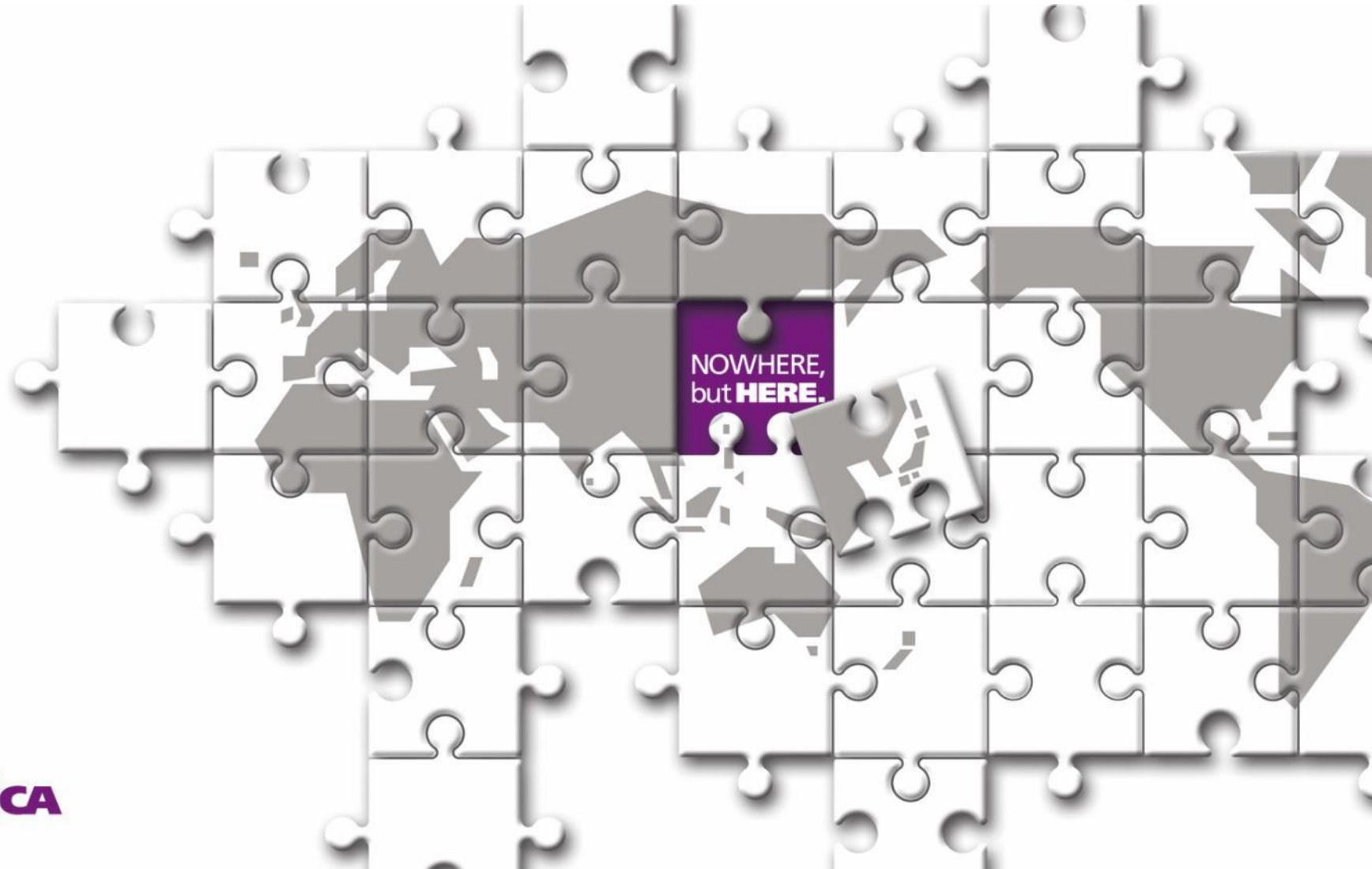


Lattice Radiant Startup Manual



はじめに



- 本マニュアルはLattice社FPGAデバイス設計ツールRadiantのオペレーションフローマニュアルです
- 新規プロジェクトの作成からデバイスへのフィッティングまでの一連のフローを解説したものです
- 各項目の詳細については、別途LatticeのユーザーマニュアルもしくはツールのHelpをご参照頂くか弊社技術サポートまでお問い合わせ下さい
- 本マニュアルとLattice社マニュアルに差異があった場合、Lattice社マニュアルを正として扱うようお願い致します



LATTICE
RADIANT
DESIGN SOFTWARE

Contents





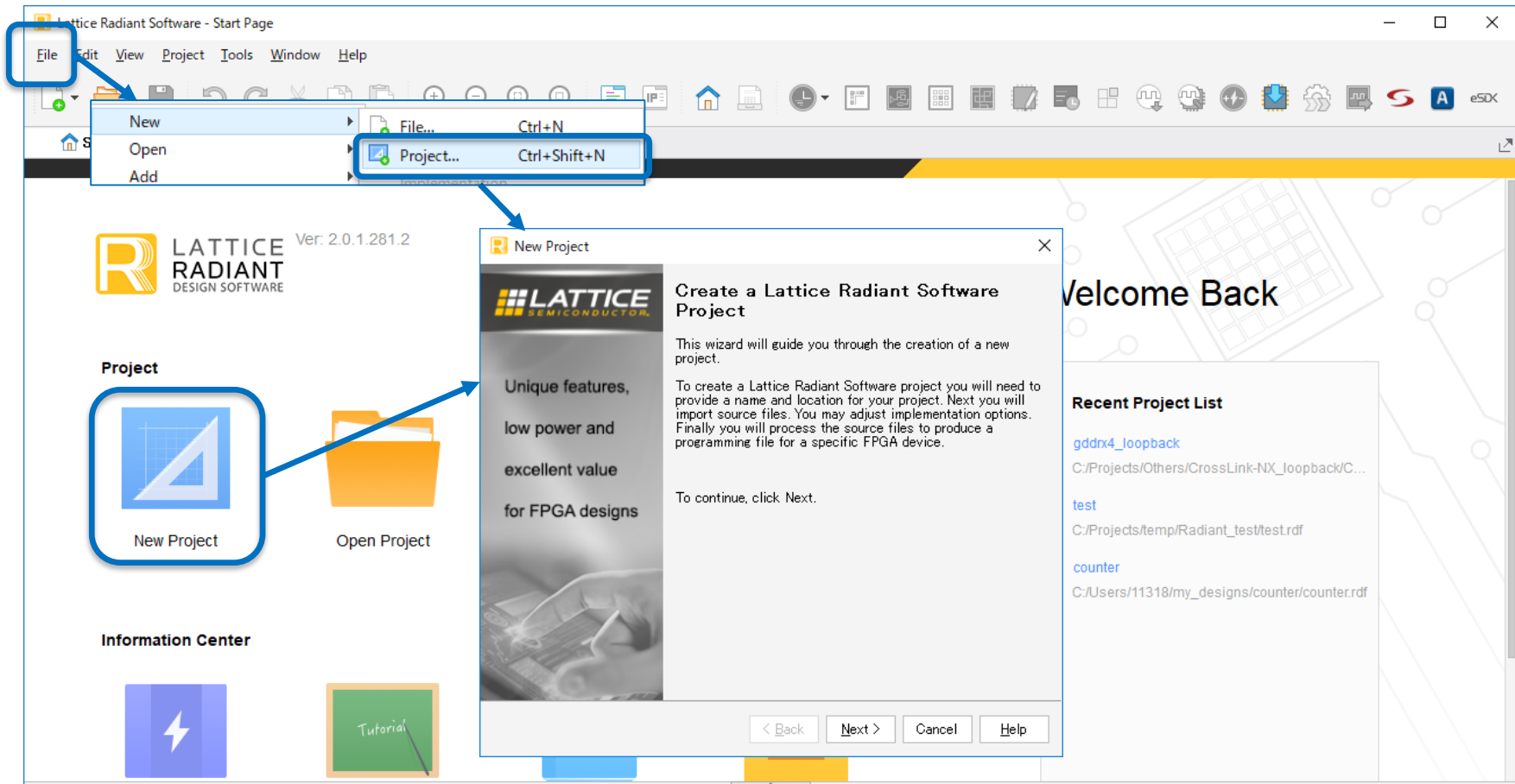
1. プロジェクト作成～デザインエントリー	・・・P3
2. タイミング制約の設定	・・・P24
3. I/Oアサイメント、デバイス全般の設定	・・・P32
4. デザインコンパイル～レポート確認	・・・P40
5. 消費電力の見積もり	・・・P60
6. Functionシミュレーション	・・・P67
7. デザイン書き込み	・・・P74
8. 実機上での内部波形確認	・・・P81

