



SoC FPGA
パケット生成論理 導入手順書

【ご注意】

- (1) 本書の内容の一部または、全部を無断転載することは禁止されています。
- (2) 本書の内容については、改良のため予告なしに変更することがあります。
- (3) 本書の内容について、ご不明な点やお気付きの点がありましたら、ご連絡ください。
- (4) 本製品を運用した結果の影響については、(3)項にかかわらず責任を負いかねますのでご了承ください。

本マニュアルに記載されている企業名、システム名、製品名は、各社の商標または登録商標です。
なお、本文中では、TM、R マークは明記していません。

©2013 DTS INSIGHT CORPORATION. All rights reserved

Printed in Japan

改訂履歴

版	発行日付	変更内容
第1版	2013.06.12	新規発行
第2版	2013.12.25	TRQerS 対応

目次

1	はじめに.....	1
2	パッケージ生成論理組み込み手順.....	2
2.1	既存プロジェクトへのパッケージ生成論理組み込み準備.....	2
2.2	Qsys でのパッケージ生成論理組み込み手順.....	3
2.3	TOP モジュールの編集.....	18
2.4	Quartus II コンパイル手順.....	21

1 はじめに

本マニュアルでは、adviceLUNA/TRQerS システムマクロトレース（以下 SMT）のパケットインターフェースに対応するためのパケット生成論理について、SoC FPGA への導入手順を説明します。

また、本マニュアルでは、アルティマ社製の Cyclon V SoC 評価ボード” Helio” を例にしています。

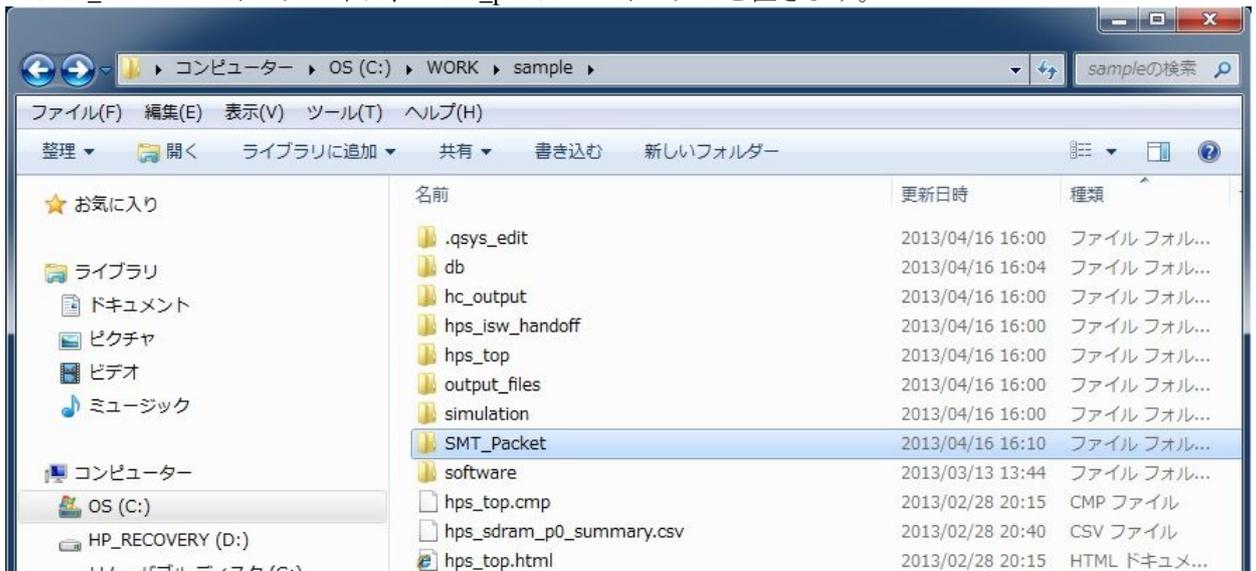
2 パケット生成論理組み込み手順

2.1 既存プロジェクトへのパケット生成論理組み込み準備

- 1) SoC FPGA 用パケット生成論理一式を用意します。

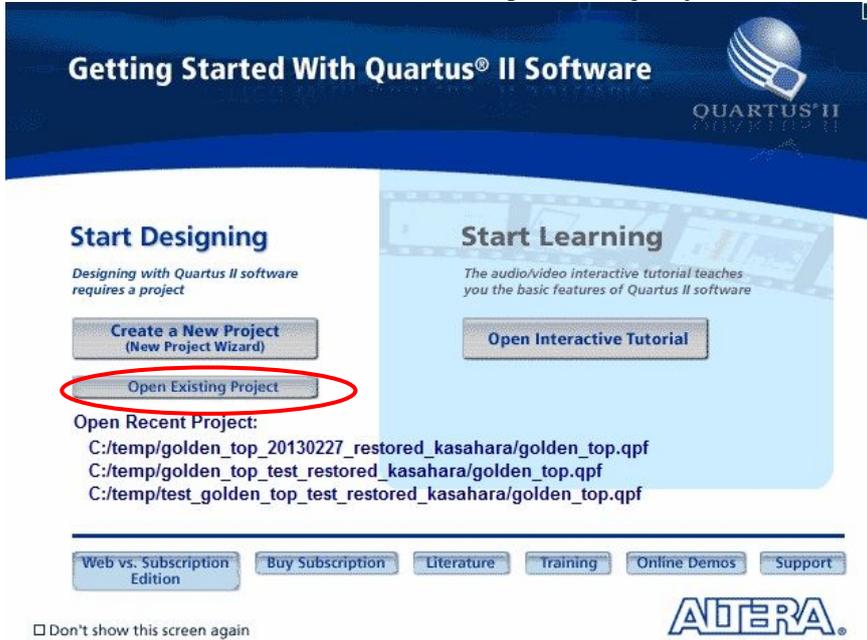


- 2) Quartus II の既存のプロジェクトがあるフォルダに、新規フォルダ “SMT_Packet” を生成し、“SMT_Packet” フォルダの中に、“smt_port32” フォルダごと置きます。

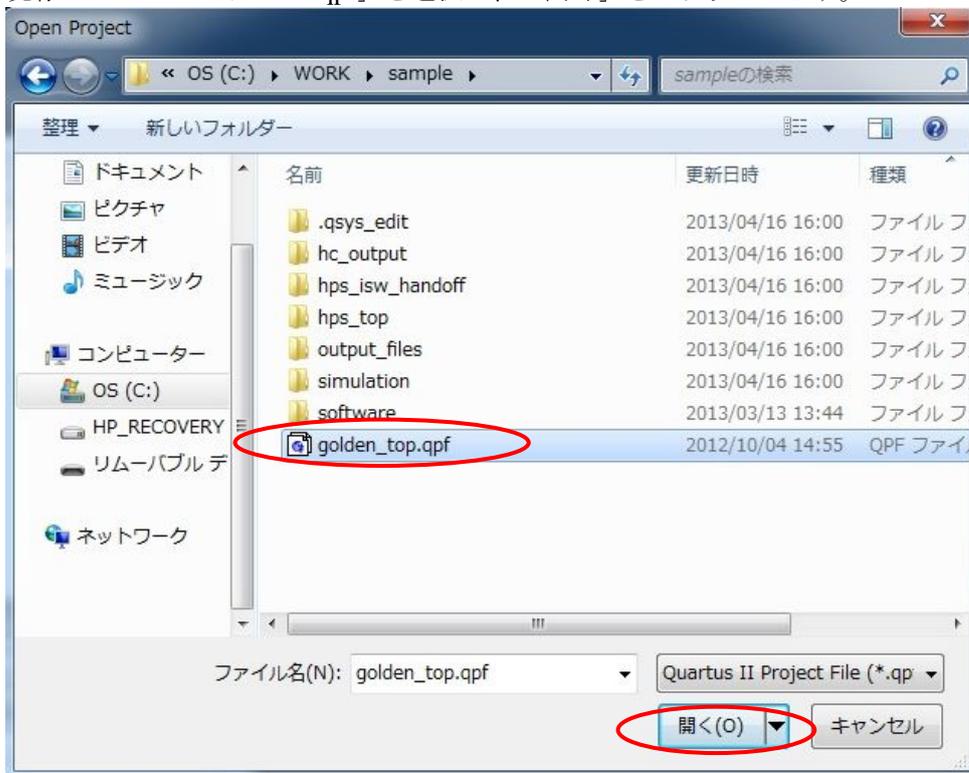


2.2 Qsys でのパケット生成論理組み込み手順

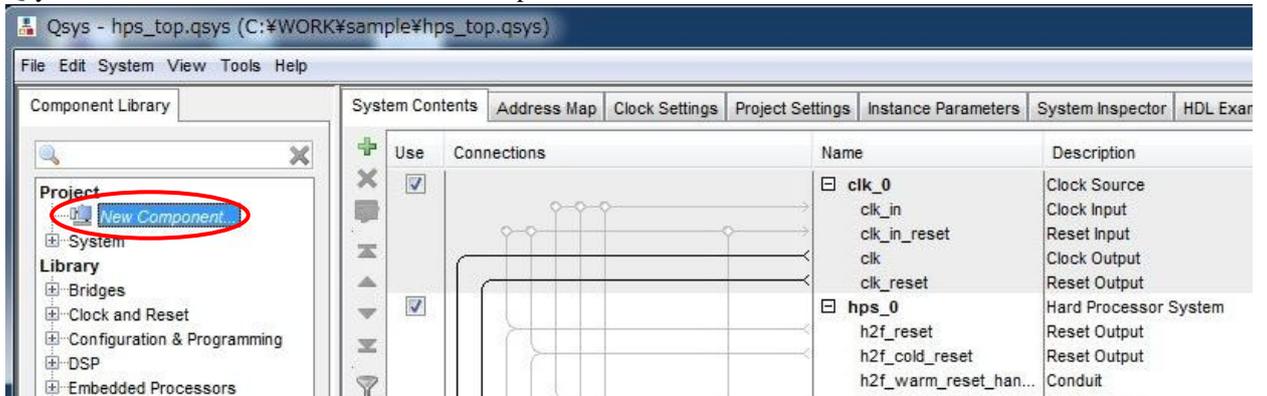
- 1) 既存のプロジェクトを立ち上げます。「Open Existing Project」をクリックします。



