Myway
it's a passion way

パワエレ技術者向けFPGAセミナー
演習資料

2020年2月21日

Mywayプラス株式会社

Myway
it's a passion way

内容

- 演習0. Vivadoインストール
- 演習1. VHDL設計の基本
  - 設計環境の基本操作
- 演習2. PWM生成回路の構成例
  - シミュレーションとデバッグ作業
- 演習3　階層化
- 演習4. モジュールのブロック化
  - IP作成による効率的な設計
- 演習5. ADインターフェースの設計
  - 順序回路によるシリアルデバイスとの通信

Myway
it's a passion way

演習0. Vivadoインストール

- 受講者は事前にVivado HIL WebPACKのインストールを行い、PCをご持参下さい（64bit OSのみ）
- インストール方法および動作環境はXilinx社により随時更新されるとともに、処理系（お客様PC）に依存します。Mywayプラス社からは対応できかねますのでご了承下さい。
- （参考）Vivadoダウンロードリンク
- https://japan.xilinx.com/support/download/index.html/content/xilinx/ja/downloadNav/vivado-design-tools.html
- ※本セミナーで使用するソースコード・設計情報等は受講者の学習用途となっております。動作についての責任は一切負いませんのでご承知おきください。

Myway
it's a passion way

演習1.
VHDL設計の基本
（設計環境の基本操作）

Myway
it's a passion way

演習1.

VHDLコード・テストベンチの基本構成を理解する。基本的な開発環境の使用方法に慣れる。

1. 開発環境Vivado立上げ
2. ソースコード説明
    1. VHDLの基本構成
    2. 組み合わせ回路と順序回路
    3. レジスタとワイヤー
    4. テストベンチ構成
3. シミュレーションの実行

下記のプロジェクトファイルをクリックしてVivadoを立ち上げて下さい
project_training_1.xpr

Myway
it's a passion way

プロジェクトの起動

- Vivado起動
  - 下記のプロジェクトファイルをクリック
  - project_training_1.xpr

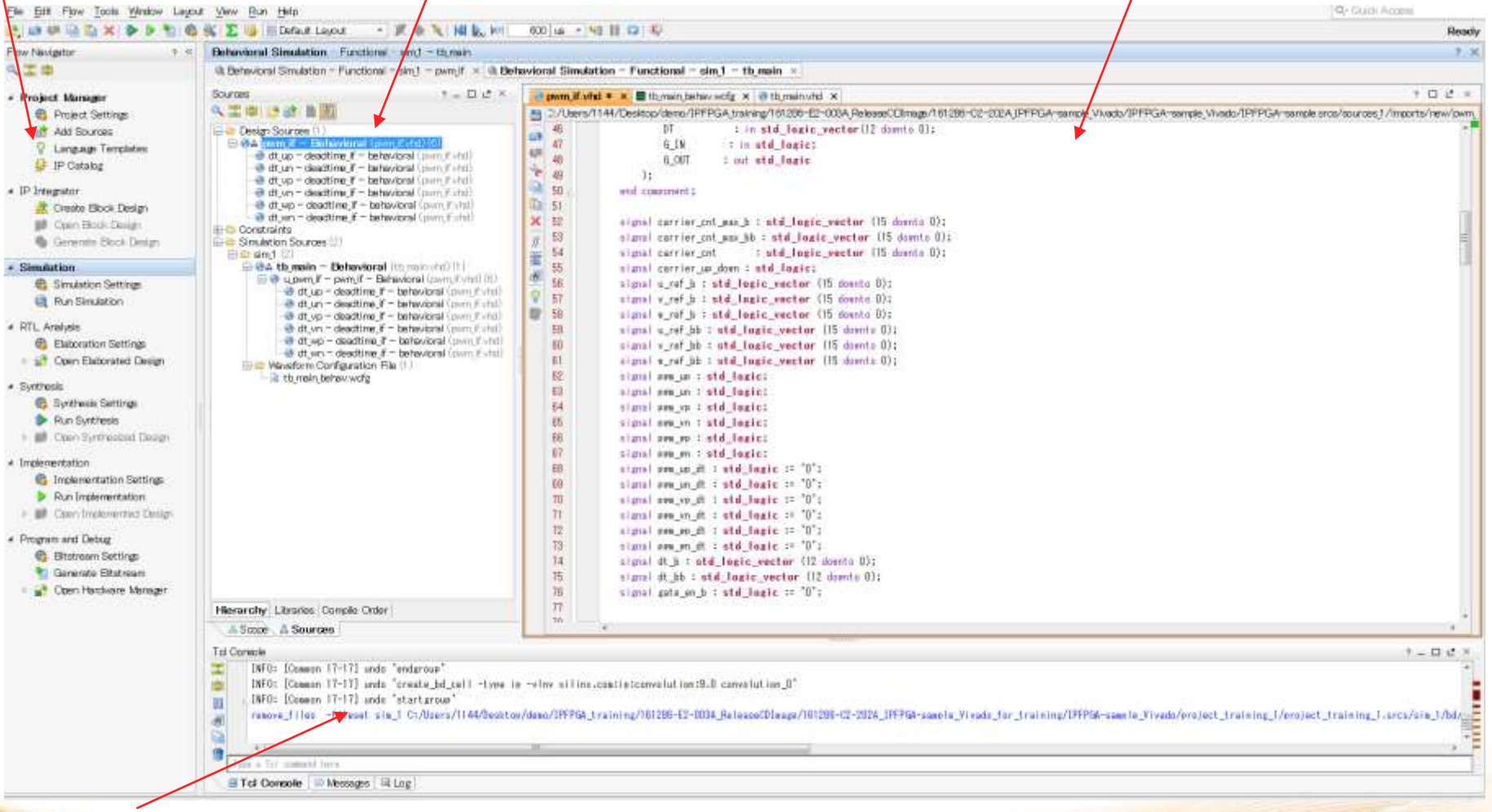
- ソースコードを見るには
  - Project Manager 内の Sources窓のHierachy(デフォルト）画面内で、pwm_ifをクリック

