション実行ファイルが GUI モードで実行されます。

ハードウェア協調シミュレーションに選択されたインスタンスには、[Instances and Processes] パネルで アイコンが表示されます。ハードウェアでインスタンスを実行すると、その内部信号およびサブモジュールをモニターすることはできません。

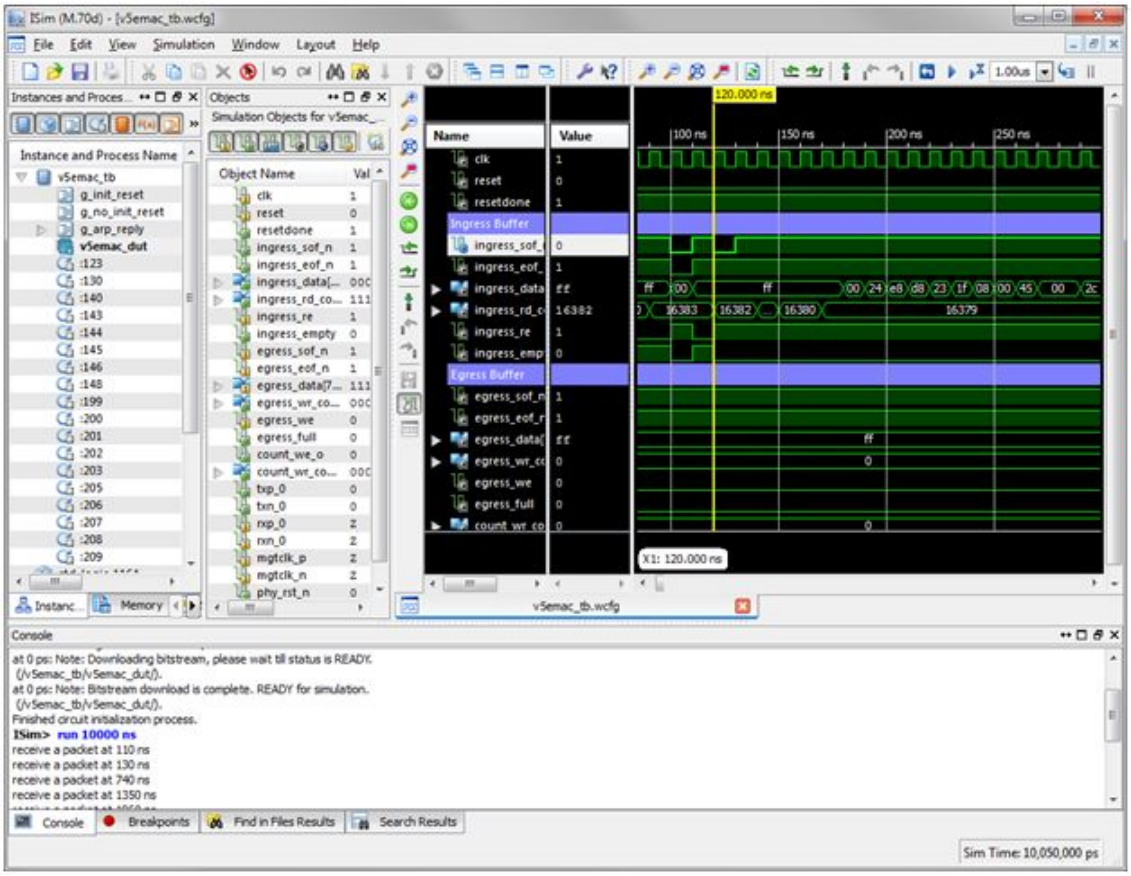


シミュレーションを開始する前に、ハードウェア協調シミュレーション用に生成されたビットストリームで FPGA がプログラムされます。ISim の [Console] パネルに、ビットストリームをダウンロード中であることを示すメッセージ「Downloading bitstream, please wait till status is READY」が表示されます。FPGA がコンフィギュレーションされると、ビットストリームのダウンロードが完了し、シミュレーションの準備ができたことを示すメッセージ「Bitstream download is complete. READY for simulation」が表示されます。この時点で、ソフトウェア シミュレーション フローと同様に、ISim GUI でシミュレーションを実行できます。

まずテストベンチにより reset 信号がアサートされ、システムがリセットされます。reset 信号がディアサートされた直後に、`resetdone` 信号が Low から High に遷移します。これは、リセット プロセスがハードウェアでフル スピードで実行されるからです。同じプロセスをソフトウェアでシミュレーションすると、時間がかかる可能性があります。



イーサネット MAC でパケットが受信されると、パケットが RX LocalLink FIFO から ingress FIFO に送信されます。テストベンチが数千ナノ秒間実行された時点で、ingress FIFO からパケットが読み出され始めます。ingress FIFO でパケット読み出しが発生すると、ISim の [Console] パネルに「receive a packet at 110 ns」のようにパケットが受信されたことを示すメッセージが表示されます。ISim の波形からパケット データ (ingress_data) を観察することもできます。[Run All] コマンドを使用して ISim シミュレーションを継続的に実行すると、パケット ストリームとパケット プロセッサでパケットが処理される様子を観察できます。パケットの有効性をチェックするため、Wireshark (http://www.wireshark.org) などのサードパーティ パケット スニファーをインストールし、ISim のテストベンチでキャプチャされたパケットとスニファーでキャプチャされたパケットを比較できます。



