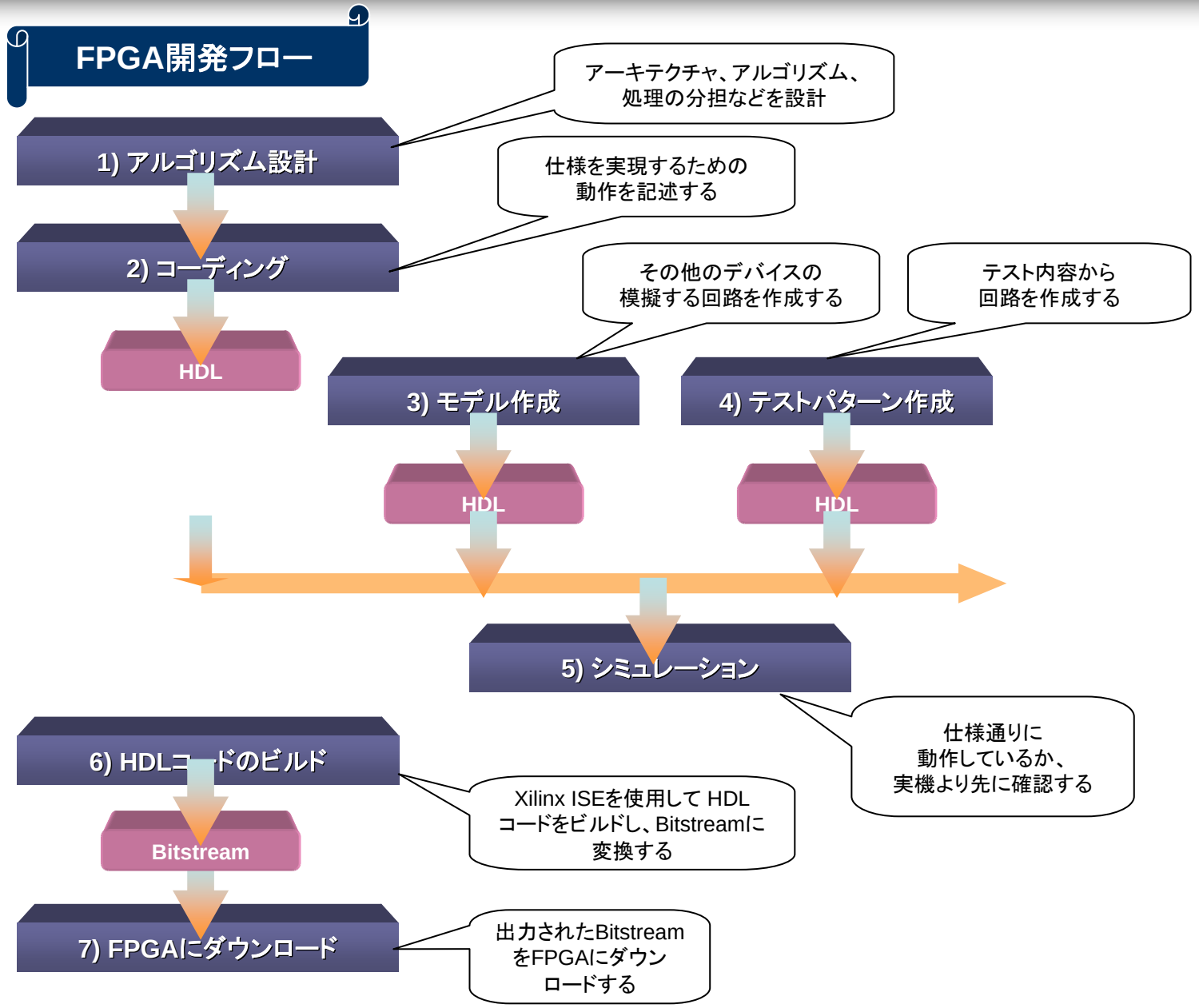
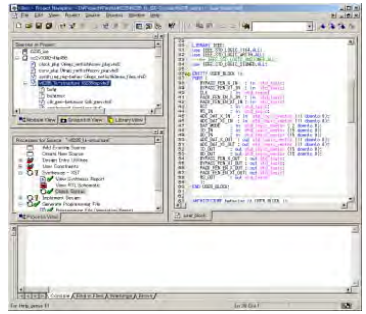
MISH
INTERNATIONAL

VHDL/Verilog による FPGA開発のアプローチ

株式会社ミッシュインターナショナル
MISH International Co., Ltd.