ウィンドウ構成（例）

基本メニュー

ファイル構成

中身（ソースコード・波形等）



エラーメッセージ等

Myway
it's a passion way

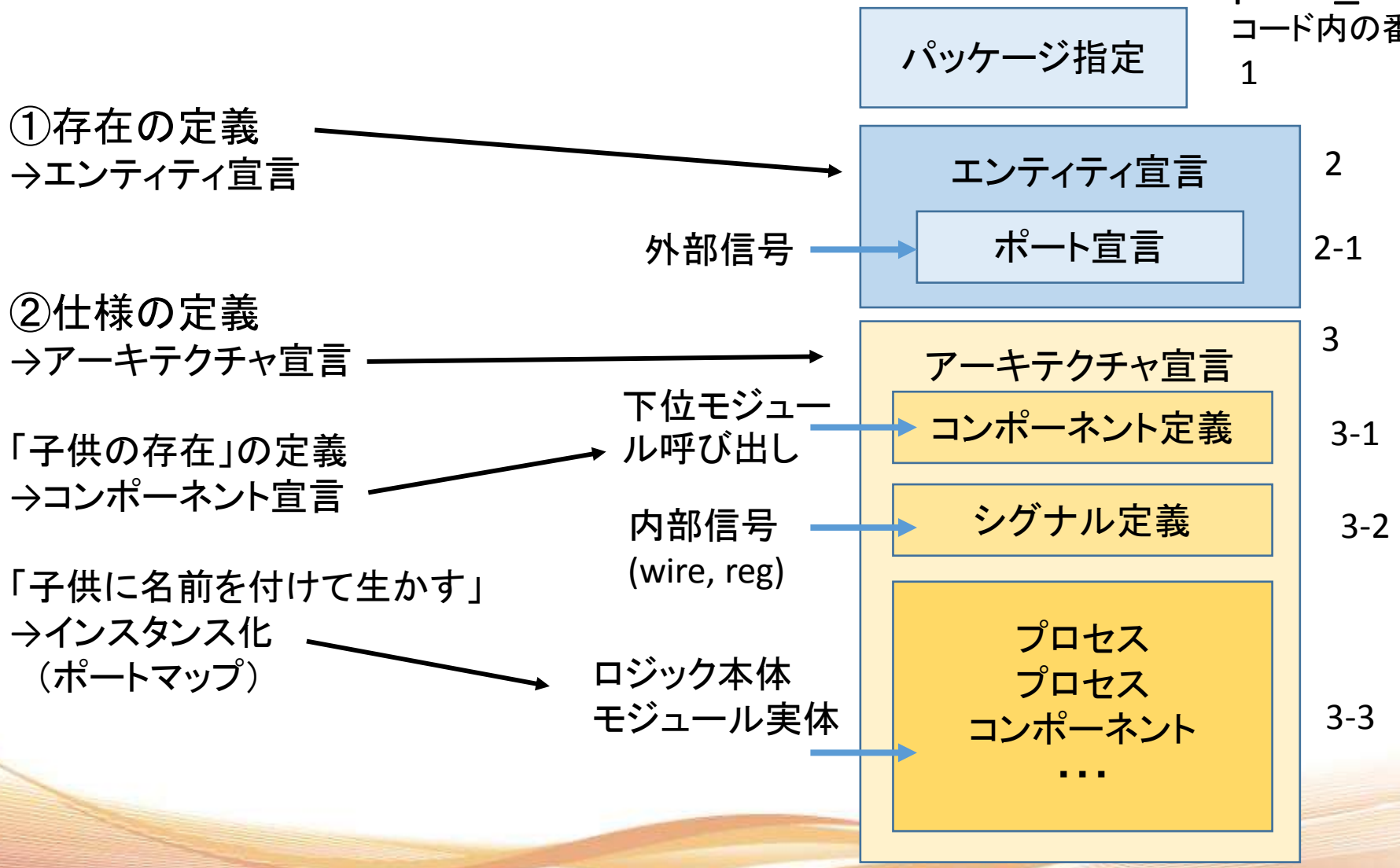
VHDLソースコードを編集する

- Source窓内で”Hierarchy”タブ（デフォルト）を指定してファイルツリーを表示
- ”pwf_if”をダブルクリック
- エディターによるソースコード編集が可能

Myway
it's a passion way

VHDLソースコード構成

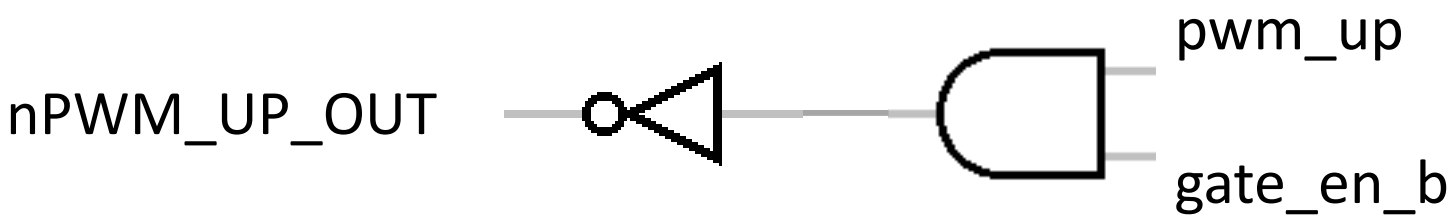
pwm_if.vhd
コード内の番号



Myway
it's a passion way

RTLの基本　組合せ回路

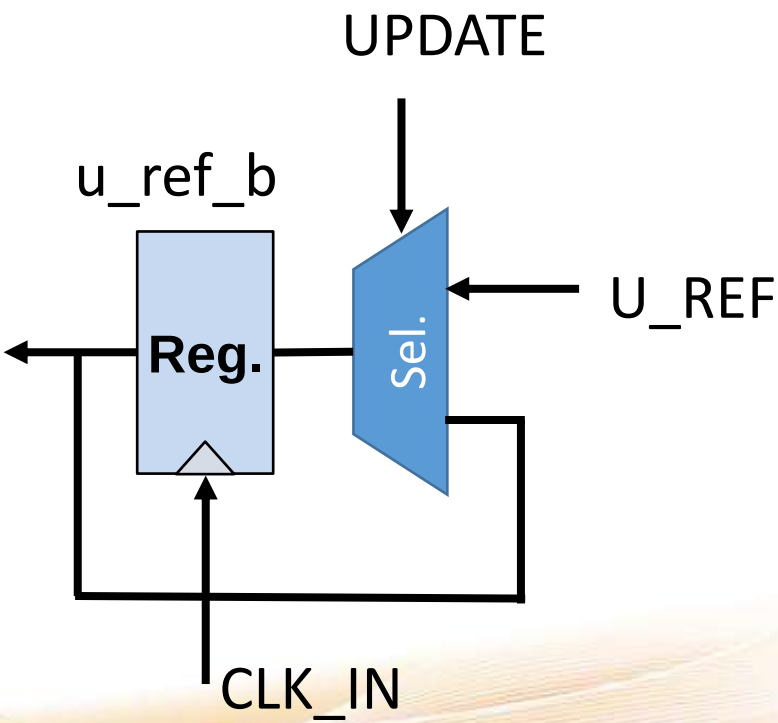
```
-- 3-3-1 Logic description ------------------------------------
   nPWM_UP_OUT <= not (pwm_up and gate_en_b);
   nPWM_UN_OUT <= not (pwm_un and gate_en_b);
```



Myway
it's a passion way

RTLの基本　順序回路

```
-- 3-3-2 Process description ----------------
   process(CLK_IN)
   begin
      if CLK_IN'event and CLK_IN = '1' then
         if RESET_IN = '1' then
            u_ref_b <= X“0000"; -- initial
         elsif UPDATE = '1' then
            u_ref_b <= U_REF;
         else
            u_ref_b <= u_ref_b;
        end if;
```