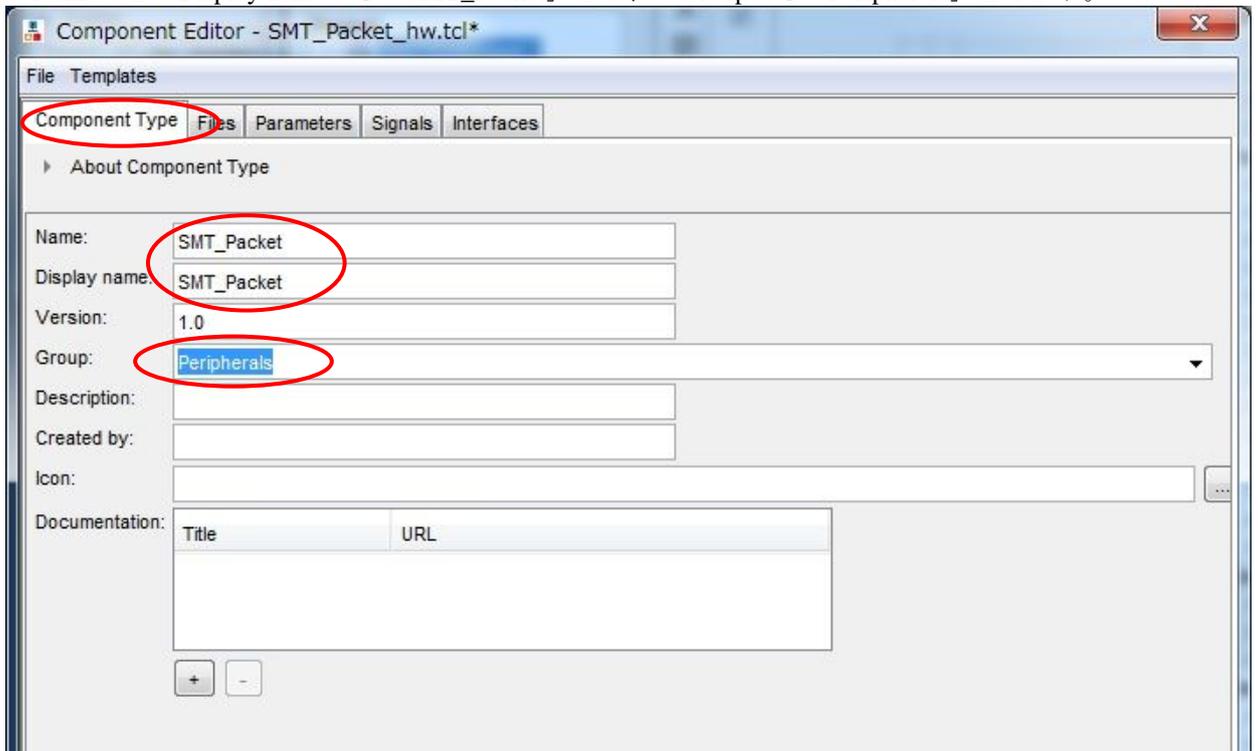
- 2) 既存のプロジェクト「xxx.qpf」を選択し、「開く」をクリックします。



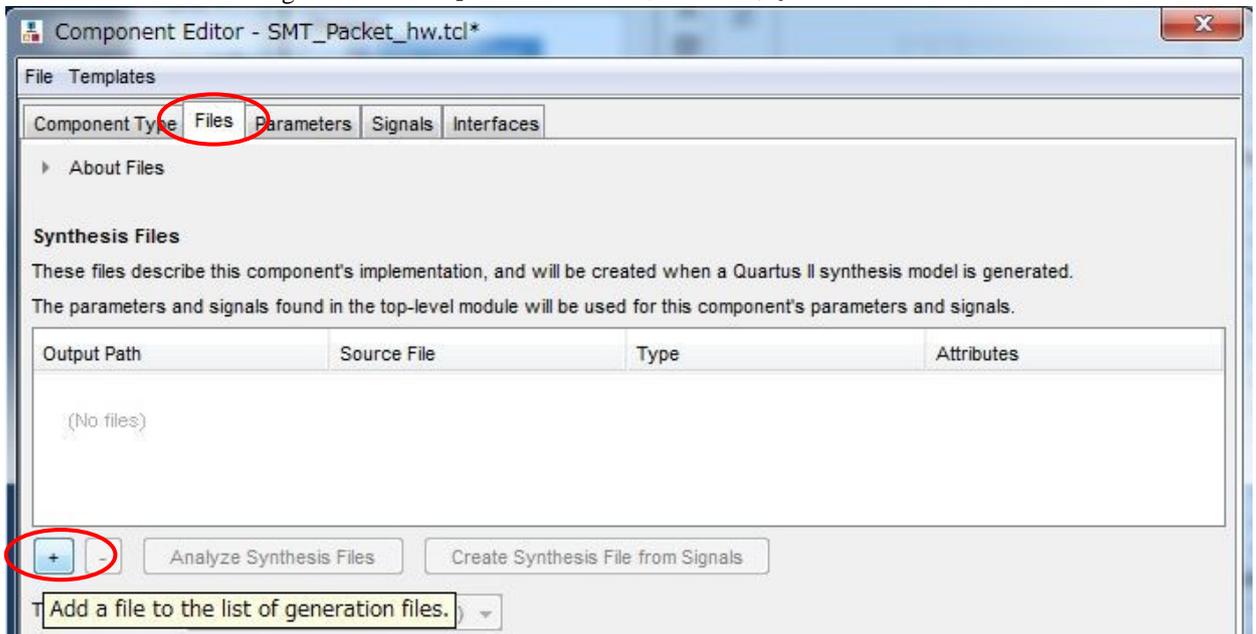
- 3) Qsys が立ち上がりますので、「New Component…」をダブルクリックします。



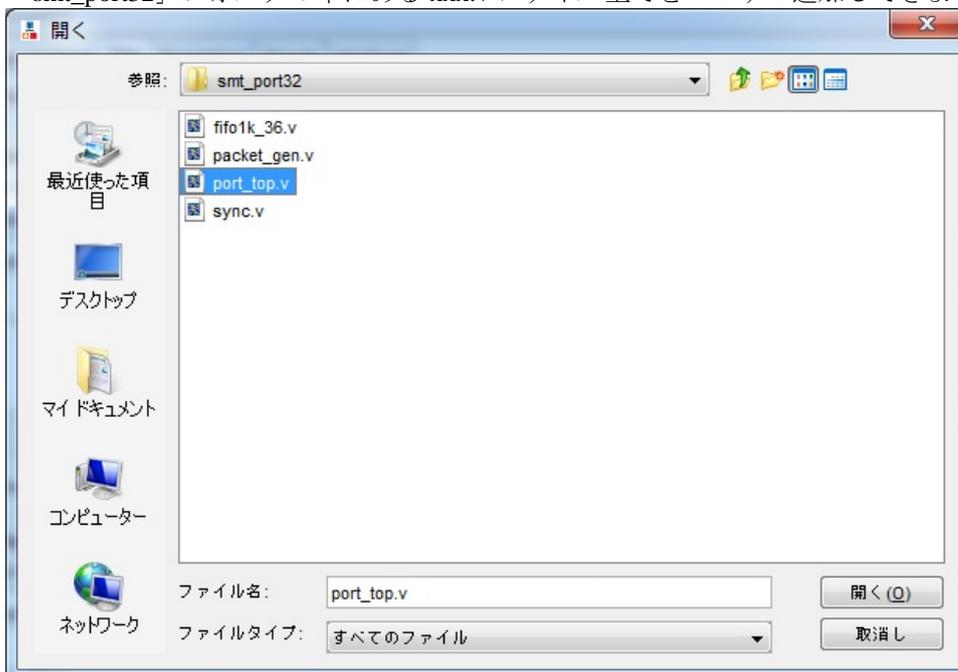
- 4) Component Editor が立ち上がりますので、「Component Type」タブの設定から開始します。
「Name」と「Display name」を「SMT_Packet」とし、「Group」を「Peripherals」とします。



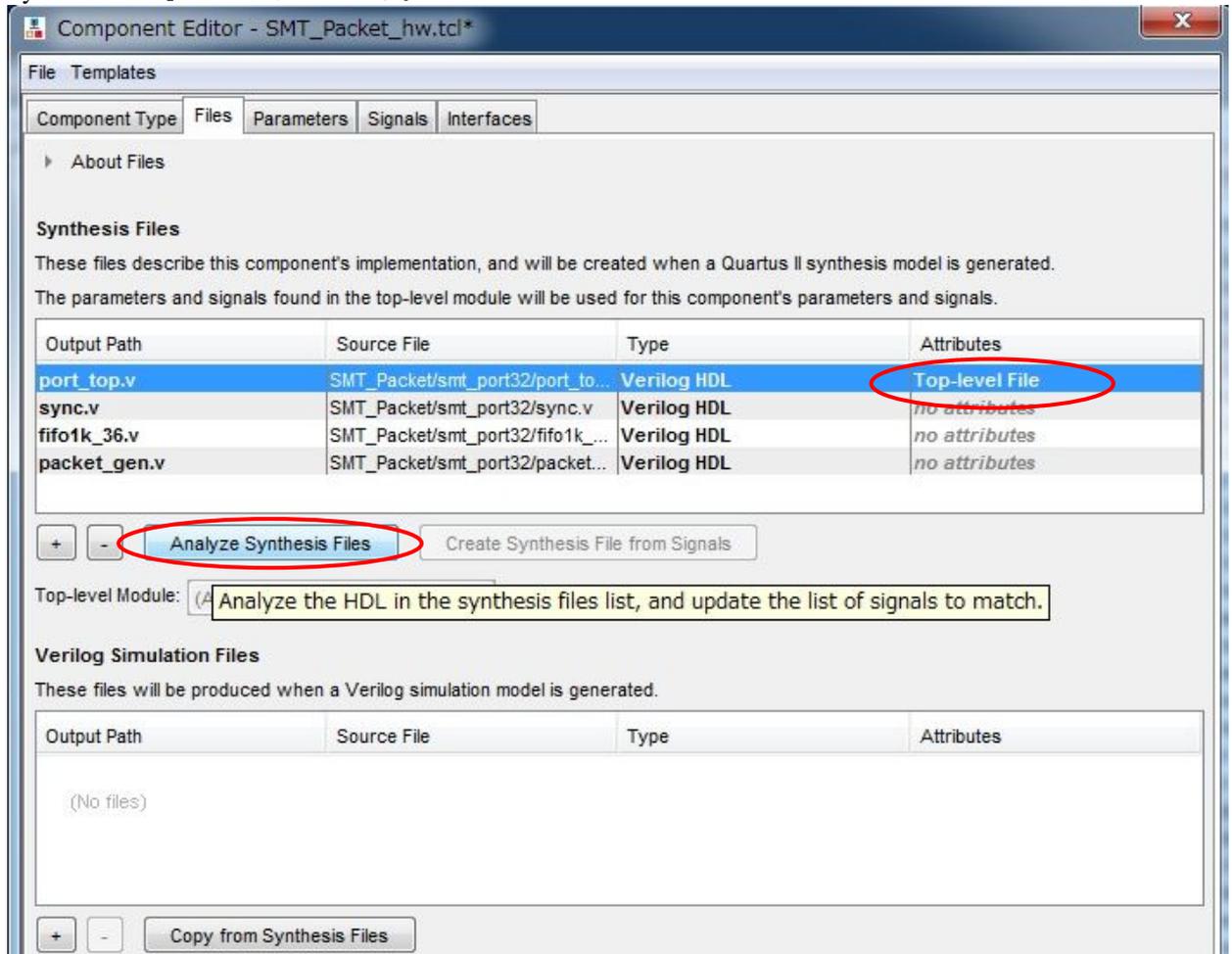
- 5) 「Files」タブでは SMT パケット論理を追加していきます。
「Add a file to the list of generation files.」 ボタンをクリックします。



