

内部ループバックが可能な 4 レーン デザイン

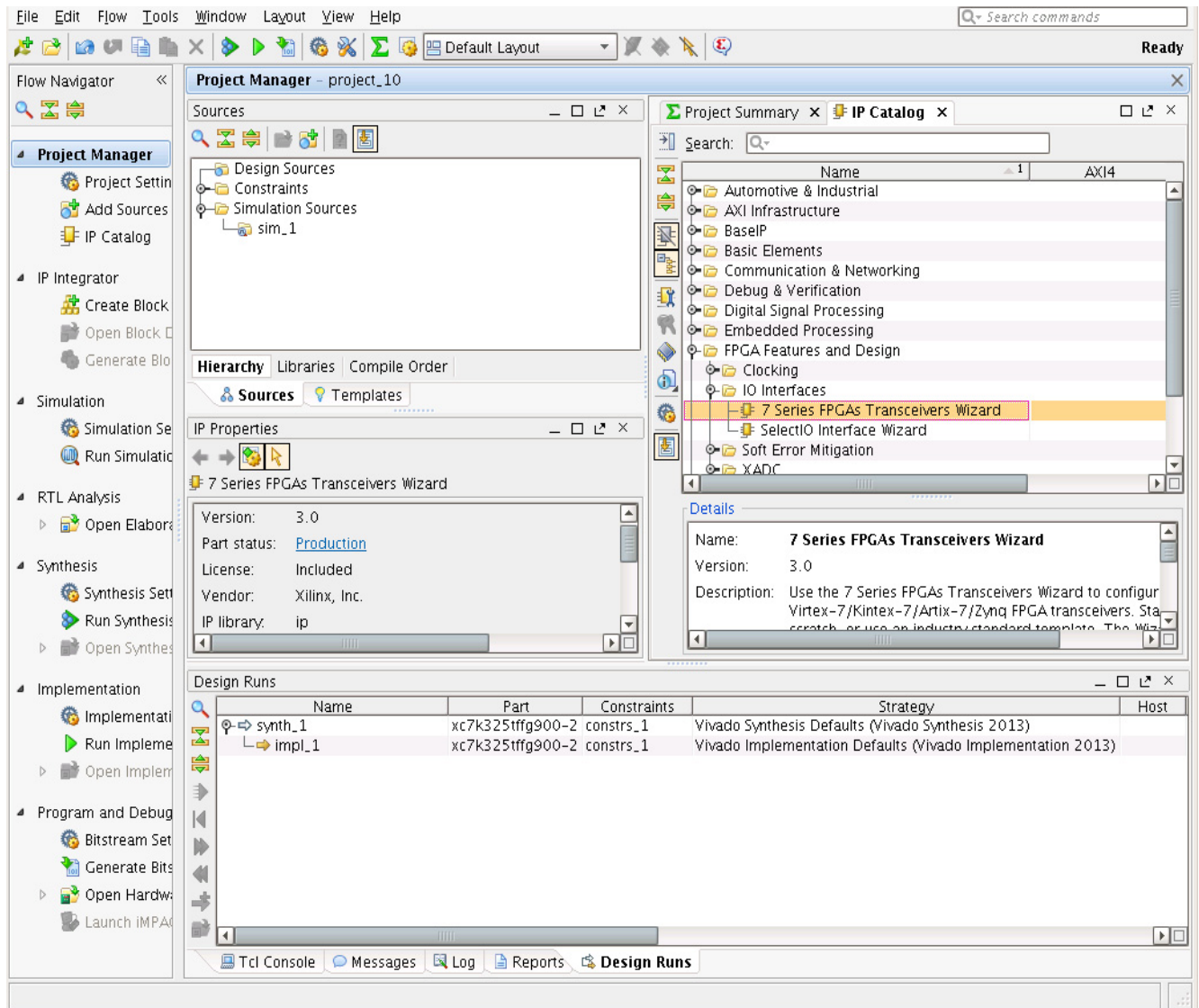
- Kintex-7 FPGA KC705 評価ボード x 1
- 12V、5A ユニバーサル 電源アダプター x 1
- 156.25MHz を生成するのに適したクロック ジェネレーター x 1
- JTAG プラットフォーム USB ケーブル x 1
- SMA-SMA コネクタ ケーブル x 2

Vivado IDE を使用したシリアル トランシーバーの作成手順

Vivado IP カタログを使用し、トランシーバー コアとサンプル デザインをカスタマイズおよび生成する手順を次に示します。この資料でデモンストレーション用にテスト済みのプロトコルは、10GBASE-R のみです。プロトコルのドロップ ダウン リストに表示される標準のプロトコルに対しても、同じウィザード ラッパーの手順を使用できます。

1. Vivado Design Suite を起動します。
2. [Create New Project] をクリックして [Next] をクリックします。
3. プロジェクト名とパスを選択して [Next] をクリックします。
4. [RTL Project] をクリックしてサンプル デザインの実行を許可し、[Do not specify sources at this time] をオンにします。その後 [Next] をクリックします。
5. [xc7k325tffg900-2] をクリックするか、または [Boards] をクリックにして [Kintex-7 FPGA KC705 Evaluation platform] をクリックします。
6. [Next] をクリックして [Finish] をクリックします。

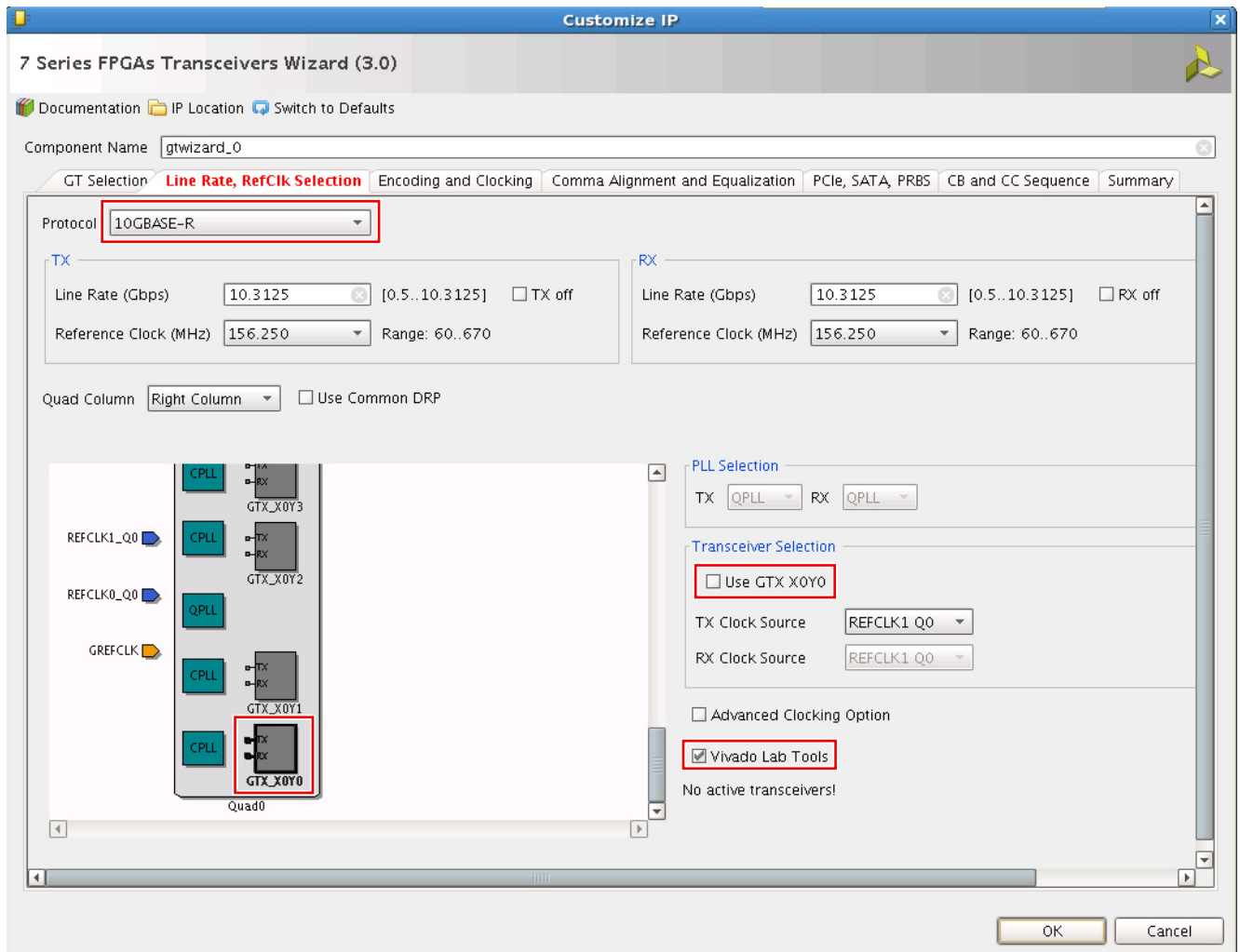
7. Flow Navigator の [Project Manager] 下にある [IP Catalog] をクリックし、「7 Series FPGAs Transceivers Wizard」を検索します (または、Vivado IP カタログで [FPGA Features and Design] → [IO Interface] から生成可能)。図 2 を参照してください。



x1200_02_021114

図 2 : [IP Catalog] ビュー

8. [7 Series FPGA Transceivers Wizard] を右クリックして [Customize IP] をクリックします。
9. [Customize IP] ダイアログ ボックスの [Line Rate, Refclk Selection] タブをクリックします。タブ内では、[Protocol] で [10GBASE-R] を選択し、[Vivado Lab Tools] をオンにして [Transceiver Selection] の [Use GTX_X0Y0] をオフにします (スクロール バーで画面を下に移動させて GTX_X0Y0 を表示させる)。図 3 を参照してください。



