ISim 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 (FFT) のシミュレーションの高速化

UG817 (v 13.3) 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
2011/11/11	13.3	リリースに合わせて変更 追加事項： · 改訂履歴 · 付録 C：その他のリソース 更新事項： · テストベンチの作成

日付	バージョン	説明
		· 全体的なグラフィックの更新 · 13.3 リリースに合わせて文書および手順を更新

目次

第 1 章

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、浮動小数点高速フーリエ変換 (FFT) のシミュレーションを高速化し、ザイリンクス ML605 ボード上で FFT インプリメンテーションを検証する方法を説明します。

DSP (Digital Signal Processing) デザインは、そのデータおよび計算量が多いため、ソフトウェアでシミュレーションすると非常に時間がかかります。

- DSP ファンクションのシミュレーションを高速化するため、高速ビット精度モデルがよく使用されますが、サイクル精度は提供されず、その他の RTL (Register Transfer Level) モジュールと統合するのも簡単ではありません。
- ビヘイビアー RTL モデルではビットおよびサイクル精度が提供されますが、シミュレーションが低速になります。構造 RTL (レジスタ転送レベル) またはゲート レベル モデルを使用すると、シミュレーション速度はさらに低下します。
- IP によっては高速ビット精度モデルは提供されておらず、ビヘイビアー RTL モデルがない場合もあり、構造/ゲート レベルのシミュレーションが唯一の方法となります。

ISim ハードウェア協調シミュレーションでは、多量の計算を FPGA で実行させることにより、ソフトウェアの負担を軽減して DSP ファンクションのシミュレーションを実行できます。合成可能な HDL コード、CORE Generator™ ツールで生成された IP コアなどの合成済みまたは保護されたネットリストを、協調シミュレーション用に FPGA に読み込むことができます。これにより、ビットおよびサイクル精度シミュレーション モデルを使用する必要はなく、シミュレーションのパフォーマンスを向上できます。複雑な DSP デザインの多くでは、デザインのシミュレーションが高速化されるだけでなく、実際のハードウェア上でのデザインのインプリメンテーションを検証できます。ISim ハードウェア協調シミュレーションは、RTL シミュレーション、合成後のシミュレーション、インプリメンテーション後のシミュレーションを補足するものです。

要件

- ISE® Design Suite
- Virtex®-6 FPGA ML605 評価キット
- デザイン ファイル : rdf0125_fft_sim_tutorial.zip

チュートリアル ファイル

ファイル	説明
fp_fft_top.v	浮動小数点 FFT コアをインスタンシエートするラッパー
fp_fft_tb.v	FFT を実行するためのテスト ベクターを生成する最上位 Verilog テストベンチ
fp_fft_tb.vhd	FFT コアを実行するためのテスト ベクターを生成する最上位 HDL テストベンチ
FloatingPointFFT.xise	このチュートリアル用の ISE® プロジェクト
sim.tcl	ISim のシミュレーション時間を計測するカスタム シミュレーション コマンド ファイル
fp_fft_tb.wcfg	カスタム波形コンフィギュレーション ファイル
fp_fft_tb.prj	コマンド ライン フロー用の ISim プロジェクト ファイル
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
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

注記 : このチュートリアルを実行する際は、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てください。

XILINX®

第 2 章

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、浮動小数点高速フーリエ変換 (FFT) のシミュレーションを高速化し、ザイリンクス ML605 ボード上で FFT インプリメンテーションを検証する方法を説明します。

4 つのセクションから構成されており、ISim ハードウェア協調シミュレーションを使用して FFT デザインを実行するのに必要な手順を示します。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™ での FFT コアの生成
2. テストベンチの作成
3. 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
4. ISim ハードウェア協調シミュレーションの実行