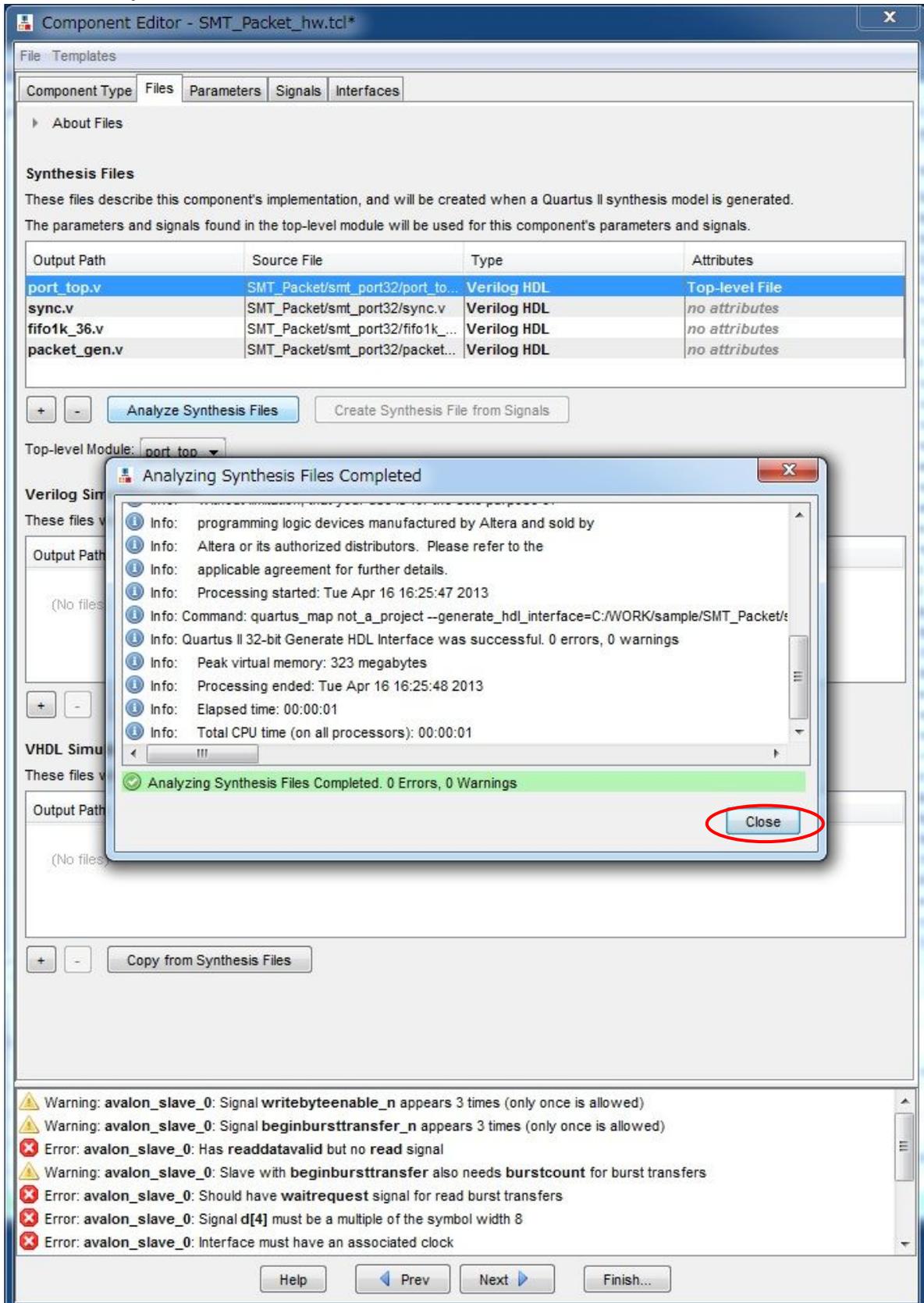
- 6) 「smt_port32」フォルダの中にある xxx.v ファイル全てを一つずつ追加していきます。



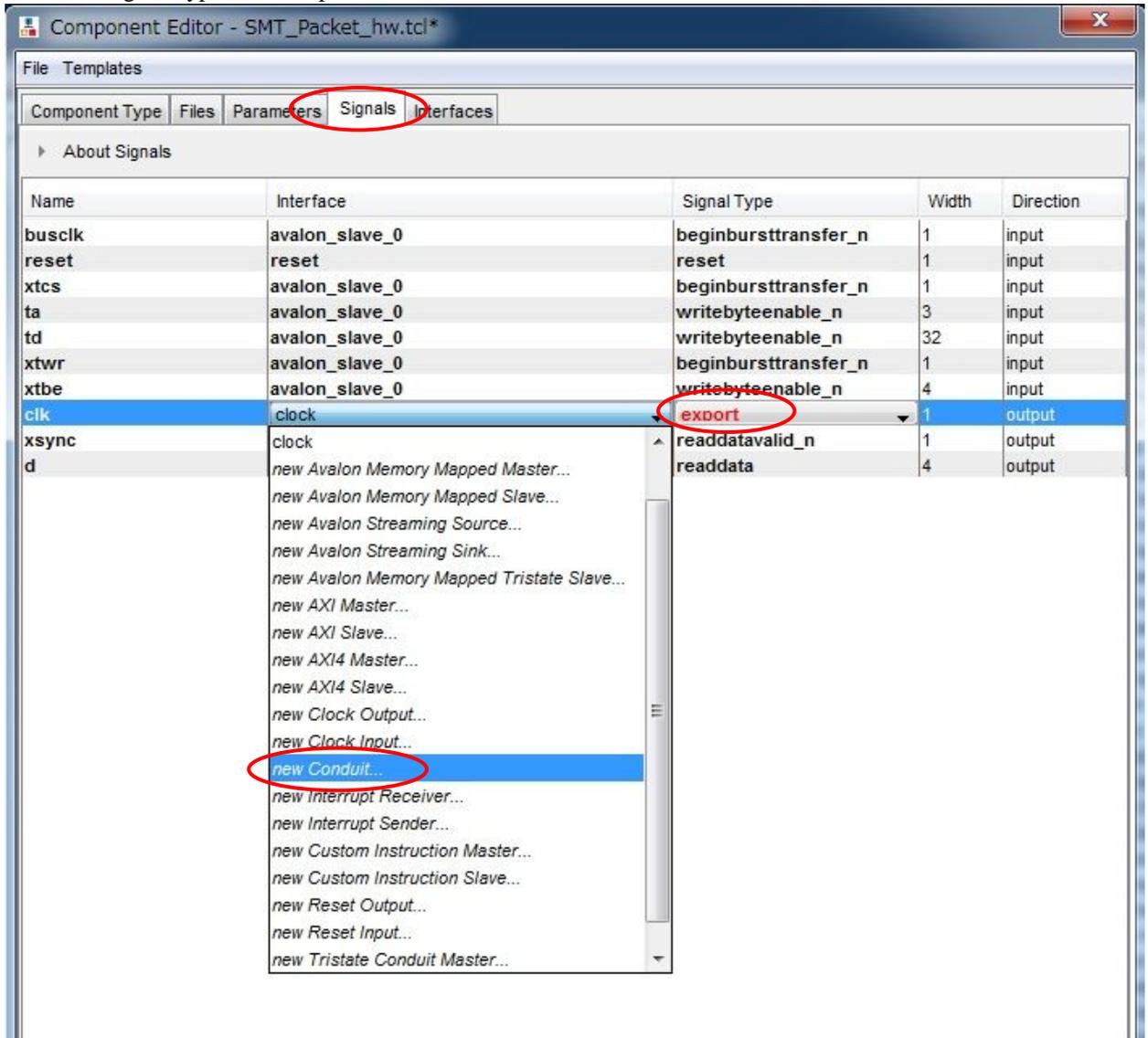
- 7) 全て登録が完了しましたら「port_top.v」が「Top-level Files」になっていることを確認し、「Analyze Synthesis Files」をクリックします。



- 8) 「Analyze Synthesis Files」が完了し、以下のダイアログが出てくるので「Close」をクリックします。ここで、Error が多数出ていますが、後の設定でこの Error は消えていきますので、ここでは気にしないでください。



- 9) 「Signals」タブにて、各信号の定義を行います。まずは、「clk」の”Interface”で「new Conduit」を選択し、”Signal Type”を「export」に設定します。



- 10) SMT の全信号に対し、以下の様に設定してください。

Name	Interface	Signal Type	Width	Direction
busclk	clock	clk	1	input
reset	reset	reset	1	input
xtcs	avalon_slave_0	chipselect_n	1	input
ta	avalon_slave_0	address	3	input
td	avalon_slave_0	writedata	32	input
xtwr	avalon_slave_0	write_n	1	input
xtbe	avalon_slave_0	byteenable_n	4	input
clk	conduit_end	export	1	output
xsync	conduit_end	export	1	output
d	conduit_end	export	4	output

【注意】 xtcs, xtwr, xtbe は、Low アクティブ信号であるため、Signal Type は「_n」付きの信号を選択してください。

11) 「Interface」タブでは、以下の設定を行います。

「avalon_slave_0」のバス設定にて、“Associated Clock”を「clock」とし、“Associated Reset”は「reset」とします。「Timing」では、“Hold”を「1」clockとし、それ以外は「0」とします。

The screenshot displays the Component Editor interface for the 'avalon_slave_0' component. The 'Interfaces' tab is active, showing the following settings:

- Name: avalon_slave_0
- Type: Avalon Memory Mapped Slave
- Associated Clock: clock
- Associated Reset: reset

The 'Block Diagram' shows the component's connections:

- xtcs to chipselect_n
- ta[2..0] to address
- td[31..0] to writedata
- xtwr to write_n
- xtbe[3..0] to byteenable_n

The 'Parameters' section shows:

- Address units: WORDS
- Associated clock: clock
- Associated reset: reset
- Bits per symbol: 8
- Burstcount units: WORDS
- Explicit address span: 0000000000000000