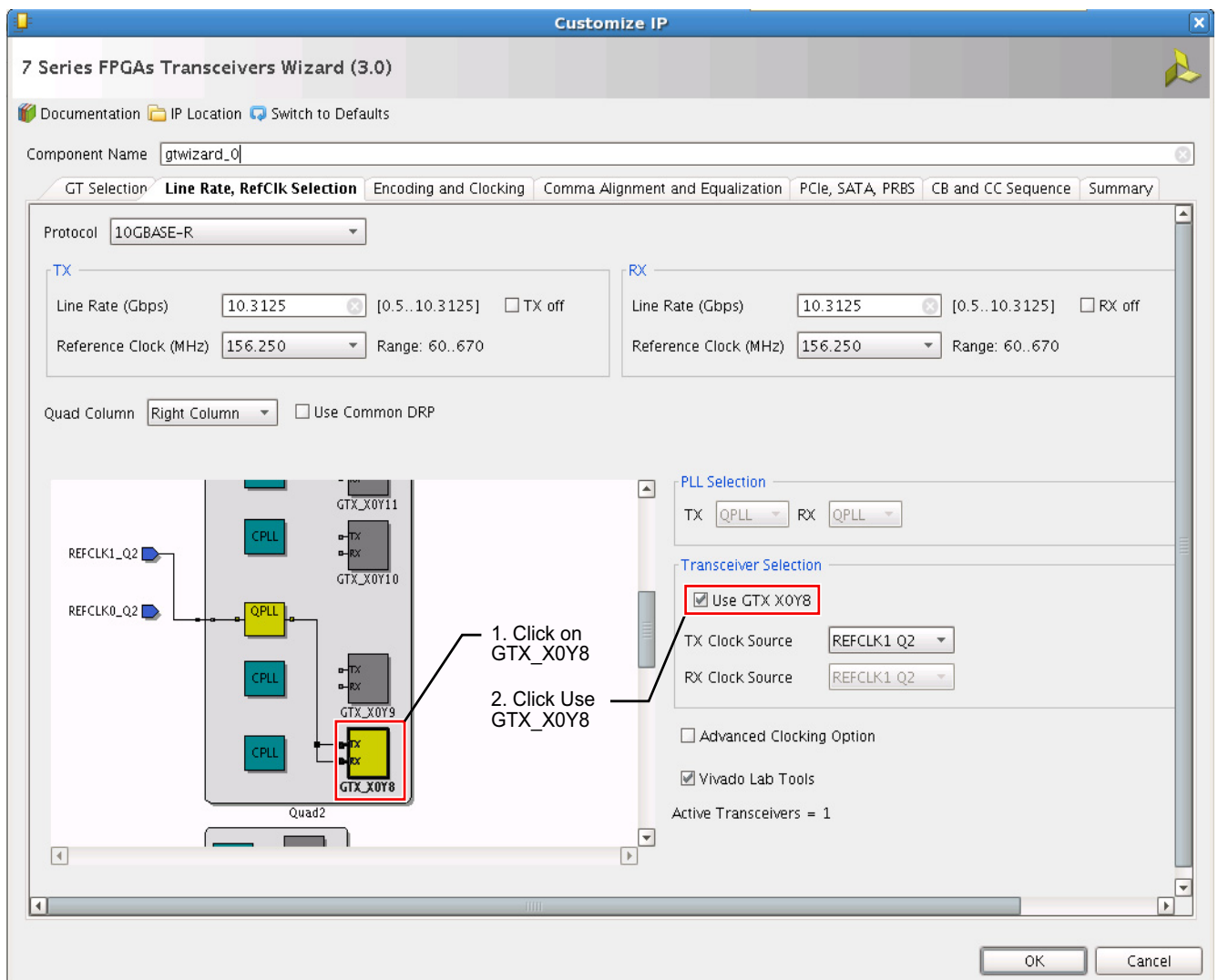
x1200_03_021114

図 3 : Vivado Lab ツールの選択

注記：手順 10～手順 17 は、シングル レーン デザインの場合にのみ実行します。内部ループバックが可能な 4 レーン デザインの場合は、手順 18 に進んでください。

シングル レーン サンプル デザイン

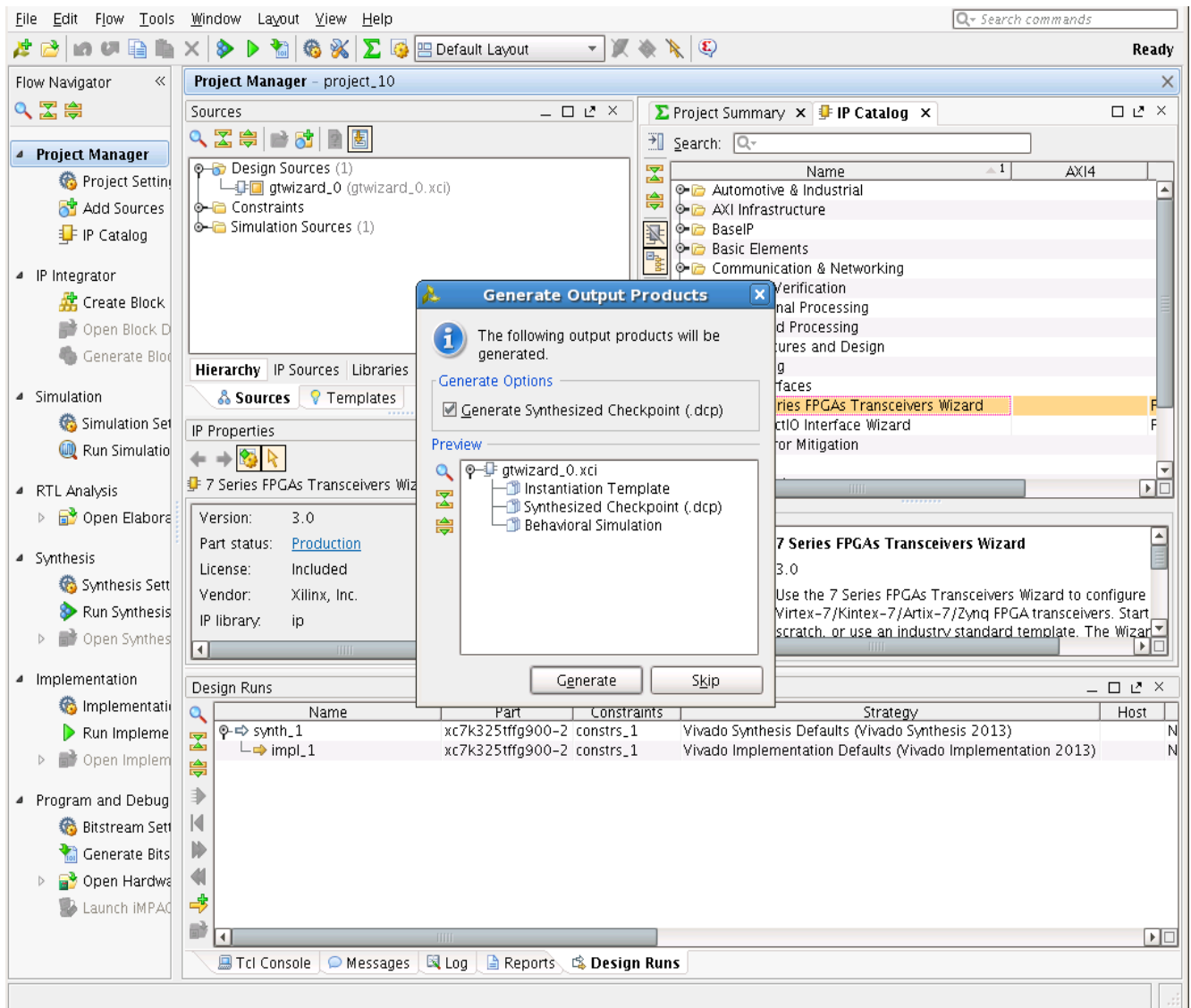
- GUI に表示される図が示す Quad2 の GTX_X0Y8 ブロックをクリックします (スクロール バーで画面を移動させて GTX_X0Y8 を表示させる)。GTX_X0Y8 ブロックをクリックし、[Transceiver Selection] の [Use GTX_X0Y8] をオンにして [OK] をクリックします。図 4 を参照してください。



x1200_04_021114

図 4 : [Customize IP] – シングル レーン デザイン用の 7 Series FPGA Transceiver Wizard

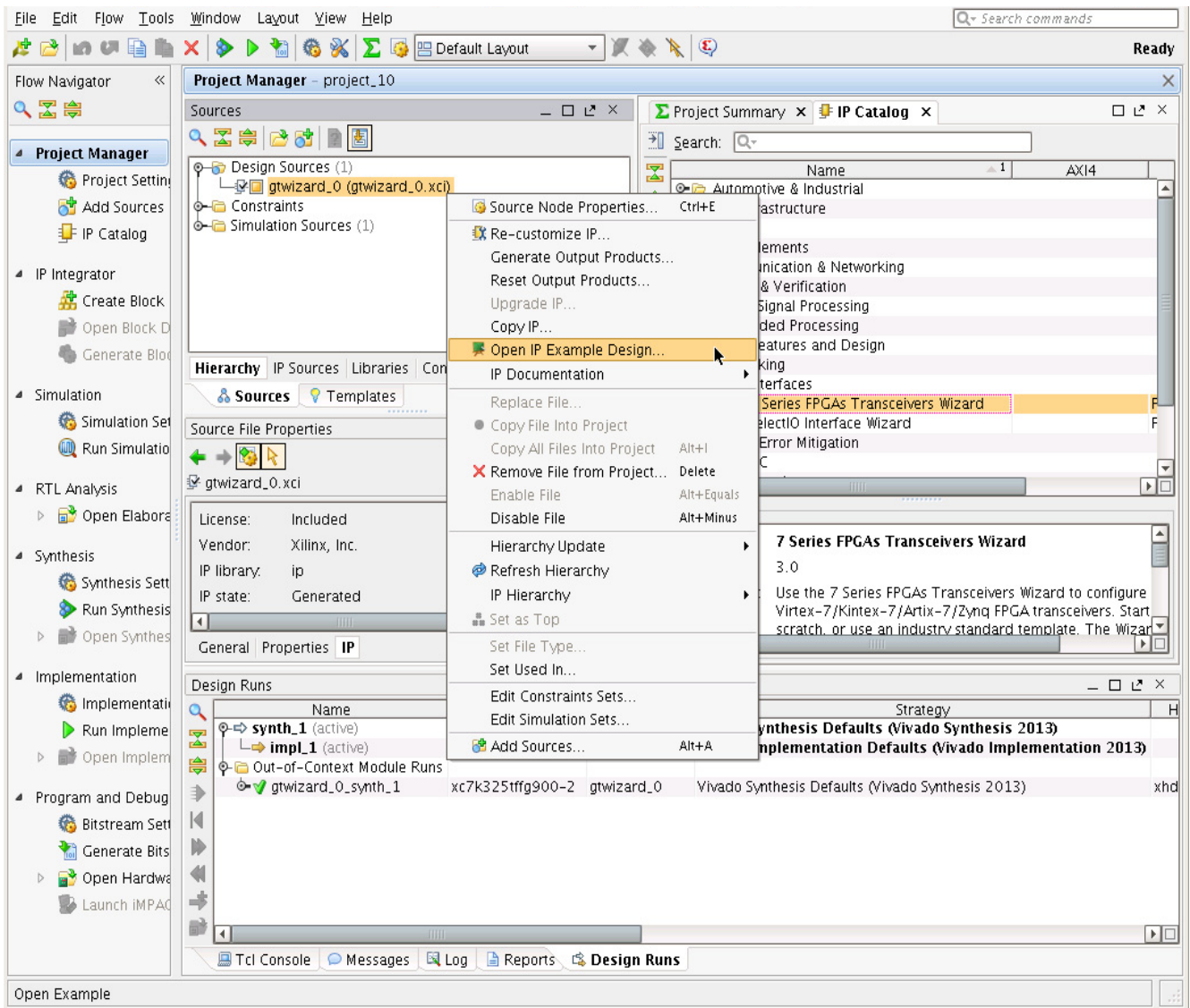
11. [Generate Output Products] ダイアログ ボックスでデフォルトで設定されていない場合は、[Generate Synthesized Checkpoint (.dcp)] をオンにして [Generate] をクリックします。図 5 を参照してください。



x1200_05_021114

図 5 : シングル レーン デザイン用の [Generate Output Products] ダイアログ ボックス

12. 出力ファイルの生成が完了したら、Vivado IDE の [Project Manager] でコア名の [gtwizard_0 (gtwizard_0.xci)] で右クリックして [Open IP Example Design] をクリックし、[OK] をクリックして既存のサンプル デザインを上書きします (図 6)。