手順 1 : CORE Generator での FFT コアの生成

このチュートリアルでは、CORE Generator™ ツールで高速フーリエ変換 (FFT) IP コアを使用し、Virtex®-6 FPGA ML605 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のスクリーン ショットは、Fast Fourier Transform v8.0 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 (FloatingPointFFT) と保存ディレクトリを入力します。[Next] をクリックします。
3. [Project Settings] ページで、次を設定します。
   - ML605 ボードで使用する [Family] に [Virtex6]、[Device] に [XC6VLX240T] を選択
   - [Package] に [FF1156] を選択
   - [Speed] に [-1] を選択
   - [Simulator] に [ISim]、[Preferred Language] に [Verilog] を選択
   - [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。[IP (CORE Generator & Architecture Wizard)] を選択し、[File name] に「fp_fft_core」と入力します。[Next] をクリックします。
5. IP リストの [By Function] または [By Name] から [Fast Fourier Transform] のバージョン 8.0 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。
6. Fast Fourier Transform コアの GUI が開いたら、[Transform Length] を [16384] に設定します。[Implementation Options] で [Pipelined, Streaming I/O] をオンにして [Next] をクリックします。
7. [Data Format] で [Floating Point] をオンにし、[Output Ordering] で [Natural Order] をオンにします。[Next] をクリックします。
8. [Complex Multipliers] で [Use 4-multiplier structure (performance optimization)] をオンにします。[Generate] をクリックしてコアを生成します。
9. 生成した fp_fft_core IP コアをインスタンシエートする最上位モジュール `fp_fft_top` を追加します。ISim ハードウェア協調シミュレーションでは、現在のところ HDL 最上位モジュールのみがサポートされています。このチュートリアルに含まれる完成した `fp_fft_top.v` を使用できます。[Project] → [Add Source] で fp_fft_top.v を選択し、[開く] をクリックしてから、[OK] をクリックします。

手順 2：テストベンチの作成

1. fp_fft_top インスタンスを実行するテスト ベクターを生成する Verilog テストベンチ モジュール `fp_fft_tb.v` を追加します。このチュートリアルで提供されている完成した `fp_fft_tb.v` を使用できます。
2. [Project] → [Add Source] をクリックし、fp_fft_tb.v を選択します。
3. [開く] をクリックしてから [OK] をクリックします。

このテストベンチには、64 ビット x 16384 アレイの `fft_xn_re_data` および `fft_xn_im_data` が含まれており、FFT 入力の実コンポーネントおよび仮想コンポーネントが格納されます。

シミュレーションが開始すると、テストベンチにより最初のフレームに `input_data.txt` データ ファイルに含まれる FFT 入力ベクターが読み込まれます。

`input_data.txt` ファイルでは、行ごとにデータ ポイント 1 つが格納されます。各データ ポイントは 32 ビットの 16 進数値 2 つ (実数部分と虚数部分) で構成されており、スペースで区切られています。この 16 進数は、IEEE 754 規格を使用した浮動小数点値を 2 進数で表現したものです。

注記： TXT データ ファイルは、シミュレーションを実行するディレクトリに配置する必要があります。

デモ テストベンチで実行される操作は、生成コアのコンフィギュレーションに適切で、次のサブセット操作から構成されています。

- フレーム 1：ファイルに含まれている生成前の入力データのフレームを駆動
- フレーム 2： 逆変換のコンフィギュレーション。フレーム 1 の出力を入力データのフレームとして駆動
- フレーム 3 のコンフィギュレーション：1 つ前の変換実行中に順変換
- フレーム 3：フレーム 2 の出力を入力データのフレームとして駆動。AXI `TVALID` (および `TREADY` 信号をときどきディアサートして、AXI のハンドシェイクを実行
- フレーム 4 ～ 7：これらのフレームを連続して実行
  - 順変換 (フレーム 4) のコンフィギュレーションに逆変換 (フレーム 5) を続け、この両方で小さいポイント数 (コンフィギュレーション可能な場合) と短いサイクリック ペリフィックス (CP) を使用
  - フレーム 4： 生成前の入力データのフレームを駆動
  - フレーム 5：フレーム 1 の出力を入力データのフレームとして駆動。同時にフレーム 6 をコンフィギュレーション最大ポイント数、比較的長いサイクリック ペリフィックス (CP)、ゼロ スケーリング スケジュール (固定スケーリング使用の場合) を使用して順変換。
  - フレーム 6：ファイルから生成前の入力データのフレームを駆動。同時にフレーム 7 をコンフィギュレーション：最大ポイント数、サイクリック プリフィックス (CP) なし、デフォルトのスケーリング スケジュール (固定スケーリング使用の場合) を使用して逆変換。
  - フレーム 7：フレーム 1 の出力を入力データのフレームとして駆動
- すべてのフレームが完了するまで待機