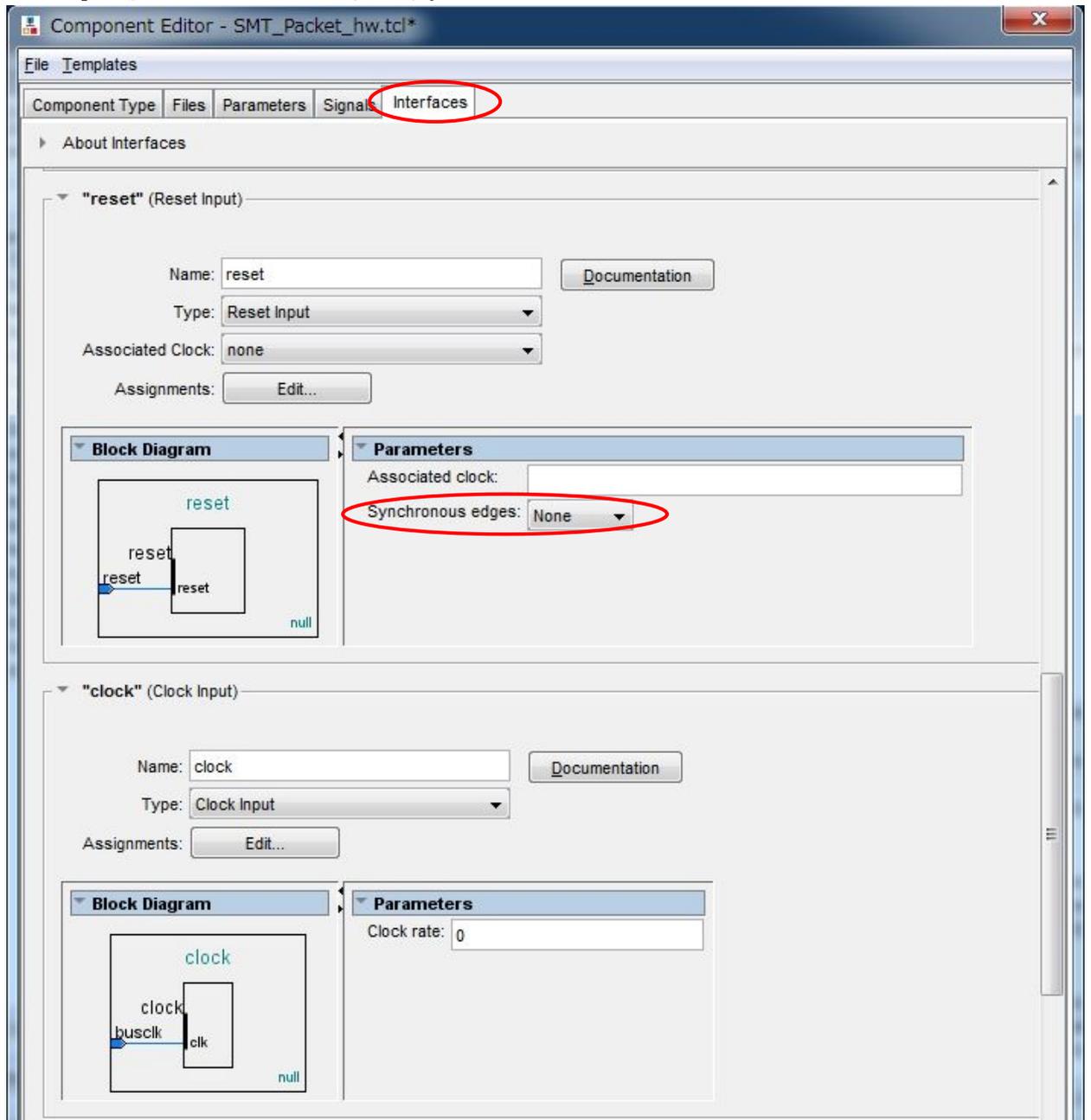
The 'Timing' section shows:

- Setup: 0
- Read wait: 0
- Write wait: 0
- Hold: 1
- Timing units: Cycles

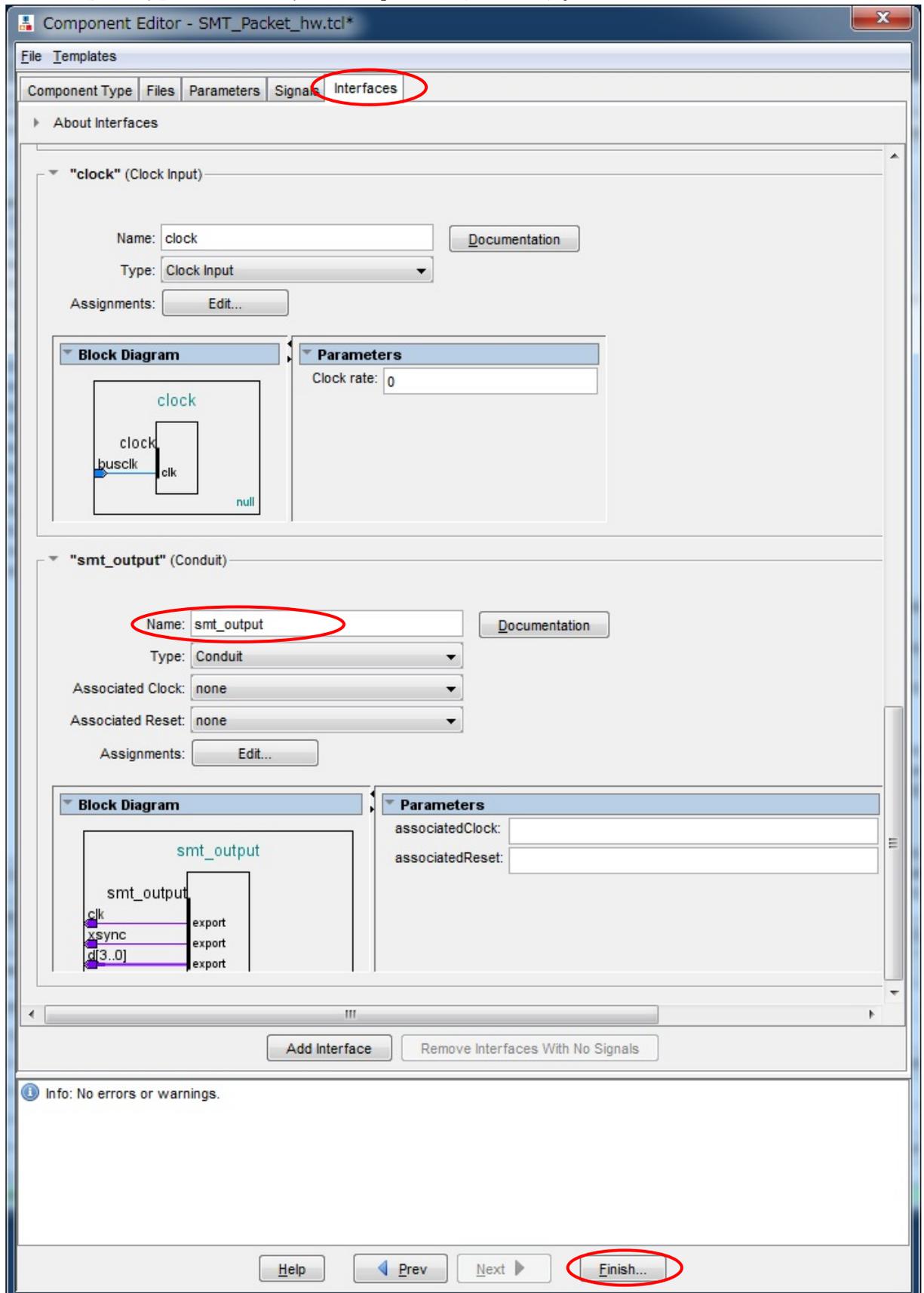
The 'Pipelined Transfers' section shows:

- Read latency: 0
- Maximum pending read transactions: 0
- Burst on burst boundaries only
- Linewrap bursts

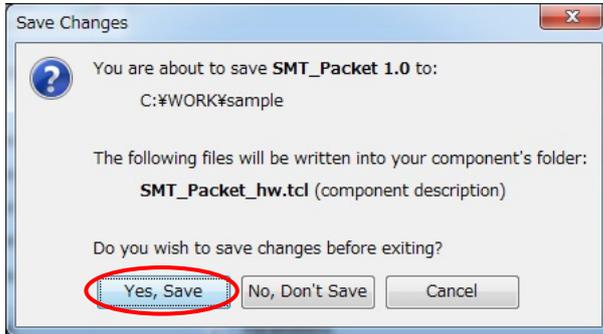
- 12) 「Interface」タブの「reset」の設定にて、“Synchronous edges”を「None」にします。
「clock」の設定はデフォルトで良いです。



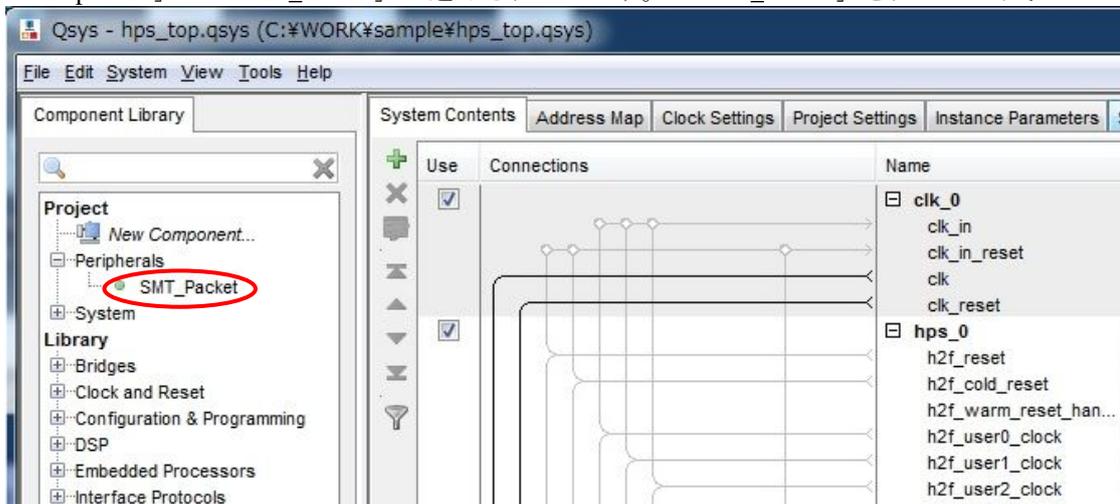
- 13) 「Interface」タブの「smt_output」の設定にて、“Name”を「smt_output」にします。
ここまで設定が完了しましたら、「Finish」をクリックします。



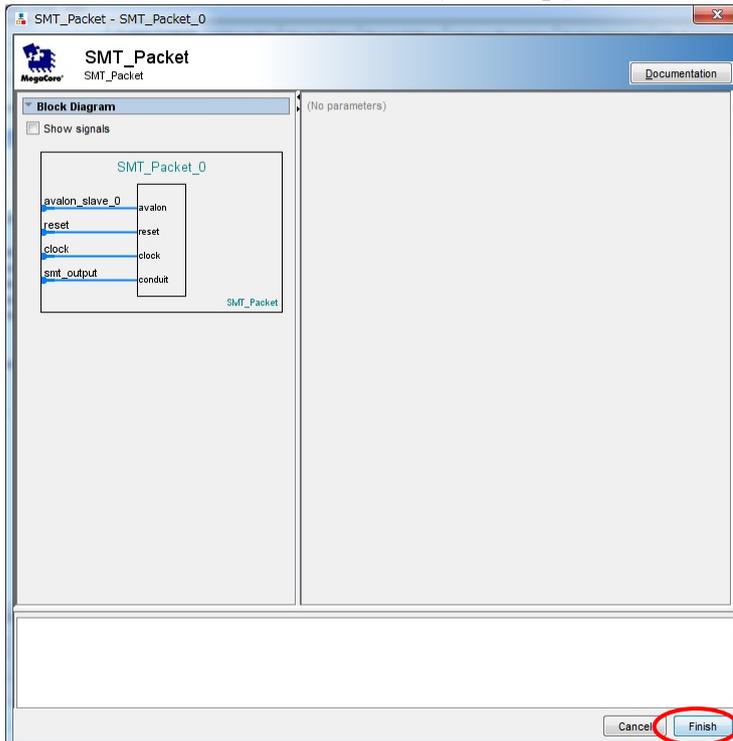
- 14) 以下のダイアログが出てくるので、「Yes, Save」をクリックします。
これでパケット論理が組み込める状態になりました。