x1200_06_021114

図 6 : シングルレーンの IP サンプル デザインを開く

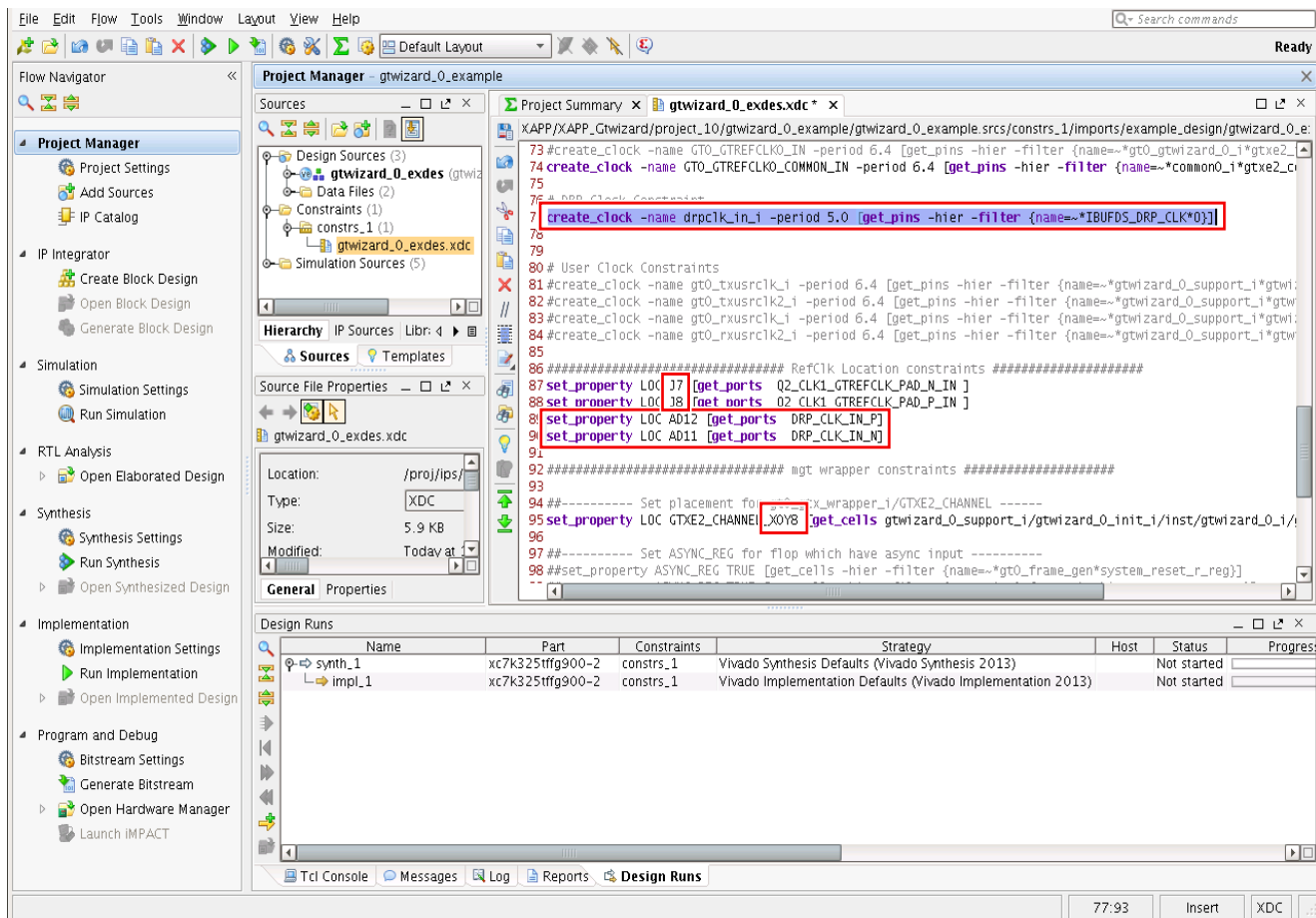
13. 新しく開いた Vivado IDE 画面の [Project Manager] で [Constraints] を展開し、gtwizard_0_exdes.xdc ファイルを開いて次のように変更します。
- 次の DRP_CLK_IN_N/P の制約を追加します。図 7 を参照してください。


```
set_property LOC AD12 [get_ports DRP_CLK_IN_P]
set_property LOC AD11 [get_ports DRP_CLK_IN_N]
```
 - Q2_CLK1_GTREFCLK_PAD_N_IN/P_IN および GTXE2_CHANNEL の位置 (J7, J8, X0Y8) が <component_name>_exdes.xdc 内で正しく設定されているかも確認してください。図 7 を参照してください。


```
set_property LOC J7 [get_ports Q2_CLK1_GTREFCLK_PAD_N_IN ]
set_property LOC J8 [get_ports Q2_CLK1_GTREFCLK_PAD_P_IN ]
set_property          LOC          GTXE2_CHANNEL_X0Y8          [get_cells
gtwizard_0_support_i/gtwizard_0_init_i/inst/gtwizard_0_i/gt0_gtwizard_0_i/gtxe2_i]
```

- c. 次に示すように DRP クロック タイミング制約を置き換え、ファイルを保存します。図 7 を参照してください。

```
create_clock -name drpclk_in_i -period 5.0 [get_pins -hier -filter {name=~*IBUFDS_DRP_CLK*O}]
```



x1200_07_021114

図 7 : シングル レーン デザインの制約ファイル

14. [Design Sources] から gtwizard_0_exdes.v ファイルを開いて次のように変更します。

- a. ポート宣言に次のポートを追加します。

```
input wire DRP_CLK_IN_P,
input wire DRP_CLK_IN_N,
```

- b. 次のポート宣言を削除します。

```
input wire DRP_CLK_IN,
```

- c. ワイヤ宣言に次の行を追加します。

```
wire DRP_CLK_IN;
```

- d. 次のモジュール インスタンスエーションを追加します。

```
IBUFDS IBUFDS_DRP_CLK
```

```
(
```

```
  .I (DRP_CLK_IN_P),
```

```
  .IB (DRP_CLK_IN_N),
```

```
  .O (DRP_CLK_IN)
```

```
);
```

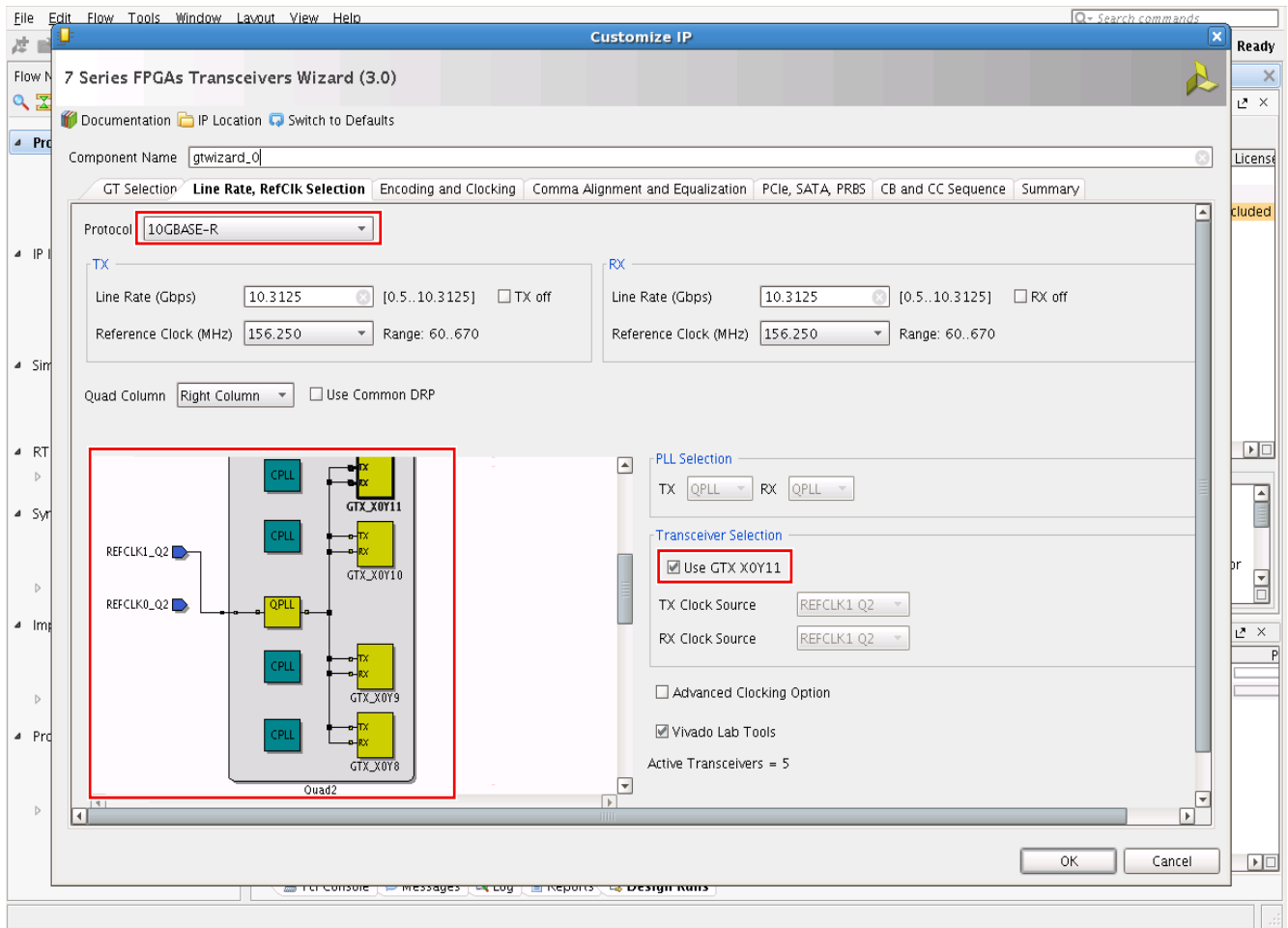
- e. wire track_data_out_i を検索し、(* mark_debug = "TRUE" *) wire track_data_out_i; に変更します。

- f. `probe_in0(tied_to_ground_i)` を検索し、`probe_in0(gt0_rxfsmresetdone_i)` に変更してファイルを保存します。
15. [Run Synthesis] をクリックし、合成が完了したら [Open Synthesized Design] をクリックします。合成の実行にしばらく時間が経過した後、合成されたネットリストのビューを開きます。合成されたデザインが開いたら、[手順 16](#)に進みます。
16. Tcl コンソールに `write_debug_probes <ltx_file_name>` と入力して Enter キーを押します。これで `<component_name>_example` フォルダに `<ltx_file_name>.ltx` が作成されます。
17. [Run Implementation] をクリックし、インプリメンテーションが完了したら [Open Implemented Design] をクリックします。ダイアログ ボックスの [Yes] をクリックしてインプリメントされたデザインを開きます。Vivado IDE で、インプリメントされたデザインのネットリスト ビューが開きます。次の 2 つの Tcl コメントを Tcl コンソールに入力し、`routed.bit` ビットストリーム ファイルを生成します。

```
set_property SEVERITY {Warning} [get_drc_checks NSTD-1]; set_property SEVERITY {Warning} [get_drc_checks UCIO-1]
write_bitstream -bitgen_options {-g UnconstrainedPins:Allow} -file routed.bit -force
```

