ChipScope Pro ILA コアと Project Navigator を使用した FPGA アプリケーションのデバッグ

UG750 (v12.3) 2010 年 11 月 5 日

XILINX®

本資料は英語版 (v 12.3) 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。

目次

XILINX®

第 1 章

はじめに

このチュートリアルでは、Integrated Logic Analyzer (ILA) コアを Project Navigator デザイン環境内に挿入して FPGA デザインをデバッグする方法について説明します。ChipScope™ Pro Analyzer の機能を利用し、デザインの問題をデバッグし、潜在的な原因を見つけることができるので、素早く問題を識別することができます。

ChipScope および Xilinx® ISE® Project Navigator 間の統合フローについては、RTL デザイン例を使用して説明します。このチュートリアルを実行するには、ISE ツール フローの基本的な知識が必要となります。

目標

このチュートリアルを終了すると、次ができるようになります。

- Project Navigator と ILA コアを使用した ChipScope、および ChipScope Analyzer を使用してデザインをデバッグします。
- ISE プロジェクトを作成し、ILA コアを使用してデザインをプローブし、Project Navigator でデザインをインプリメントする方法について理解します。
- ChipScope Analyzer を使用してデザインをデバッグし、Project Navigator デザイン環境と SP601 プラットフォームを使用してデザインを繰り返し実行します。

チュートリアルに必要な要件

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。

- ザイリンクス ISE Design Suite 12.3 (Logic、DSP、Embedded、または System Edition)
- SP601 プラットフォーム
- SP601 プラットフォームに含まれる JTAG ケーブル

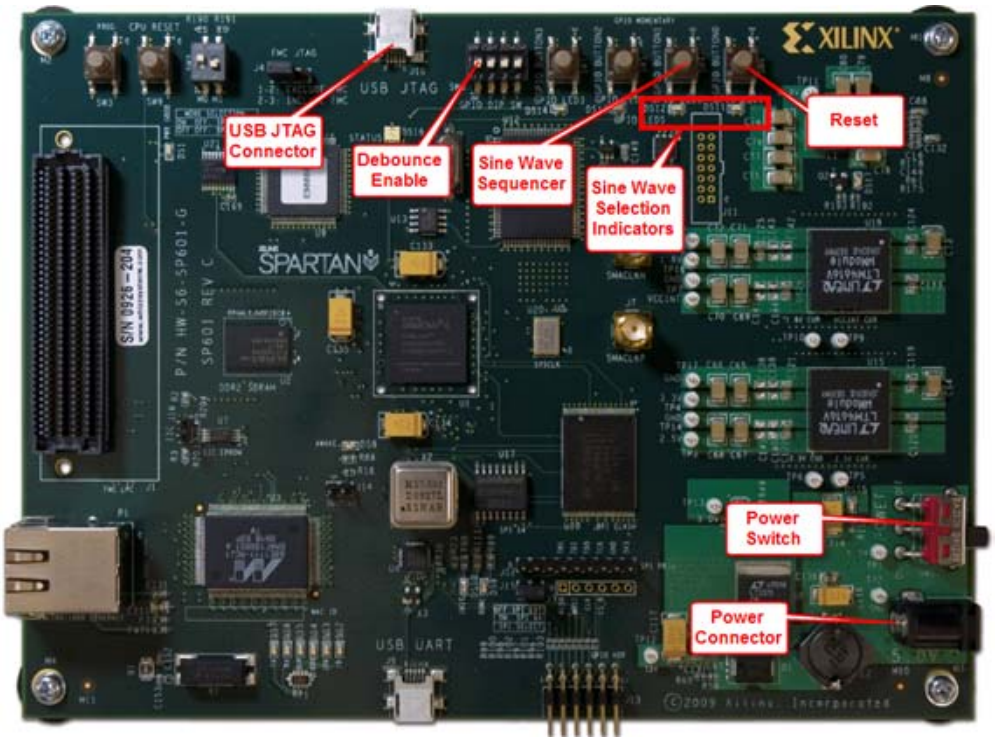

図 1-1 : SP-601 プラットフォーム

デザインの説明

RTL デザイン例の最上位レベルのブロック図を次に示します。デザインには、シンプルな制御ステート マシン、複数のサイン波ジェネレーター、共通のプッシュ ボタン (GPIO_BUTTON)、DIP スイッチ (GPIO_SWITCH)、LED ディスプレイ (GPIO_LED) が含まれます。

プッシュ ボタン スイッチ

プッシュ ボタン スイッチは、デバウンス回路や制御ステート マシン回路への入力として使用されます。スイッチを押すと、High から Low への遷移パルスが生成されます。生成された出力パルスは、それぞれステート マシンへの入力として使用されます。

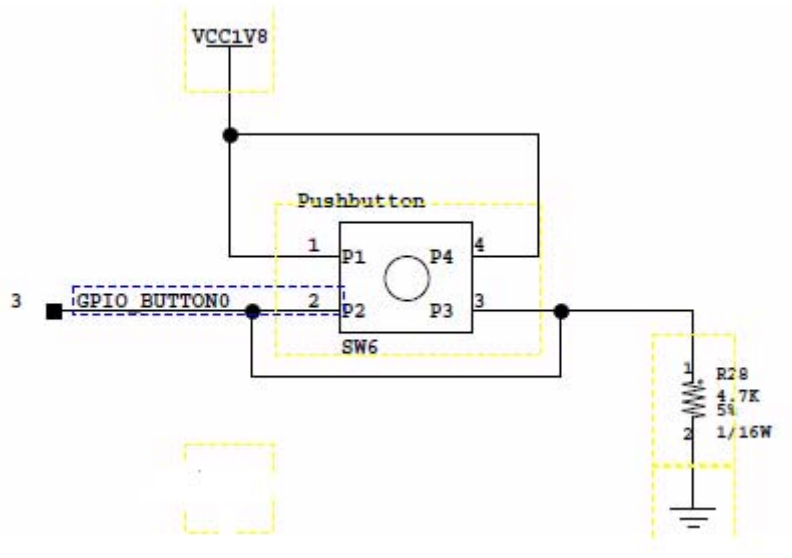

図 1-2 : DIP スイッチ - デバウンス回路をイネーブルまたはディスエーブルにするスイッチ

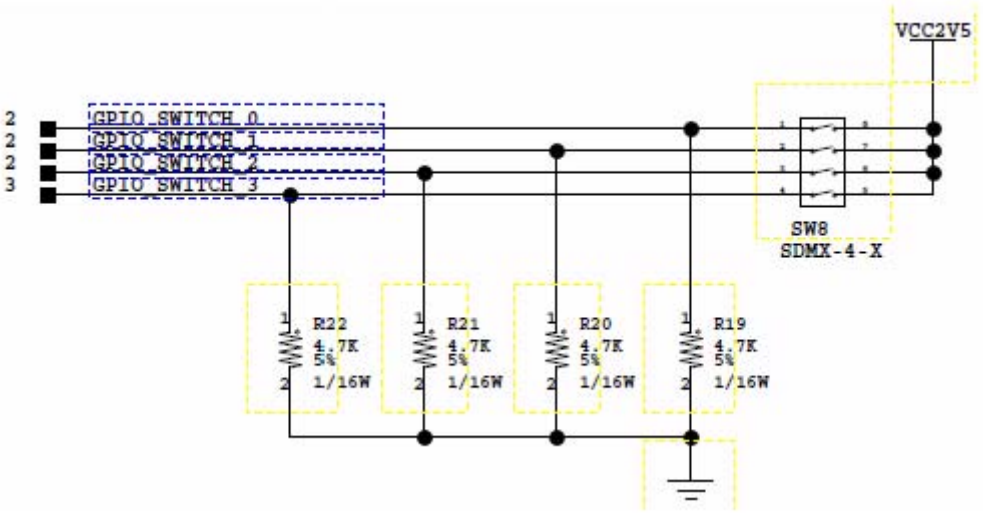

図 1-3 : デバウンス 回路

デバウンス 回路

イネーブルになると、このデザイン例の場合、デバウンス回路によりクリーンなパルスまたは High から Low への遷移が提供され、ボタンを押してから放すと、一連のスパイクやグリッチが削除されます。

制御ステート マシン

制御ステーと マシンは、2 つのプッシュ ボタン スイッチからの入力パルスをキャプチャーしてデコードするために使用され、00、01、10、11 (0 ～ 3) 間のサイン波選択回路およびインジケーター回路を提供します。

LED ディスプレイ

GPIO_LED_0 および GPIO_LED_0 はステート マシンの出力からの選択ステータスを表示し、それぞれ高、中、低周波の異なるサイン波周波数を表します。

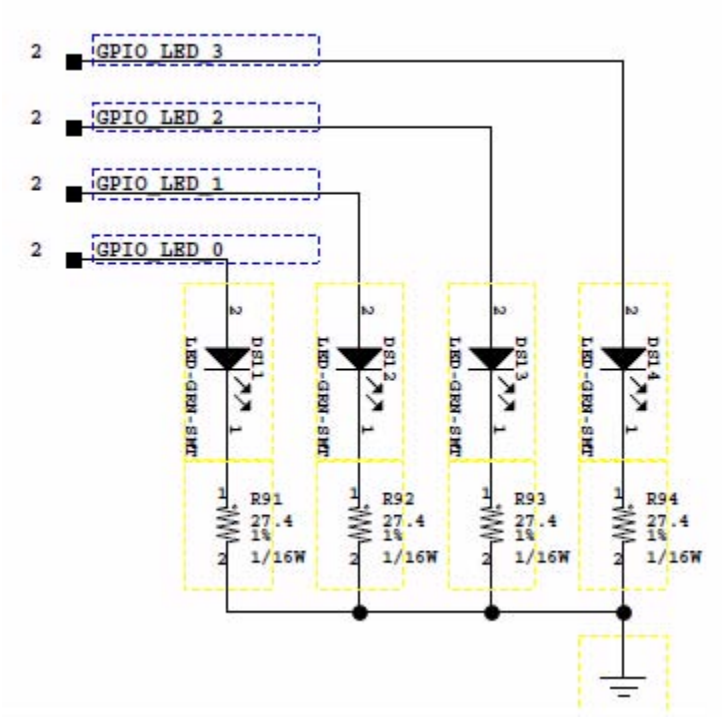

図 1-4 : LED ディスプレイ

チュートリアルに含まれるファイル

このチュートリアルには、次のファイルおよびフォルダが含まれています。

- debounce.vhdl - デバウンス回路
- fsm.vhdl - 制御ステート マシン
- sinegen_demo - サイン波ジェネレーターのラッパー
- sinewave_demo - 最上位レベルのラッパー デザイン
- sine_high.xco、sine_mid.xco、sine_low.xco - ザイリンクス CORE Generator ファイル
- sinegen_demo_sp601.ucf - UCF 制約ファイル