- 15) 「Peripherals」に「SMT_Packet」が追加されています。「SMT_Packet」をダブルクリックします。



- 16) 以下のダイアログが出てきますので、「Finish」をクリックします。



17) 「SMT_Packet_0」が組み込まれました。ここで Error が出ていますが、接続が完了すればこの Error は消えますので、ここでは気にしないでください。

ここからバスの接続を行います。まずは見やすい様に IP の順番を入れ替えます。

「SMT_Packet_0」を選択し、上矢印ボタンを押して「hps」の近くに持っていきます。

The screenshot shows the Qsys IDE interface. The main window displays the System Contents, which is a hierarchical view of the system components. The component list on the right is as follows:

Use	Connections	Name	Description	Exp
<input checked="" type="checkbox"/>		led_pio	PIO (Parallel I/O)	
<input checked="" type="checkbox"/>		dipsw_pio	PIO (Parallel I/O)	
<input checked="" type="checkbox"/>		button_pio	PIO (Parallel I/O)	
<input checked="" type="checkbox"/>		jtag_uart_0	JTAG UART	
<input checked="" type="checkbox"/>		nios2_qsys_0	Nios II Processor	
<input checked="" type="checkbox"/>		ram	On-Chip Memory (RAM or ROM)	
<input checked="" type="checkbox"/>		SMT_Packet_0	SMT_Packet	

The Messages window at the bottom shows the following errors and warnings:

Description	Path
2 Errors	
SMT_Packet_0.clock must be connected to a clock output	System.SMT_Packet_0
SMT_Packet_0.reset must be connected to a reset source	System.SMT_Packet_0
2 Warnings	
SMT_Packet_0.avalon_slave_0 must be connected to an Avalon-MM master	System.SMT_Packet_0
SMT_Packet_0.smt_output must be exported, or connected to a matching conduit.	System.SMT_Packet_0
2 Errors, 2 Warnings	

- 18) 「hps_0」の「h2f_lw_axi_master」バスに、「SMT_Packet_0」の「avalon_slave_0」を接続します。
 この接続により、ツールの方でlw_axi-avalonのブリッジが自動生成されます。
 clk,reset も以下の様に接続を行います。

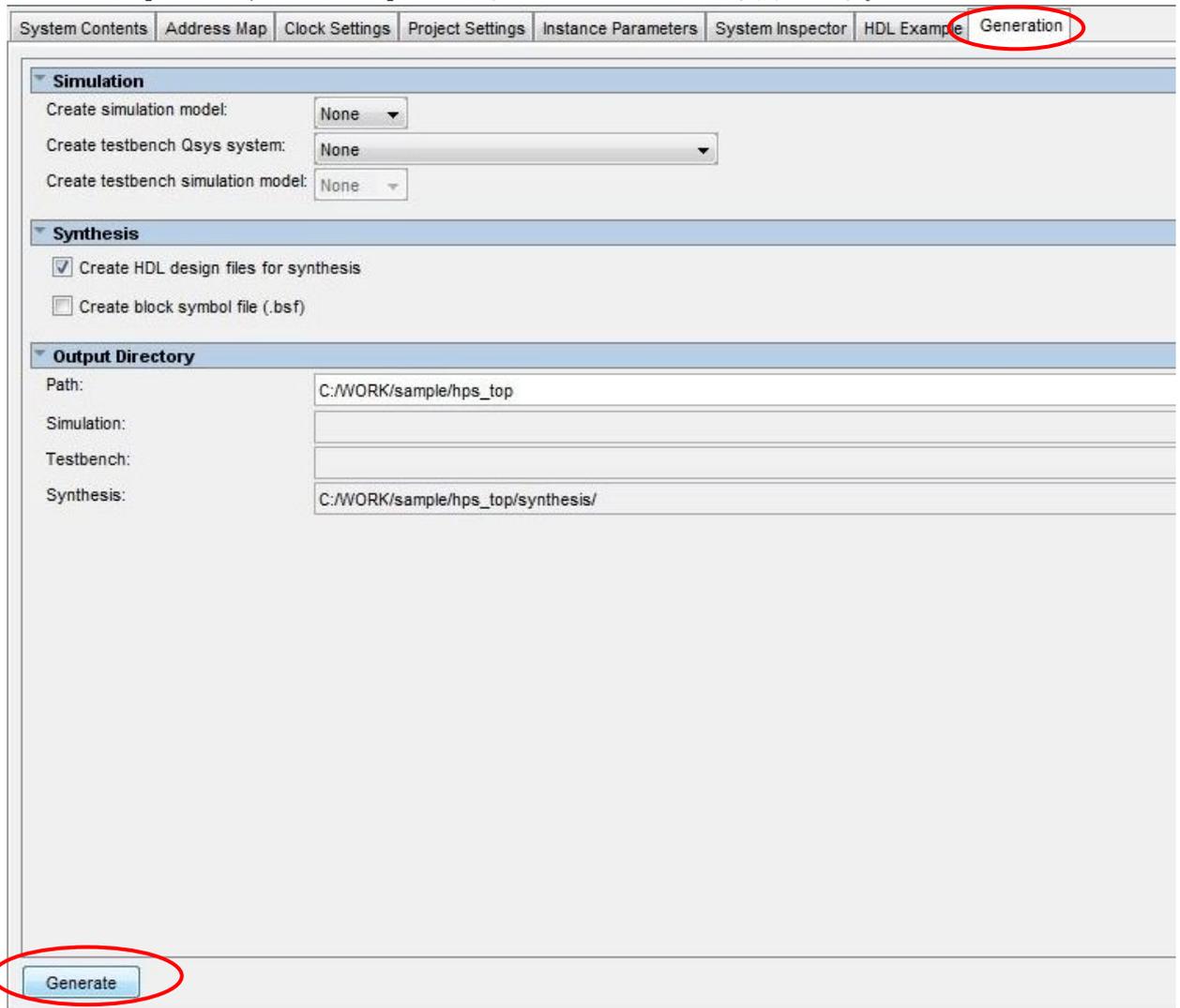
smt_output は「Export」欄をダブルクリックし、外部信号に設定します。

Use	Connections	Name	Description	Export
<input checked="" type="checkbox"/>		clk_0	Clock Source	clk
<input checked="" type="checkbox"/>		clk_in	Clock Input	clk_0_clk_in_reset
<input checked="" type="checkbox"/>		clk_in_reset	Reset Input	Double-click to export
<input checked="" type="checkbox"/>		clk	Clock Output	Double-click to export
<input checked="" type="checkbox"/>		clk_reset	Reset Output	Double-click to export
<input checked="" type="checkbox"/>		hps_0	Hard Processor System	
<input checked="" type="checkbox"/>		h2f_reset	Reset Output	hps_0_h2f_reset
<input checked="" type="checkbox"/>		h2f_cold_reset	Reset Output	hps_0_h2f_cold_reset
<input checked="" type="checkbox"/>		h2f_warm_reset_han...	Conduit	hps_0_h2f_warm_reset_h...
<input checked="" type="checkbox"/>		h2f_user0_clock	Clock Output	hps_0_h2f_user0_clock
<input checked="" type="checkbox"/>		h2f_user1_clock	Clock Output	hps_0_h2f_user1_clock
<input checked="" type="checkbox"/>		h2f_user2_clock	Clock Output	hps_0_h2f_user2_clock
<input checked="" type="checkbox"/>		h2f_axi_clock	Clock Input	Double-click to export
<input checked="" type="checkbox"/>		h2f_axi_master	AXI Master	Double-click to export
<input checked="" type="checkbox"/>		h2f_lw_axi_clock	Clock Input	Double-click to export
<input checked="" type="checkbox"/>		h2f_lw_axi_master	AXI Master	Double-click to export
<input checked="" type="checkbox"/>		f2h_sdram0_data	AXI Slave	Double-click to export
<input checked="" type="checkbox"/>		f2h_sdram0_clock	Clock Input	Double-click to export
<input checked="" type="checkbox"/>		memory	Conduit	memory
<input checked="" type="checkbox"/>		hps_io	Conduit	hps_io
<input checked="" type="checkbox"/>		SMT_Packet_0	SMT_Packet	
<input checked="" type="checkbox"/>		avalon_slave_0	Avalon Memory Mapped Slave	Double-click to export
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to export
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to export
<input checked="" type="checkbox"/>		smt_output	Conduit	Double-click to export
<input checked="" type="checkbox"/>		onchip_memory2_0	On-Chip Memory (RAM or ROM)	
<input checked="" type="checkbox"/>		clk1	Clock Input	SMT_Packet_0.smt_output
<input checked="" type="checkbox"/>		s1	Avalon Memor...	Conduit [conduit_end 12.1]
<input checked="" type="checkbox"/>		reset1	Reset Input	Double-click to export

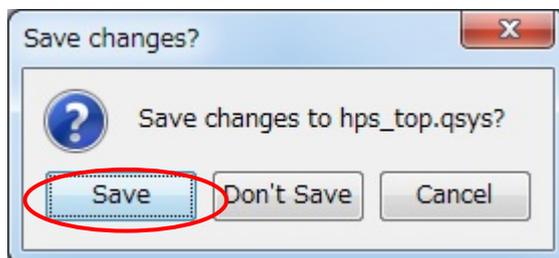
名称は、デフォルト (SMT_Packet_0_smt_output) としました。

<input checked="" type="checkbox"/>		SMT_Packet_0	SMT_Packet	
<input checked="" type="checkbox"/>		avalon_slave_0	Avalon Memory Mapped Slave	Double-click to export
<input checked="" type="checkbox"/>		reset	Reset Input	Double-click to export
<input checked="" type="checkbox"/>		clock	Clock Input	Double-click to export
<input checked="" type="checkbox"/>		smt_output	Conduit	SMT Packet 0 smt output