内部ループバックが可能な 4 レーン デザイン

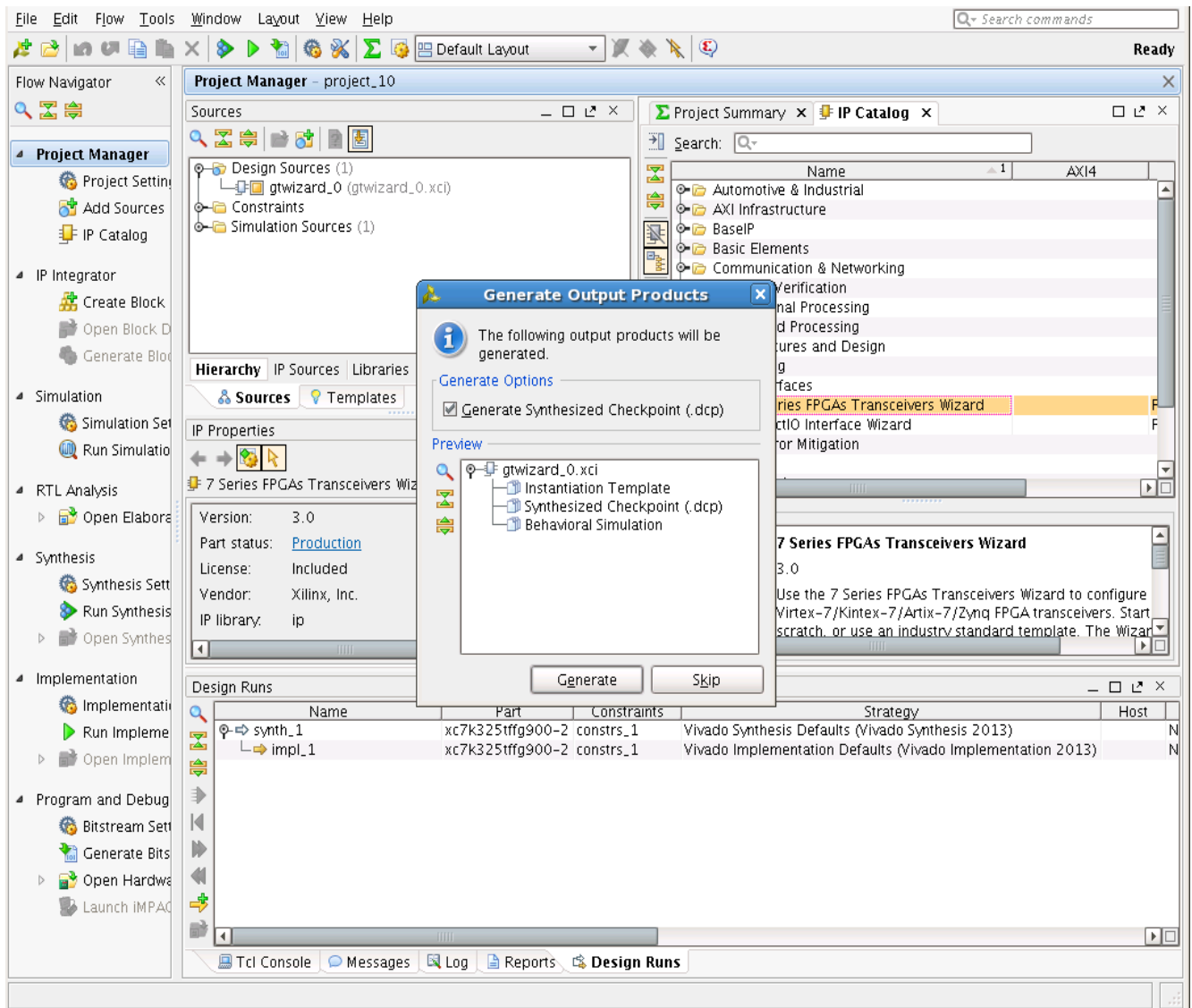
18. Quad2 の GTX_X0Y8 をクリックし、[Transceiver Selection] の [Use GTX X0Y8] をオンにします (手順 10 および図 4)。同様に、Quad2 の GTX_X0Y9、GTX_X0Y10 および GTX_X0Y11 について [Transceiver Selection] の [Use GTX X0Y9]、[Use GTX X0Y10]、[Use GTX X0Y11] をそれぞれオンにします。続いて [OK] をクリックします。図 8 を参照してください。



x1200_08_021114

図 8 : [Customize IP] – 4 レーン デザイン用の 7 Series FPGA Transceiver Wizard

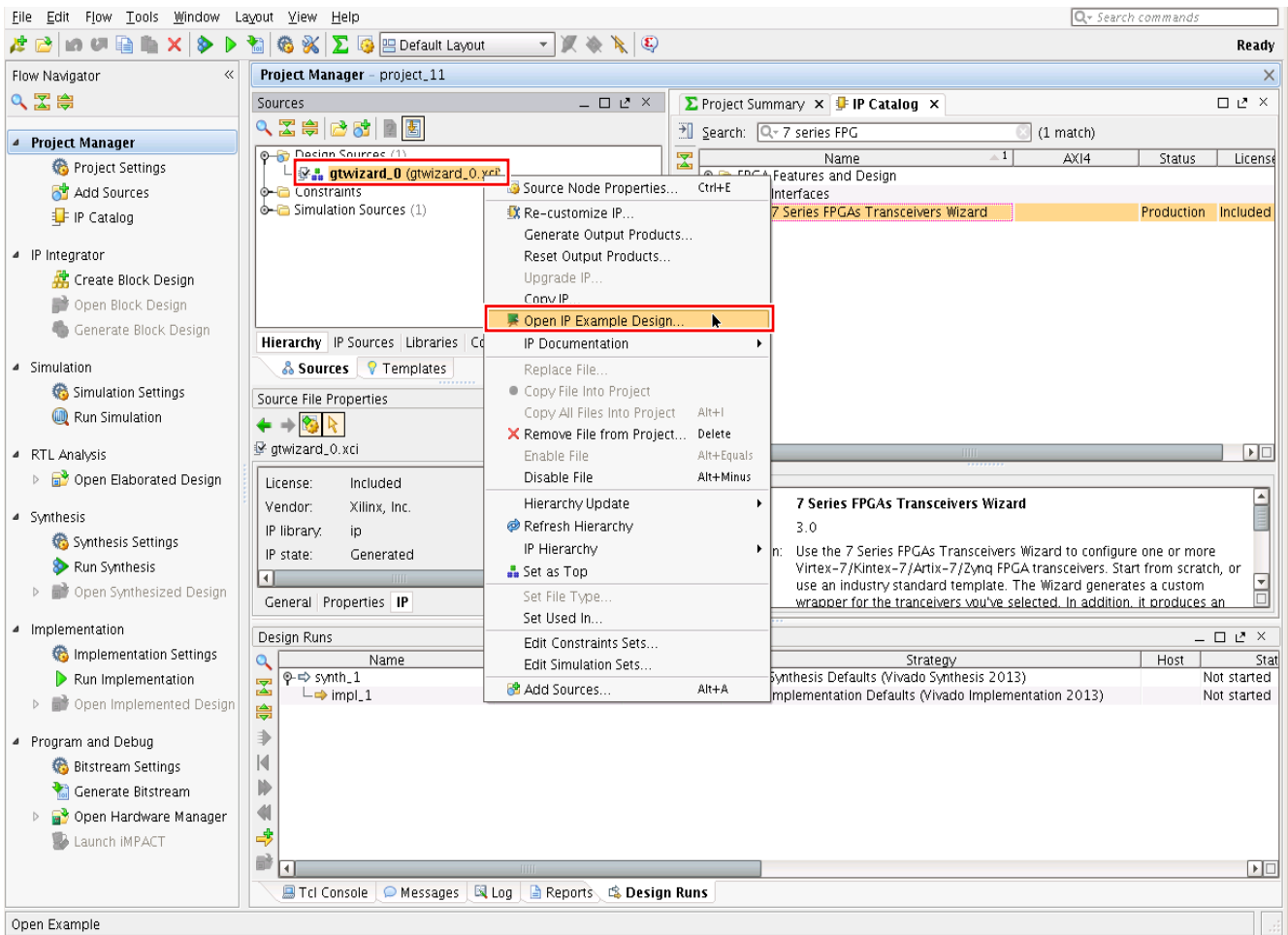
19. [Generate Output Products] ダイアログ ボックスでデフォルトで設定されていない場合は、[Generate Synthesized Checkpoint (.dcp)] をオンにして [Generate] をクリックします。図 9 を参照してください。



x1200_09_021114

図 9 : 4 レーン デザイン用の [Generate Output Products] ダイアログ ボックス

20. 出力ファイルの生成が完了したら、Vivado IDE の [Project Manager] でコア名の [gtwizard_0 (gtwizard_0.xci)] で右クリックして [Open IP Example Design] をクリックし、[OK] をクリックして既存のサンプル デザインを上書きします (図 10)。



x1200_10_021114

図 10 : 4 レーンの IP サンプル デザインを開く

21. 新しく開いた Vivado IDE 画面の [Project Manager] で [Constraints] を展開し、gtwizard_0_exdes.xdc ファイルを開いて次のように変更します。
- 次の DRP_CLK_IN_N/P の制約を追加します。図 11 を参照してください。


```
set_property LOC AD12 [get_ports DRP_CLK_IN_P]
set_property LOC AD11 [get_ports DRP_CLK_IN_N]
```
 - Q2_CLK1_GTREFCLK_PAD_N_IN/P_IN および GTXE2_CHANNEL の位置 (J7, J8, X0Y8, X0Y9, X0Y10, X0Y11) が <component_name>_exdes.xdc 内で正しく設定されているかも確認してください。図 11 を参照してください。

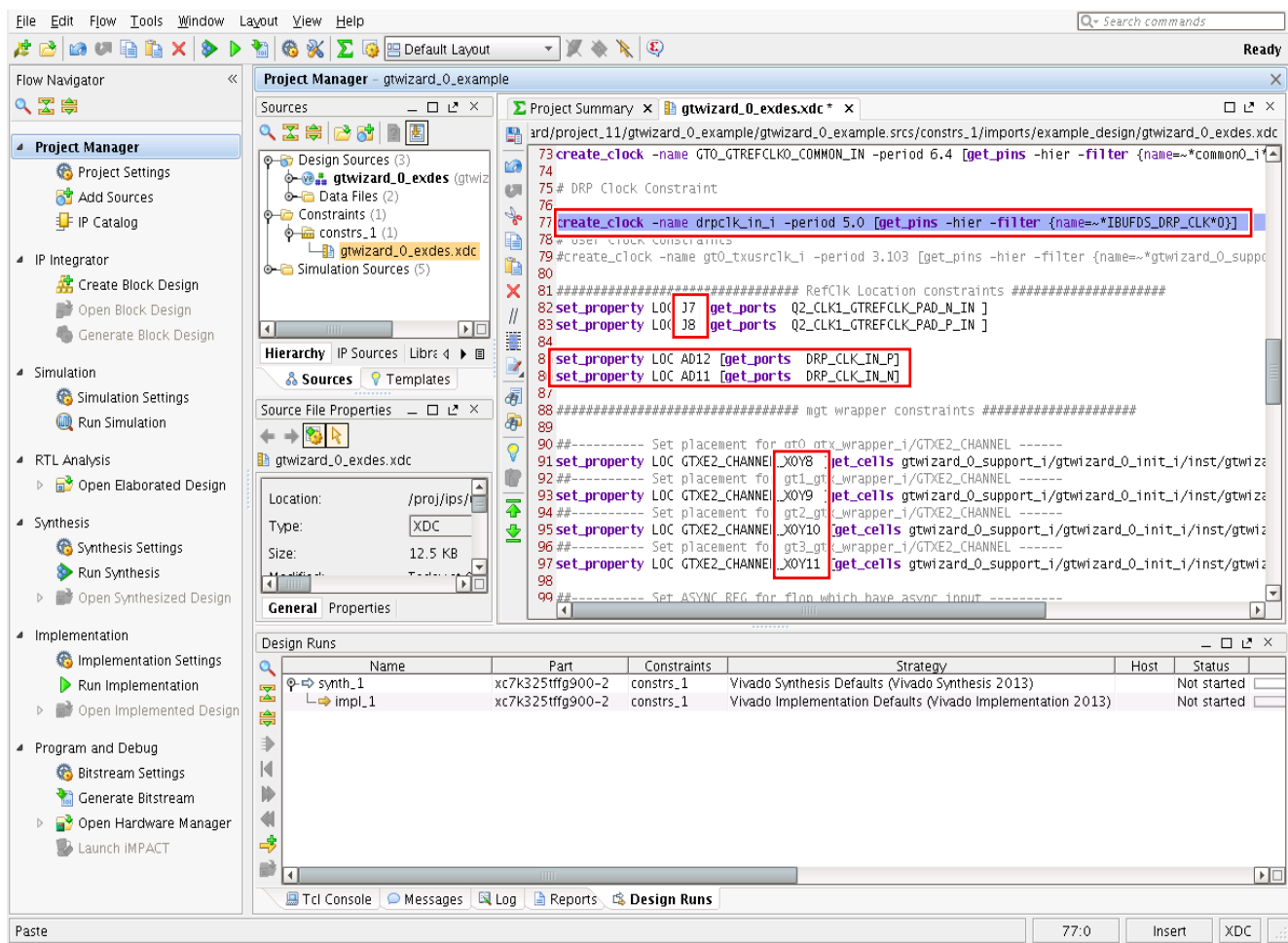

```
set_property LOC J7 [get_ports Q2_CLK1_GTREFCLK_PAD_N_IN ]
set_property LOC J8 [get_ports Q2_CLK1_GTREFCLK_PAD_P_IN ]
set_property          LOC          GTXE2_CHANNEL_X0Y8          [get_cells
gtwizard_0_support_i/gtwizard_0_init_i/inst/gtwizard_0_i/gt0_gtwizard_
0_i/gtxe2_i]
set_property          LOC          GTXE2_CHANNEL_X0Y9          [get_cells
gtwizard_0_support_i/gtwizard_0_init_i/inst/gtwizard_0_i/gt1_gtwizard_
0_i/gtxe2_i]
```

```
set_property LOC GTXE2_CHANNEL_X0Y10 [get_cells
gtwizard_0_support_i/gtwizard_0_init_i/inst/gtwizard_0_i/gt2_gtwizard_0_i/gtxe2_i]
```

```
set_property LOC GTXE2_CHANNEL_X0Y11 [get_cells
gtwizard_0_support_i/gtwizard_0_init_i/inst/gtwizard_0_i/gt3_gtwizard_0_i/gtxe2_i]
```