メモ：このチュートリアルでは、SP605 と ML605 の 2 つのザイリンクス プラットフォームもサポートされています。このチュートリアルの対象を SP605 か ML605 に変更する場合は、次の表のピン配置情報を参照してください。

表 1-1：ザイリンクス プラットフォームのピン配置情報

	ピン配置の位置			ファンクション
	SP601	**SP605**	**ML605**	
CLK_N	K16	K22	H9	クロック
CLK_P	K15	K21	J9	クロック
GPIO_BUTTONS0	P4	F3	A19	リセット
GOIP_BUTTONS1	F6	G6	G26	シーケンサー
GPIO_SWITCH	D14	C18	D22	デバウンス回路セレクター
GPIO_LED_0	E13	D17	AC22	Selector[0]
GPIO_LED_0	C14	AB4	AC24	Selector[1]

手順

このチュートリアルでは、次の 3 つのタスクを実行します。

1. 「Project Navigator でのプロジェクトの作成とインプリメント」
2. 「デザインへの ChipScope ILA コアの追加」
3. 「ChipScope Pro Analyzer を使用したデザインのデバッグ」

XILINX®

第 2 章

Project Navigator でのプロジェクトの作成とインプリメント

RTL デザインの作成およびインプリメント

この手順では、Project Navigator で RTL デザインを素早くインプリメントする方法とProject Navigator で SP601 プラットフォームをターゲットにした ISE® プロジェクトを作成する方法について学びます。

1. 提供されているソース ファイルを `C:\ChipScope_ProjNav\` で解凍します。
2. Project Navigator を起動し、[File] → [New Project] をクリックして新しいプロジェクトを作成します。名前、ディレクトリ、プロジェクト タイプを指定し、[Next] をクリックします。

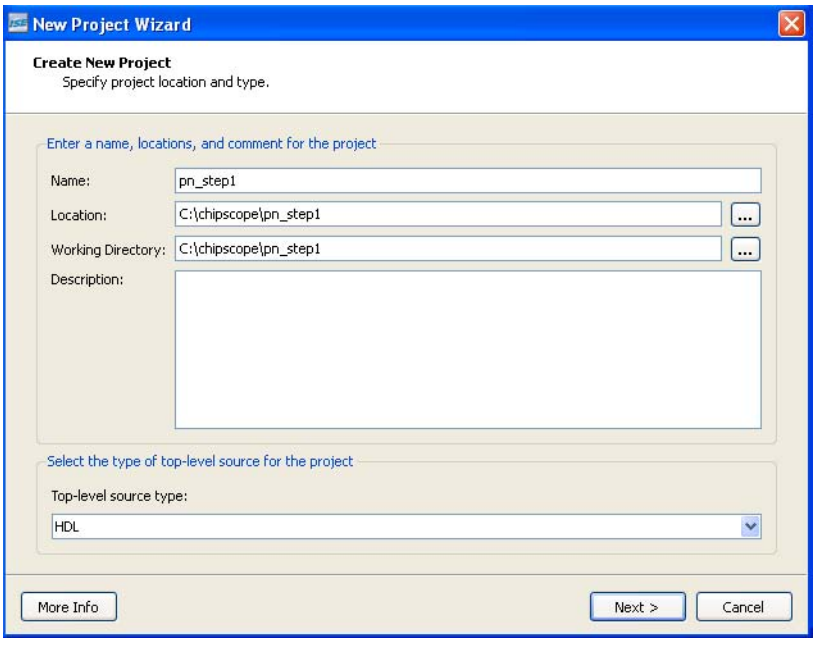

図 2-1 : [Create New Project] ページ

3. デバイスとプロジェクト プロパティを図のように指定し、[Next] をクリックします。

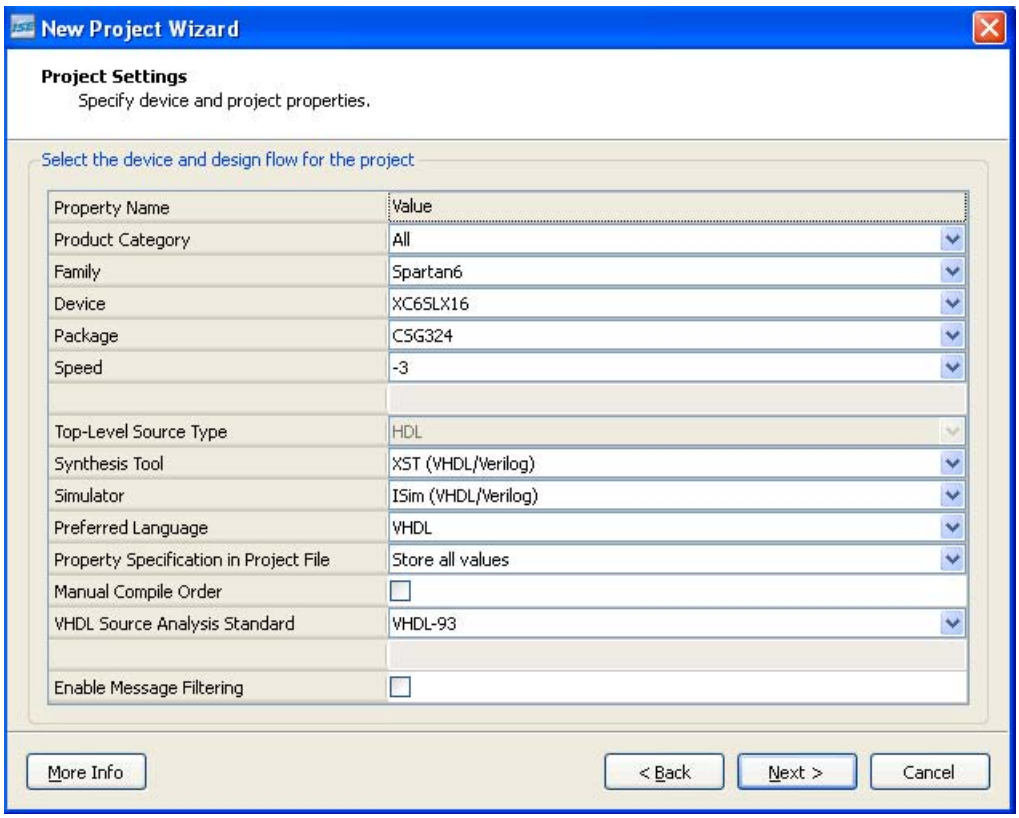

図 2-2 : [Project Settings] ページ

4. サマリを確認したら、[Finish] をクリックします。
5. pn_step1 プロジェクトを右クリックして [Add Source] をクリックし、VHDL ソース ファイルを新規プロジェクトに追加します。