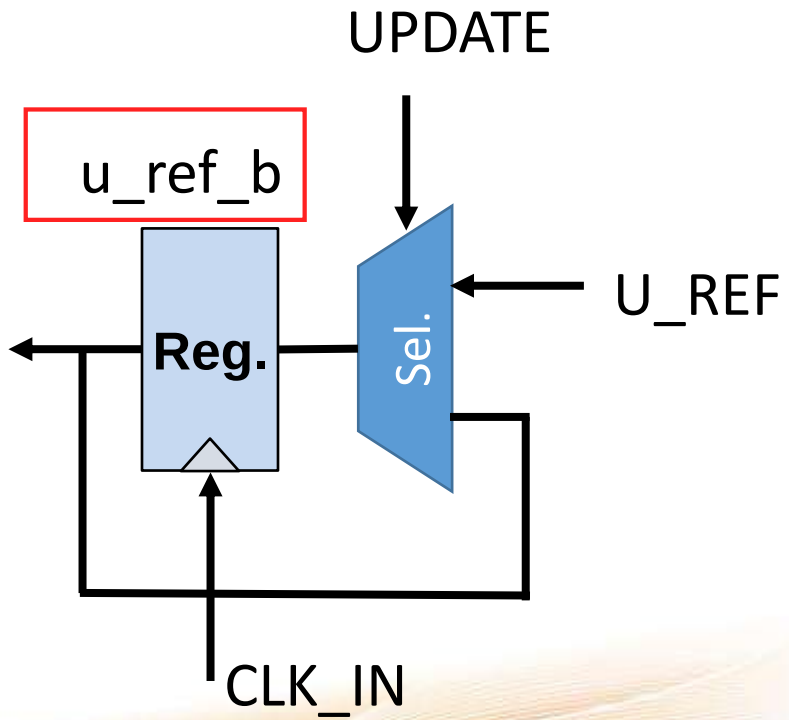


Myway
it's a passion way

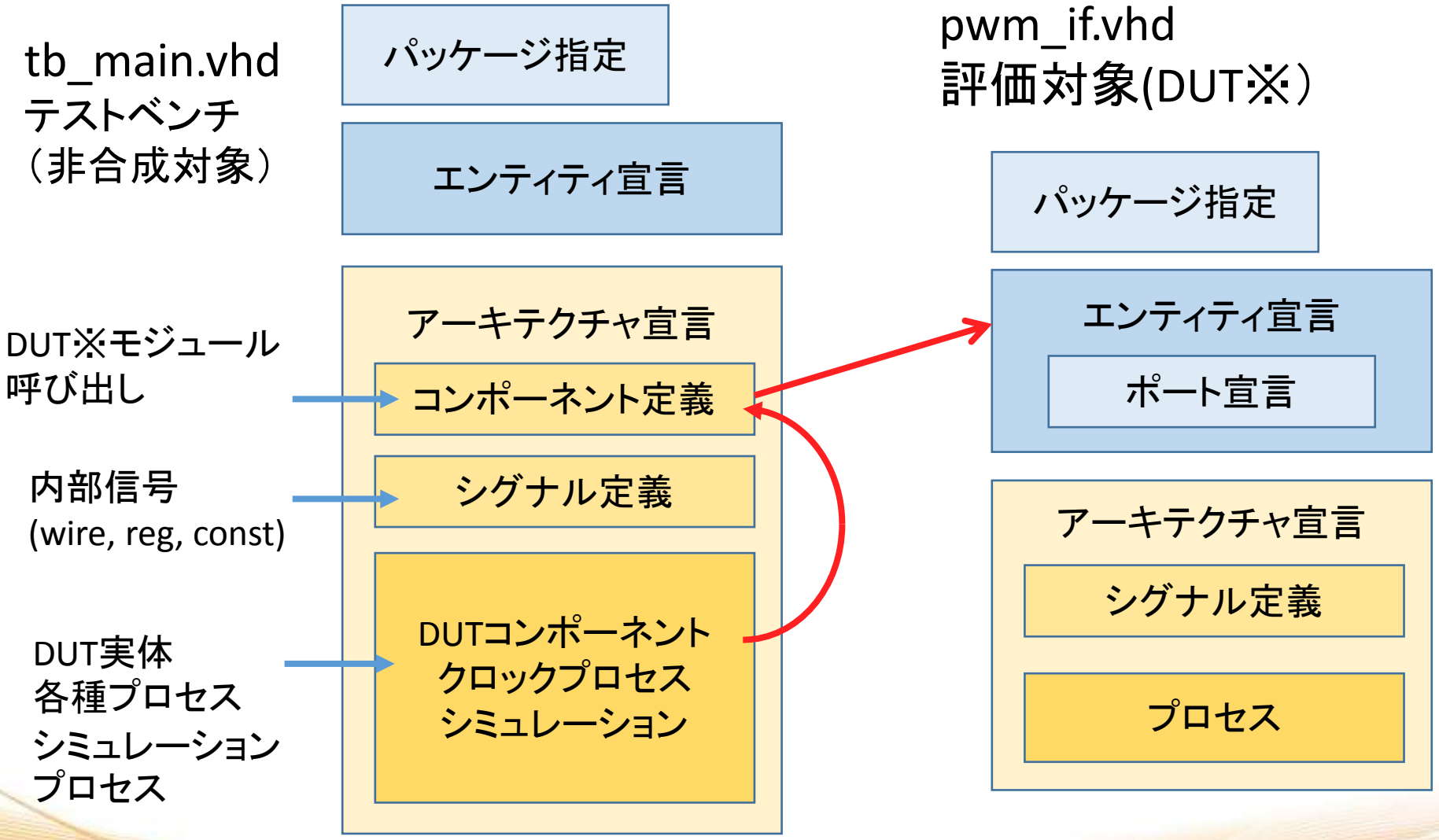
RTLの基本　レジスタとワイヤー

どちらも signal (VHDL ※)　　(verilogでは regまたはwire)

```
process(CLK_IN)
 begin
   if CLK_IN'event and CLK_IN = '1' then
  u_ref_b <= U_REF;
```



VHDLテストベンチ構成



※ DUT: Design Under Test

Myway
it's a passion way

シミュレーションの実行

- 左側のFlow Navigator内のSimulation下のRun Simulationをクリック
- Run Behavioal Simulationをクリック
- 画面上部の▶(T)マーク（時間を300usに指定)を押してさらに実行
- 波形を確認

- 観測したい信号はObjects窓内の信号をドラッグして波形表示窓にドロップ

Myway
it's a passion way

演習２.
PWM生成回路の構成例（シミュレーションとデバッグ作業）

Myway
it's a passion way

演習2

コンパイルエラーおよびシミュレーション結果から、ソースコードの記述ミスを見つけて修正する

1. 三角波生成のカウンタ回路の解説
2. デバッグ①
3. デバッグ②

Myway
it's a passion way

キャリアカウンター（三角波生成用）

```
-- 3-3-2-3 carrier counter increment or decrement ---
1.初期化    if RESET_IN = '1' then
              carrier_cnt <= X"0000";
2.上り      elsif carrier_up_down = '1' then
              carrier_cnt <= carrier_cnt + 1;
3.下り      else
              carrier_cnt <= carrier_cnt - 1;
            end if;
```

Myway
it's a passion way

三角波の山と谷の処理

```
-- 3-3-2-4 reference, carrier count and dead time --------
if RESET_IN = '1' then
（省略）
elsif  carrier_cnt = X"0001" and carrier_up_down = '0' then
     carrier_up_down    <= '1';

elsif carrier_cnt >= (carrier_cnt_max_bb -1) and carrier_up_down = ‘1’
     carrier_up_down    <= '0';
     carrier_cnt_max_bb  <= carrier_cnt_max_b;
end if;
```

谷の時

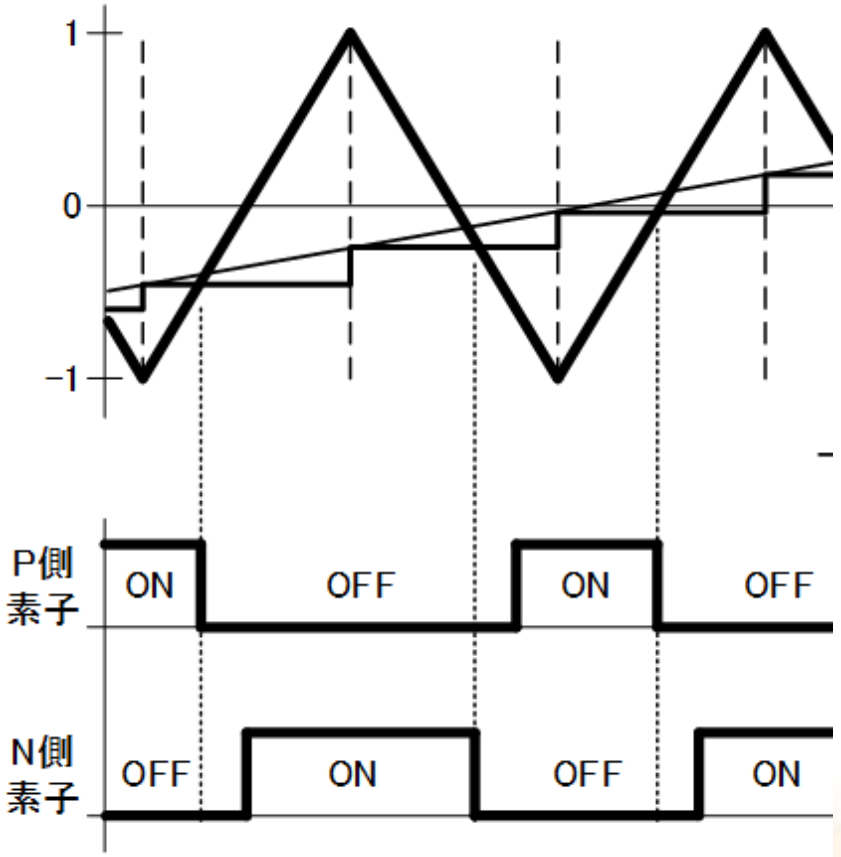
山の時

山・谷以外のときは前状態保持

Myway
it's a passion way

ゲートパルス生成

```
-- 3-3-3 Process description ----------------------------------
if RESET_IN = '1' then
（省略）
 elsif carrier_cnt >= u_ref_bb then
            pwm_up <= '0';
            pwm_un <= '1';
     else
            pwm_up <= '1';
            pwm_un <= '0';
      end if;
```