[Enable Incremental Implementation] ハードウェア協調シミュレーション プロパティをオンにすると、テストベンチでパケット プロセッサを変更し、ISim テストベンチを再コンパイルできます。これにより、HDL の開発、コンパイル、デバッグ サイクルにかかる時間を大幅に短縮できます。

XILINX®

付録 A

その他のリソース

- ザイリンクス用語集：http://japan.xilinx.com/company/terms.htm
- ザイリンクス資料：http://japan.xilinx.com/support
- ザイリンクス サポート：http://japan.xilinx.com/support
- 『ML505/ML506/ML507 評価プラットフォーム ユーザー ガイド』(UG347)
- Virtex®-6 ML605 資料
- Spartan®-6 ボードとキット
- ISim ユーザー ガイド
- 『ISE ハードウェア協調シミュレーション チュートリアル：浮動小数点高速フーリエ変換のシミュレーションの高速化』(UG817)
- 『ISE ハードウェア協調シミュレーション チュートリアル：Spartan-6 メモリ コントローラーとオンボードの DDR2 メモリ間の通信』(UG818)
- 『ISE ハードウェア協調シミュレーション チュートリアル：Virtex-5 エンベデッド イーサネット MAC を介したライブ イーサネット トラフィックの処理』(UG819)

XILINX®

付録 B

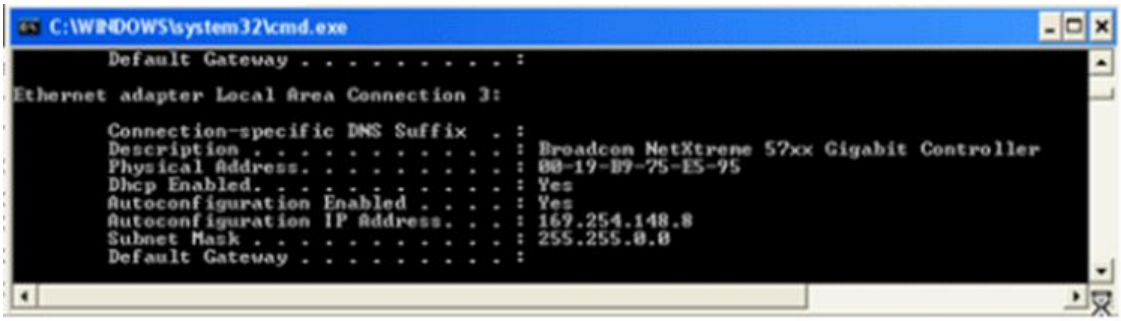
イーサネット ポートの決定

複数のイーサネット インターフェイスが存在する場合にイーサネット ベースのハードウェア協調シミュレーションを実行するには、協調シミュレーションを実行するイーサネット インターフェイスを選択する必要があります。

以前のハードウェア協調シミュレーションをポイント ツー ポイント インターフェイス オプションで実行すると、次のエラー メッセージが表示されます。

```
"ERROR: In process wrapper AHIL_INITIALIZE
Failed to open hardware co-simulation instance.
Error in Point-to-point Ethernet Hardware Co-simulation.
There are multiple Ethernet interfaces available.
Please select an interface."
```

次の手順に従ってイーサネット ポートを決定し、イーサネットのアドレスを設定、確認して、シミュレーション run を検証します。手順 1 は、次の図を参照してください。



1. 協調シミュレーション ボードが接続されているイーサネット ポートを決定します。
   a. システムのコマンド プロンプトでコマンド ターミナル ウィンドウを開きます (**cmd**)。
   b. コマンド ウィンドウで「**ipconfig -all**」と入力して、イーサネット ポートおよびその接続のリストを表示します。
   c. 協調シミュレーション ボードに接続されているイーサネット ポートの物理アドレスを検索します。
   d. 物理アドレスの区切り文字をダッシュ (-) からコロン (:) に変更します。
      例 : `00:19:B9:75:E5:95`

2. 次の手順に従い、ISim でイーサネット ポートを設定、確認します。
   a. ISim GUI を起動します。
   b. DUT (Design Under Test、被試験デバイス) を選択します。
   c. Tcl コンソールを表示します。
   d. Tcl コンソールに次のコマンドを入力します。
      i. イーサネット アドレスを設定します。

```
hwcosim set ethernetInterfaceID
<##:##:L#:##:L#:##> <physical address>
```

      ii. イーサネット アドレスを確認します。

```
hwcosim get ethernetInterfaceID
```

      iii. シミュレーションが実行されるか確認します。

```
run 10us
```

次の図では、ISim GUI でのプロセスが示されています。