1. プロジェクト作成～デザインエントリー

Lattice Radiant Softwareの起動～新規Project作成



- Windowsのスタート > 全てのプログラム> Lattice Radiant Software x.x > Radiant Softwareまたは、デスクトップやタスクバーに配置したショートカットアイコン  からRadiantを起動します。
- 画面上のNew Projectアイコン  をクリック、またはFile > New > Projectでプロジェクト作成Windowを開きます。

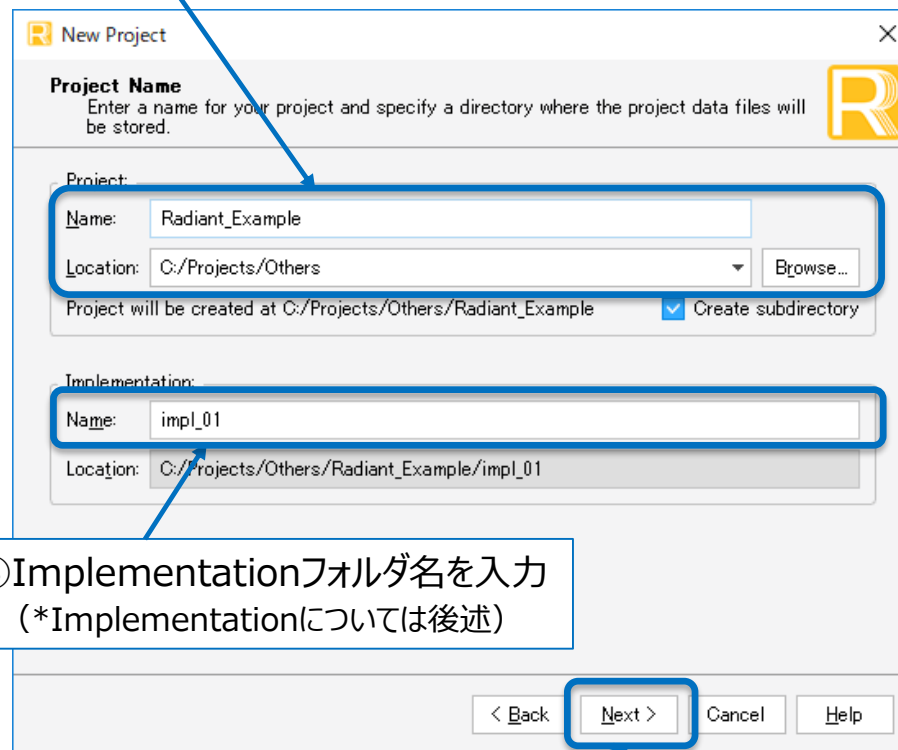
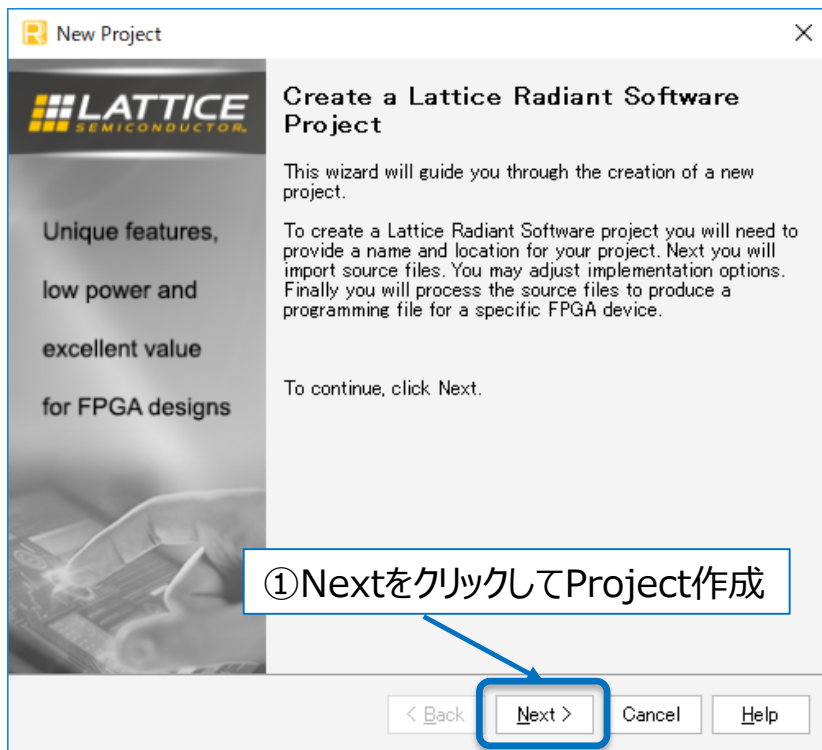