Myway
it's a passion way

デバッグ①

ゴール：コンパイルエラーから、記述ミスを見つけてコードを修正してください

1. 演習2プロジェクト project_training_2.xprを開いて下さい。
2. Source窓でpwm_ifをダブルクリックしてソースコードを修正
   エディタ右端の赤い部分のエラーメッセージを参考にデバッグ(pwm_if.vhdを修正してfile->saveにて保存する）
3. シミュレーションを実行→エラーが出たらCritical Message -> Open Message View ->Tcl Consoleのメッセージを下から上に見てゆくERRORメッセージを参考に、pwm_if.vhdを修正→save
4. 上記をエラーがなくなるまで繰り返す

Myway
it's a passion way

デバッグ①（ヒント）

- ３つのコンパイルエラー
- 69行、207行、272行
- コメントアウトされている隣の行を参照

Myway
it's a passion way

デバッグ②

ゴール：シミュレーション結果を波形表示し正しい波形となるようにソースコードの修正を行う

1. シミュレーションを実行し、V相のゲート制御波形tb_pwm_vp_n, tb_pwm_vn_nの波形を表示して下さい。
2. U相とV相のpwm波形を確認→異常は？
3. U相と同様に正しいPWMパルスが出るようにpwm_if.vhdを修正して下さい。

Myway
it's a passion way

デバッグ②ヒント

tb_pwm_vp_n, tb_pwm_vn_nの波形を表示して下さい。

（ヒント）Objects窓からtb_pwm_up_nとtb_pwm_un_nをドラッグして波形の窓の信号名が並んでいる領域にドロップ

U相と同様に正しいPWMパルスが出るようにpwm_if.vhdを修正して下さい。

（ヒント）189行

Myway
it's a passion way

演習３　階層化

モジュール呼び出し

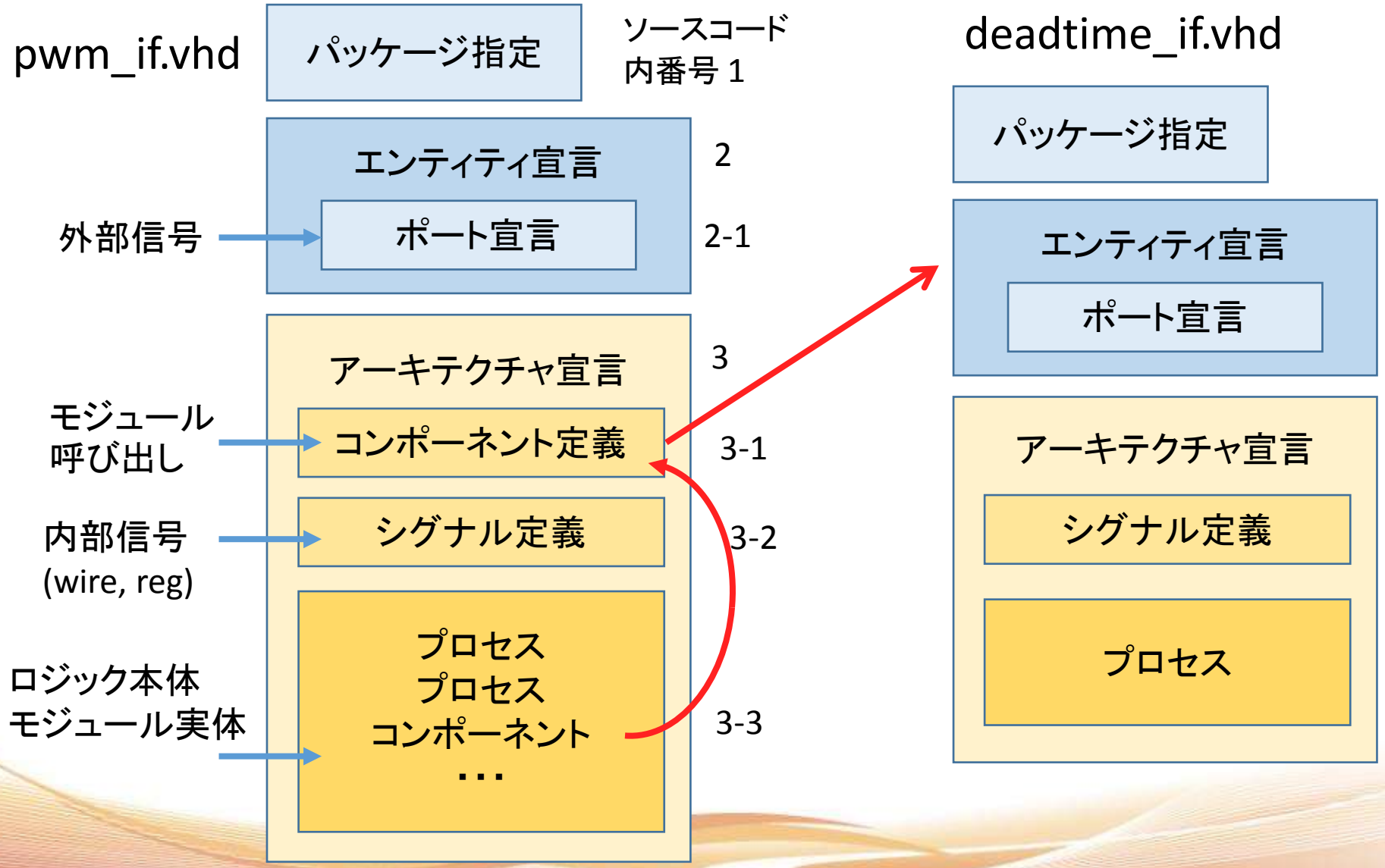
Myway
it's a passion way

演習3　階層化

- VHDLの階層化の理解

1. 階層構造について
2. Deadtime機能の追加

Deadtime生成回路



Myway
it's a passion way

演習　デッドタイムロジック追加

- デバッグ済みの演習２のロジックに、デッドタイムロジックを追加して下さい
- SourceのDesign Sourceのツリー構造の変化を確認してください

→deadtime_ifが下位にインスタンス化されていますか？

- シミュレーションを実行してデッドタイムが挿入されていることを確認

Myway
it's a passion way

演習　ヒント

（ヒント）

- 223行のデッドタイムモジュールの呼び出しを利用（現状は一部コメントアウト、追加でインスタンス記述）
- 96行～101行のゲート出力(nPWM_UP_OUT等）の入力信号をデッドタイム付の信号に変更

Myway
it's a passion way

演習4　モジュールのブロック化

IP作成による効率的な設計

Myway
it's a passion way

モジュールのブロック化

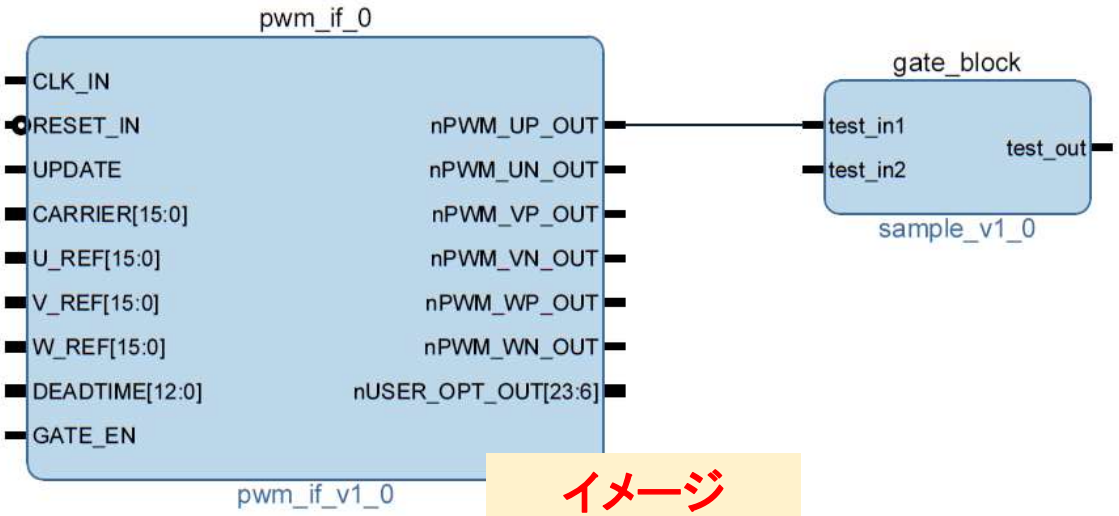
RTLモジュール(pwm_ifおよび下位モジュール）のブロック化を行い、ブロックを用いた回路設計とシミュレーションを行います