- c. 次に示すように DRP クロック タイミング制約を変更し、ファイルを保存します。図 11 を参照してください。

```
create_clock -name drpclk_in_i -period 5.0 [get_pins -hier -filter
{name=~*IBUFDS_DRP_CLK*0}]
```



x1200_11_021114

図 11 : 4 レーン デザインの制約ファイル

22. [Design Sources] から gtwizard_0_exdes.v ファイルを開いて次のように変更します。

- a. ポート宣言に次のポートを追加します。

```
input wire DRP_CLK_IN_P,
input wire DRP_CLK_IN_N,
```

- b. 次のポート宣言を削除します。

```
input wire DRP_CLK_IN,
```

- c. ワイヤ宣言に次の行を追加します。

```
wire DRP_CLK_IN;
wire rxfsmresetdone;
```

- d. 次のモジュール インスタンスエーションを追加し、IBUFDS を使用してボードの差動クロックをシングルエンド クロックに変換します。

```
IBUFDS IBUFDS_DRP_CLK
```

```
(
    .I (DRP_CLK_IN_P),
    .IB (DRP_CLK_IN_N),
    .O (DRP_CLK_IN)
);
```

```
assign rx fsmresetdone = (gt0_rxfsmresetdone_i & gt1_rxfsmresetdone_i &
gt2_rxfsmresetdone_i & gt3_rxfsmresetdone_i);
```

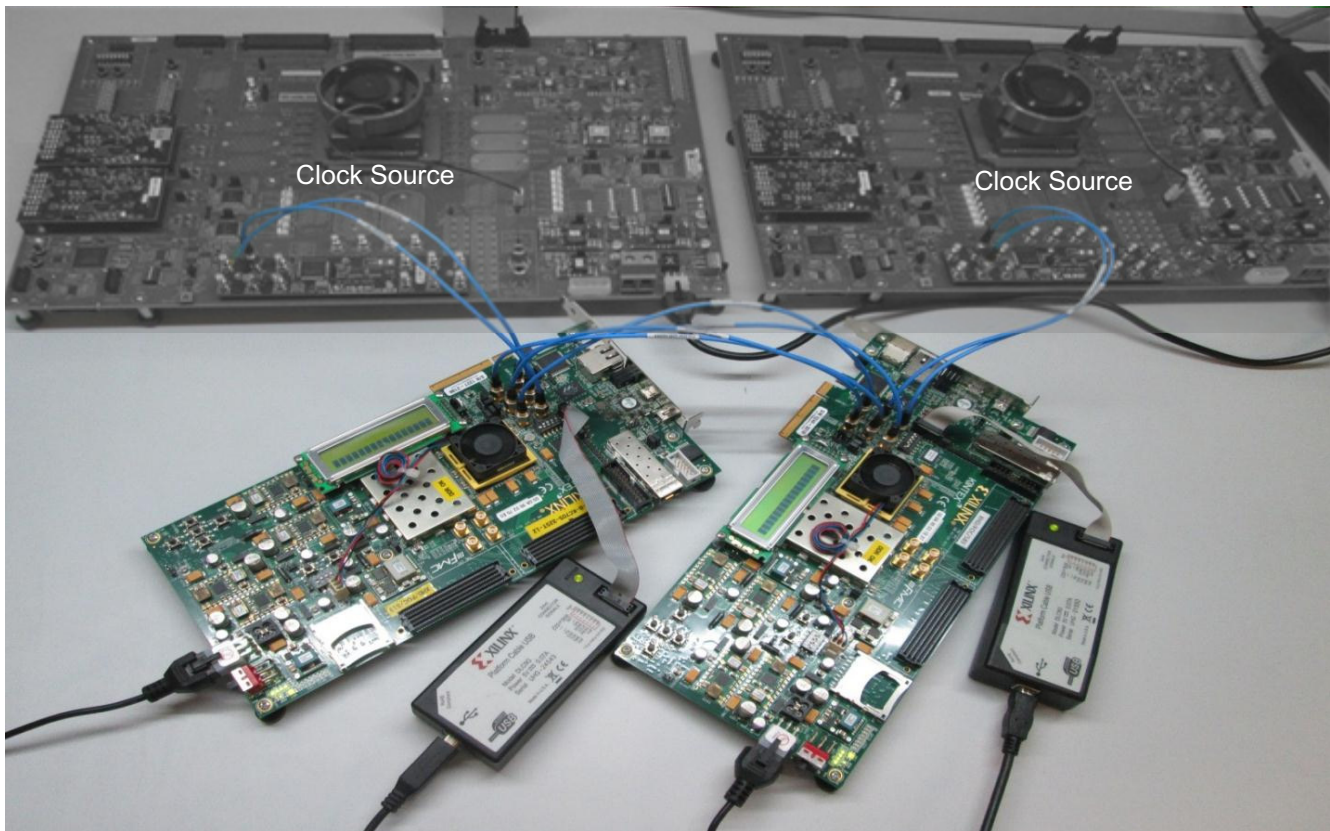
- e. `wire track_data_out_i` を検索し、その行を (`* mark_debug = "TRUE" *`) `wire track_data_out_i`; 用に変更します。
 - f. 内部ループバック テスト用に次に示すようにループバック ポートの入力値をすべて変更します。これにより、近端 PMA ループバック テストが有効になります。
 - i. `(gt0_loopback_i) to (3'b010)`
 - ii. `(gt1_loopback_i) to (3'b010)`
 - iii. `(gt2_loopback_i) to (3'b010)`
 - iv. `(gt3_loopback_i) to (3'b010)`
 - g. `probe_in0(tied_to_ground_i)` を検索し、`probe_in0(gt0_rxfsmresetdone_i)` に変更してファイルを保存します。
23. [Run Synthesis] をクリックし、合成が完了したら [Open Synthesized Design] をクリックします。合成の実行にしばらく時間が経過した後、合成されたネットリストのビューを開きます。合成されたデザインが開いたら、Tcl コンソールに `write_debug_probes <ltx_file_name>` と入力して Enter キーを押します。これで `<component_name>_example` フォルダに `<ltx_file_name>.ltx` が作成されます。
24. [Run Implementation] をクリックし、インプリメンテーションが完了したら [Open Implemented Design] をクリックします。ダイアログ ボックスの [Yes] をクリックしてインプリメントされたデザインを開きます。Vivado IDE で、インプリメントされたデザインのネットリスト ビューが開きます。次の 2 つの Tcl コメントを Tcl コンソールに入力し、`routed.bit` ビットストリーム ファイルを生成します。

```
set_property SEVERITY {Warning} [get_drc_checks NSTD-1]; set_property
SEVERITY {Warning} [get_drc_checks UCIO-1]
write_bitstream -bitgen_options {-g UnconstrainedPins:Allow} -file
routed.bit -force
```

シングル レーン デザインのハードウェア セットアップ接続

次の手順に従って、2 つの KC705 ボード (ボード 1 およびボード 2) 間のハードウェア接続をセットアップします。

1. ボード 1 の TXP はボード 2 の RXP に接続し、ボード 1 の TXN はボード 2 の RXN に接続します。
2. 同様に、ボード 2 の TXP はボード 1 の RXP に接続し、ボード 2 の TXN はボード 1 の RXN に接続します。
3. 各 KC705 ボードへの基準クロックは、それぞれ別のクロック ソースから供給する必要があります (図 12)。図 12 に示すセットアップでは、VC7222 ボードを使用して 156.25MHz クロックを生成しています。ユーザー自身のクロック ソースを使用してこのクロックを生成することもできます。



x1200_12_021114

図 12: シングル レーン ボードのセットアップ

注記: 4 レーンを使用する場合、図 12 に示す接続は不要となります。代わりに、図 13 に示す単純なセットアップになります。

4. USB ケーブルをホスト PC から USB JTAG ポートに接続します。適切なデバイス ドライバーがインストールされていることを確認します。
5. 電源アダプターを KC705 ボードに接続します。

4 レーン デザインのハードウェア セットアップ接続

1. 1つのクロック ソースから供給される基準クロックを KC705 ボードに接続しますが、このクロックは別のソースから供給する必要があります (図 13)。ユーザー自身のクロック ソースを使用してこのクロックを生成することもできます。
2. USB ケーブルをホスト PC から USB JTAG ポートに接続します。適切なデバイス ドライバーがインストールされていることを確認します。
3. 電源アダプターを KC705 ボードに接続します。