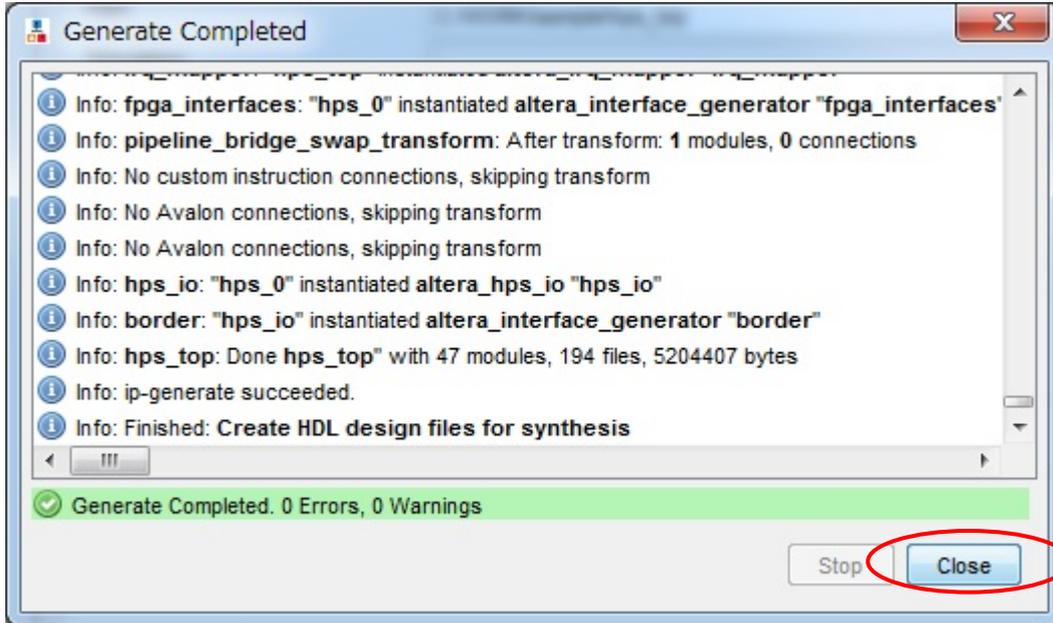
- 22) 「Generation」 タブで、「Generate」をクリックしてコンパイルを開始します。



- 23) 以下のダイアログが出てきますので、「Save」を押します。

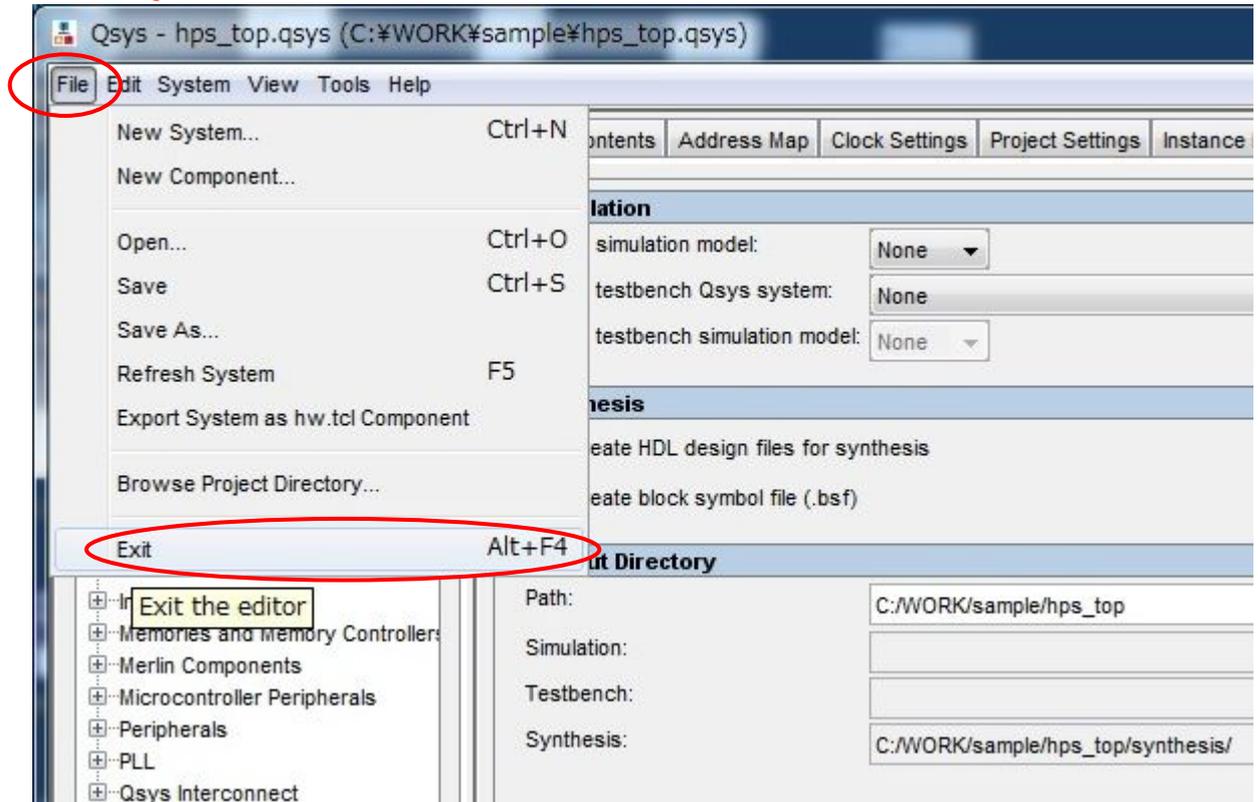


24) コンパイルが完了すると、以下のダイアログの内容となります。「Close」をクリックします。



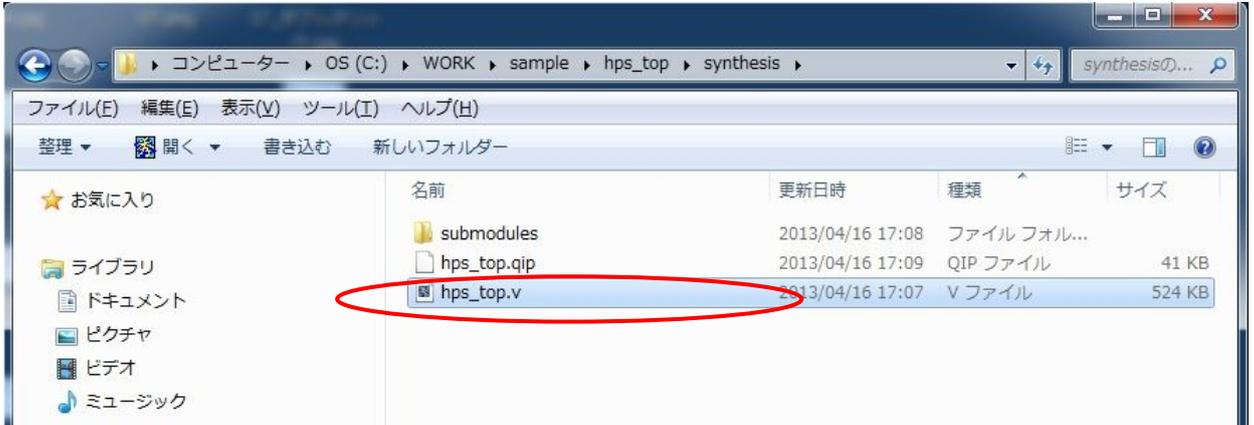
25) これで Qsys 側の作業は終了となります。「File」→「Exit」で、Qsys を閉じます。