1. サブモジュールで記述されたpwm_if.vhdの確認
   Triangle_count.vhd, pwm_gen.vhd, deadtime_if.vhd
2. ブロック設計によるIP化
3. シミュレーションの実行

ブロック図設計（IP化・IP活用）

直感的な理解・設計に有利

ブロック図記述 (.bd)



イメージ

テキスト情報による管理に有利

VHDLコード (.vhd)

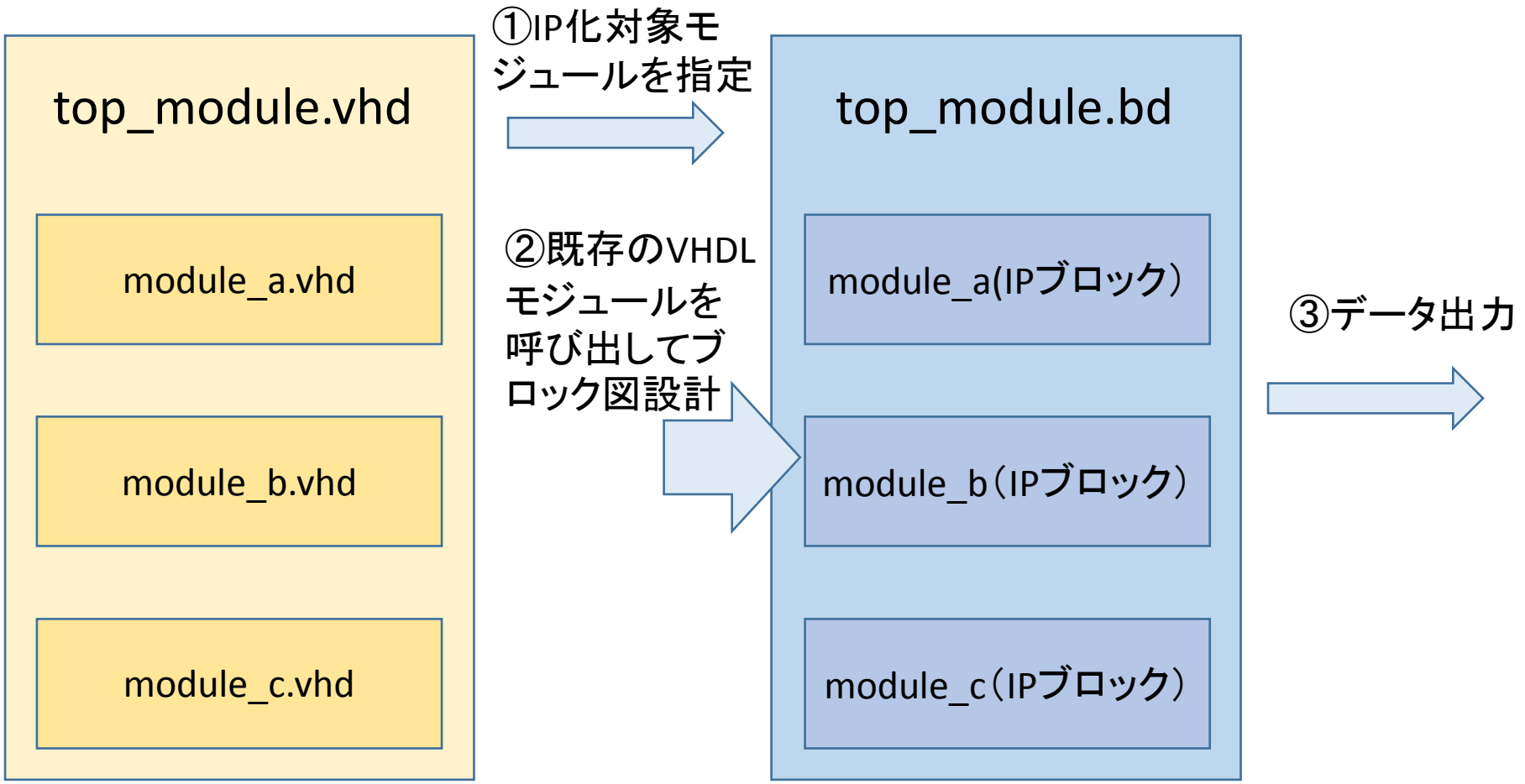
```
entity pwm_if is
port (
CLK_IN          : in std_logic;
RESET_IN        : in std_logic;
nPWM_UP_OUT     : out std_logic; --nUSER_OPT_OUT(0)
nPWM_UN_OUT     : out std_logic; --nUSER_OPT_OUT(1)
nPWM_VP_OUT     : out std_logic; --nUSER_OPT_OUT(2)
nPWM_VN_OUT     : out std_logic; --nUSER_OPT_OUT(3)
nPWM_WP_OUT     : out std_logic; --nUSER_OPT_OUT(4)
nPWM_WN_OUT     : out std_logic; --nUSER_OPT_OUT(5)
nUSER_OPT_OUT : out std_logic_vector (23 downto 6);
UPDATE    : in std_logic;
CARRIER   : in std_logic_vector (15 downto 0);
U_REF     : in std_logic_vector (15 downto 0);
V_REF     : in std_logic_vector (15 downto 0);
W_REF     : in std_logic_vector (15 downto 0);
DEADTIME : in std_logic_vector (12 downto 0);
GATE_EN  : in std_logic
);
end pwm_if;
```

イメージ

IPインテグレータによりVHDLをブロック化

Myway
it's a passion way

VHDL・ブロックのデータ構成



IPブロックデータを生成するとラッパー等既存のRTLコードとの整合を保つデータも生成され、既存の環境でシミュレーション・コンパイルが可能です

Myway
it's a passion way

ブロック設計IP化実習(1/4)