6. ソース ファイルのある src ディレクトリを指定し、すべての VHDL ファイルを選択して開きます。

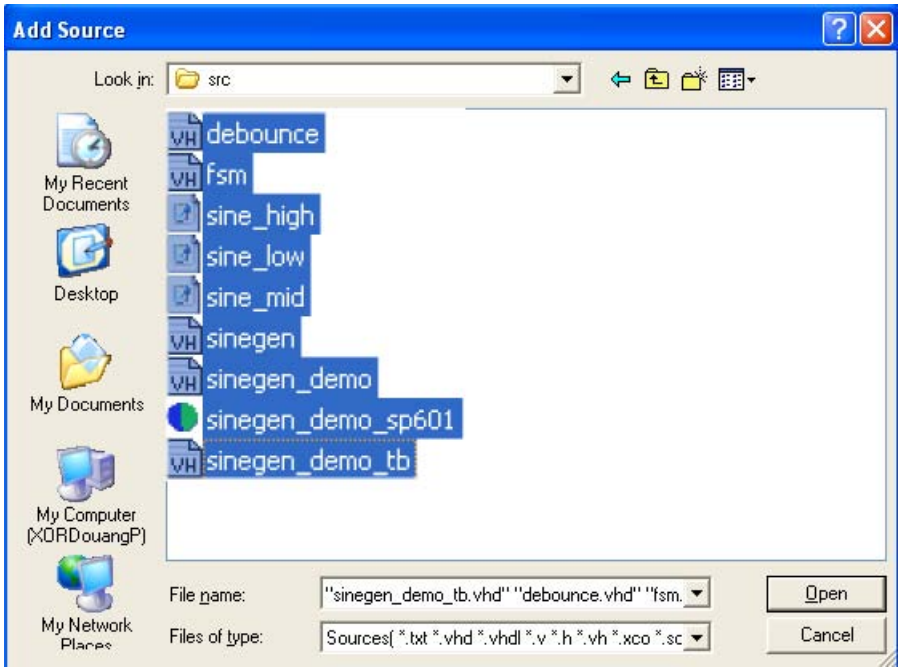

図 2-3 : ソース ファイル

7. [Adding Source Files] ウィンドウにリストされるファイルを確認し、[OK] をクリックします。

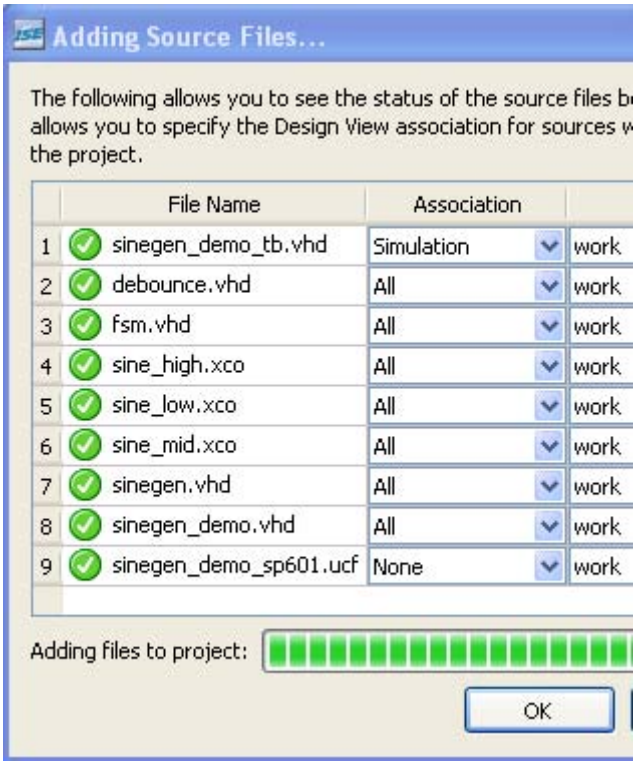

図 2-4 : [Adding Source Files] ウィンドウ

8. sinegen_demo を右クリックし、[New Source] をクリックして新しい定義と接続 (.cdc) ファイルをプロジェクトに追加します。

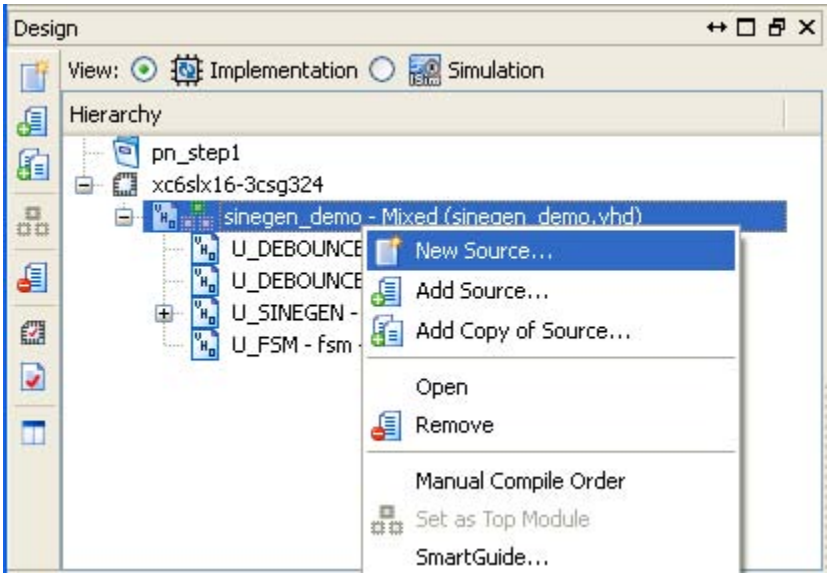

図 2-5 : 新規ソースの選択

9. [Select Source Type] ウィンドウで [ChipScope Defination and Connection File] を選択し、ファイル名に pn_step1.cdc と入力して [Finish] をクリックします。
10. サマリ情報を確認して、[Finish] をクリックします。

    次に、3 つのサイン波ジェネレーターのコアすべてを生成します。

11. [Design] ウィンドウで U_SINEGEN の下の 3 つすべての .xco ファイルを選択し、[Regenerate All Cores] プロセスをダブルクリックします。

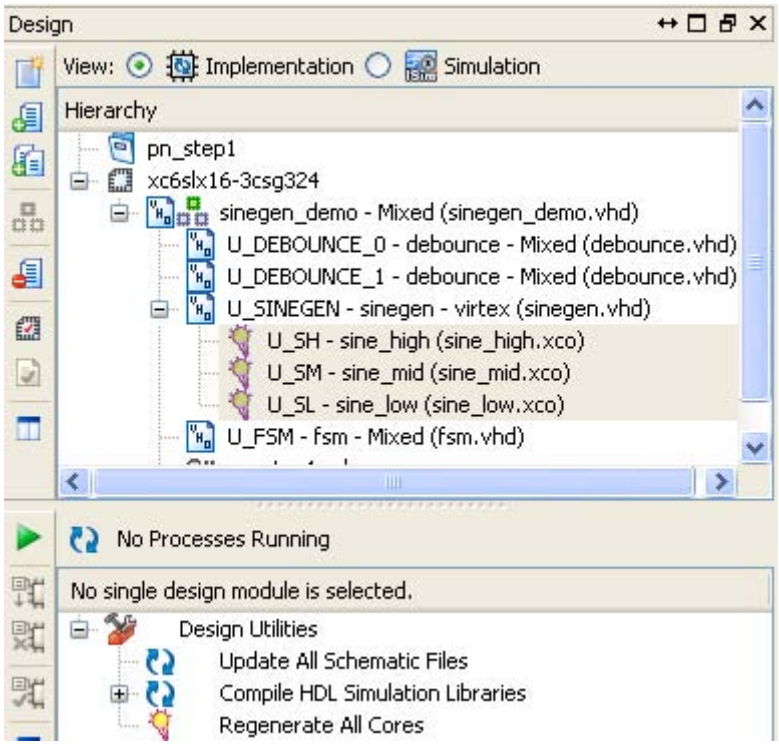

図 2-6 : [Design] ウィンドウのサイン波ジェネレーターのファイル

12. デザインを合成する前に、[Keep Hierarchy] オプションを [Soft] に設定し、デザイン階層が保持され、XST で階層最適化が実行されないようにしておきます。これを設定するには、[Synthesize] プロセスを右クリックし、[-Keep_hierarchy] オプションを [Soft] にします。
13. [Apply] をクリックします。

    これで、合成準備が完了です。
14. [Synthesize] プロセスをダブルクリックします。
15. [File] → [Copy Project] をクリックし、プロジェクト名を pn_step2 にして保存します
16. プロジェクト名とディレクトリを入力したら、[OK] をクリックします。

この段階までで、Project Navigator を使用して ISE プロジェクトを作成し、デザインを合成しました。

質問

1. 手順 1 で実行したことを簡単に説明してください。

   _____
2. このチュートリアルで使用した主な回路は何ですか。

   _____
3. ほかのザイリンクス ボードをターゲットにする場合、どのソース ファイルを修正する必要がありますか。

   _____

デザインへの ChipScope ILA コアの追加

ChipScope ILA コアの追加

この手順では、Project Navigator と ChipScope Pro Core Inserter ツール間の統合フローを使用して、デザインに ChipScope™ ILA コアを追加します。以前はユーザーが ILA コアを手動で RTL デザインに挿入する必要がありました。この場合、デザインのソース ファイルを変更する必要があったため、手間がかかる上にミスをする可能性がありました。

今回からは、元の RTL ソース ファイルを変更せずに、コアを追加し、信号を接続詞、デザインをプローブできます。

1. [Hierarchy] ウィンドウで `pn_step1.cdc` ファイルをダブルクリックし、ChipScope Pro Core Inserter ツールを起動します。

   最初のウィンドウには、デバイス　オプションが表示されます。ターゲット　デバイスは既に Project Navigator で設定したので、デバイス オプションが自動的に設定されています。これにより、ChipScope Pro Core Inserter と Project Navigator 間で異なるデバイスを選ぶミスを避けることができます。