【注意】 Quartus II はまだ閉じないでください

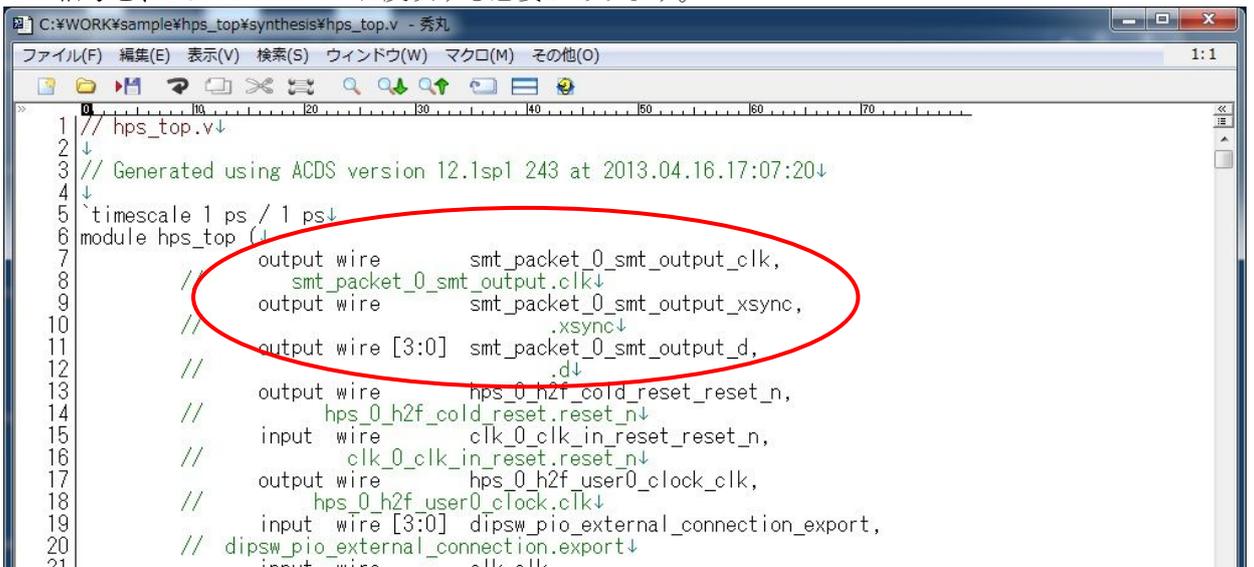


2.3 TOP モジュールの編集

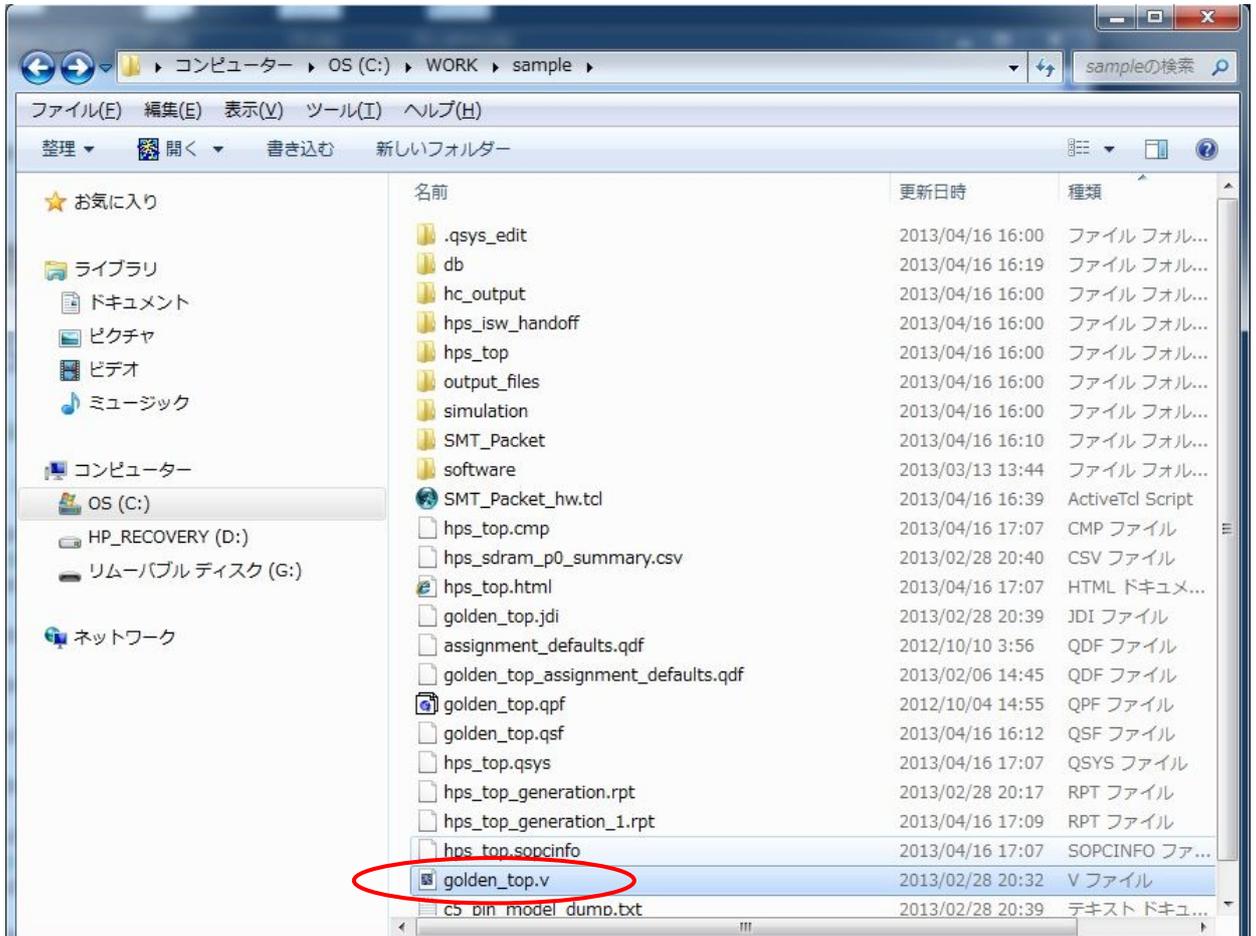
- 1) “ワークフォルダ¥hps_top¥synthesis¥hps_top.v” を開き、SMT パケット生成論理の出力信号を確認します。



- 2) パケット生成論理の出力信号が反映されていることを確認します。
この信号を、TOP モジュールに反映する必要があります。



- 3) 本プロジェクトの TOP モジュールを開きます。



- 4) hps_top.v の SMT パケット論理の出力信号を、TOP モジュールに接続していきます。
下記の様な変更（赤字の追記）を行います。

※入出力定義の箇所（先頭）

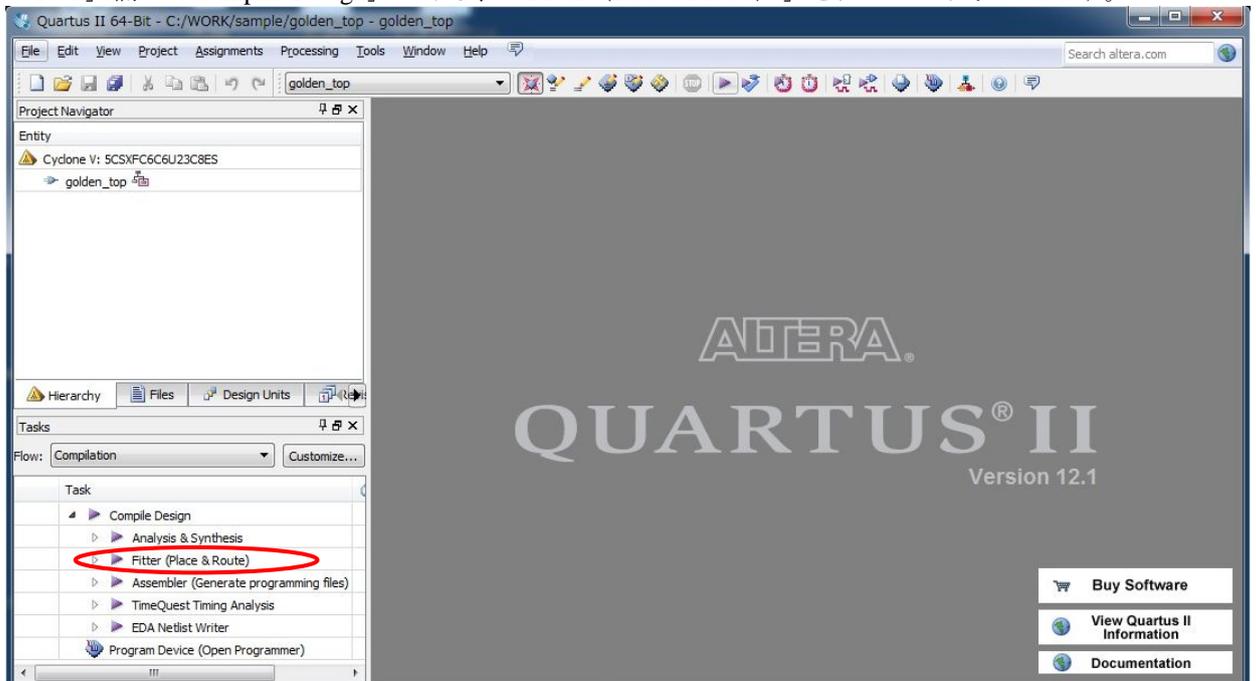
```
module golden_top (
  input      uart_rx,           //3.3V LV //UART Receive
  output     uart_tx,           //3.3V //UART Transmit
  inout     conv_hps_usb_n,     //3.3V //Placed on HPS GPIO
  inout     i2c_scl_hps,        //3.3V //HPS I2C Clock output
  inout     i2c_sda_hps,        //3.3V //HPS I2C Data Input/Output
  :
  // @YDC add begin
  // SMT Signals-----
  output     smt_packet_0_smt_output_clk, // clk
  output     smt_packet_0_smt_output_xsync, // xsync
  output [3:0] smt_packet_0_smt_output_d, // d
  // @YDC add end
  output     overtemp,          //2.5V
  output [2:0] test_pad         //2.5V
);
```

※hps_top モジュールへの接続の箇所

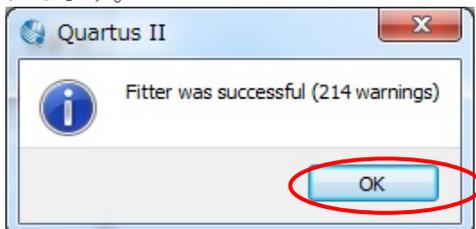
```
hps_top u0 (  
  //@YDC add begin  
  .smt_packet_0_smt_output_clk(smt_packet_0_smt_output_clk),      // clk  
  .smt_packet_0_smt_output_xsync      (smt_packet_0_smt_output_xsync),      //xsync  
  .smt_packet_0_smt_output_d  (smt_packet_0_smt_output_d),// d  
  //@YDC add end  
  .clk_clk                          (clk_50m_fpga),      // clk.clk  
  .clk_0_clk_in_reset_reset_n      (cpu_resetn),      // clk_0_clk_in_reset.reset_n  
  :  
  :  
  .hps_0_h2f_user0_clock_clk      (h2f_user0_clock),  
  .hps_0_h2f_user1_clock_clk      (h2f_user1_clock),  
  .hps_0_h2f_user2_clock_clk      (h2f_user2_clock)  
);
```

2.4 Quartus II コンパイル手順

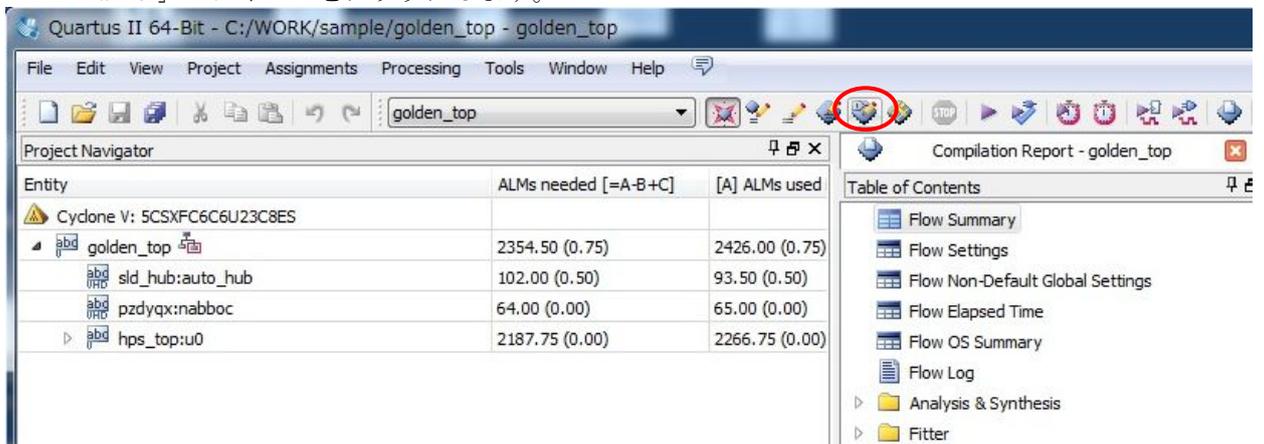
- 1) Quartus IIにて、「Fitter (Place&Route) 」まで実行します。
「Task」欄の「Compile Design」にある、「Fitter (Place&Route) 」をダブルクリックします。