Lattice Radiant Softwareの起動～新規Project作成



- 以下のフローに従い新規プロジェクトを作成します
- フォルダ名は半角英数のみで入力し、フォルダパスに全角やスペースが入らないようにして下さい

②Projectフォルダ名とProjectフォルダのロケーションを入力
ロケーションはBrowseで選択することが可能です

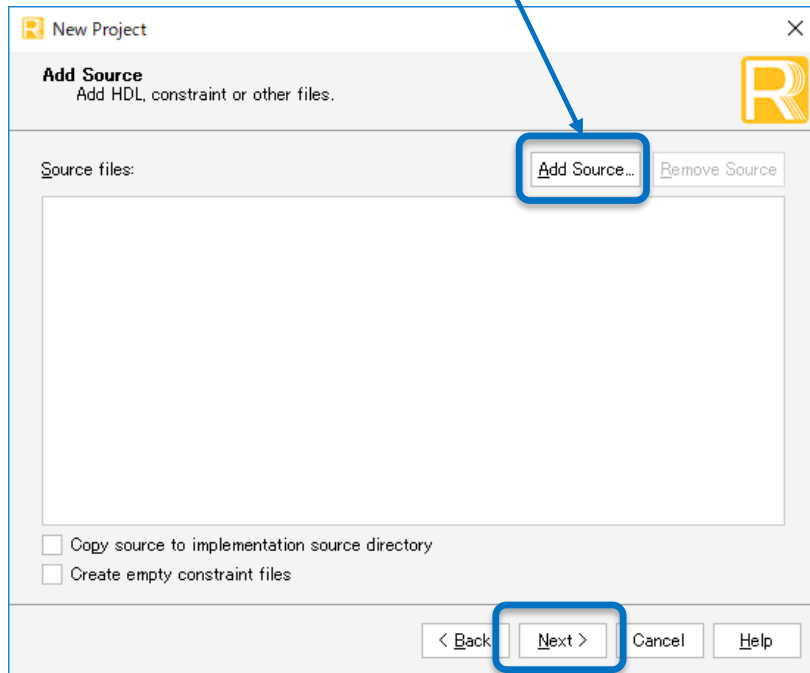


Lattice Radiant Softwareの起動～新規Project作成



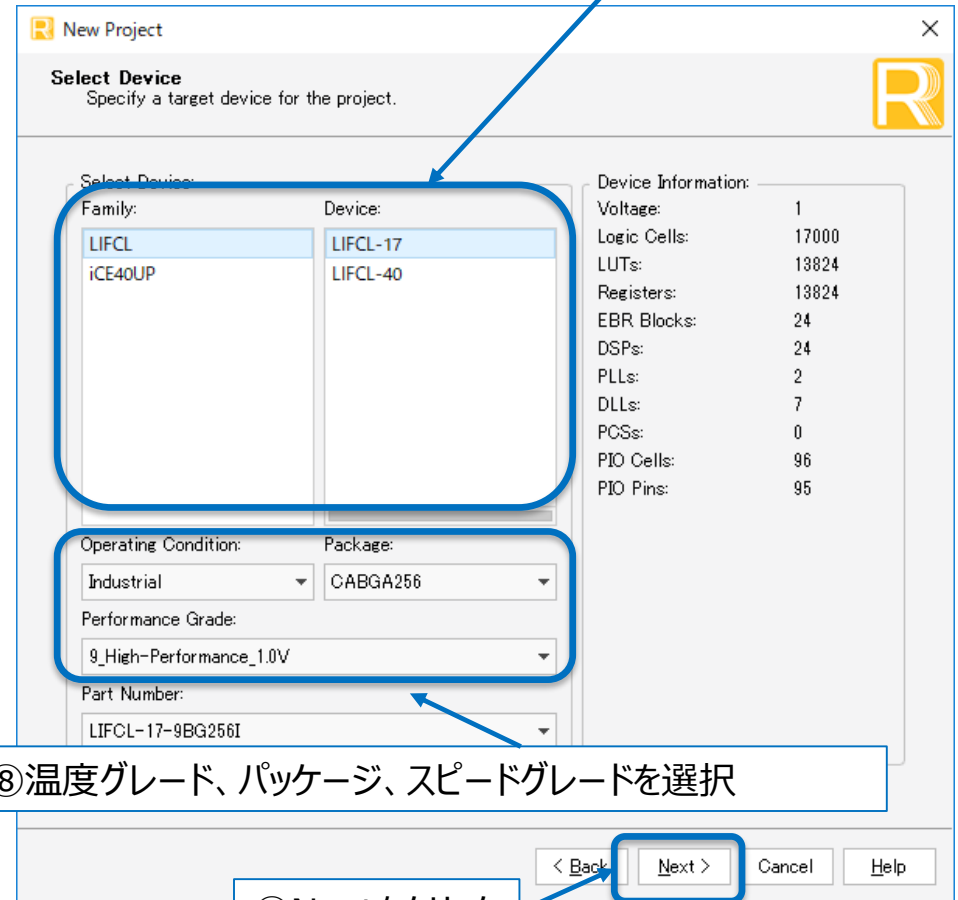
- 以下のフローに従い新規プロジェクトを作成します

⑤プロジェクトにインポートするソースファイルを選択
(プロジェクト作成後でもインポート可能です)