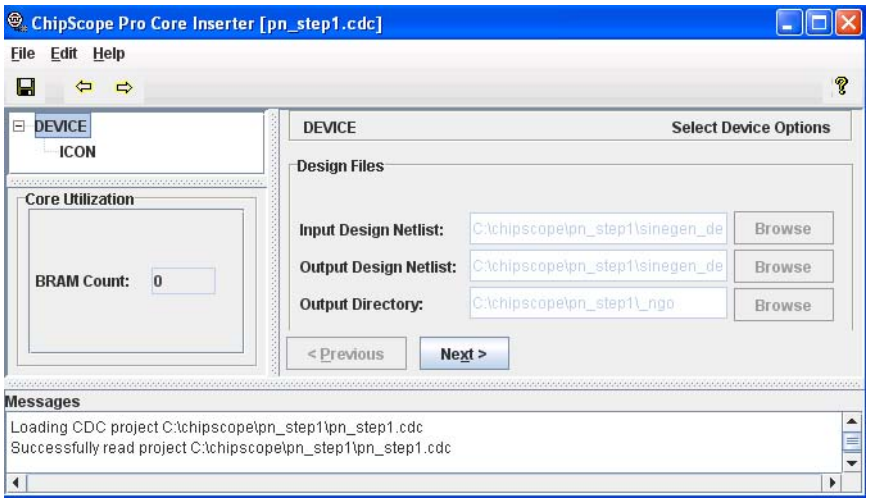

図 3-1 : ChipScope Pro Core Inserter

2. [Next] をクリックします。ウィンドウに ICON オプションが表示されます。

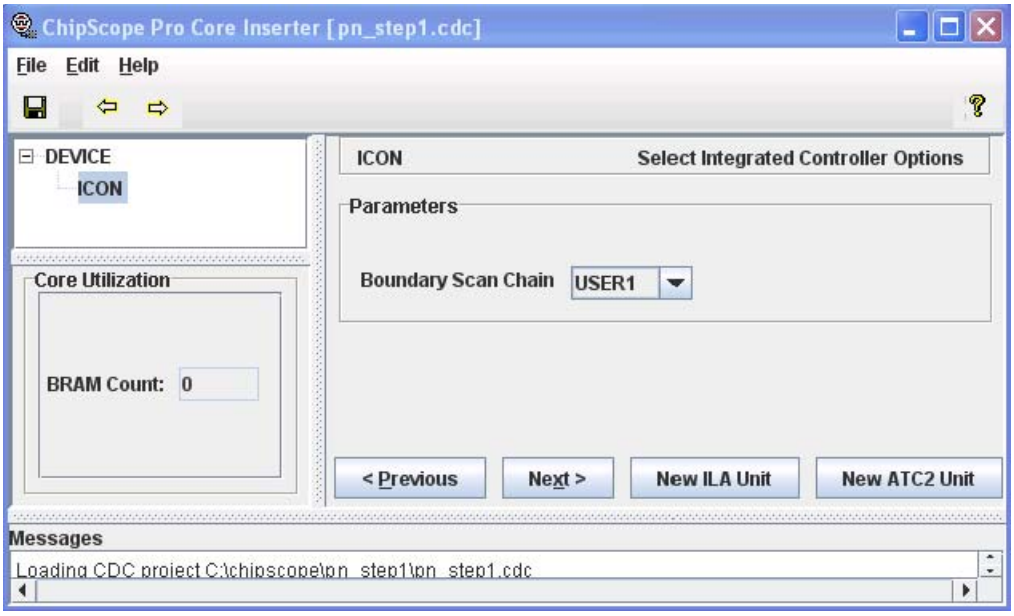

図 3-2 : ICON オプション

3. [Next] をクリックして ILA コアを追加します。
4. [ILA Options] ウィンドウの [Trigger Parameters] タブでトリガー パラメータを設定します。

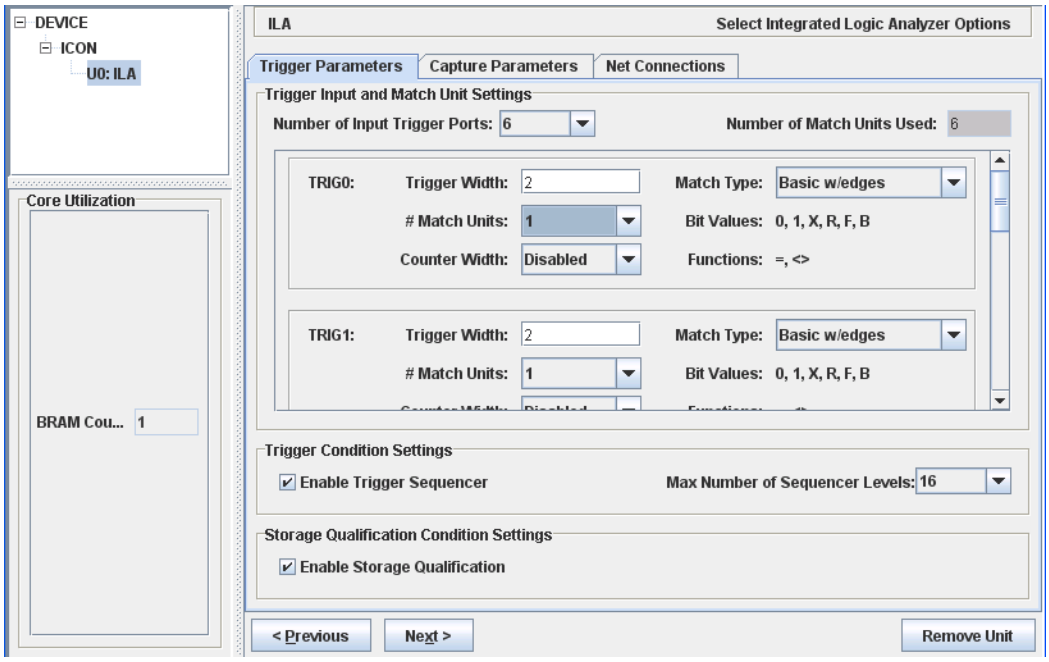

図 3-3 : [ILA Options] ウィンドウの [Trigger Parameters] タブ

この例では、次のようにトリガー パラメータを設定します。

表 3-1 : トリガー パラメーター

トリガー名	トリガー幅	マッチ タイプ
TRIG0	2	Basic w/ edge
TRIG1	2	Basic w/ edge
TRIG2	2	Basic w/ edge
TRIG3	20	Basic
TRIG4	2	Basic w/ edge
TRIG5	3	Basic w/ edge

5. [Capture Parameters] タブをクリックし、設定したトリガー パラメータを確認します。

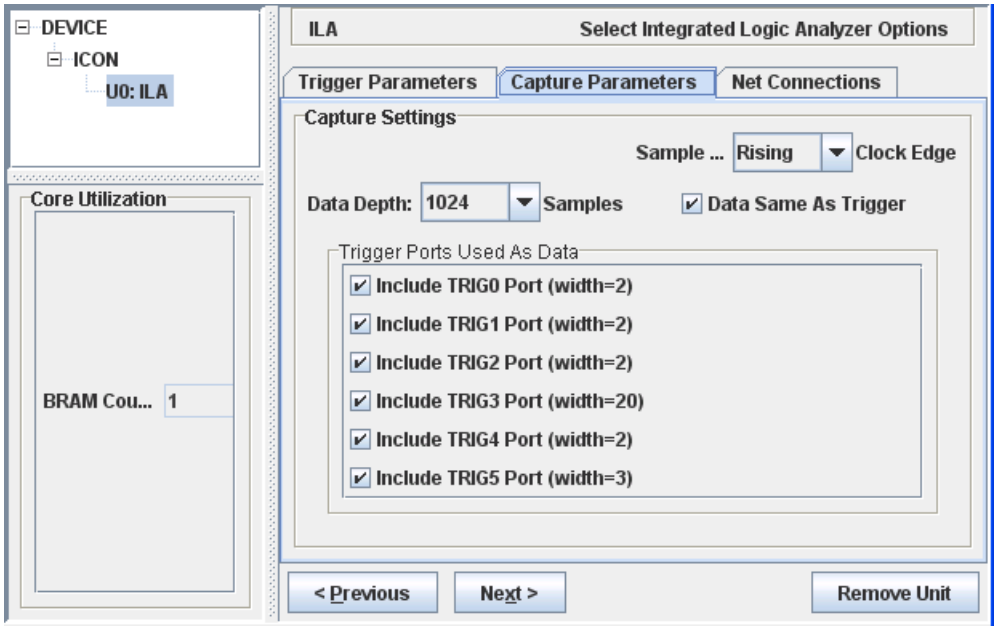

図 3-4 : [Capture Parameters] タブ

6. 各ポートをデバッグ ネットに接続します。次の手順に従います。
   a. [Net Connections] タブをクリックします。
   b. [Modify Connections] をクリックします。[Select Net] ウィンドウが表示されます。
   c. [Select Net] ウィンドウで sinegen_demo 階層から CLK_BUFG ネットを検索します。

      CLK_BUFG ネットを検索するには、[Pattern] フィールドに clk_bufg と入力し、[Filter] をクリックします。

d. 検索結果から `clk_BUFG` を選択し、[Make Connections] をクリックします。

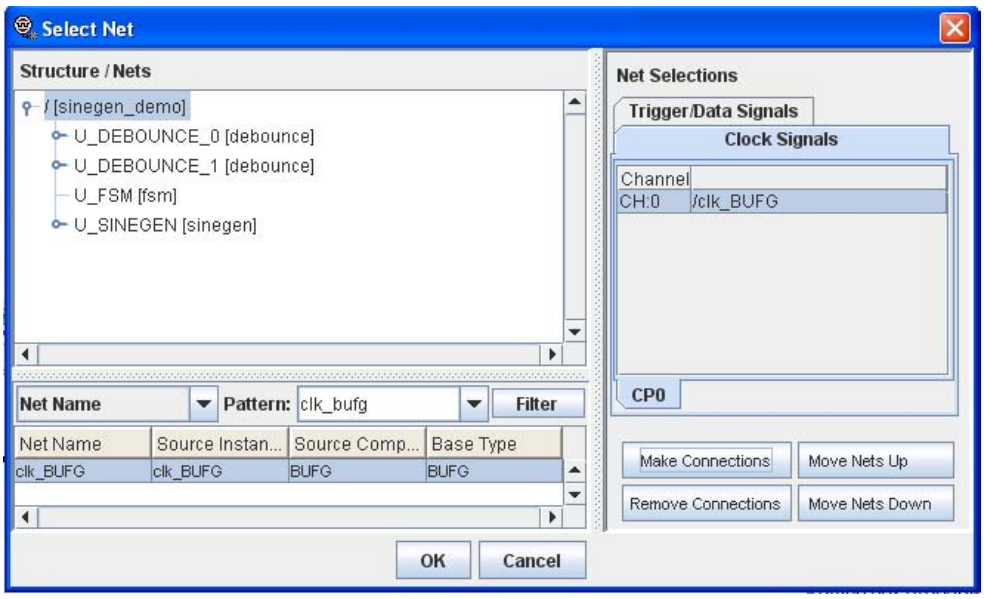

図 3-5 : [Select Net] ウィンドウ

7. 手順 6 を繰り返し、残りのトリガー ポートも接続します。
   - TRIG0 : U_SINEGEN 階層で *sel* を検索
   - TRIG1 : sinegen_demo 階層で *GPIO_BUTTONS_re* を検索
   - TRIG2 : sinegen_demo 階層で *GPIO_BUTTONS_dly* を検索
   - TRIG3 : sinegen_demo 階層で *SINE* を検索
   - TRIG4 : sinegen_demo 階層で *GPIO_BUTTONS_db<0>* を検索
   - TRIG4 : sinegen_demo 階層で *GPIO_BUTTONS_db<1>* * を検索
   - TRIG5 : sinegen_demo 階層で *GPIO_BUTTONS_0_IBUF* を検索
   - TRIG3 : sinegen_demo 階層で *GPIO_BUTTONS_1_IBUF* を検索
   - TRIG3 : sinegen_demo 階層で *GPIO_SWITCH_IBUF* を検索