FPGA開発フロー


FPGA開発フロー
アーキテクチャ、アルゴリズム、
処理の分担などを設計
1) アルゴリズム設計
仕様を実現するための
動作を記述する
2) コーディング
HDL
その他のデバイスの
模擬する回路を作成する
テスト内容から
回路を作成する
3) モデル作成
4) テストパターン作成
HDL
HDL
5) シミュレーション
仕様通りに
動作しているか、
実機より先に確認する
6) HDLコードのビルド
Xilinx ISEを使用して HDL
コードをビルドし、Bitstreamに
変換する
Bitstream
7) FPGAにダウンロード
出力されたBitstream
をFPGAにダウン
ロードする



開発環境に関して

- **ISE Foundation**
  - Xilinx社製、FPGA統合開発環境
    http://japan.xilinx.com/ise/logic_design_prod/foundation.htm
- **ChipScope**
  - Xilinx社製、オンチップロジックアナライザ
    http://japan.xilinx.com/ise/optional_prod/cspro.htm
- **ModelSim**
  - MentorGraphics 社製、HDL シミュレーション環境
    http://www.mentorg.co.jp/solution/fpga-pld/simulation/
- **Active-HDL**
  - ALDEC 社製、統合開発(HDL シミュレーション)環境
    http://www.aldec.co.jp/products/active-hdl/

1) アルゴリズム設計

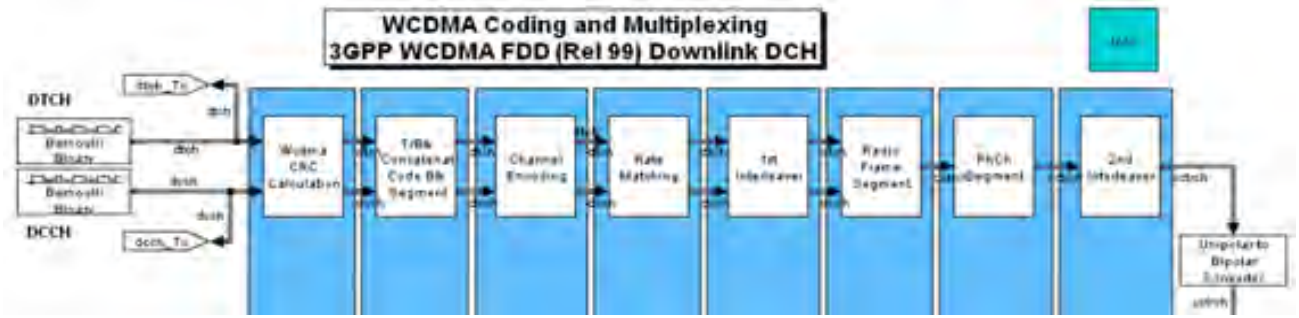
- FPGA でどのように処理を行うかを設計
  - 仕様から、必要な処理と流れを作ります。
  - メモリを使用する場合は、必要に応じて内部メモリマップも作成します。
  - 必要に応じて、デバッグ機能も検討します。
  - ソフトウェアとの処理の分担も、ここまでで決定します。