⑥Nextをクリック

⑦設計対象のデバイスファミリー、デバイス型名を選択



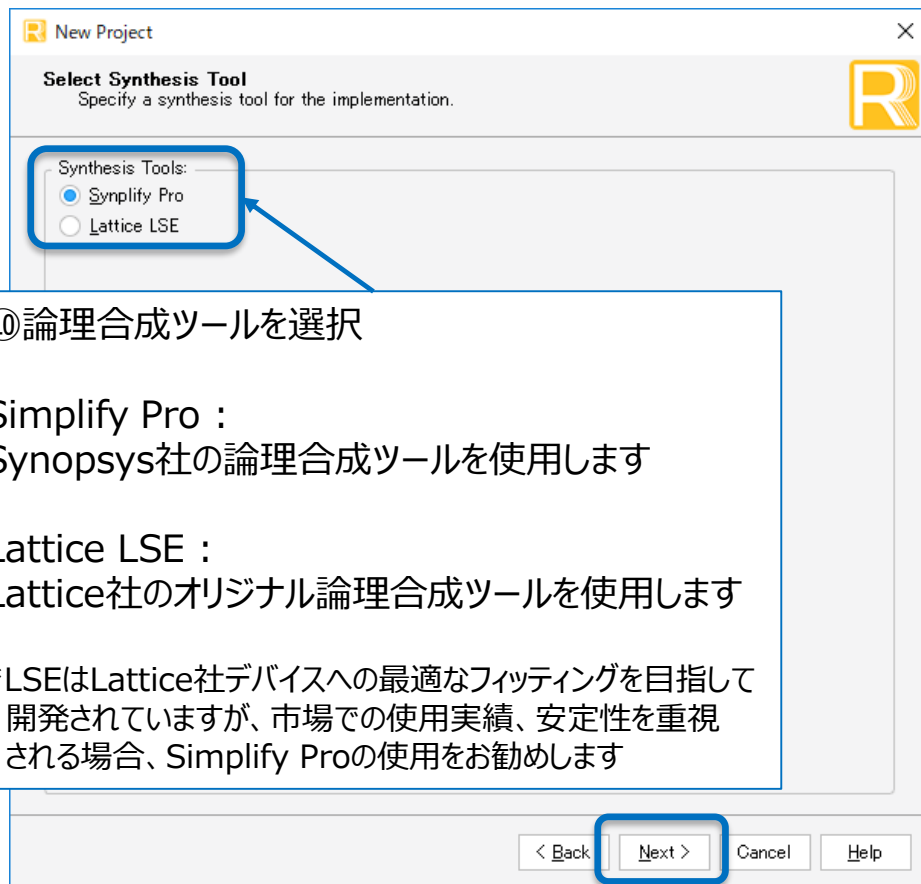
⑧温度グレード、パッケージ、スピードグレードを選択

⑨Nextをクリック

Lattice Radiant Softwareの起動～新規Project作成



- 以下のフローに従い新規プロジェクトを作成します



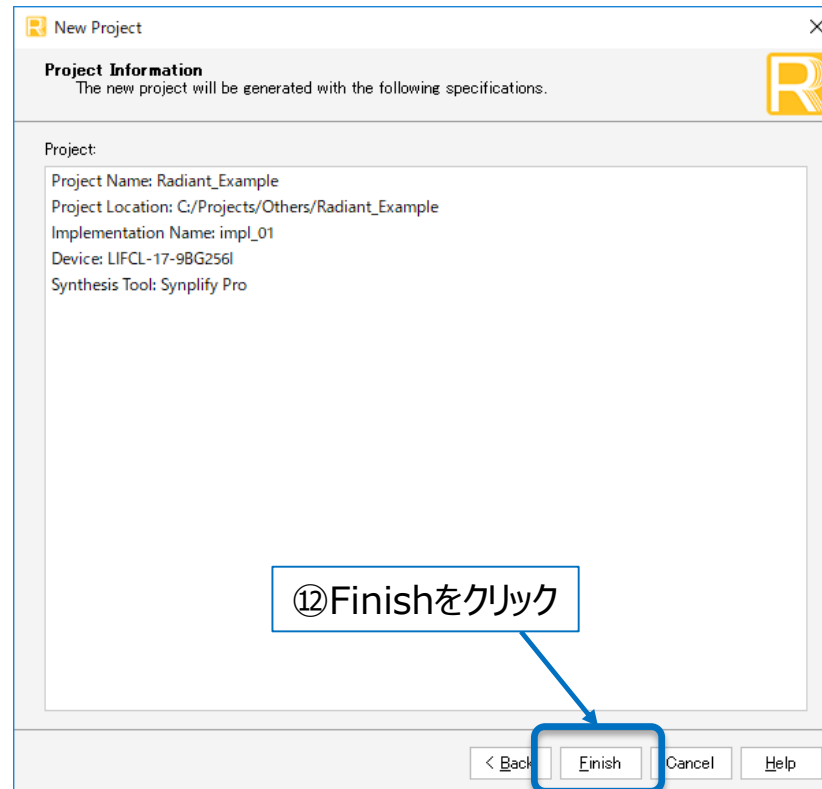
⑩論理合成ツールを選択

Simplify Pro :
Synopsys社の論理合成ツールを使用します

Lattice LSE :
Lattice社のオリジナル論理合成ツールを使用します

*LSEはLattice社デバイスへの最適なフィッティングを目指して開発されていますが、市場での使用実績、安定性を重視される場合、Simplify Proの使用をお勧めします

⑪Nextをクリック



⑫Finishをクリック

(参考) Implementationについて



- Radiantは作成したProject単位で管理を行います
- Projectの下にさらに複数のサブProjectを構成でき、サブProjectごとに違う設計データを持たせてデザインの管理を行うことができます。このサブProjectをImplementationと呼びます
- Implementation毎にStrategy（コンパイルオプション）や制約を持たせることができ、全てのImplementationを並列にコンパイルさせ、それぞれのコンパイル結果を比較することもできます

New Project

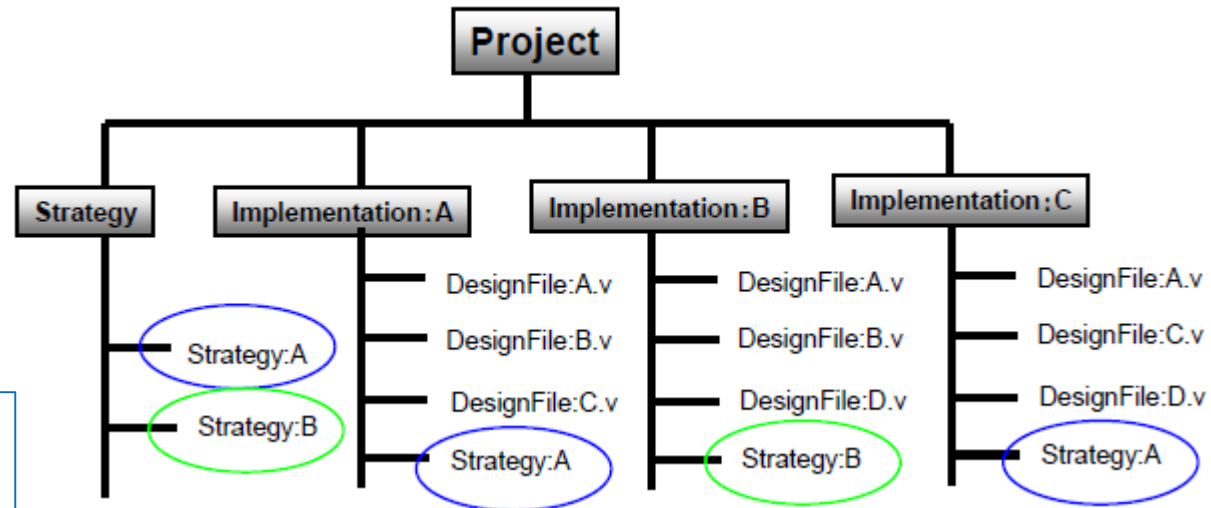
Project Name
Enter a name for your project and specify a directory where the project data files will be stored.

Project:
Name: Radiant_Example
Location: C:/Projects/Others Browse...
Project will be created at C:/Projects/Others/Radiant_Example Create subdirectory


Implementation:
Name: impl_01
Location: C:/Projects/Others/Radiant_Example/impl_01