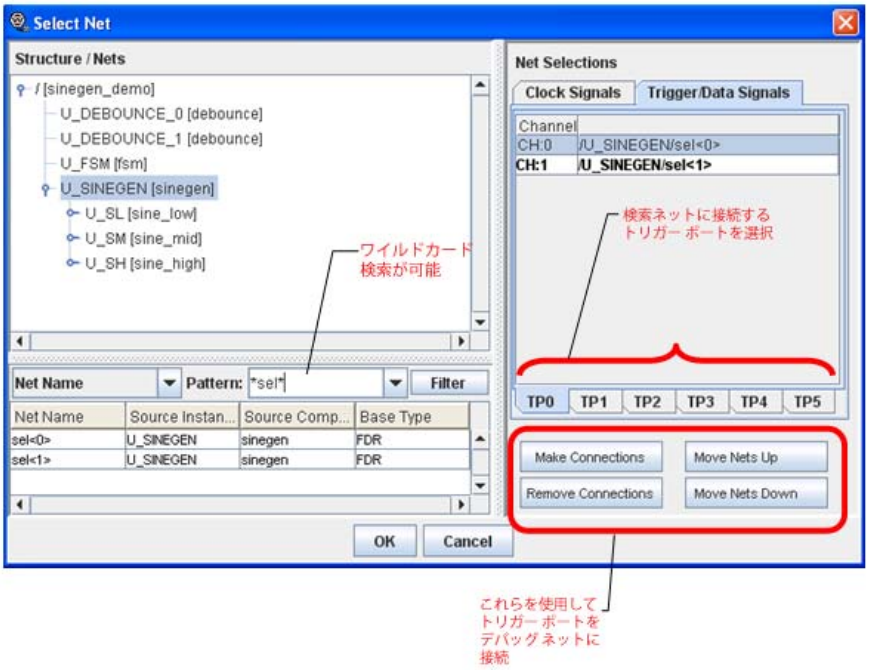

図 3-6 : トリガー /データ信号

8. 終了したら、すべてのポートが赤色から黒色に変わり、すべてのクロックおよびトリガー ポートをデバッグ ネットに接続したことを示します。

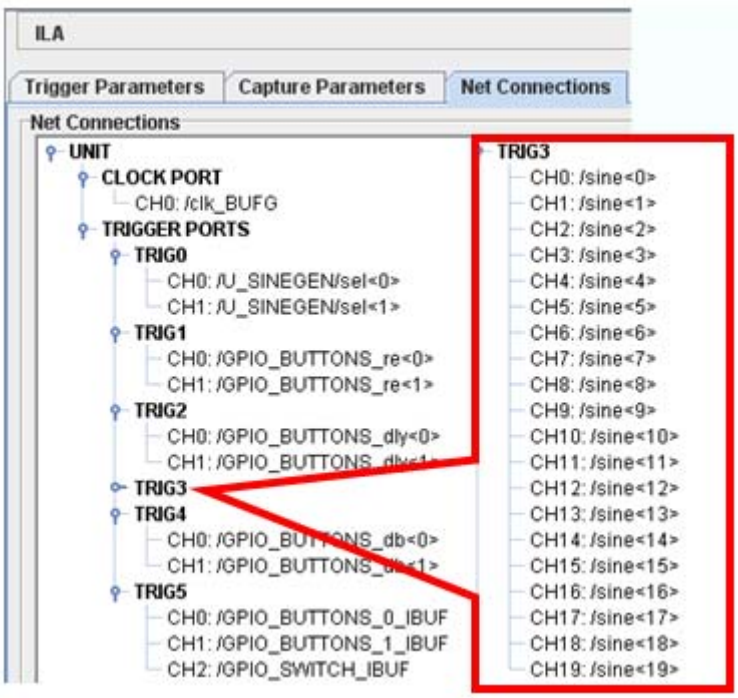

図 3-7 : ネット接続の終了

9. `pn_step1.cdc` ファイルを保存して閉じます。

   ChipScope Pro Analyzer ツールを使用してビットストリームをデバイスにダウンロードする前に、ビットストリーム生成オプションが正しく設定されているかどうか確認します。

10. [Generate Programming File] プロセスを右クリックし、[Properties] をクリックします。
11. [Startup Options] カテゴリで `-g StartUpClk` オプションを [JTAG Clock] に設定します。

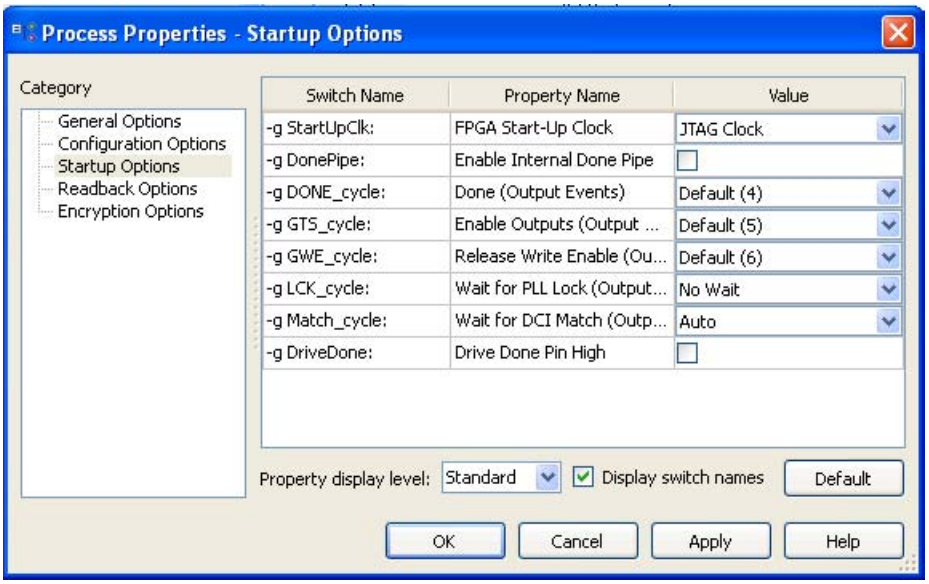

図 3-8 : [Startup Options] カテゴリ

これで、プログラム ファイルの生成を開始できます。

12. [Analyze Design Using ChipScope] プロセスをダブルクリックします。

    プロセスが終了すると、ChipScope Pro Analyzer ツールが起動されます。

質問

4. デバッグ プローブを HDL デザイン ファイルに追加するのではなく、合成後のネットリストに挿入する主な利点は何ですか。

   _____

   ここまでで ChipScope ILA コアを挿入し終わったので、ChipScope Pro Analyzer を使用してデザインをデバッグできるようになりました。

XILINX®

第 4 章

ChipScope Pro Analyzer を使用したデザインのデバッグ

デザインのデバッグ

ここでは、Xilinx® ChipScope™ Pro Analyzer を使用してデザインをデバッグし、エラーを検出して修正してから繰り返し実行する方法について説明します。また、デザインから一部のデータをトリガーおよびキャプチャーする方法についても説明します。

ChipScope Pro Analyzer を使用して、サイン波ジェネレーターが正しく動作しているかどうかを確認します。ここでの主な目的は、次の 2 つです。

- ☐ すべてのサイン波選択が正しいかどうか確認
- ☐ 選択ロジックが正しく動作しているかどうか確認

次の手順に従います。

1. JTAG チェーンを USB ケーブルにし、通信パラメータを設定します。
   a. [JTAG Chain] → [Xilinx Platform USB Cable] をクリックします。
   b. [ChipScope Pro Analyzer] ダイアログ ボックスが開くので、速度とポートのパラメータを設定します。

      このチュートリアルでは、[Speed] を 3 MHz に [Port] を USB21 にします。
   c. [OK] をクリックします。
   d. [ChipScope Pro Analyzer] ダイアログ ボックスで、デバイスの詳細を確認し、[OK] をクリックします。

2. JTAG チェーンへの接続を確認します。

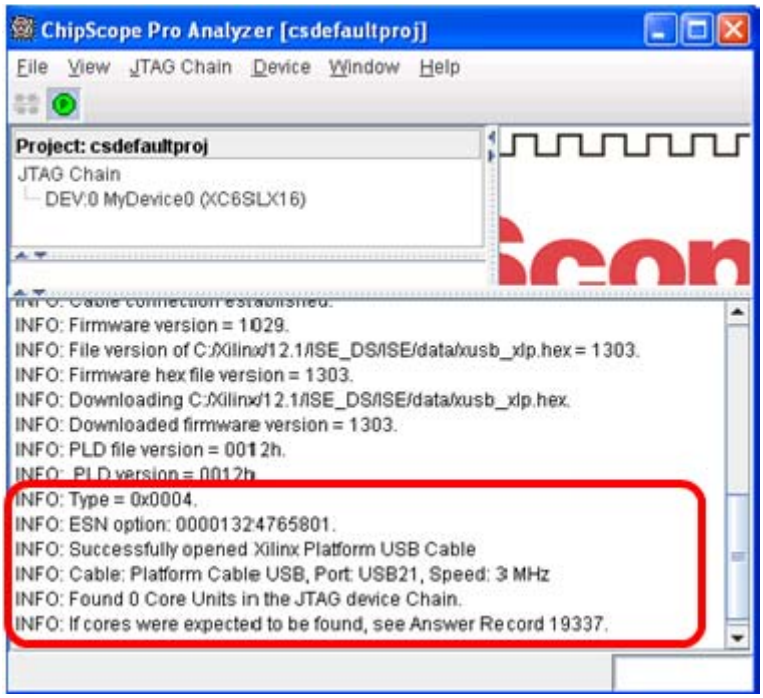

図 4-1 : JTAG チェーン接続の詳細

3. [Project] リストでデバイス名を右クリックし、[Configure] をクリックしてデバイスをコンフィギュレーションします。
4. [Configuration] ウィンドウで、Project Navigator からのデフォルトの BIT と CDC ファイルを選択します。