- プロセッサとの通信が必要な場合はレジスタマップを決定
  - ソフトウェアとの処理の分担が決まればそこでやりとりするデータも決まります。

- 大まかな規模を求める

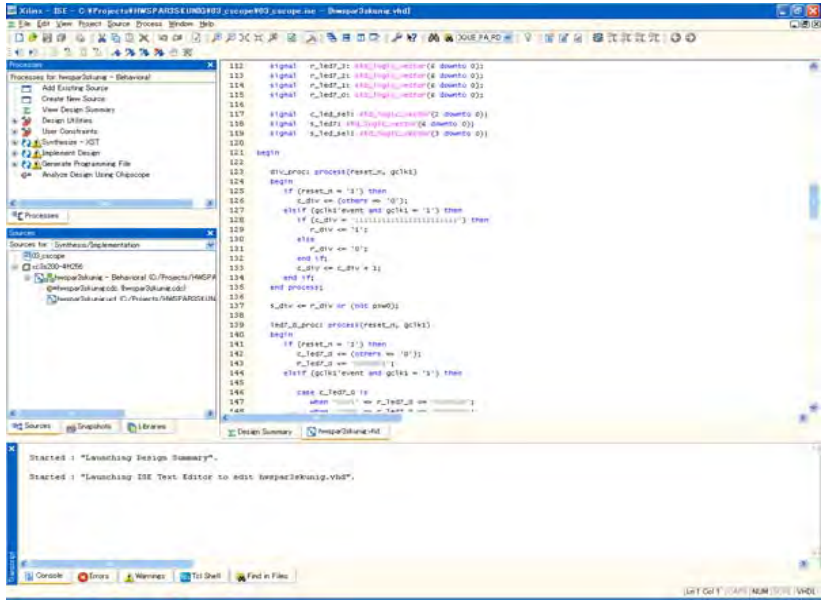
デバイスに収まるかどうか、また収まらない場合の対応（別デバイスを選択、機能に制約を加える等）を検討します。
※規模は、経験則より
　　ロジックエレメント数＝フリップフロップ数×１６
で求められます。ここで

- ロジックエレメントは、ルックアップテーブル(LUT)とフリップフロップの組み合わせです。
- フリップフロップ数は、各処理の内容から大まかに数えたものです。
  例えば、16bit カウンタなら 16 と数えます。
  16bit パラレルーシリアルコンバータなら、データ 16bit ＋カウンタ 4bit＋出力レジスタ 1bit で 21 と数えるのが目安です。



2) コーディング

- HDL でコーディング
  - VHDL 、Verilog HDL がよく使われます。
  - System C、Spec C、Impulse C、CyberWorkbench 等の C 言語開発もあります。
- IP (Intellectual Property / 完成した回路ブロック) が使用可能
  - 外部メモリや PCI 等のインタフェース、フィルタ回路などがあります。
  - デバイスメーカーやサードパーティが提供しています。
  - 有償、無償のものがあります。
- HDL 開発のサポートソフトウェア
  - Xilinx SystemGenerator 等、Simulink モデルから HDL を出力するソフトウェアがあります。
  - ステートマシンから HDL を出力するソフトウェアがあります。

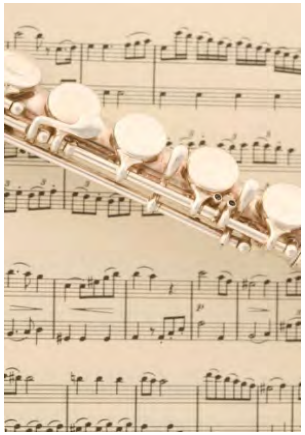


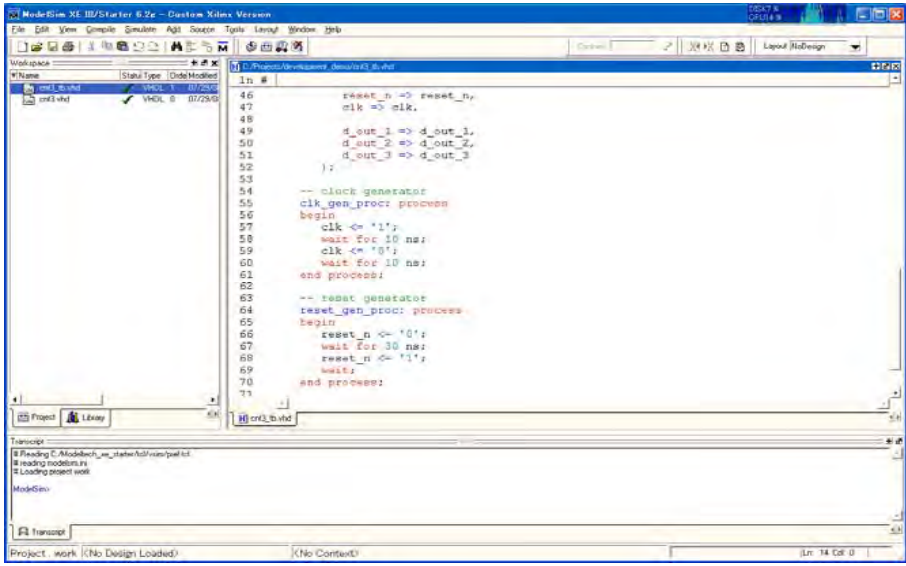
3) モデル作成
4) テストパターン作成

- テスト、デバッグのためにシミュレーション環境を構築する
  - テストベンチと呼びます。
  - 対象の FPGA のコードとモデルを接続したもので、HDL で記述します。
  - テストしたい内容毎に、テストベンチを作成します。