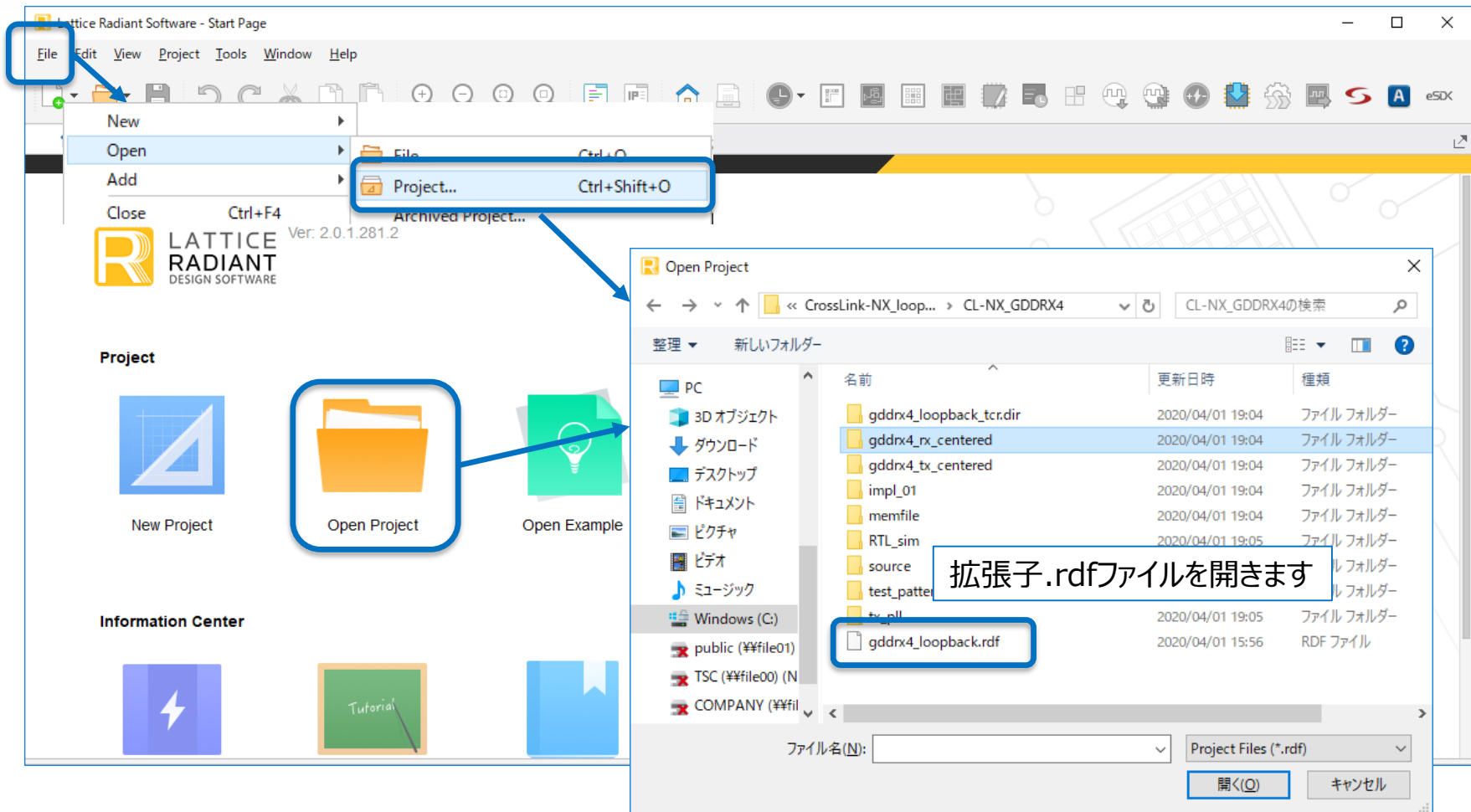
Implementation = サブProject

プロジェクト作成時はデフォルトのImplementationを作成します
作成後に他のImplementationを追加することが可能です



既存プロジェクトのオープン

- 画面上のOpen Projectアイコンをクリック、またはFile > Open > Projectで既存プロジェクトの拡張子.rdfファイルを開きます



Project Navigator



- プロジェクトを新規作成、または既存プロジェクトをオープンすると以下のProject NavigatorのGUIが開きます

The screenshot displays the Lattice Radiant Software Reports window. The interface includes a menu bar (File, Edit, View, Project, Tools, Window, Help), a toolbar with various icons, and a progress bar with steps: Synthesize Design, Map Design, Place & Route Design, and Export Files. The main area is divided into a left sidebar and a central content area.

Left Sidebar (Project Navigator):

- lvds71
 - LIFCL-17-9MG1211
 - Strategies
 - Area
 - Timing
 - Strategy1
 - impl_01 (Synplify Pro)
 - Input Files
 - source/lvds71_loopback_top.v
 - source/lvds71_rx_module.v
 - source/lvds71_tx_module.v
 - source/rx_data_mapper.v
 - source/tx_data_mapper.v
 - source/pattern_gen/simple_pattern_gen.v
 - gddr71_tx/gddr71_tx.ipx
 - RTL Files
 - Constraint Files
 - Testbench Files

Central Content Area (Reports):

Reports

- Project Summary**
- Synthesis Reports
- Map Reports
- Place & Route Reports
- Export Reports

Lvds71 Project Summary

Implementation Name:	impl_01	Performance Grade:	9_High-Performan
Strategy Name:	Strategy1	Operating Condition:	IND
Part Number:	LIFCL-17-9MG1211	Synthesis:	Synplify Pro
Family:	LIFCL	Timing Errors:	
Device:	LIFCL-17	Project Created:	2020/04/14 12:05:
Package:	CSFBGA121	Project Updated:	2020/04/01 18:58:
Project File:	C:/Projects/Others/CrossLink-NX_71LVDS/lvds71.rdf		
Implementation Location:	C:/Projects/Others/CrossLink-NX_71LVDS/impl_01		

Bottom Panel (Messages):

0 Errors, 0 Warnings, 77 Infos

Project (77 infos)

- 2049992 INFO - C:/Projects/Others/CrossLink-NX_71LVDS/source/lvds71_loopback_top.v(1,8-1,27) (VERI-1018) compiling module lvds71_loopback_top

Project Navigator



■ Process Toolbar

論理合成、マッピング、配置配線、ファイル出力のプロセスはここで実行します

■ Toolbar

ファイル追加、制約設定、消費電力見積もり、シミュレーション等を実行する各種ツールのアイコン

The screenshot displays the Lattice Radiant Software interface with several key components highlighted by blue boxes and callouts:

- Process Toolbar:** Located at the top, it contains icons for Synthesize Design, Map Design, Place & Route Design, and Export Files.
- Project View:** On the left, a tree view shows the project structure for 'lvds71', including 'LIFCL-17-9MG1211', 'Strategies', and 'Input Files'.
- Tool View:** The central area shows a 'Reports' sidebar and a 'Lvds71 Project Summary' table. The table contains the following data:

Lvds71 Project Summary			
Implementation Name:	impl_01	Performance Grade:	9_High-Performar
Strategy Name:	Strategy1	Operating Condition:	IND
- Message View:** At the bottom, a 'Message' pane shows project information, including '0 Errors', '0 Warnings', and '77 Infos'.

■ Project View


- ・インポート済みファイルリスト/ソース階層表示
 - ・ソーステンプレートの表示
 - ・専用モジュール/IPの生成画面表示
- をタブで切り替えて表示します

■ Tool View

制約設定ツール、配置配線状況確認ツール、レポートビューワ等各種ツールが開きます。複数ツールの表示をタブで切替えます


■ Message View

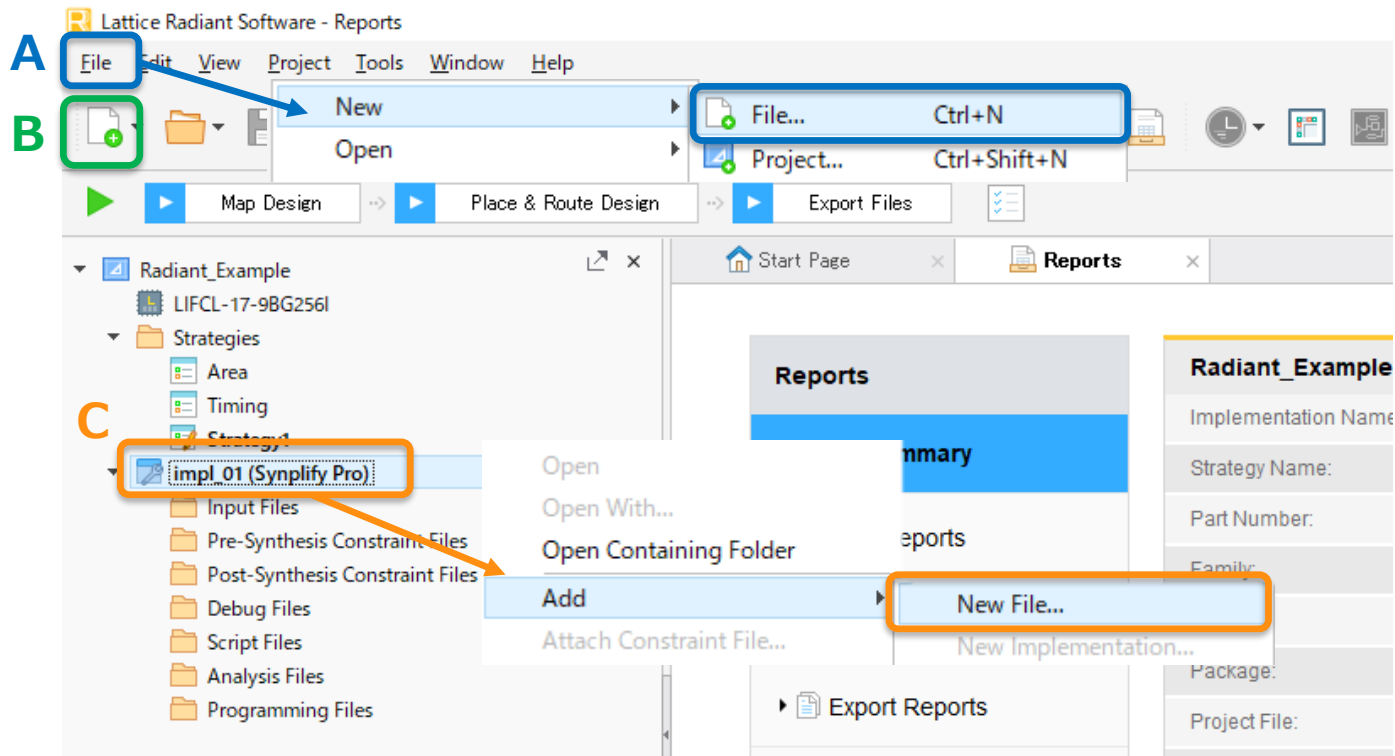
プロセス進行状況、エラー、ワーニングメッセージ等表示されます。確認したいメッセージをタブで切り替えます