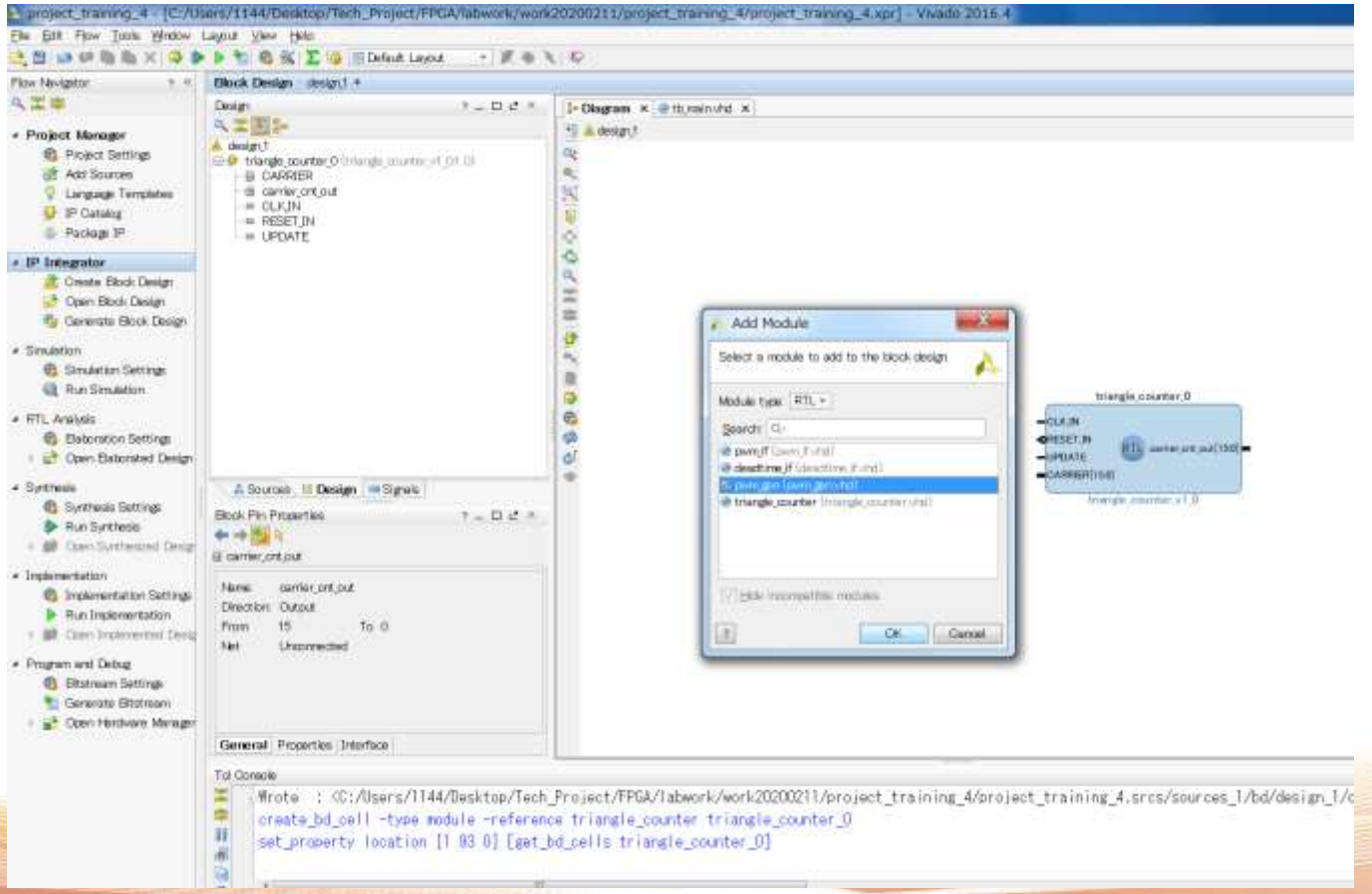
0. 下位がモジュール化されたpwm_if.vhdコードを確認
1. (上部タブ）Tools→Create and Package New IP
   Package your current project（一番上）→ Next
   include xci files（一番上）→Next (適時 overwrite, OK, Finish)
2. 続いて表示されるPackaging Stepsの画面にて
   Review and Package（一番下）を選択
   Review and Package窓の中央一番下
   →Package IPまたは(Re-Package IP) ボタン →OK
3. Flow Navigator（一番左）
   IP Integrator → Create Block Design → そのままOK

Myway
it's a passion way

ブロック設計IP化実習(2/4)

“design_1”のデザイン画面で右クリック→add module

Triangle_counter, pwm_if, deadtime_if (6個）を追加



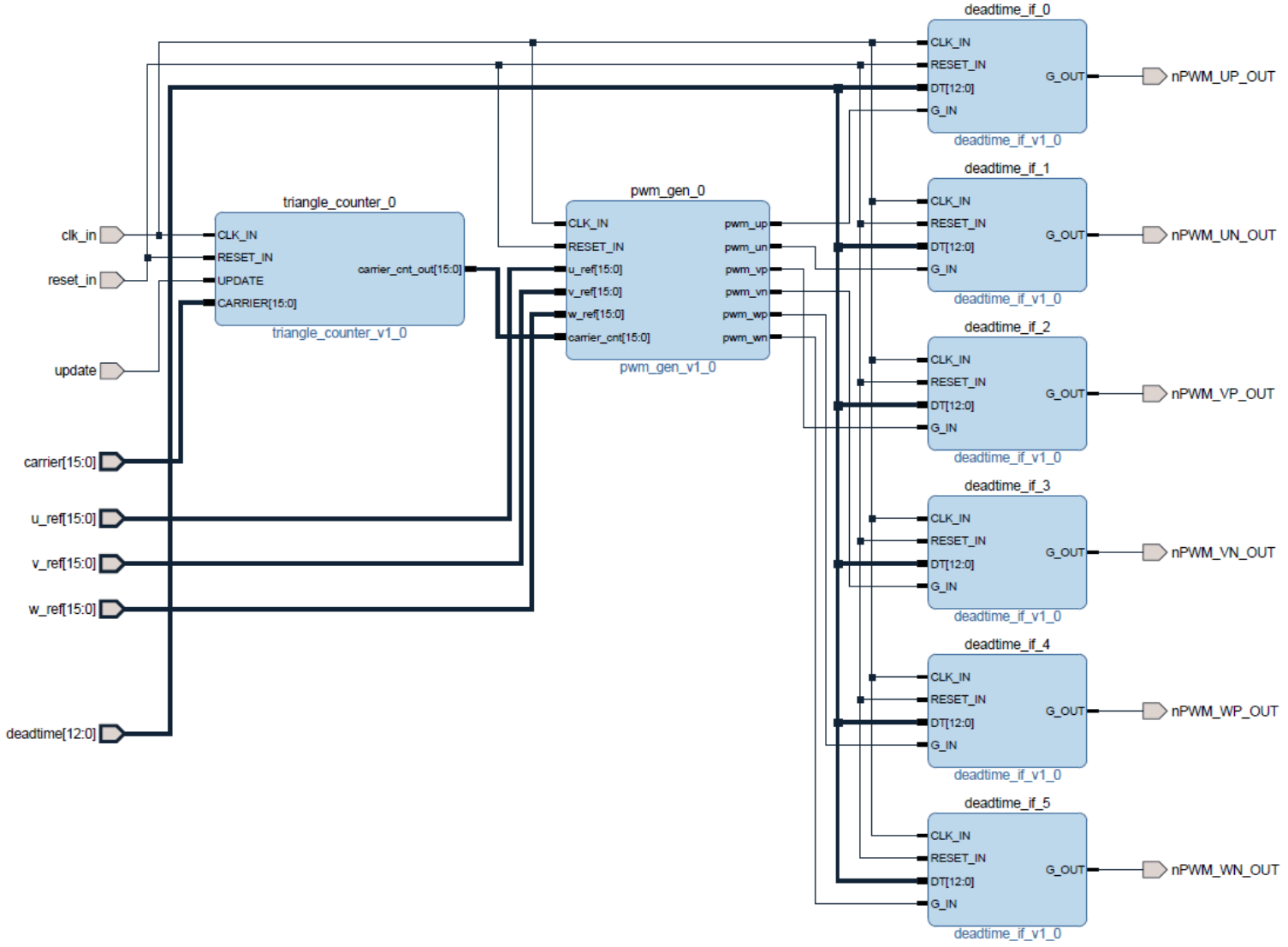
Myway
it's a passion way

ブロック設計IP化実習(3/4)

- ポート入力
  - 右クリックCreat Port
    →クロック(100MHz)、リセット定義、その他はOthers
    →Input, Output, バス設定についても入力
- 配線
  - マウスで選択配線
  - ポートを選択して右クリック make connectionで複数ポートは一度で配線

Myway
it's a passion way

pwm_if.vhdのブロック設計完了



Myway
it's a passion way

ブロック設計IP化実習(4/4)

1. (Flow Navigator) IP Integrator →Generate Block Design
   IPブロックのデータ出力
2. (Flow Navigator) Project Manager →Package IP
   再利用のためIP Catalogへの登録　→再利用が可能

Myway
it's a passion way

シミュレーションの実行

- 最上位のテストベンチ(tb_main.vhd)の呼び出しモジュール名に注意
  - Component宣言
    - （前回）　component　pwm_if
    - （ブロック図）　component　design_1
  - インスタンス化、ポートマップ宣言
    - （前回）　u_pwm_if : pwm_if
    - （ブロック）　u_pwm_if : design_1
- 同様の結果が得られますか？