- シミュレーション用のモデルを作成する
  - 例えばプロセッサと接続する FPGA を設計している場合、テストベンチ内に仮想の CPU を用意して、実機に近い環境を用意することでシミュレーションの精度を高めます。
  - 一般的には、FPGA に接続された機能デバイス（外部メモリ、プロセッサなど）全てのモデルを用意します。
  - プロセッサなどの複雑なデバイスは、簡略化したモデルを用意します。
  - モデルは市販もされています。

- テストパターンを作成する
  - 最低でも、入力パターンは必要です。
  - 入力データ、出力データはファイルからの読み込みが可能です。
  - 出力データやタイミングは、テストベンチ内でチェックできます。

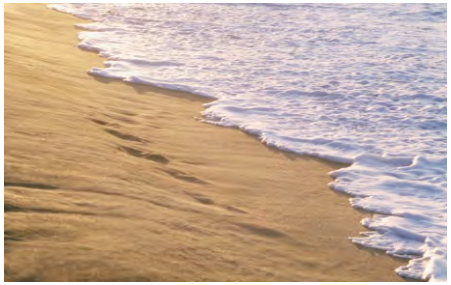


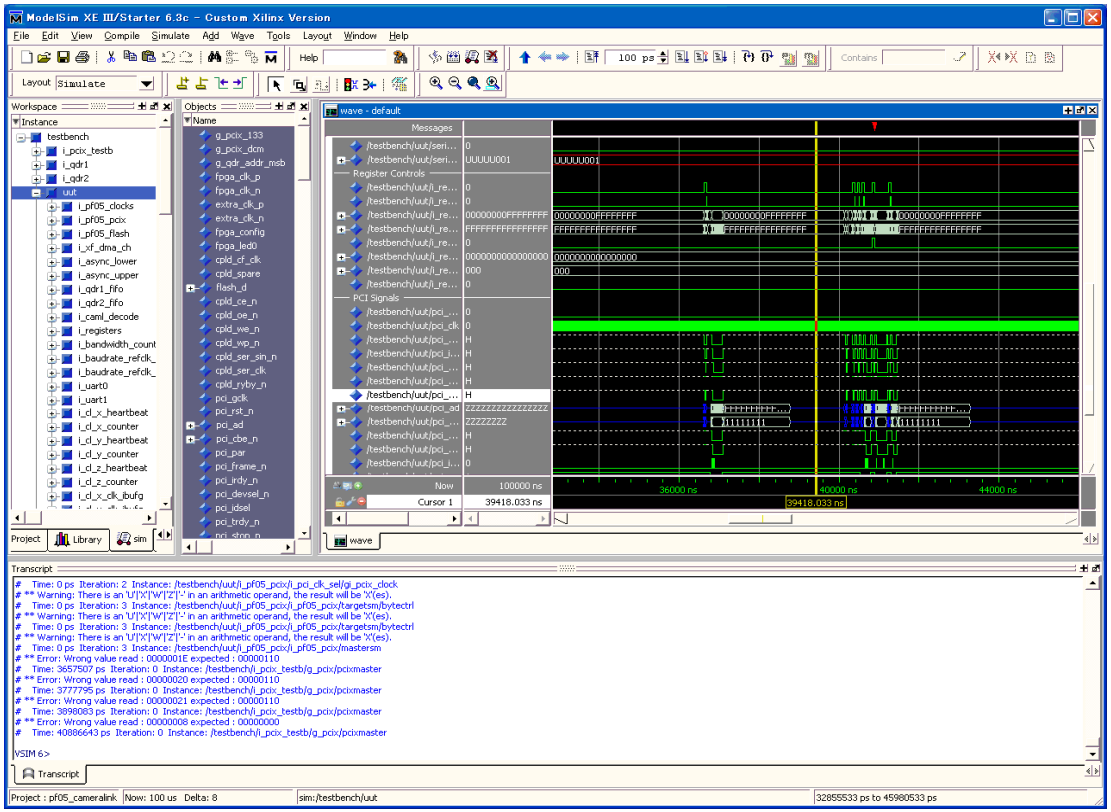


MISH INTERNATIONAL

5) シミュレーション

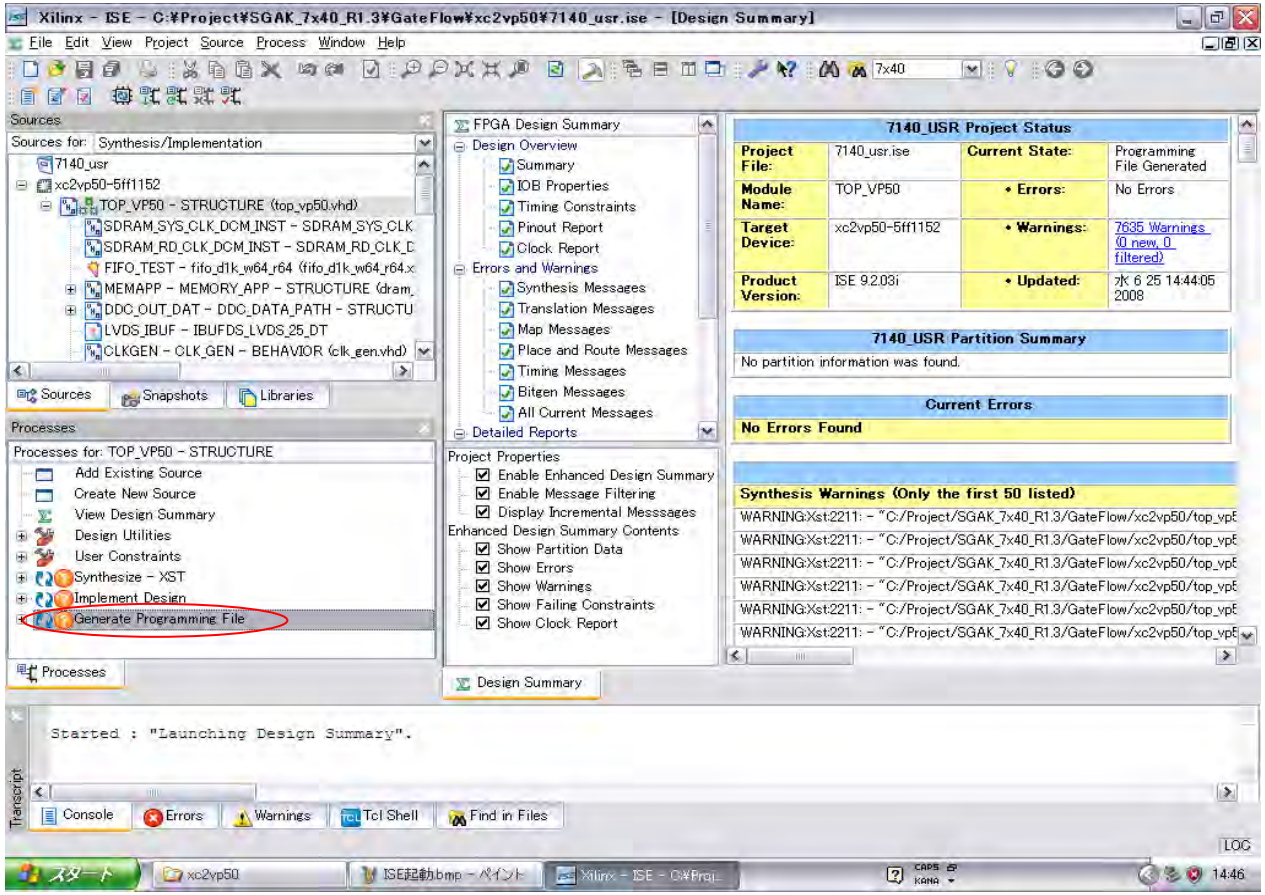
- HDL シミュレータでシミュレーションを行う
  - シミュレーションが完了すると、動作波形が表示されます。
- バッチシミュレーションも可能
  - テストベンチで合否判定ができる場合、又は出力ファイルを後から一括でチェックできる場合は、バッチファイルでの連続シミュレーションが可能です。