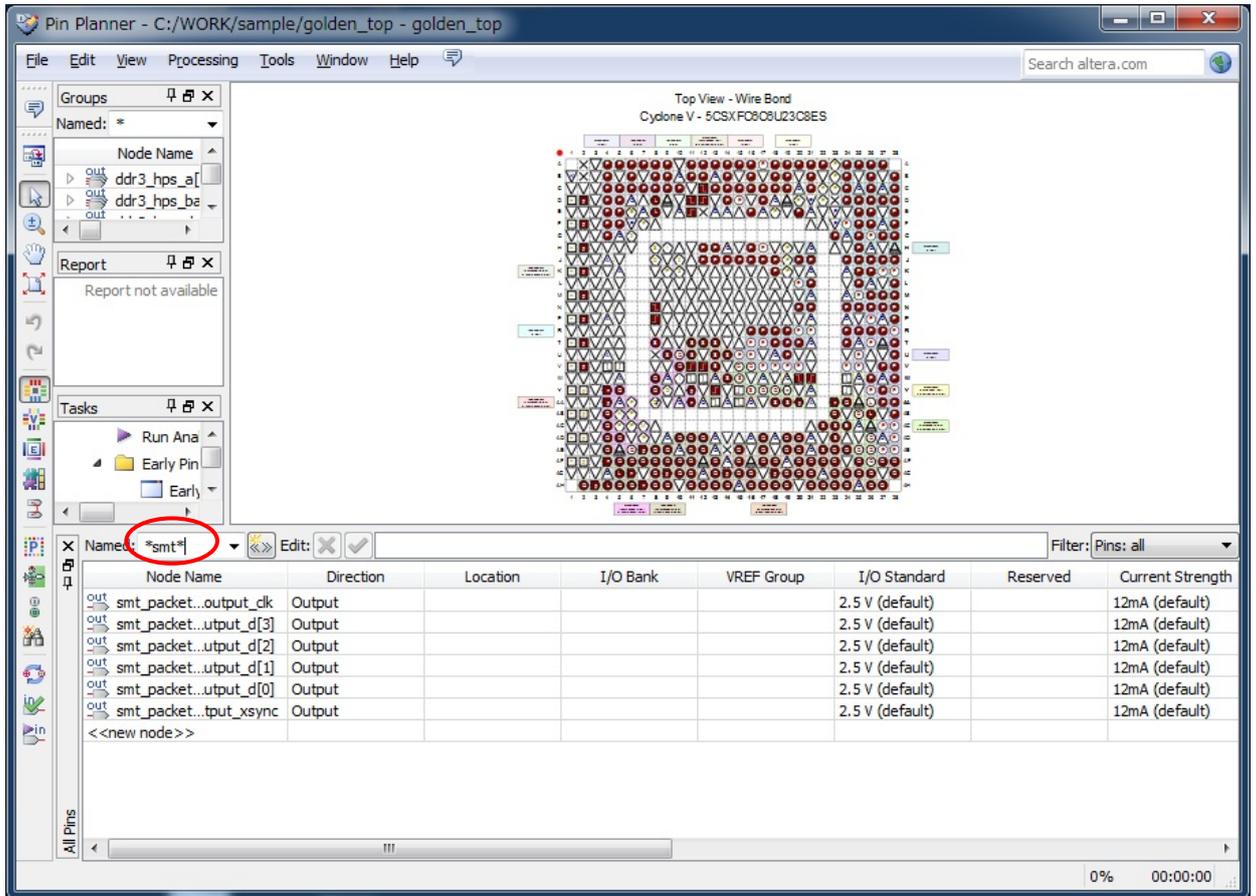
- 2) 「Fitter (Place&Route) 」まで無事に完了すると以下のダイアログが出てくるので、「OK」をクリックします。



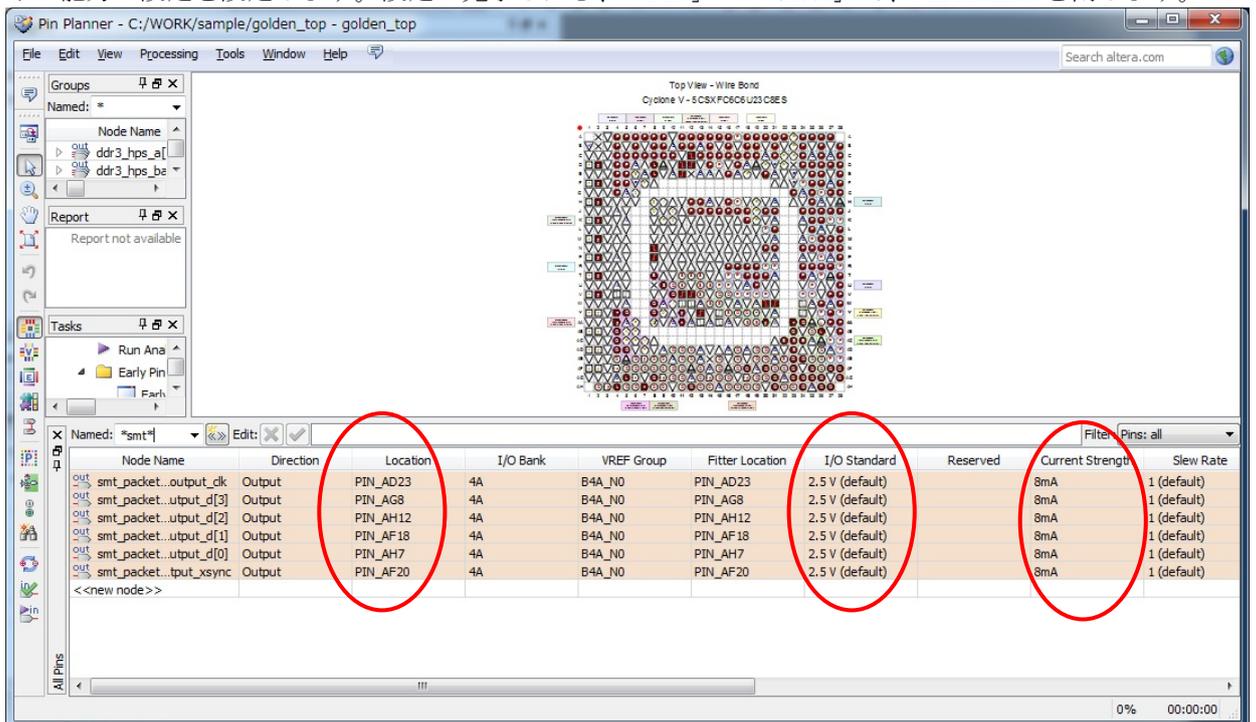
- 3) 「Pin Planner」のアイコンをクリックします。



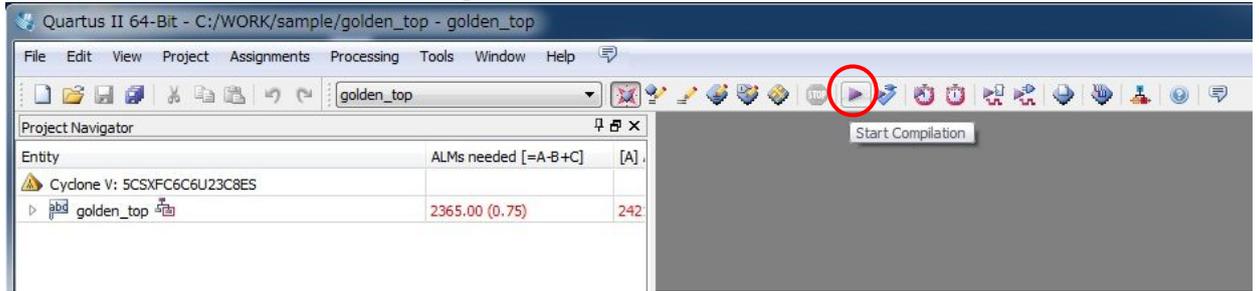
- 4) 信号一覧にて、「Named:」で「*smt*」を入力し、SMT パケット論理出力信号のみ表示させます。



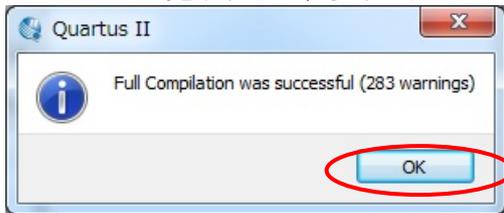
- 5) 「Location」で pin アサイン, 「I/O Standard」で信号の電圧レベル, 「Current Strength」で信号のドライブ能力の設定を設定します。設定が完了したら、「File」→「Close」で、Pin Planner を閉じます。



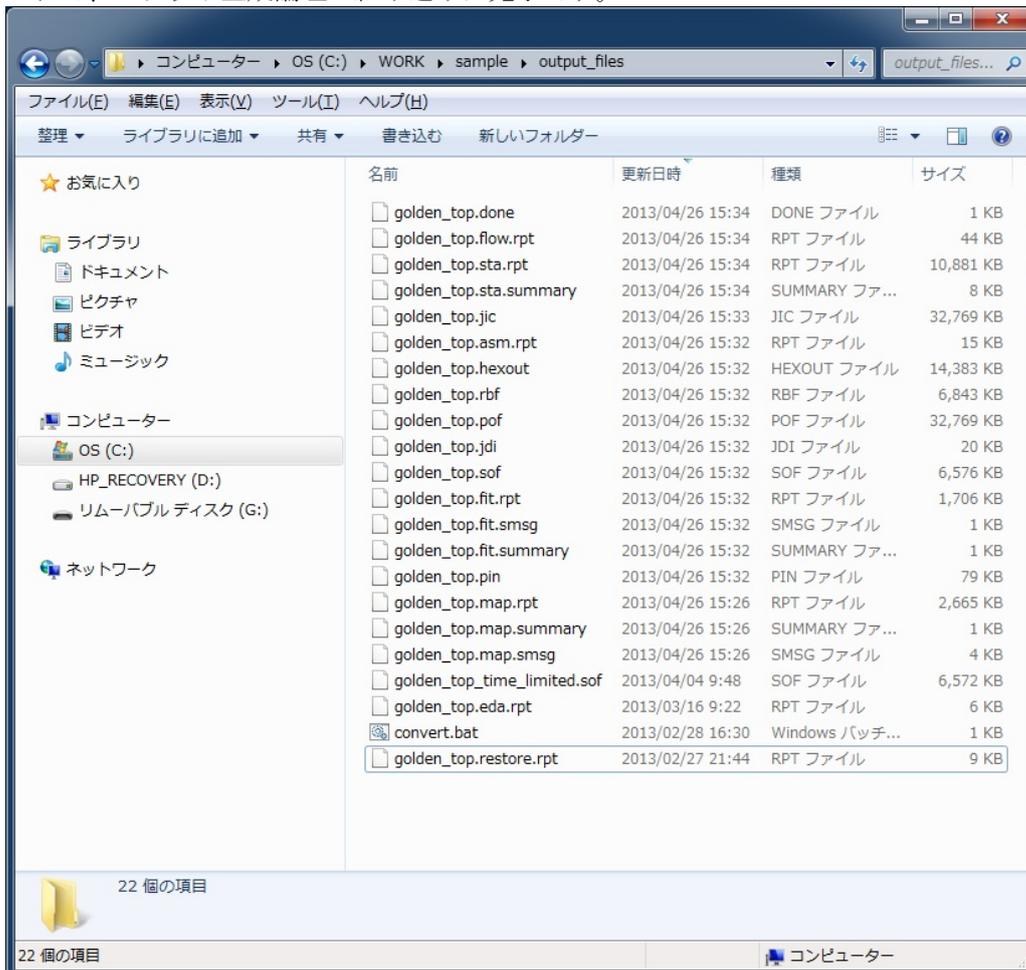
- 6) コンパイルを実施します。「Start Compilation」ボタンをクリックし、コンパイルを開始します。



- 7) コンパイルが完了すると、以下のダイアログが出てくるので「OK」をクリックします。



- 8) “” ワークフォルダ¥hps_top¥synthesis¥hps_top.v sof ファイル, Pof ファイルが無事に生成されていることを確認します。
これで、パケット生成論理の組み込みは完了です。



以上。