手順 3 : 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 で [Simulation] ビューに切り替えます。[Hierarchy] ペインで `[fp_fft_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. [Source Properties] ダイアログ ボックスで、次を実行します。
   - [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 [aclk] に設定します。
   - [Target Board for Hardware Co-Simulation] を [ML605 (JTAG)] に設定します。
   - [Enable Incremental Implementation] はオフのままにして、[OK] をクリックします。

   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、特殊なアイコンが付きます。 

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たら、[Enable Incremental Implementation] を使用できます。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、このオプションをオンにすると、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 `[fp_fft_tb]` インスタンスをクリックします。
4. [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。
5. [Process Properties – ISim Properties] ダイアログ ボックスで [Property display level] を [Advanced] に設定してから、[OK] をクリックします。すべての値を確認します。このチュートリアルでは、デフォルトの Advanced 設定を使用します。[OK] をクリックします。
6. [Processes] ペインで `dut - fp_fft_tb` インスタンスに対して [Simulate Behavioral Model] を実行します。

コマンド ラインでのデザインのコンパイル

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。fuse を実行するときには、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する必要があります。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、次に示す引数も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 <constraints file>
     -hwcosim_incremental [0|1]
```

- -hwcosim_instance : ハードウェア協調シミュレーションを実行する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。
- -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 FFT 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 fp_fft_tb.prj fp_fft_tb
     –o fp_fft_tb.exe
     -hwcosim_instance /fp_fft_tb/dut
     -hwcosim_clock aclk
     -hwcosim_board ml605-jtag
```

手順 4：ISim ハードウェア協調シミュレーションの実行

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、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、次のメッセージが表示されます。

“Downloading bitstream, please wait till status is READY”.

FPGA がコンフィギュレーションされると、次のメッセージが表示されます。

“Bitstream download is complete.READY for simulation.”

注記： コンピューターに複数のイーサネット ポートがある場合は、シミュレーション プロセスで使用するポートを識別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。

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。

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

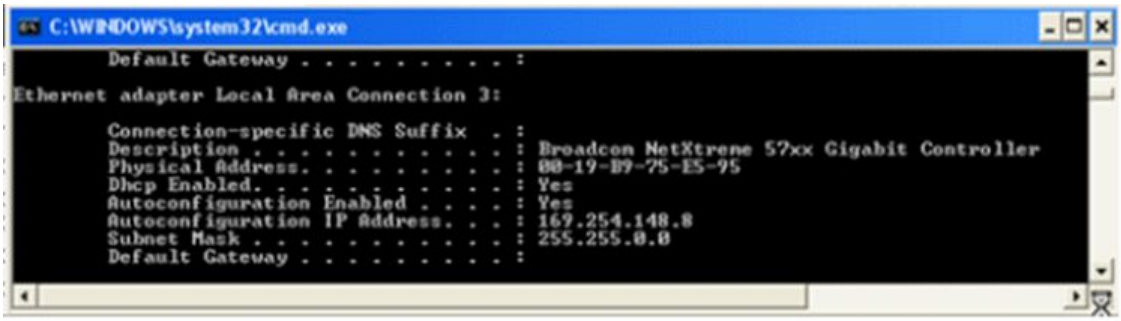
イーサネット ポートの決定

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。