5. デバイス コンフィギュレーションと ILA コアを確認します。

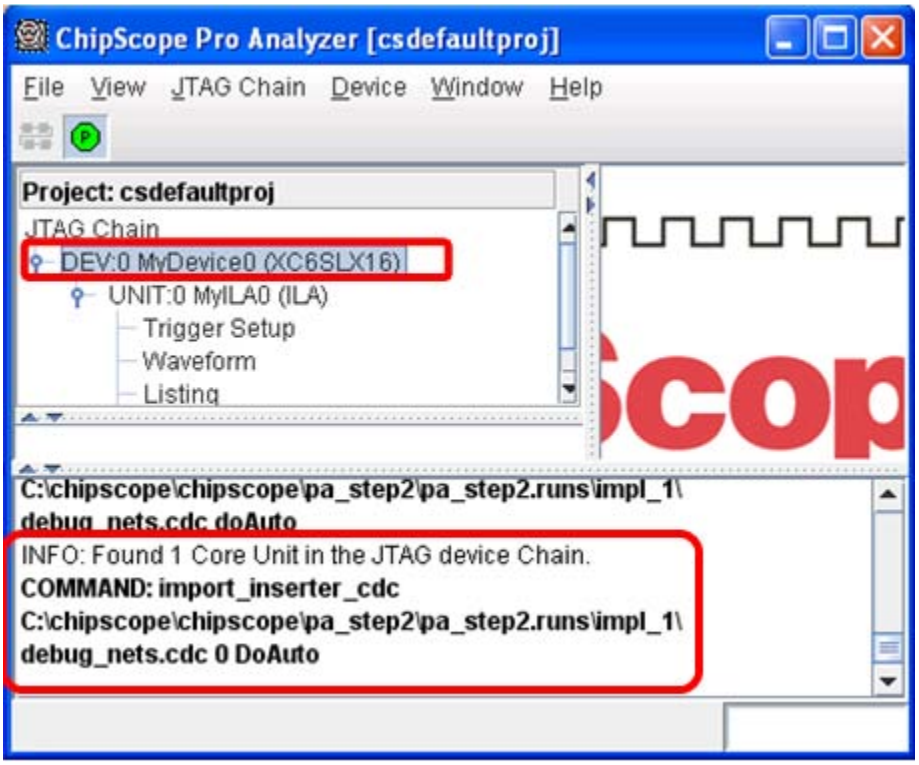

図 4-2 : デバイス コンフィギュレーションの詳細

6. [Trigger Setup] および [Waveform] をそれぞれダブルクリックし、[Trigger Setup] および [Waveform] ウィンドウを開きます。