*各ビュー右上のDetach Viewボタン  でビューを別ウィンドウに切り離して大きく表示できます

新規デザインエントリー



- New Fileウィンドウを以下のいずれかの方法で開きます
 - A) File > New > File を選択
 - B) Newアイコンをクリック
 - C) Implementation名を右クリック > Add > New File



新規デザインエントリー



- New Fileウィンドウが開いたら以下の手順で新規ソースコードを作成します

① Source Filesの一覧から使用する言語に応じてVerilog Files またはVHDL Filesを選択します

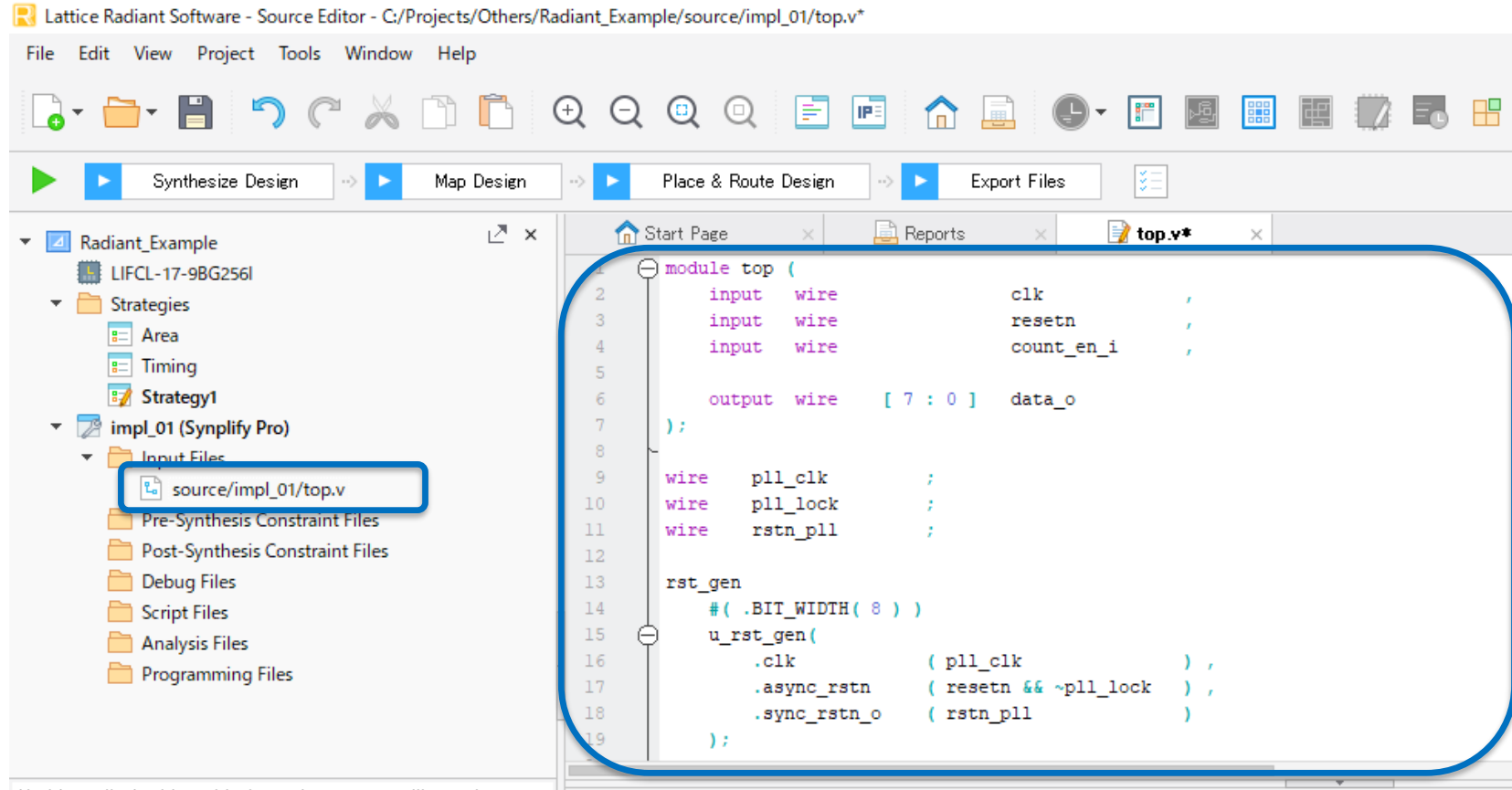
② ファイル名を入力します

③ Newをクリックします

新規デザインエントリー



- 新規ソースコードは自動でプロジェクトにインポートされます
- 同時にSource Editorが開き、そのままソースコード編集が可能です



既存デザインエントリー

- 既存のソースコードはFile > Add > Existing FileもしくはImplementation名を右クリックしてAdd > Existing Fileを選択します
- Add Existing Fileウィンドウが開きますので、インポートしたいソースコードを選択します


The screenshot illustrates the steps to add an existing file to a design entry in Lattice Radiant Software. The main window shows the 'File' menu with 'Add' > 'Existing File...' selected. The 'impl_01 (Synplify Pro)' entry is selected in the project tree, and its context menu is open with 'Add' > 'Existing File...' selected. The 'Add Existing File' dialog box is open, showing the file list with 'rst_gen.v' selected. The 'File name' field contains 'rst_gen.v' and 'counter.v'. The 'Files of type' is set to 'Input Files (*.vhd *.v *sv *h *.ipx *.sbx *.vm)'. The 'Copy file to directory' checkbox is checked, and the path is 'C:/Projects/Others/Radiant_Example/source/impl_01'.