Myway
it's a passion way

演習５　ADインターフェースの設計

順序回路によるシリアルデバイスとの通信

Myway
it's a passion way

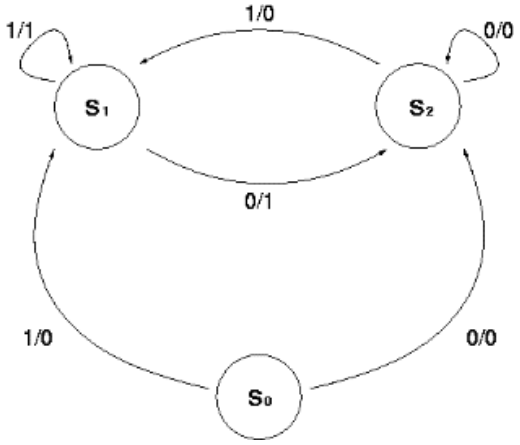
ADインターフェースの設計

基本的なステートマシーンの構造を理解して、ADインターフェース回路のシミュレーションを実行する

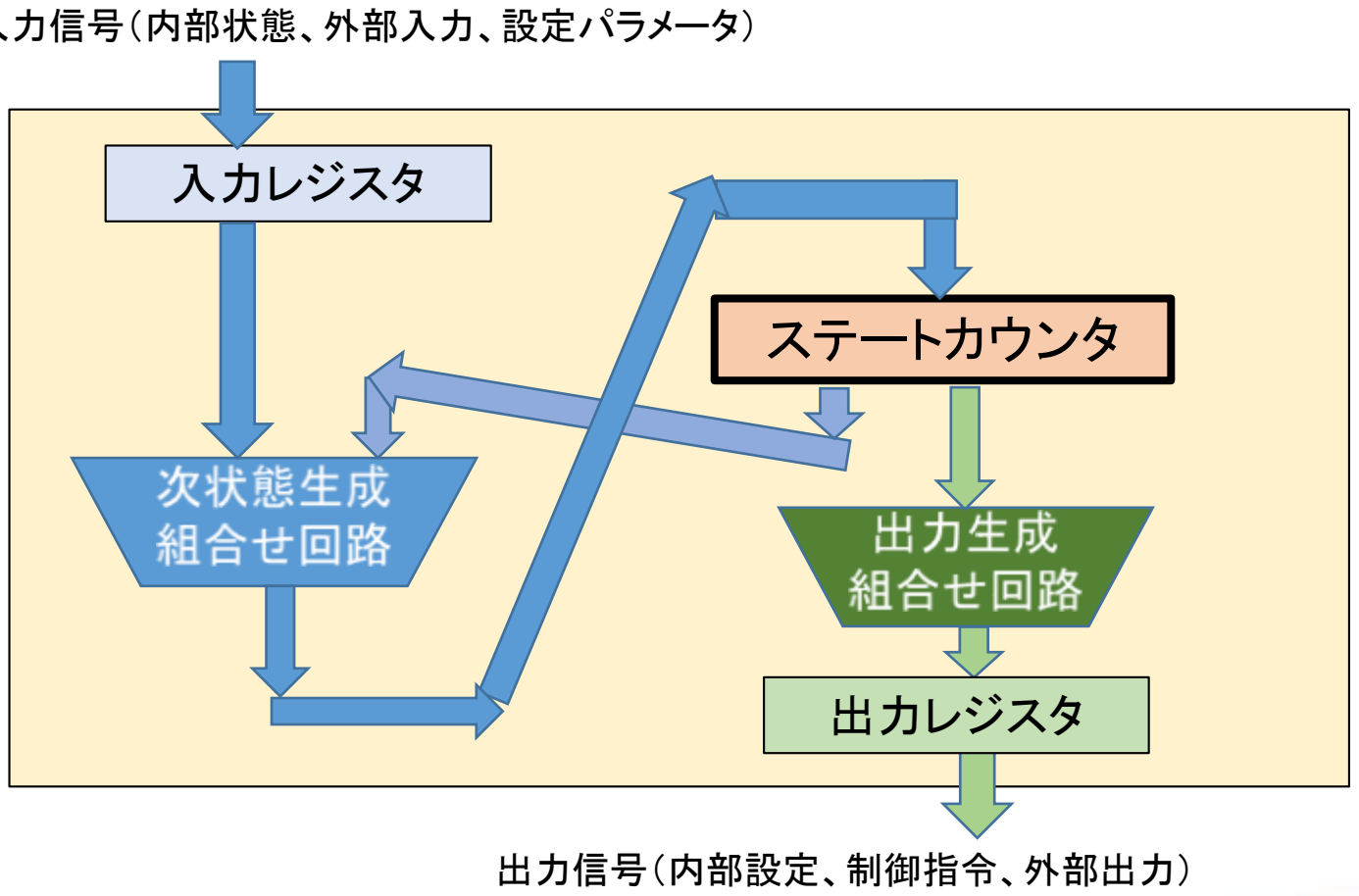
1. ステートマシーンの構造
2. ADインターフェースの仕様
3. ステートマシーンのデバッグ