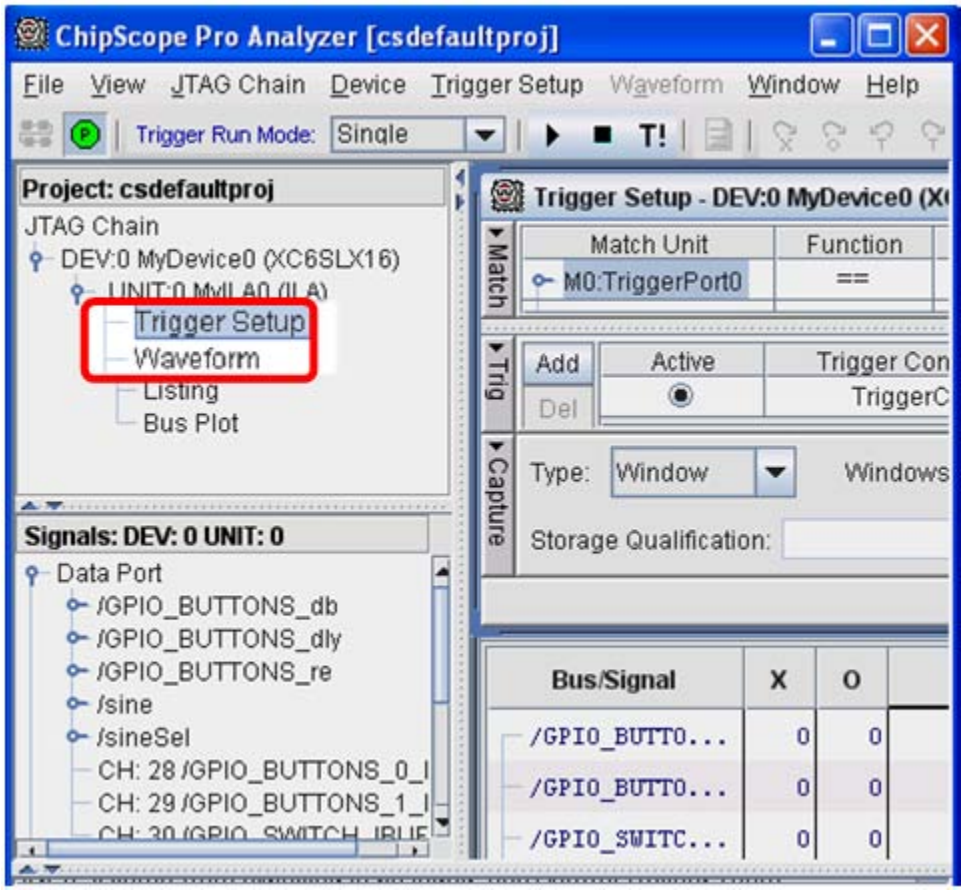

図 4-3 : [Trigger Setup] および [Waveform] の位置

7. [Trigger Setup] → [Trigger Immediate] をクリックします。
8. サイン波で動きがあることを確認します。

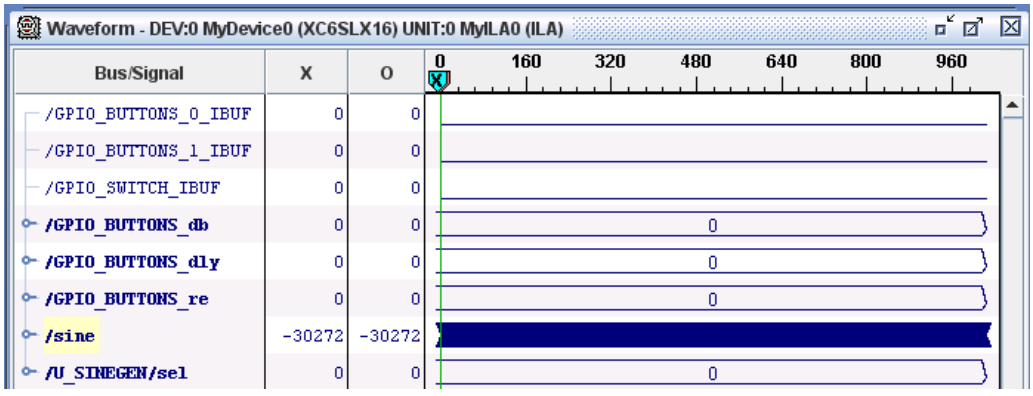

図 4-4 : サイン波の動作

9. [Bus Plot] をダブルクリックし、[Bus Plot] ウィンドウを開きます。

10. [Bus Plot] ウィンドウで [/sine] チェック ボックスをオンにしてサイン波を表示します。

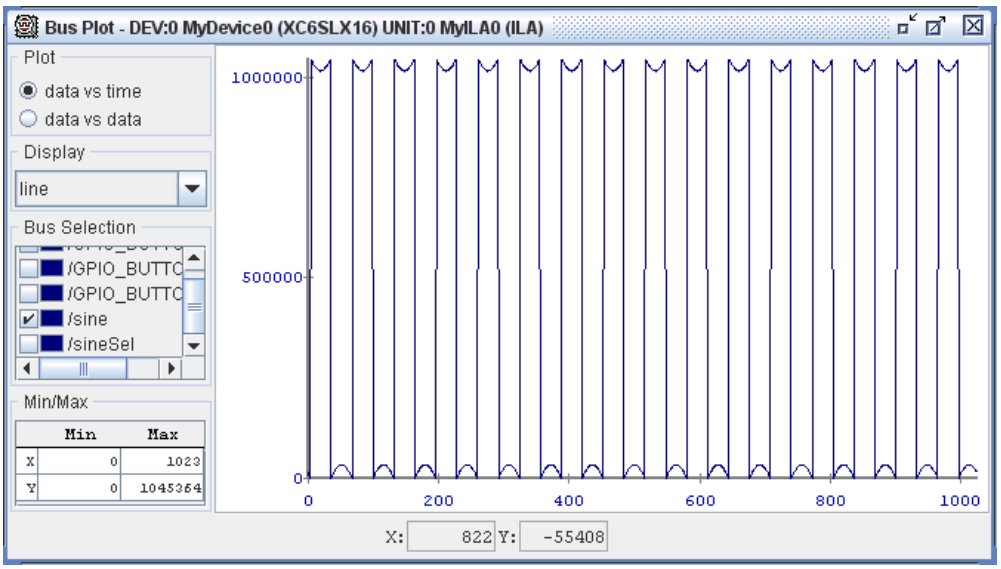

図 4-5 : サイン波の表示

この波形はサイン波のようには見えません。これは、基数設定を 16 進数から符号付き 10 進数へ変更すると正しく表示されます。

11. [/sine] を右クリックし、[Bus Radix] をクリックします。[Signed Decimal] チェック ボックスをオンにします。

12. [Trigger Immediately] ボタン T! をクリックし、高周波数のサイン波のバス プロットを表示します。

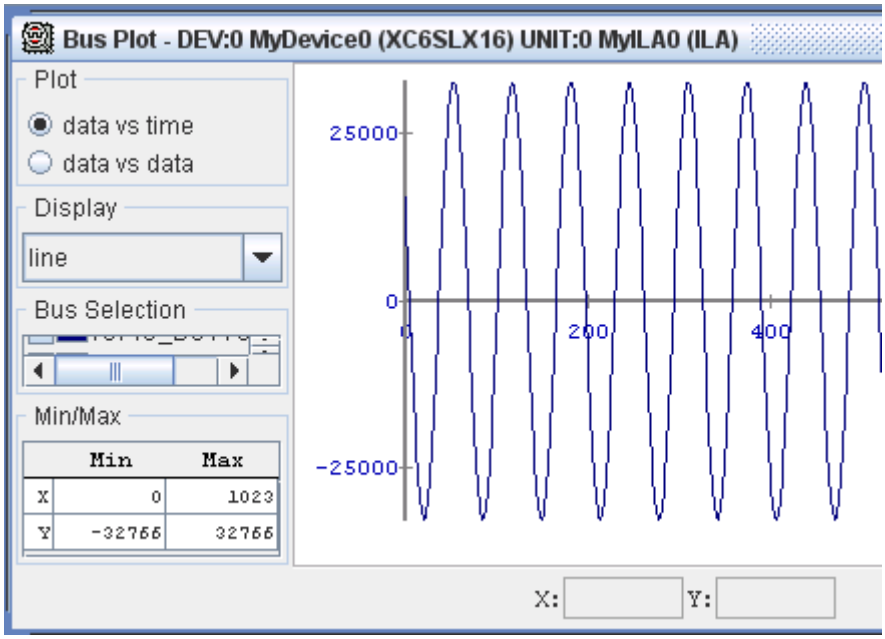

図 4-6 : 高周波数のサイン波のバス プロット

13. ボードのサイン波選択インジケーターの LED が 「off,on, 01」と表示されるまで、ボードで [Sine Wave Sequencer] ボタン (図 1-1) を押し続けます。

    メモ : シーケンサーは正しく動作していません。予測される動作では、ボタンを押すごとにカウントするシンプルな 2 ビット カウンター (00, 01, 10, 11...) になるはずです。この問題の原因については、チュートリアルの後半でデバッグします。

14. [Trigger Immediately] ボタンをもう 1 度クリックし、中周波数のサイン波のバス プロットを表示します。

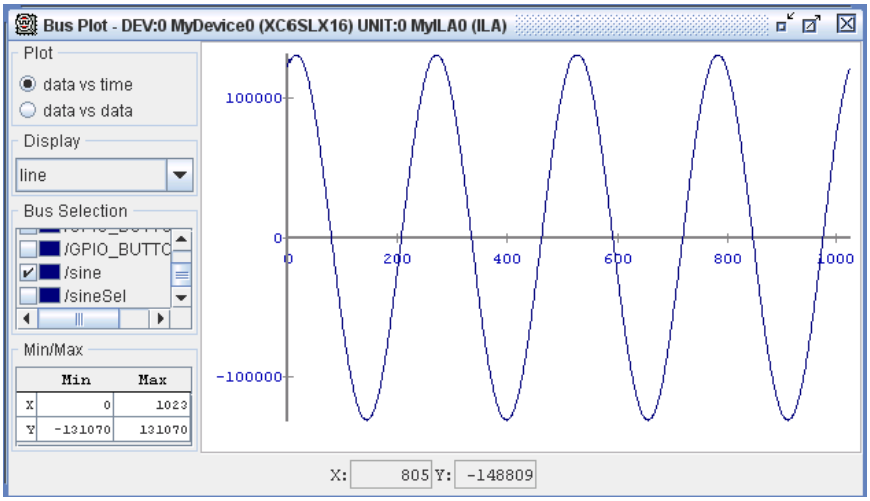

図 4-7 : 中周波数のサイン波のバス プロット

15. ボードのサイン波選択インジケーターの LED が「on, off, 10」と表示されるまで、ボードで [Sine Wave Sequencer] ボタンを押し続けます。

16. [Trigger Immediately] ボタンをもう 1 度クリックし、低周波数のサイン波のバス プロットを表示します。

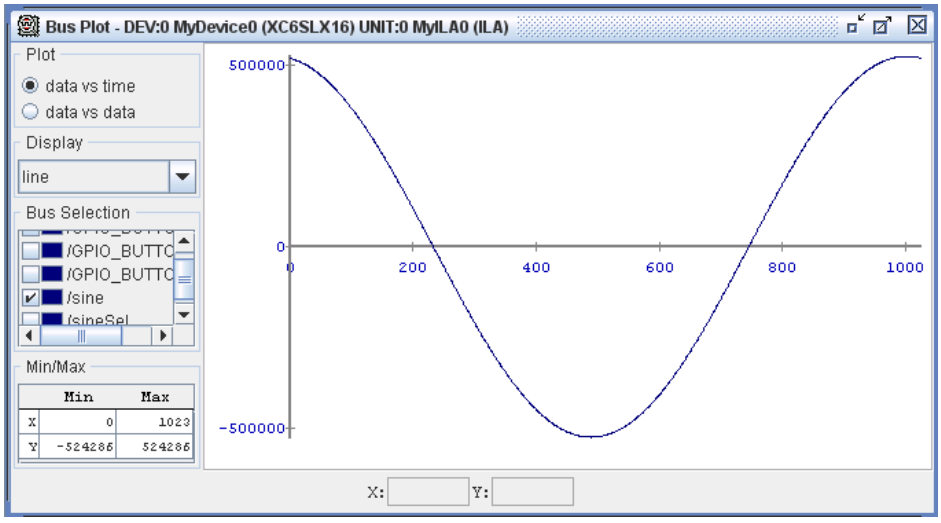

図 4-8 : 低周波数のサイン波のバス プロット

17. ボードのサイン波選択インジケーターの LED が「on, on, 11」と表示されるまで、ボードで [Sine Wave Sequencer] ボタンを押し続けます。

18. [Trigger Immediately] ボタンをクリックし、混合サイン波のバス プロットを表示します。

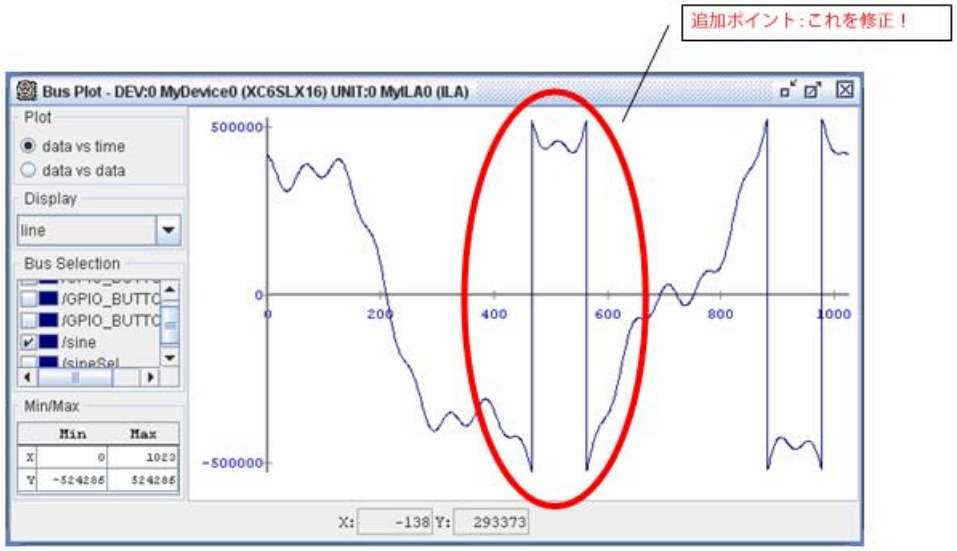

図 4-9 : 混合サイン波のバス プロット

すべてのサイン波選択が正しいかどうか確認しましたが、選択ロジック回路はまだ正しく動作していません。

- ☑ すべてのサイン波選択が正しいかどうか確認します。
- ☐ 選択ロジックが正しく動作しているかどうかは、次の手順で確認できます。
  - ☐ ステート マシンが正しく遷移し、出力が正しいかどうかを確認します。
  - ☐ ステート マシンの入力が正しいかどうかを確認します。

次に選択ロジック回路のデバッグを開始します。

19. 次のパラメータを設定してください。

[Match] : [TriggerPort1] を [RX] に設定 -- GPIO_BUTTONS_re[1] の立ち上がりエッジが FSM が遷移する原因となっているので検索