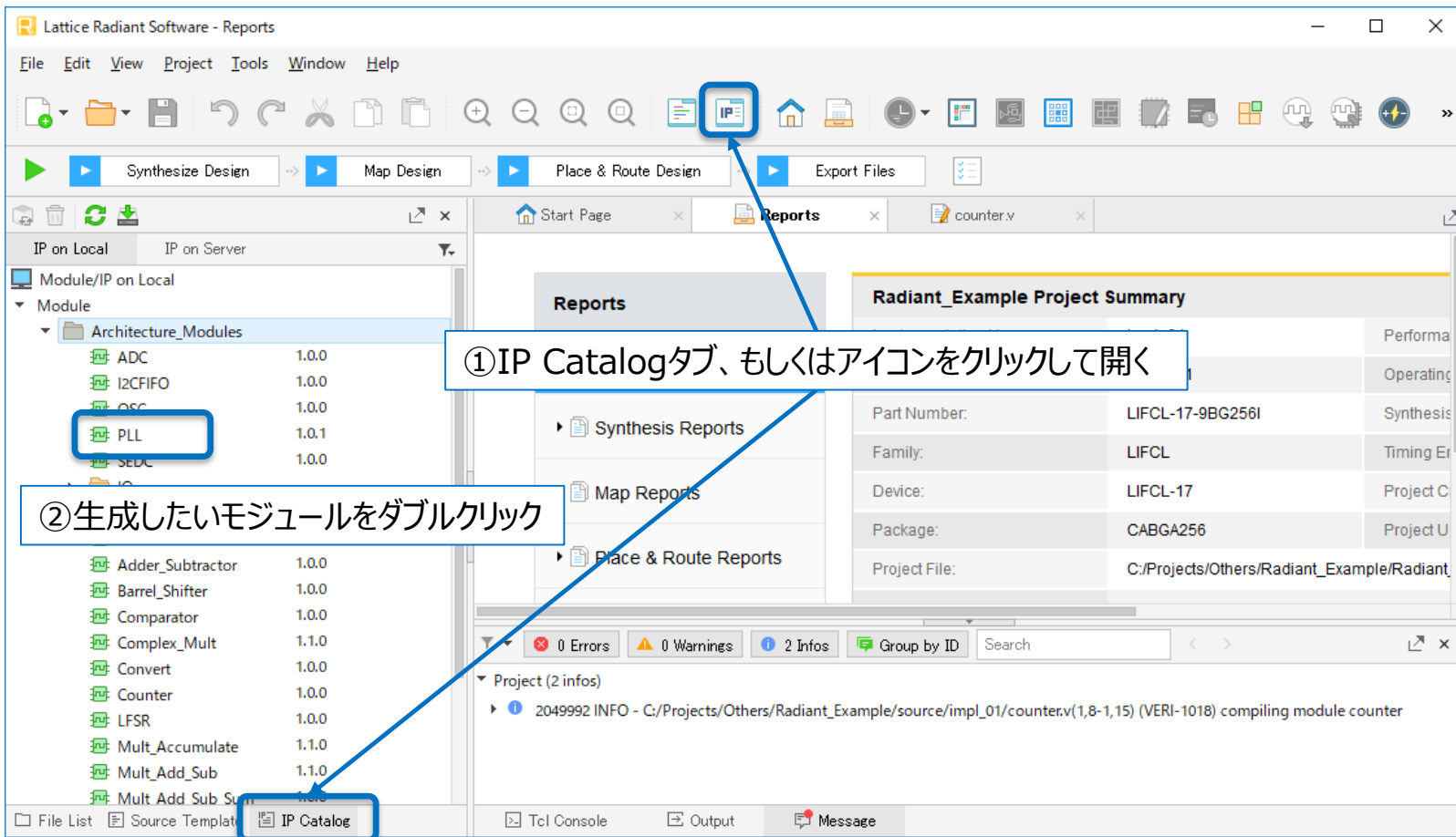
Name	Size	Type	Date Modified
counter.v	408...tes	v File	2020/0... 17:18
rst_gen.v	1 KB	v File	2020/0... 18:28
top.v	634...tes	v File	2020/0... 19:11

Unable to display hierarchical tree due to source file parsing

専用モジュールの生成



- Project View下部のタブからIP Catalogを選択、もしくはToolbarからIP Catalogアイコンをクリックします
- 生成したいモジュールをダブルクリック（ここでは例としてPLLを選択）



The screenshot shows the Lattice Radiant Software Reports window. The toolbar at the top contains an IP Catalog icon (a document with 'IP' on it) which is circled in blue. A blue arrow points from this icon to a callout box containing the text: ①IP Catalogタブ、もしくはアイコンをクリックして開く. In the left-hand pane, the 'Module/IP on Local' tree is expanded to 'Architecture_Modules', and the 'PLL' module is highlighted with a blue box. A second blue arrow points from this box to another callout box containing the text: ②生成したいモジュールをダブルクリック. At the bottom of the window, the 'IP Catalog' tab is also highlighted with a blue box. The main area of the window displays a 'Radiant_Example Project Summary' table with the following data:

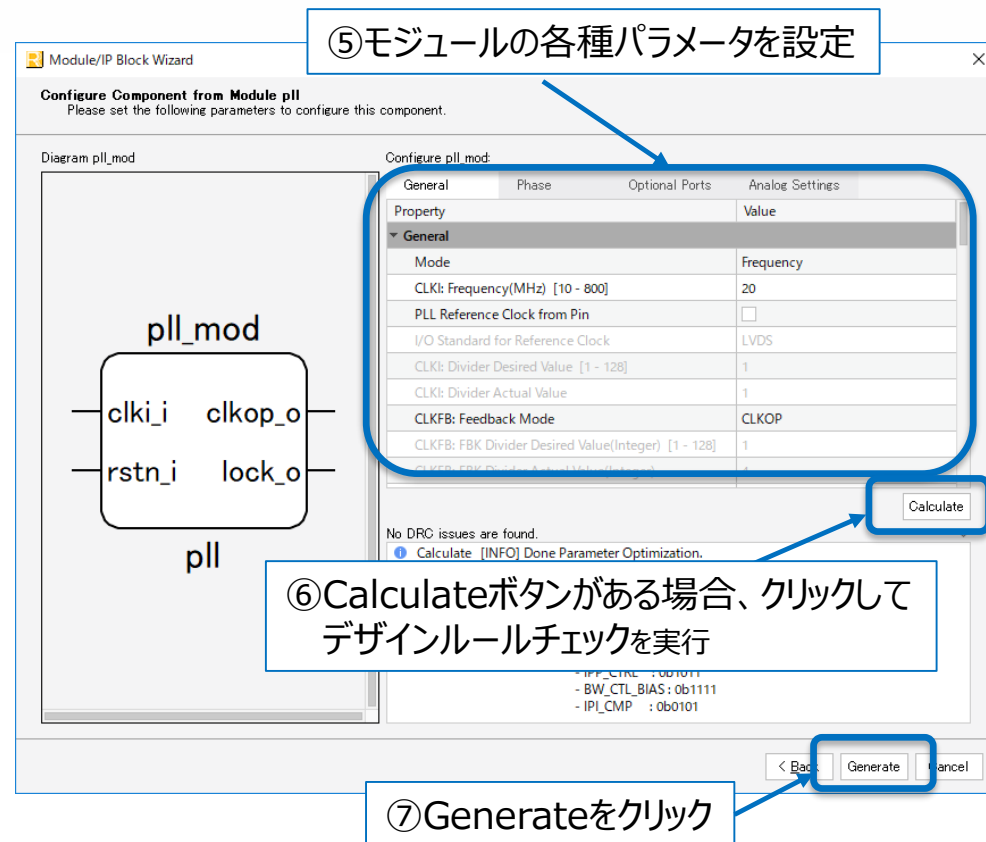
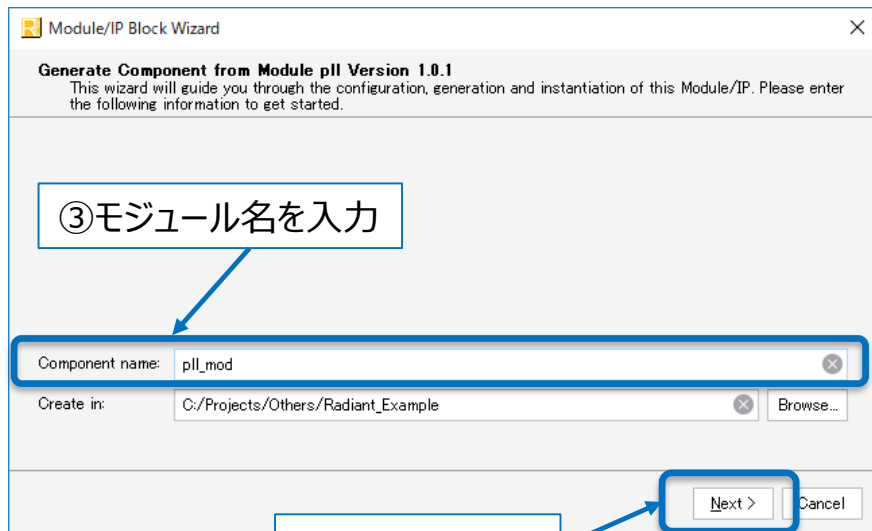
Property	Value	Category
Part Number:	LIFCL-17-9BG256I	Performance
Family:	LIFCL	Operating
Device:	LIFCL-17	Synthesis
Package:	CABGA256	Timing Er
Project File:	C:/Projects/Others/Radiant_Example/Radiant	Project C
		Project U