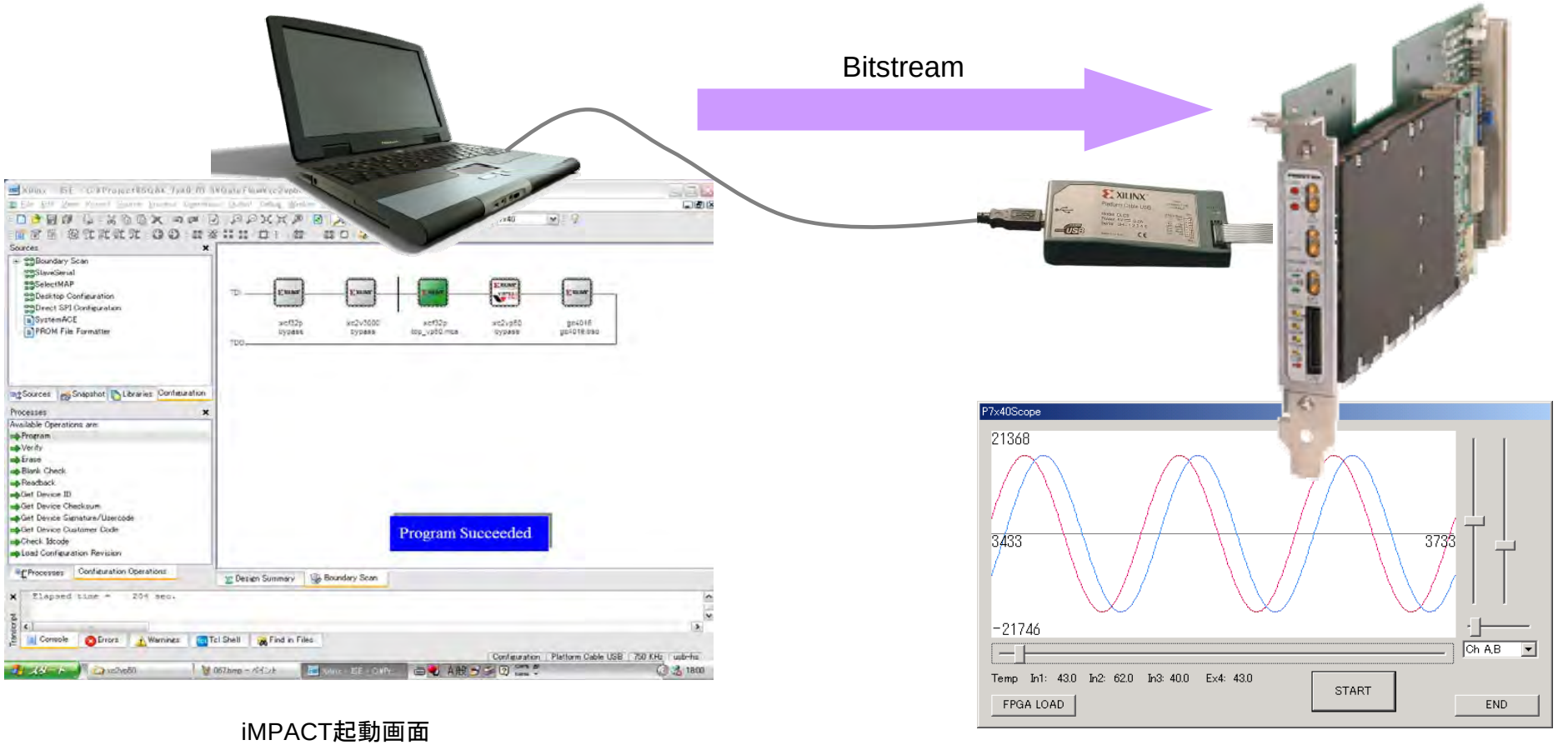
6) VHDLコードのビルド

- ISE FoundationでISEプロジェクトをビルドする
  - Xilinx社製統合開発環境ISE Foundationを使用して論理合成・配置配線・ビットストリーム生成を行いビルドします
  - ビルドが完了すると"*.bit"ファイルが出力されます（ROMに焼きこむ場合はmcsファイルに変換します）