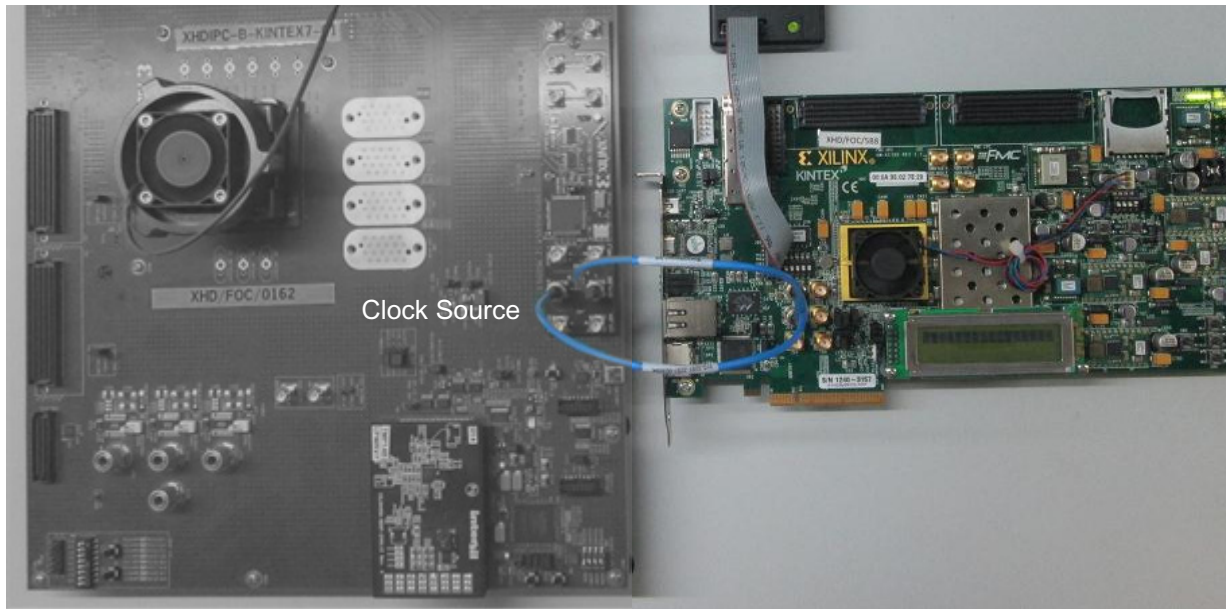
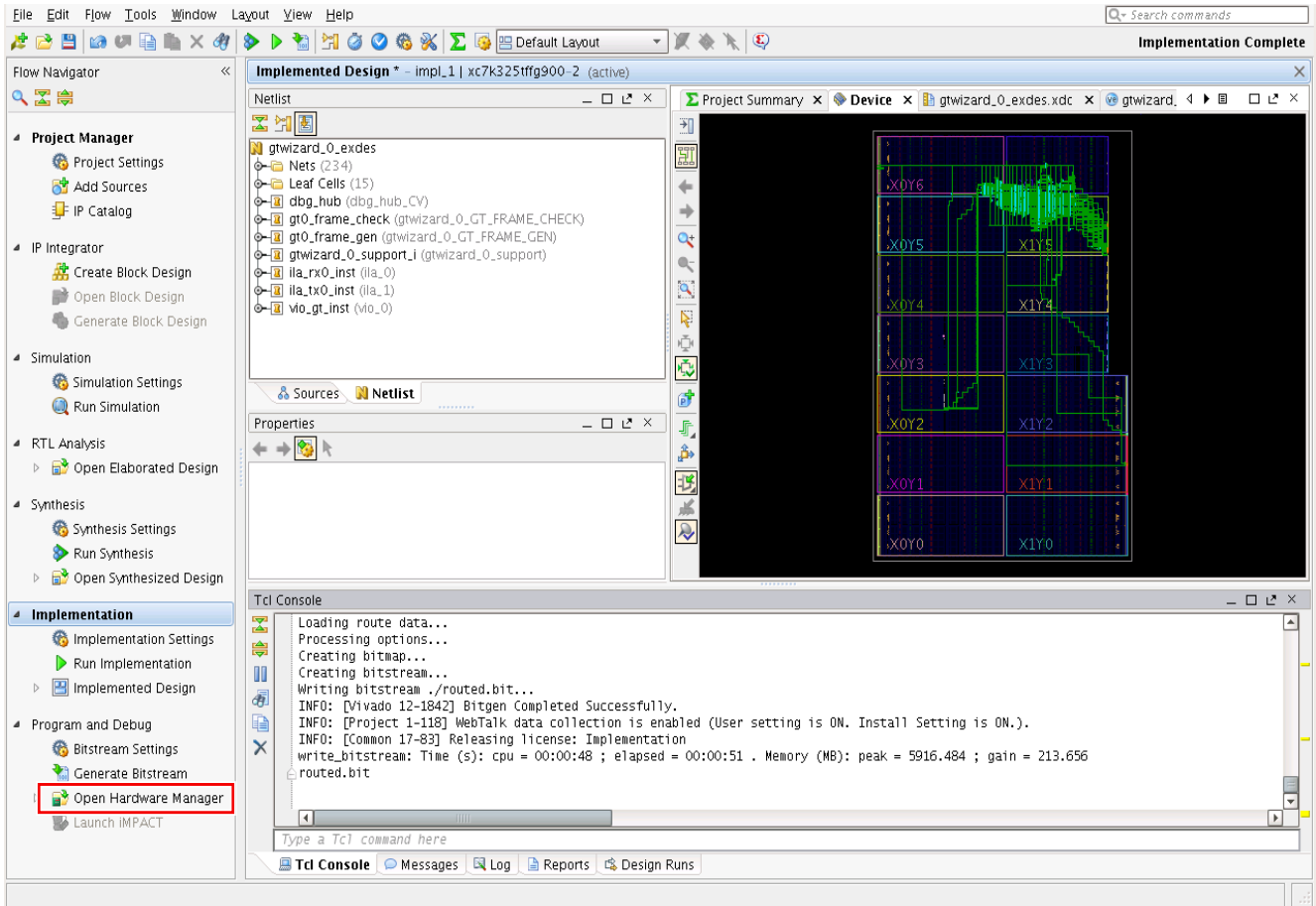


図 13 : 4 レーン ボードのセットアップ

デバイスのプログラム

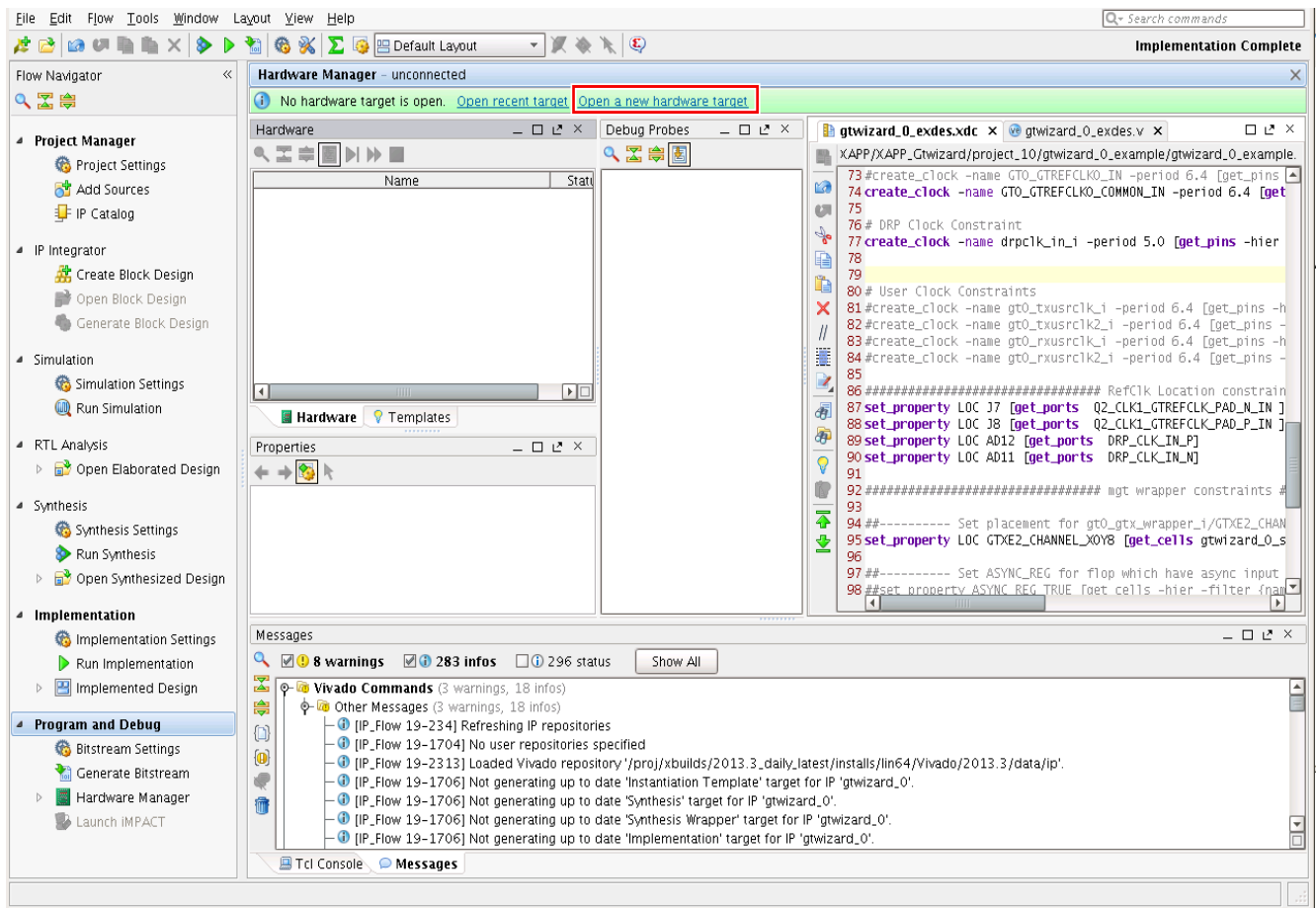
1. Vivado IDE の Flow Navigator の [Program and Debug] 下にある [Open Hardware Manager] をクリックします (図 14)。



x1200_14_021114

図 14 : ハードウェア マネージャーを開く

2. [Hardware Manager] ページの上部にある [Open a new hardware target] をクリックして [Next] をクリックします (図 15)。

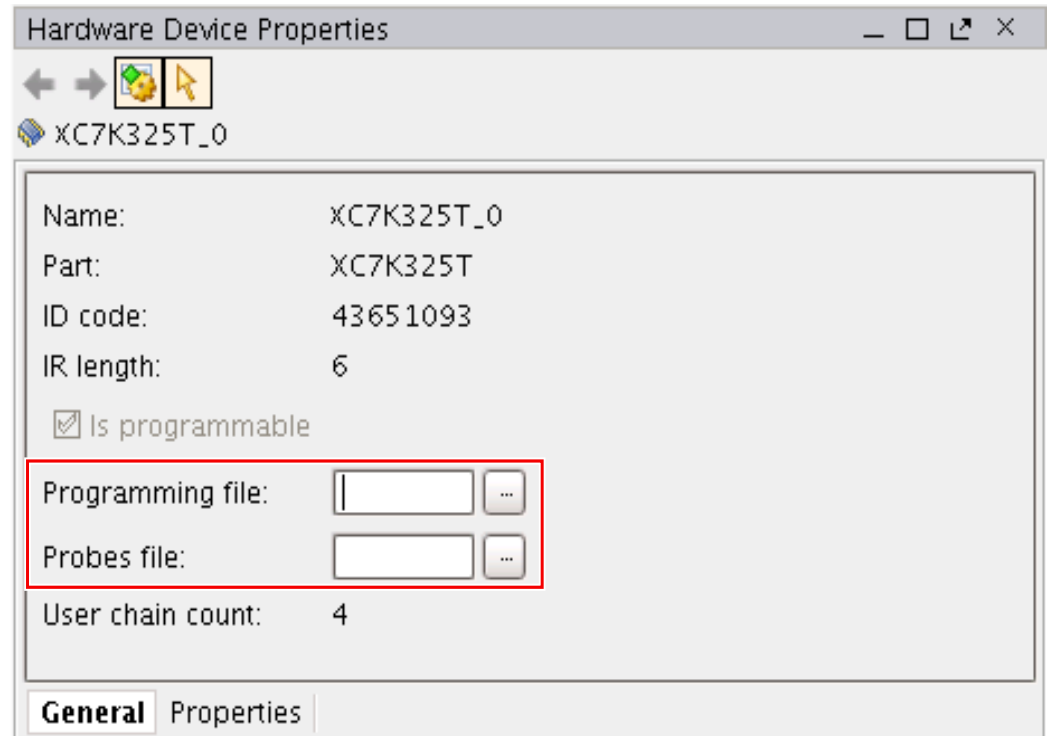


x1200_16_021114

図 15 : 新しいハードウェア ターゲットを開く

3. サーバー名 <host[:port]> の設定を [localhost:60001] のままにして [Next] をクリックします。
4. [Hardware Targets] にリストされているプラットフォーム ボードの 1 つをクリックしてハイライト表示し、[Next] をクリックして [Finish] をクリックします。