Myway
it's a passion way

ステートマシーン部

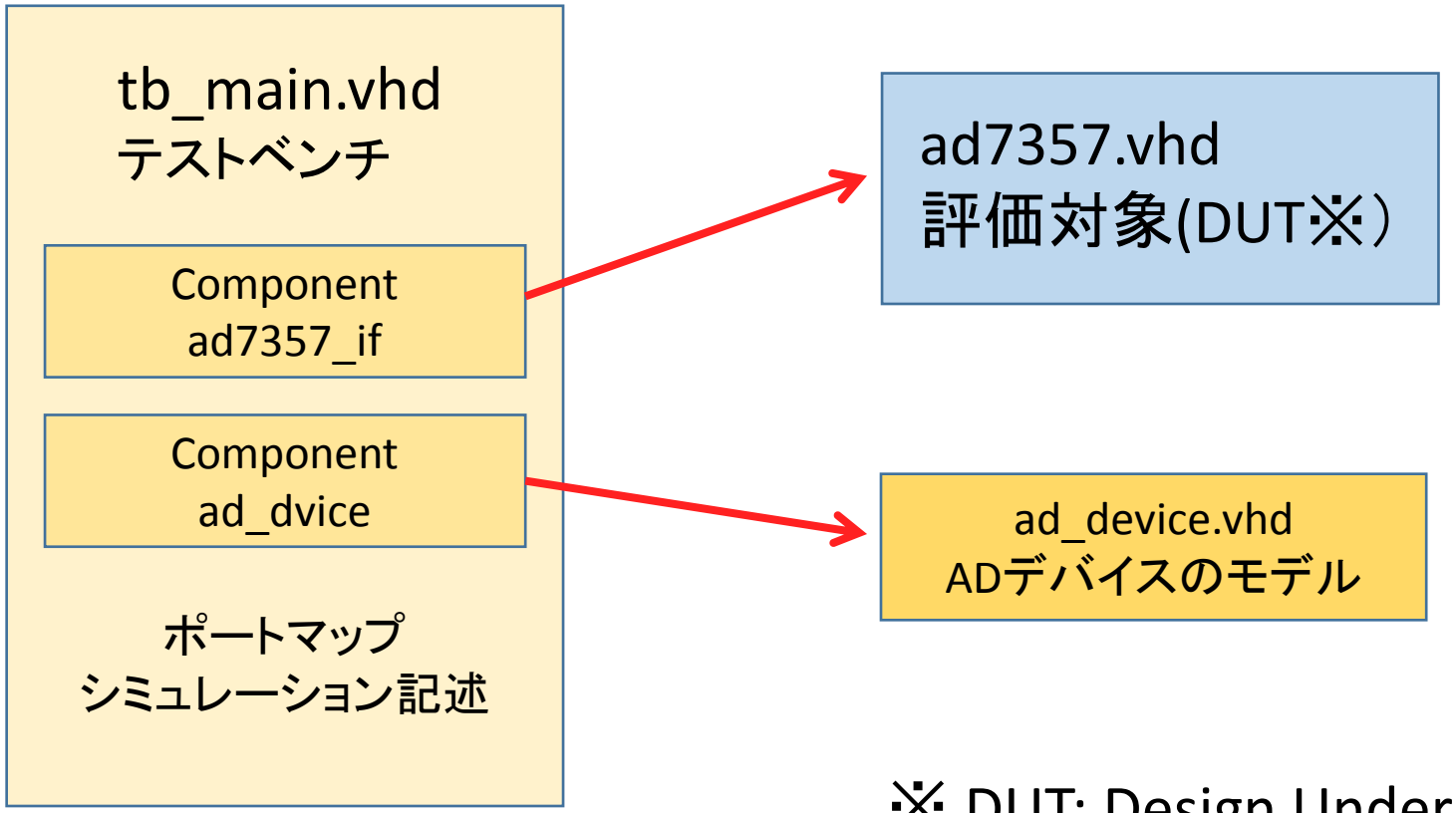


ステートマシーン
（例）
（状態遷移図）



原則ムーアマシン（入出力の間にステートカウンタを経由する）を使用
一方、ミーリマシンはステートカウンタを介さずに入力が直接出力に影響する

VHDLテストベンチ構成



※ DUT: Design Under Test

Myway
it's a passion way

ADインターフェース仕様

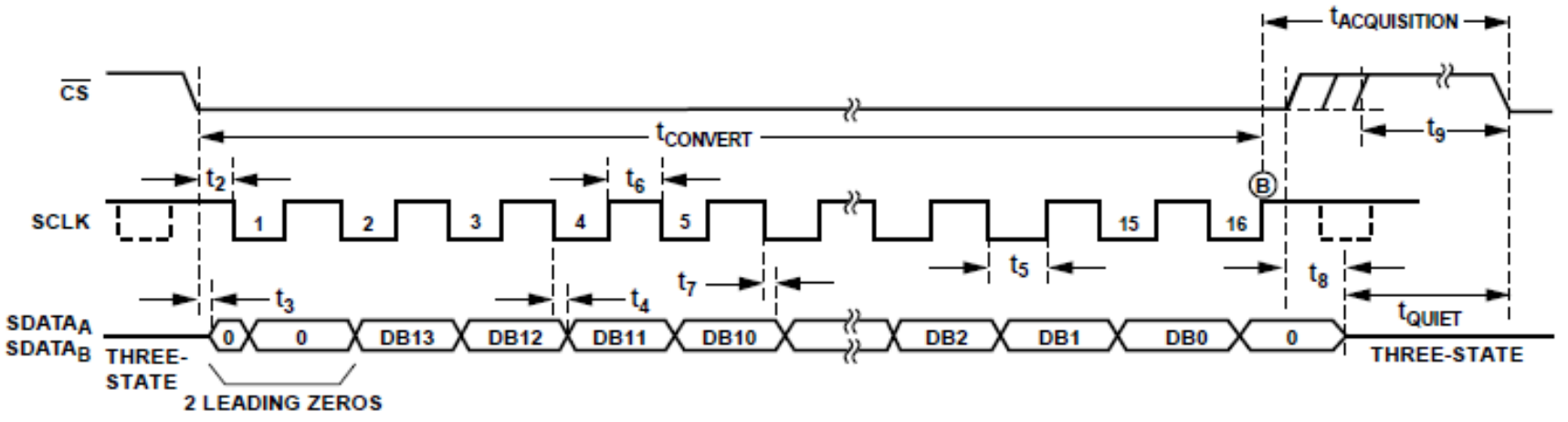


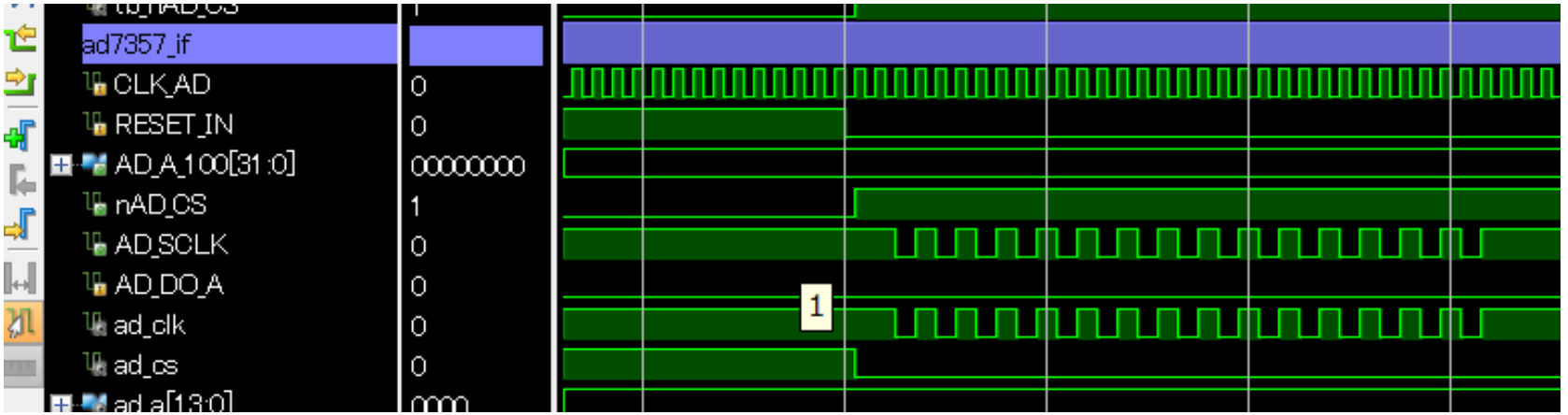
Figure 31. Serial Interface Timing Diagram

7357データシートより

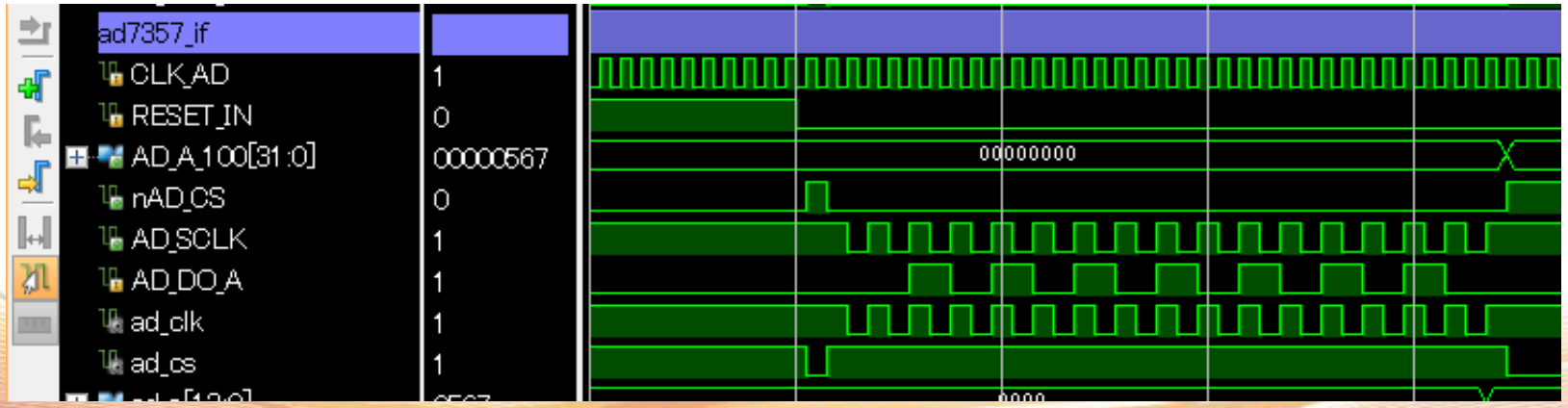
Myway
it's a passion way

演習　ステートマシーンのデバッグ

ADインターフェース回路のシミュレーションを実行してステートマシーンロジックのデバッグを行います



現在の波形



正しい波形

Myway
it's a passion way

ヒント

- CSは正しく生成されていますか？(59,60行）
- ad_clk(SCLK)の個数は正しい（１６）ですか？(104,106行）