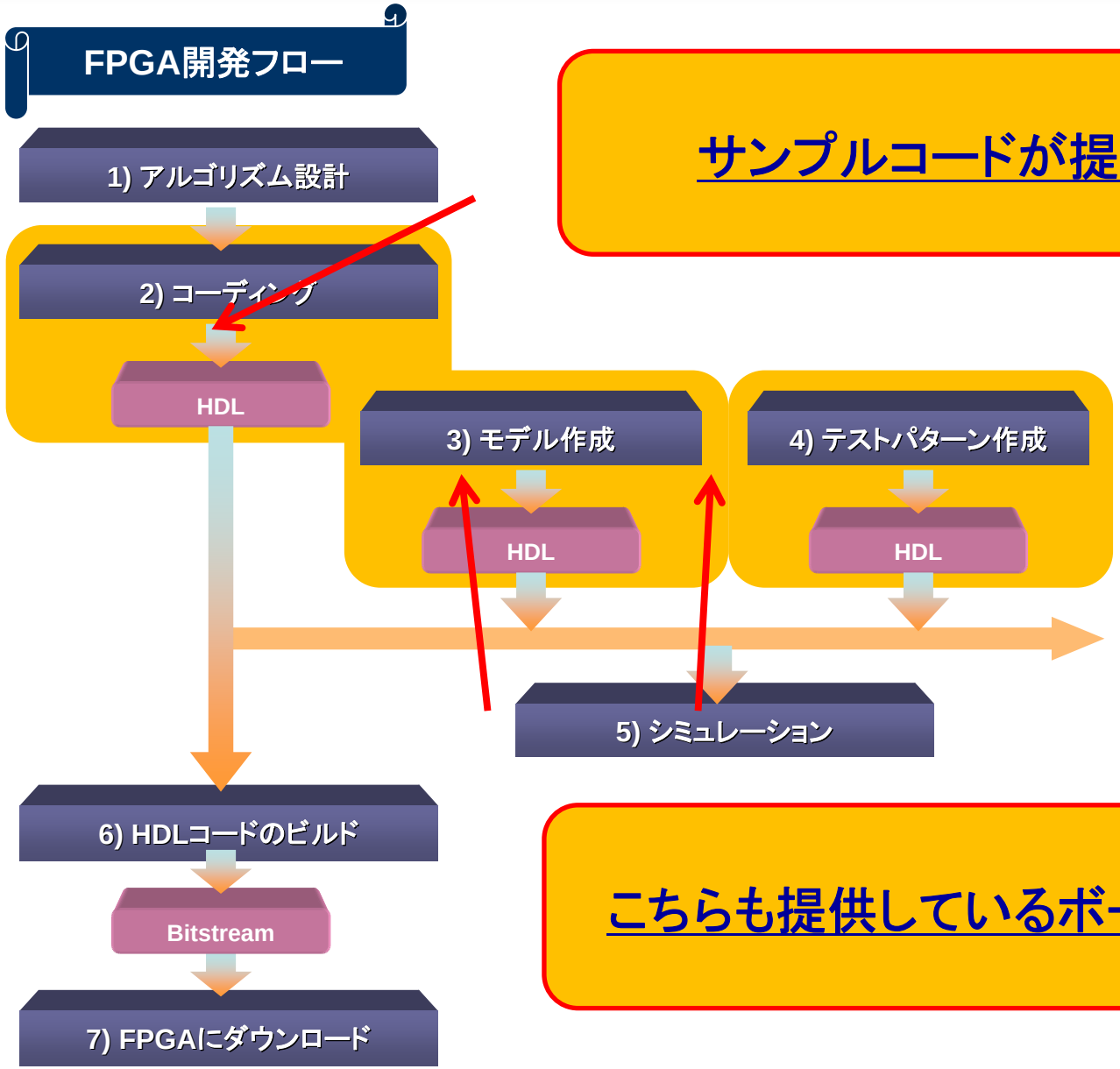
7) FPGAコードのダウンロード

- **iMPACTでFPGAにダウンロード**
  - ISEに書き込みツールとして含まれているiMPACTを使用して、生成されたBitstreamをJTAGケーブル（又はプログラム経由）でFPGAにダウンロードします