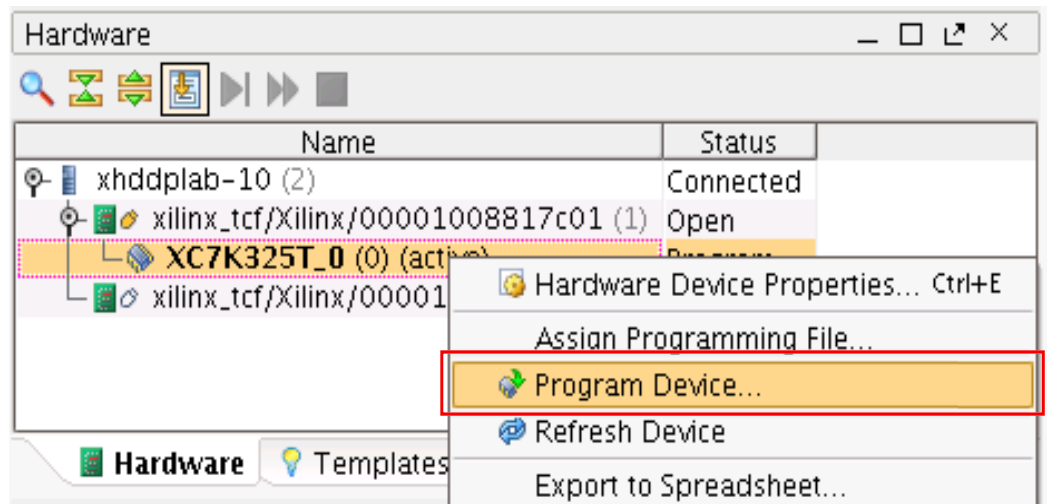
- [Hardware] ビューにリストされているアクティブなデバイス [XC7K325T_0(0) (Active)] をクリックします。[Hardware Device Properties] ビューで、[Programming file] にビットストリーム ファイル名 (routed.bit) を指定し、[Probes file] にプローブ ファイル名 (<ltx_file_name>.ltx) を指定します (図 16)。



x1200_17_021114

図 16 : [Hardware Device Properties] ビュー

- [Hardware] ビューの (アクティブな) デバイスを右クリックし、[Program Device] をクリックします (図 17)。ビットストリーム ファイルのパスと名前が正しいことを確認して [OK] をクリックします。

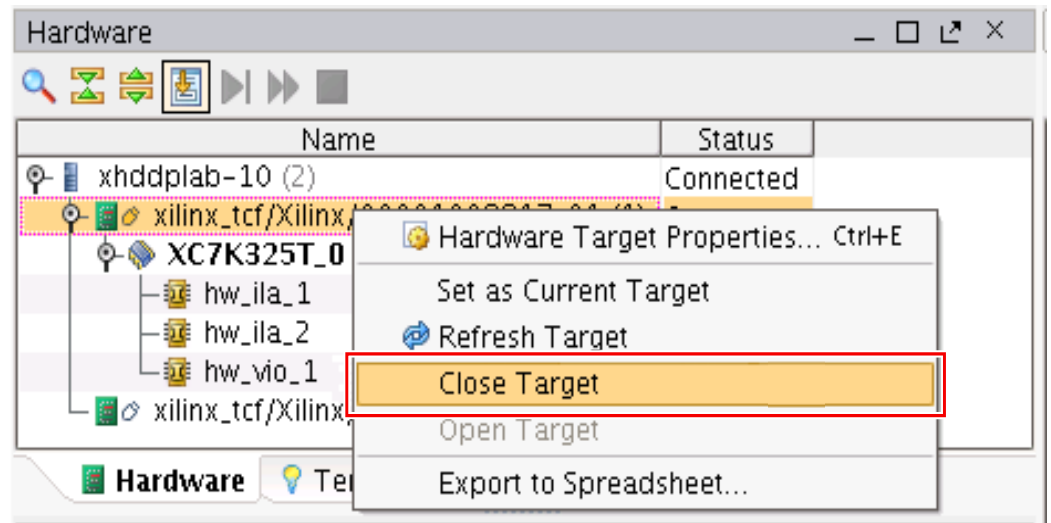


x1200_18_021114

図 17 : [Program Device] の選択

注記 : 4 レーン ループバック テストでは、手順 7、手順 8、および手順 9 を実行する必要はありません。

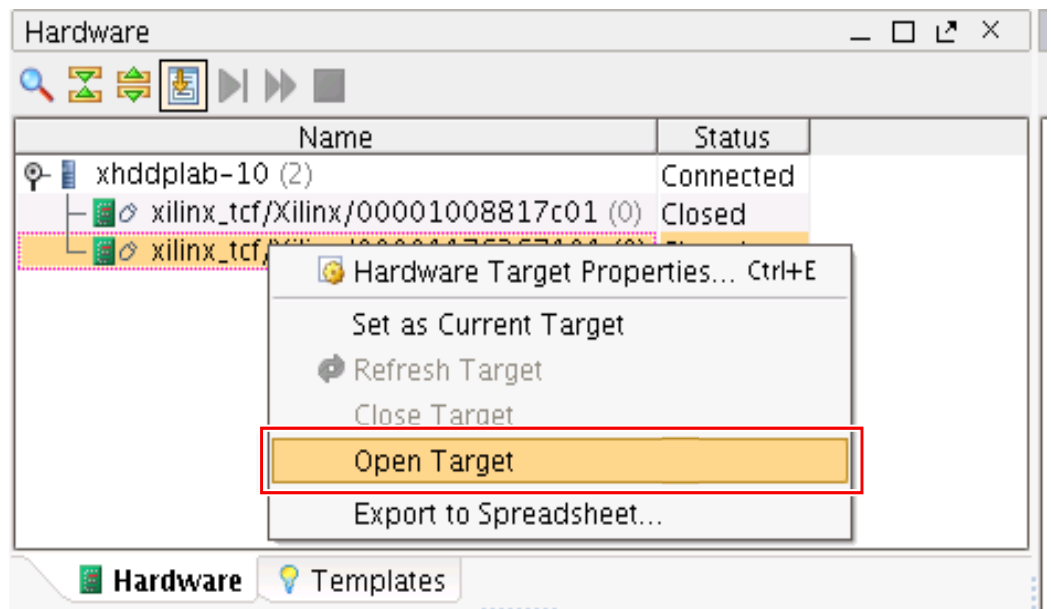
7. プログラムが完了したら [Hardware] ビューでプログラムされたターゲット デバイスを右クリックし、[Close Target] をクリックします (図 18)。



x1200_19_021114

図 18 : [Close Target] の選択

8. [Hardware] ビューで 2 つ目のターゲット プラットフォームを右クリックし、[Open Target] をクリックします (図 19)。

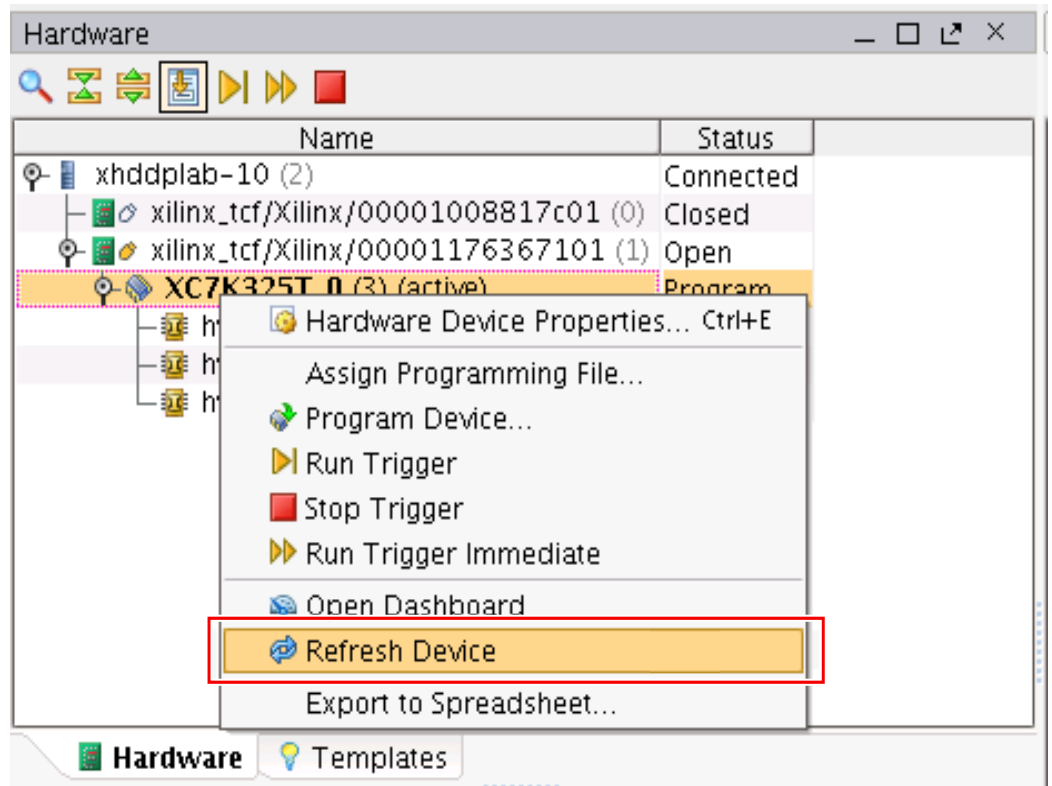


x1200_20_021114

図 19 : [Open Target] の選択

9. 最初のターゲットで使用したビットストリーム ファイルとプローブ ファイルを用いて手順 5 および手順 6 を繰り返します。

10. プログラムが完了したら [Hardware] ビューでプログラムされたターゲット デバイスを右クリックし、[Refresh Device] をクリックします (図 20)。