[Trigger Conditions] : M1 -- トリガー式を M1 に設定

[Capture Settings] : Windows = 10、Depth = 2

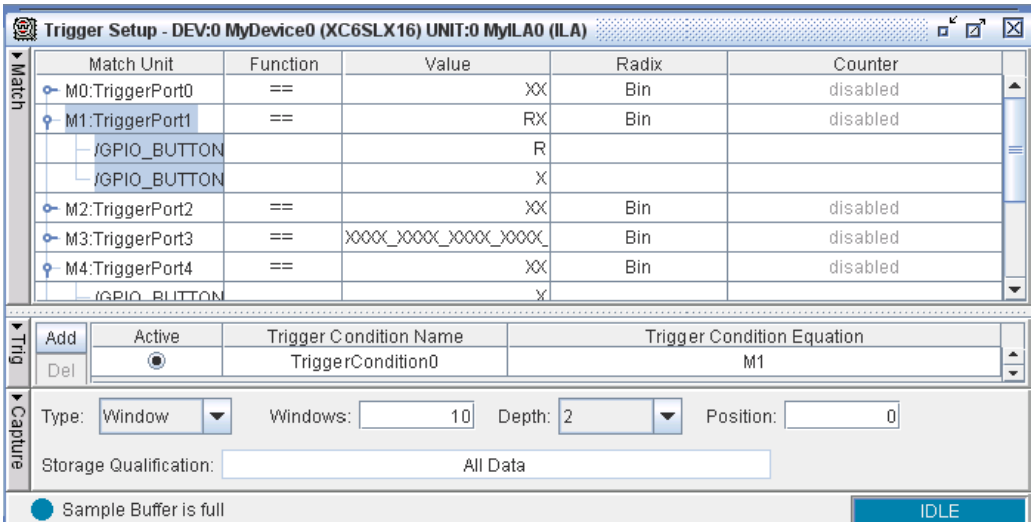

図 4-10 : トリガの設定ウィンドウ

メモ : GPIO_BUTTONS_re[1] ネットの実際の TriggerPort 番号はこのチュートリアルとは異なることもあります。

20. [Run Trigger] ボタン をクリックしてトリガーを開始し、待ちます。

    進捗状況は、ウィンドウの一番したに表示されます。

21. ボードの [Sine Wave Sequencer] ボタンを押します。

22. キャプチャーされたウィンドウ数を確認します。

    - キャプチャーされたウィンドウが 1 つだけの場合は、21 と22 を繰り返します。
    - 複数のウィンドウがキャプチャーされた場合は、次の手順へ進みます。

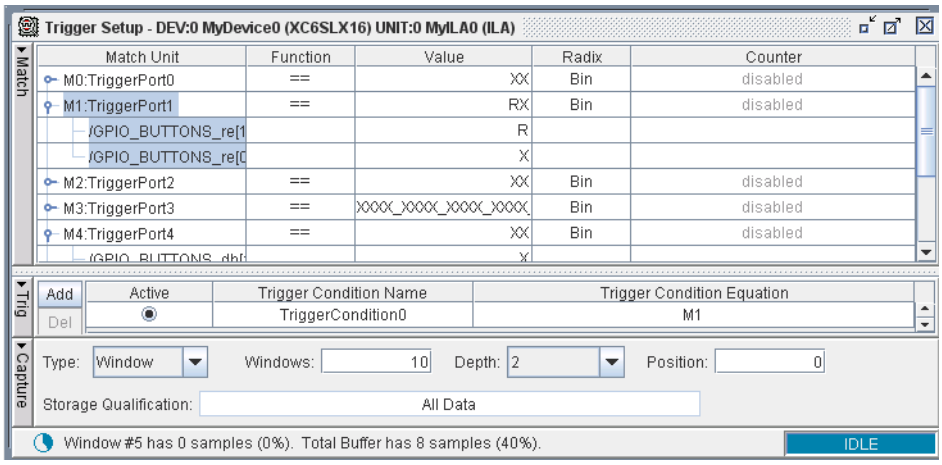

図 4-11 : キャプチャーされたウィンドウの確認

23. [Stop Trigger] ボタン をクリックし、キャプチャーされたデータを確認します。

    [Sine Wave Seqencer] ボタンを押すたびに複数の立ち上がりエッジがあることがわかります。また、ステート マシンが正しく動作しているかどうかを示す sineSel が正しく遷移していることもわかります。

ここまでで、サイン波ジェネレーターが正しく動作しているかどうかを確認しました。

- ☑ すべてのサイン波選択が正しいかどうか確認します。
- ☐ 選択ロジックが正しく動作しているかどうかは、次の手順で確認できます。
  - ☑ ステート マシンが正しく遷移し、出力が正しいかどうかを確認します。
  - ☐ ステート マシンの入力が正しいかどうかを確認します。

24. トリガー モードおよび状況を指定する次のパラメータを設定します。
    - [Trigger Run Mode] :Repetitive
    - [Match] : [TriggerPort5] を[XRX] に設定 -- ボードの [Sine Wave Sequencer] ボタンの入力バッファである GPIO_BUTTONS_1_IBUF の立ち上がりエッジを検索.
    - [Trigger Conditions] :M5
    - [Capture Settings] :
      - Windows = 1
      - Depth = 1024
      - Position = 512
25. [Run Trigger] ボタン をクリックし、GPIO_BUTTONS_1_IBUF 信号で複数の遷移があるまで、ボードの [Sine Wave Sequencer] ボタンを押し続けます。

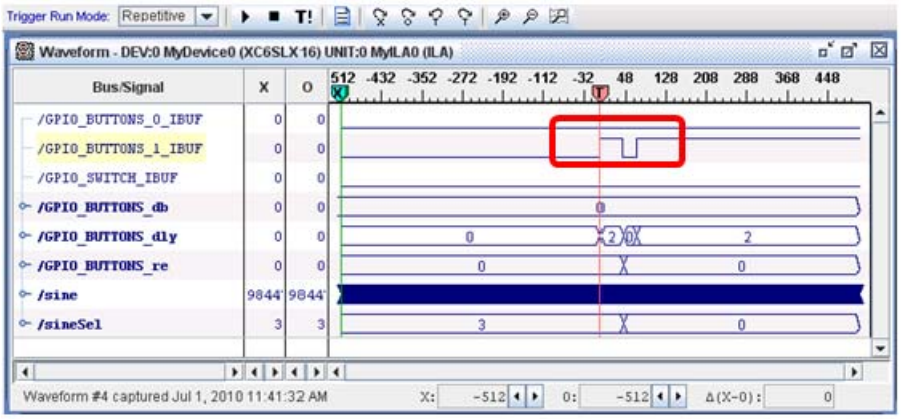

図 4-12：波形の表示

メモ：波形には、信号グリッチが図と同じ位置に表示されないこともあります。これは、トリガー実行モードを [Repetitive] にした場合の利点です。

ここまでで、サイン波ジェネレーターが正しく動作しているかどうかを確認しました。ただし、ステート マシンの入力は正しくありません。これらの入力はプッシュ ボタン スイッチから直接接続されています。

- ☑ すべてのサイン波選択が正しいかどうか確認します。
- ☒ 選択ロジックが正しく動作しているかどうかは、次の手順で確認できます。
  - ☑ ステート マシンが正しく遷移し、出力が正しいかどうかを確認します。
  - ☒ ステート マシンの入力が正しいかどうかを確認します。

図 4-12 に示すように、プッシュ ボタン スイッチが押されて離されるごとにグリッチが生成されるので、問題の原因はプッシュ ボタン スイッチにあるようです。複数の遷移を引き起こすこれらのグリッチを削除するため、プッシュ ボタン スイッチにはそれぞれデバウンス回路が必要です。

デバウンス回路は、既にデザイン例に含まれています。デバウンス回路をイネーブルにするには、DIP スイッチ 1 をオンにし、24 と25 を繰り返し、ボタンを 1 度押すごとに遷移が 1 つだけになることを確認します。

質問

5. 手順 18 の問題を解決して追加ポイントを取得する時間はありましたか。
   _____
6. このチュートリアルでデバウンス回路が必要な理由は何ですか。
   _____

XILINX®

第 5 章

チュートリアルのまとめ

まとめ

このチュートリアルでは ChipScope Pro ILA Core Inserter と Project Navigator の統合フローを紹介し、Project Navigator で IP コアのネットリストを生成してデザインを合成し、ILA Core Inserter を使用して ChipScope ILA コアをデザインに追加する方法について説明しました。また、デバッグ プロセスについても詳しく説明し、ChipScope Pro Analyzer を使用し、さまざまなトリガー設定でデザインをデバッグする方法を紹介しました。

これにより、基本的なデザイン フローの一部と Project Navigator と ChipScope Pro 間の統合について理解いただけたはずです。

質問の解答

1. 手順 1 で実行したことを簡単に説明してください。

   PlanAhead の RTL プロジェクトを作成し、VHDL の ChipScope™ デザインに読み込んで、デザインをインプリメントしました。

2. このチュートリアルで使用した主な回路は何ですか。

   - debounce.vhdl - デバウンス回路
   - fsm.vhdl - 制御ステート マシン
   - sinegen_demo - サイン波ジェネレーターのラッパー

3. ほかのザイリンクス ボードをターゲットにする場合、どのソース ファイルを修正する必要がありますか。

   UCF 制約ファイル

4. デバッグ プローブを HDL デザイン ファイルに追加するのではなく、合成後のネットリストに挿入する主な利点は何ですか。

   元の HDL ファイルを直接変更する必要がないので、間違ってミスを引き起こす可能性はなくなりました。

5. 手順 18 の問題を解決して追加ポイントを取得する時間はありましたか。

   ヒント：Analyzer の波形から判断すると、sinegen_demo.vhd および sinegen.vhd モジュールで指定されたビット ベクターが足りないために、出力サイン波が切り捨てられている可能性があるようです。現在のところ 20 ビット ベクターに指定されていますが、これは 22 ビットまで拡張する必要があります。これらの 2 つのモジュールを変更し、デザインを繰り返してください。

6. このチュートリアルでデバウンス回路が必要な理由は何ですか。

   このデザイン例の場合、デバウンス回路によりクリーンなパルスまたは High から Lpw への遷移が提供され、ボタンを押してから放すと、一連のスパイクやグリッチが削除されます。