iMPACT起動画面

8) MISH 取扱 FPGA ボードの特徴

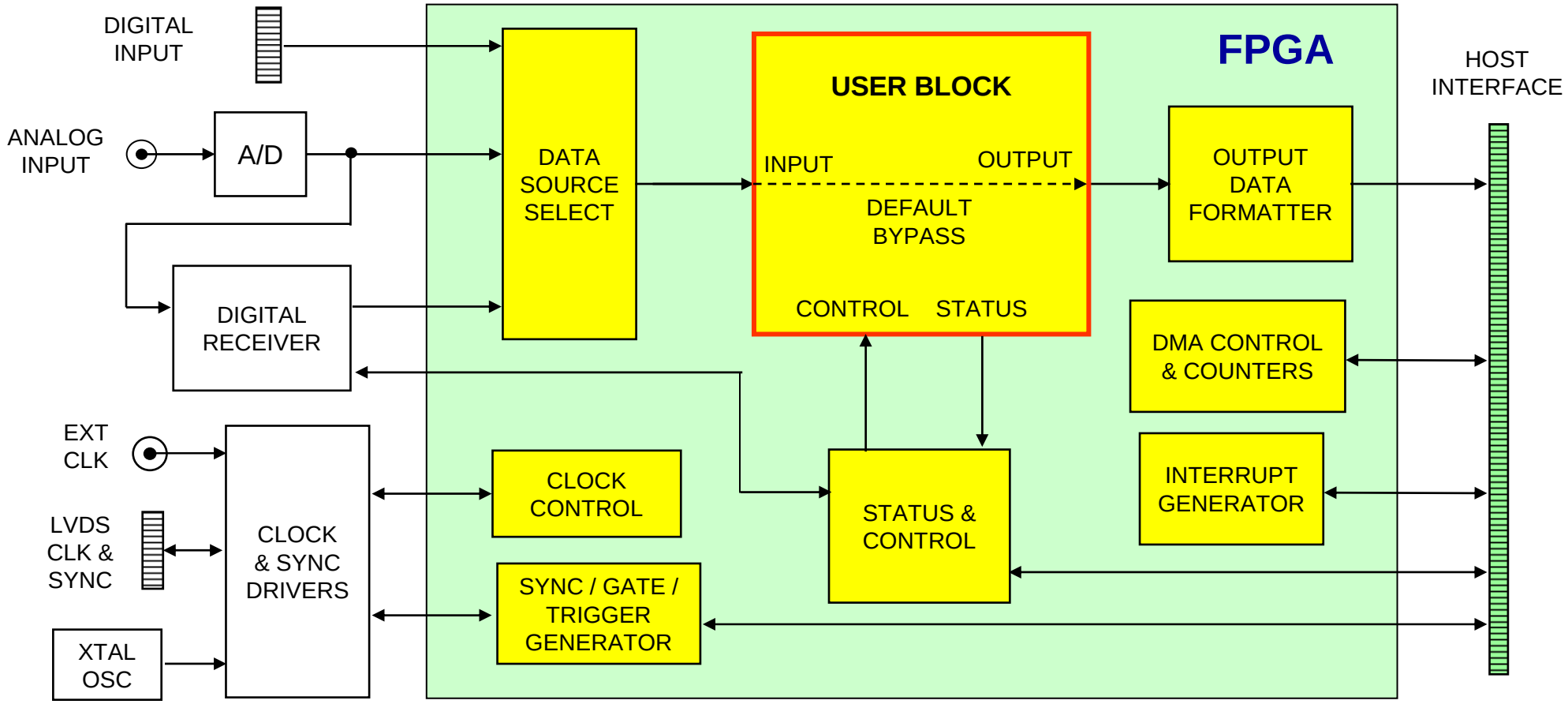


サンプルコードが提供されます。

こちらも提供しているボードもあります。

8) MISH 取扱 FPGA ボードの特徴

- **提供されているVHDLサンプルコードについて**
  - サンプルコードにはユーザーアルゴリズムを実装しやすいようにユーザーブロックがVHDLコードで予め準備されているものもあります。これによりユーザーはユーザーブロックのエリアのみ開発する事でFPGAにアルゴリズムを実装する事ができます



8) MISH 取扱 FPGA ボードの特徴

ホストインタフェースや A/D 、D/A コンバータなど
外部のデバイスとのインタフェースを検証済みのソースファイルを提供しています。
→ ユーザーアルゴリズムの開発に集中できるので、
開発期間の大幅な短縮が可能です。

サンプルソースファイルは最低限の処理の流れができていますので、
A/D、D/Aボードなどはそのまま使用することもできます。

シミュレーション用のテストベンチやモデルも提供しているボードもあります。
これらの提供がないボードでも、ユーザーアルゴリズムを組み込むための
「ユーザーブロック」が提供されます。
この部分の入出力を汎用化することで、開発部分の切り分けを容易にしています。