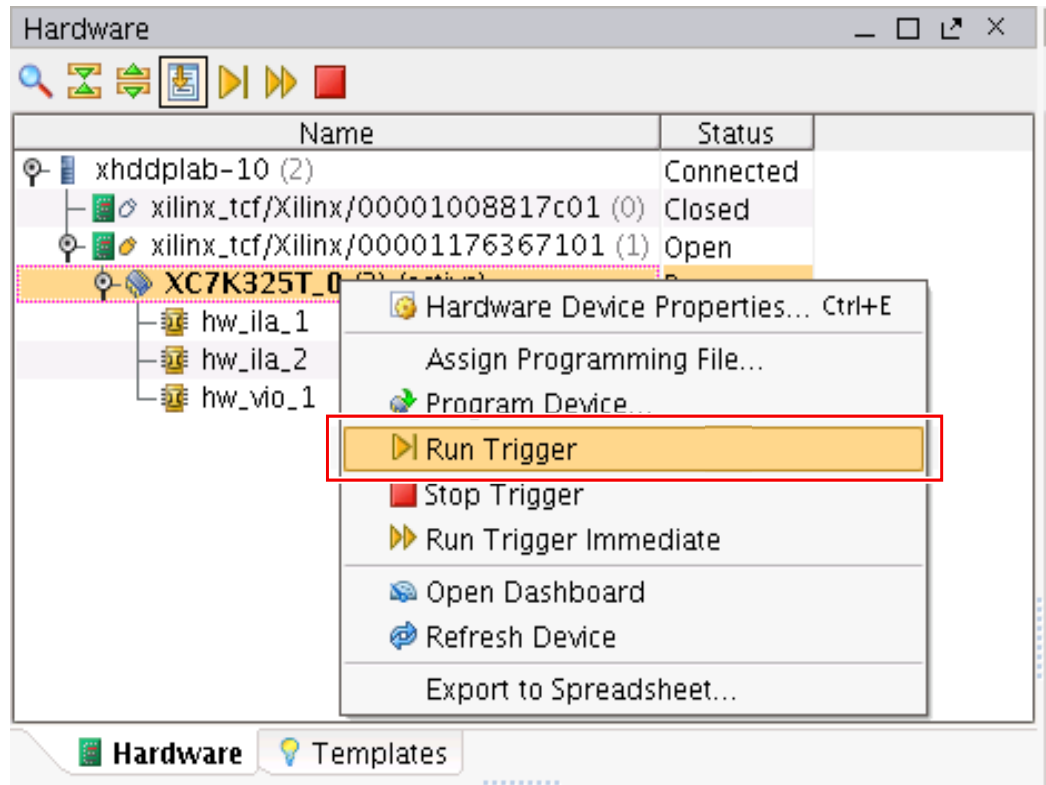


x1200_21_021114

図 20 : [Refresh Device] の選択

デザインの実行

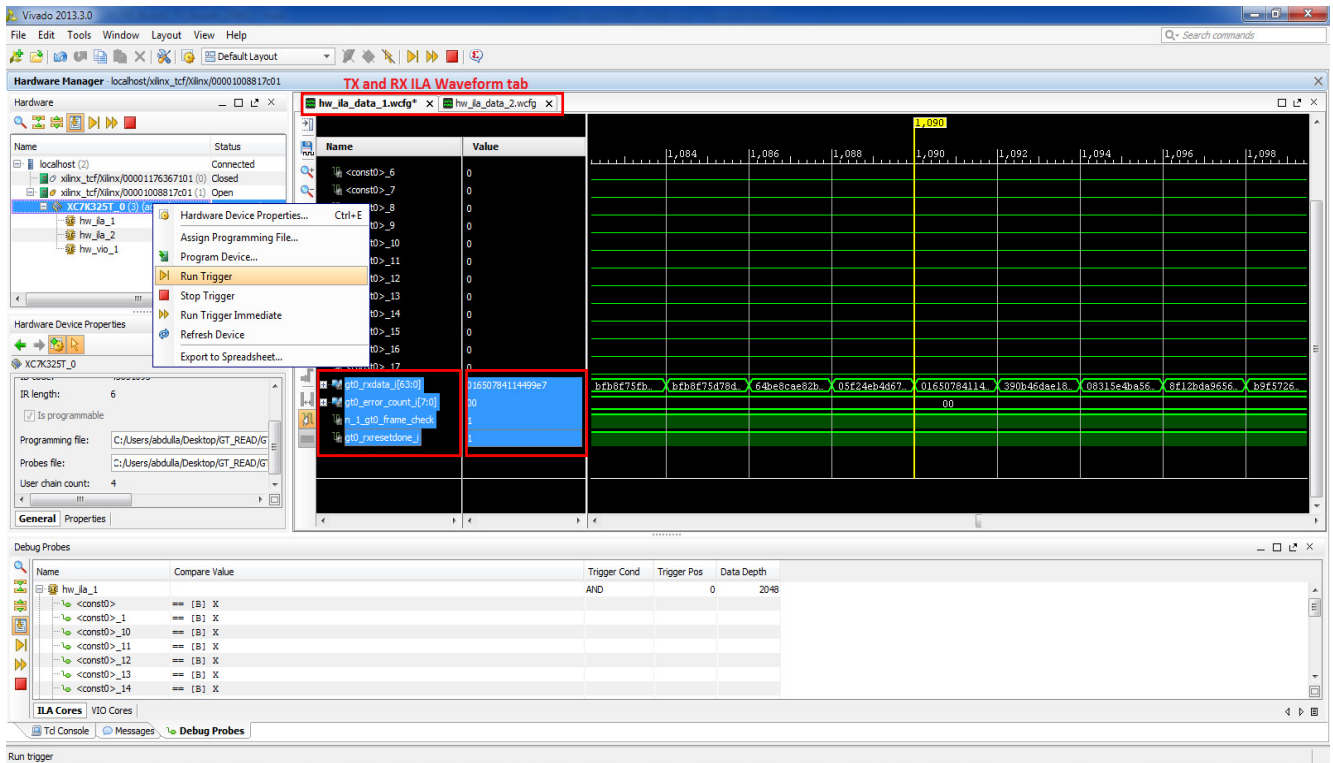
1. [Hardware] ビューに表示されているデバイスを右クリックし、[Run Trigger] をクリックします (図 21)。ここでは、ILA コアに追加されているすべてのネットが確認できるはずですが。



x1200_22_021114

図 21 : [Run Trigger] の選択

2. 表示された波形画面で、track_data_out_i 信号、gt0_rxreset_done_ila 信号、gt0_txresetdone_ila 信号が High であることを確認します (図 22)。



x1200_23_021114

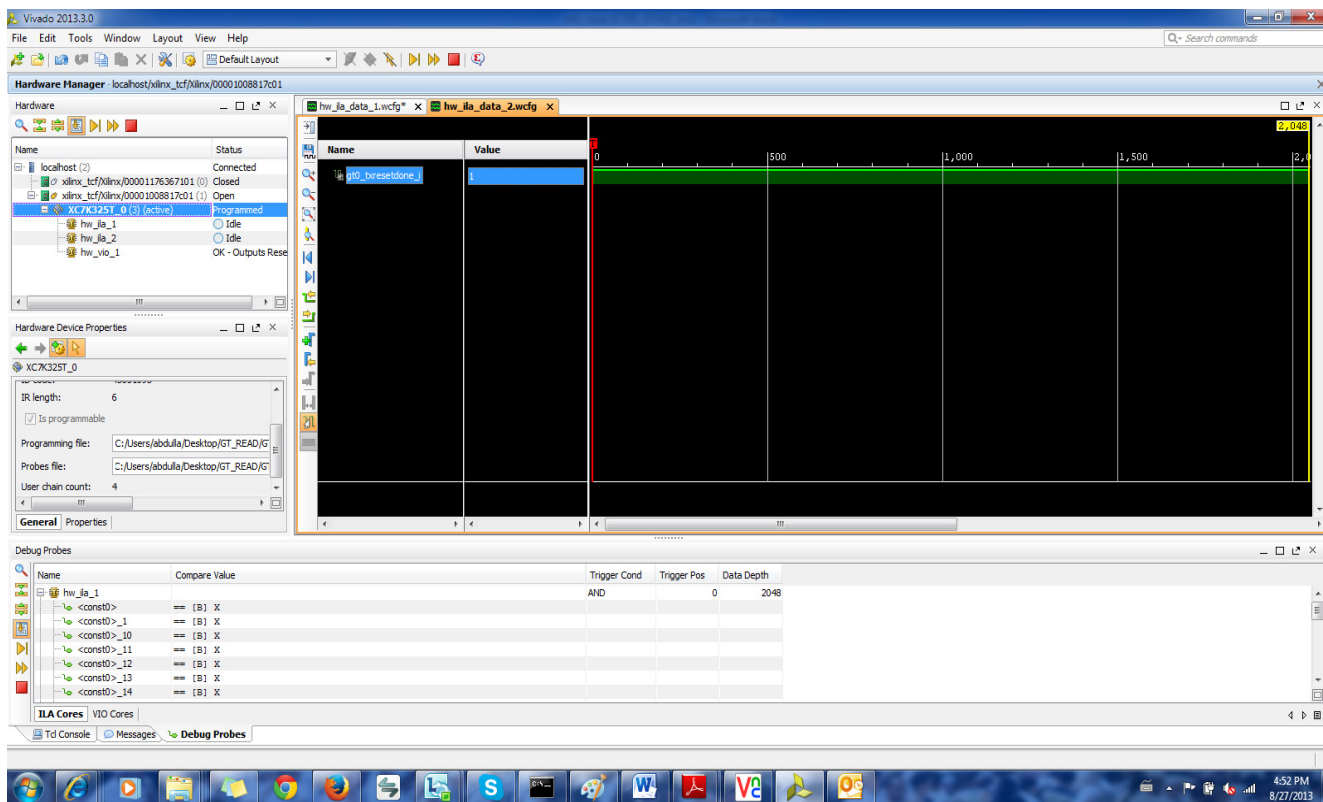
図 22 : RX ILA 波形画面

注記 : 図 22 に示す RX ILA 波形画面の信号が次の値となることを確認します。

gt0_rxdata_i[63:0] = ランダムな 64 ビット値
 gt0_rxreset_done_ila = 1
 track_data_out_i = 1

注記：同様に、図 23 に示す TX ILA 波形画面の信号が次の値となることを確認します。

gt0_txresetdone_ila = 1



x1200_24_021114

図 23 : TX ILA 波形画面

4K リセット テスト

次の手順に従って Gtwizard 4K リセット テストを実行します。

1. reset_fast.tcl ファイルをプロジェクト エリアにコピーします。
2. KC705 ボードのビット ファイル (.bit) およびプローブ ファイル (.ltx) を読み込みます (すでに読み込まれている場合はこの手順は無視する)。
3. Vivado Tcl コンソールでソース <dirpath>/reset_fast.tcl を実行します。
4. 次の 4K RESET TEST PASS OR FAIL の結果が Tcl コンソールに表示されます。

TCL CONSOLE OUTPUT :

```

..
Status @Reset Assertion::TRACK_DATA_OUT ::0
Status @Reset Deassertion::TRACK_DATA_OUT ::1
Info:Iteration :4094 :::LANES ON DUT2 are UP
Reset Test Iteration :4095 ::::::::::::::
Status @Reset Assertion::TRACK_DATA_OUT ::0
Status @Reset Deassertion::TRACK_DATA_OUT ::1
Info:Iteration :4095 :::LANES ON DUT2 are UP
Reset Test Iteration :4096 ::::::::::::::
Status @Reset Assertion::TRACK_DATA_OUT ::0
Status @Reset Deassertion::TRACK_DATA_OUT ::1

```

```
Info:Iteration :4096 ::::LANES ON DUT2 are UP
Iteration :4096 ::TEST PASSED :::::
TCL_OK
```

リファレンス デザイン

この資料に記載の手順に従ってデザインを作成できます。このデザインは、ハードウェアで完全に検証およびテストされています。

表 1: リファレンス デザインの詳細

| パラメーター | 説明 |
|---|---|
| 全般 | |
| 開発者 | Dinesh Kumar, Ramachandra Thupalli |
| ターゲット デバイス (ステッピング レベル、ES、プロダクション、スピード グレード) | Kintex7 FPGA、KC705 評価ボード |
| ソース コードの提供 | あり |
| ソース コードの形式 | VHDL/Verilog (一部は暗号化済み) |
| 既存のザイリンクス アプリケーション ノート/リファレンス デザイン、CORE Generator ツール、サードパーティからデザインへのコード/IP の使用 | 提供されているリファレンス デザインでは、Vivado Design Suite 2013.3 で生成されたコアを使用 |
| インプリメンテーション | |
| 使用した合成ツール/バージョン | Vivado Design Suite 2013.3 |
| 使用したインプリメンテーション ツール/バージョン | Vivado Design Suite 2013.3 |
| スタティック タイミング解析の実施 | あり (PAR/TRCE のタイミングにパス) |
| ハードウェア検証 | |
| ハードウェア検証の実施 | あり |
| 使用したハードウェア プラットフォーム | KC705 評価ボード |

参考資料

この文書の参考資料は次のとおりです。

- 『Kintex-7 FPGA 用 KC705 評価ボード ユーザー ガイド』([UG810](#))
- 『エンベデッド システム ツール リファレンス マニュアル: EDK 14.4』([UG111](#))
- 『Vivado Design Suite ユーザー ガイド : リリース ノート、インストールおよびライセンス』([UG973](#))

改訂履歴

次の表に、この文書の改訂履歴を示します。

| 日付 | バージョン | 内容 |
|-----------------|-------|----|
| 2014 年 3 月 10 日 | 1.0 | 初版 |

Notice of Disclaimer

The information disclosed to you hereunder (the “Materials”) is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available “AS IS” and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at <http://www.xilinx.com/warranty.htm>; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: <http://www.xilinx.com/warranty.htm#critapps>.

本資料は英語版 (v1.0) 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。