Below the table, there is a status bar showing '0 Errors', '0 Warnings', and '2 Infos'. A message pane at the bottom displays: Project (2 infos) > 2049992 INFO - C:/Projects/Others/Radiant_Example/source/impl_01/counter.v(1,8-1,15) (VERI-1018) compiling module counter.

専用モジュールの生成



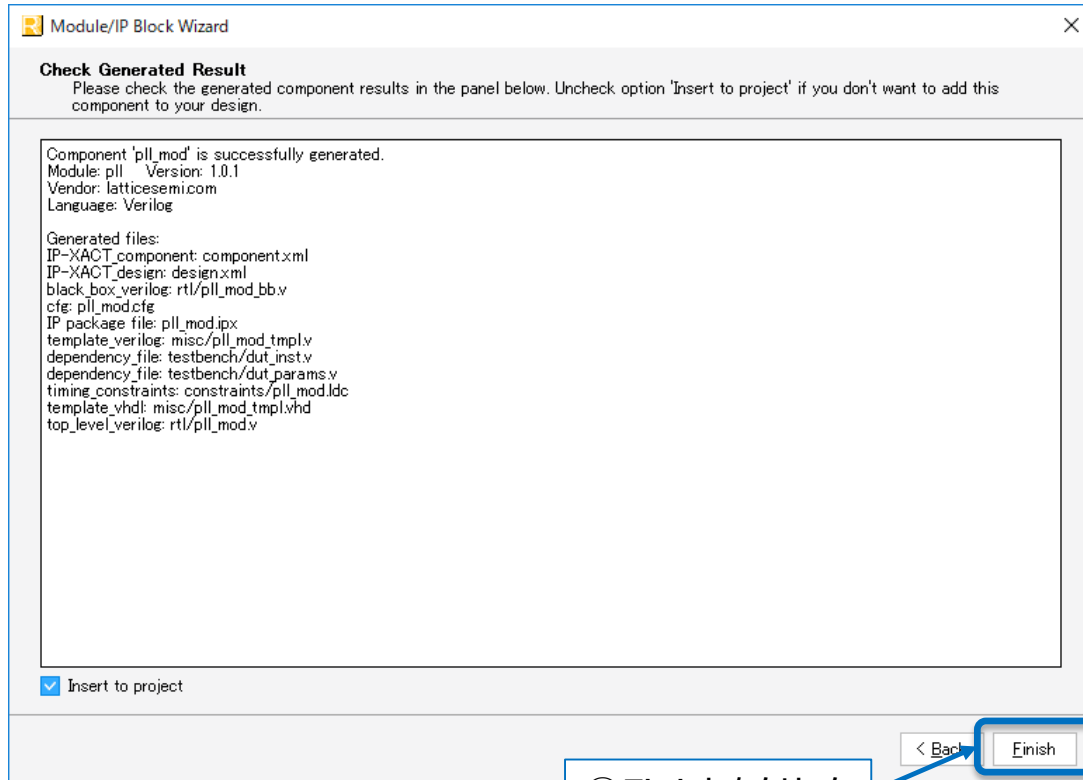
- Module/IP Block Wizardが開きますので、以下の手順でモジュールを生成します



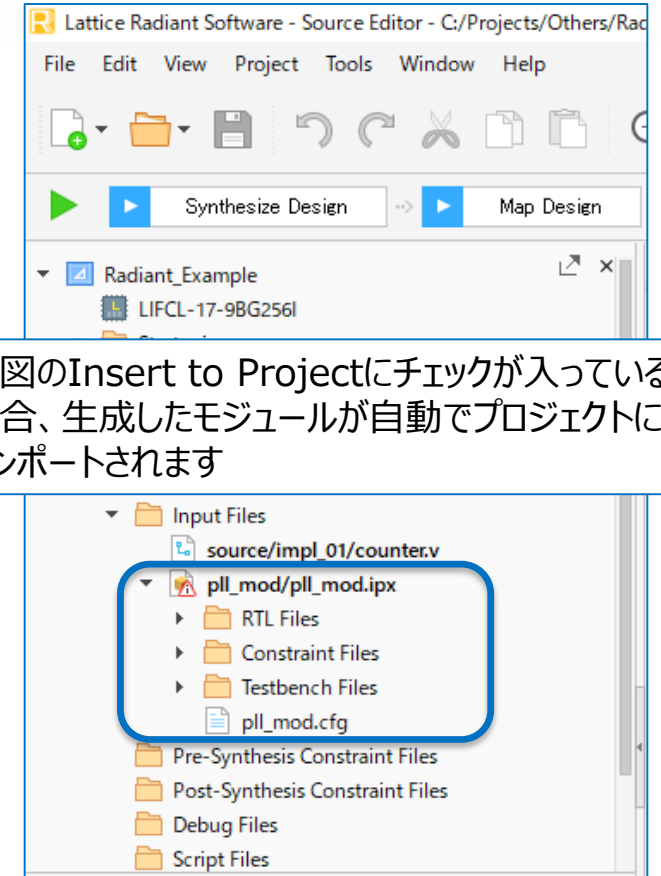
*モジュールのパラメータ内容の詳細に関しては Lattice社HPに掲載されている各デバイスの専用モジュール向けに用意されたApplication Noteや Project Navigator上部のHelpからLattice Radiant Software Helpをご参照下さい

専用モジュールの生成

NOWHERE,
but **HERE.**

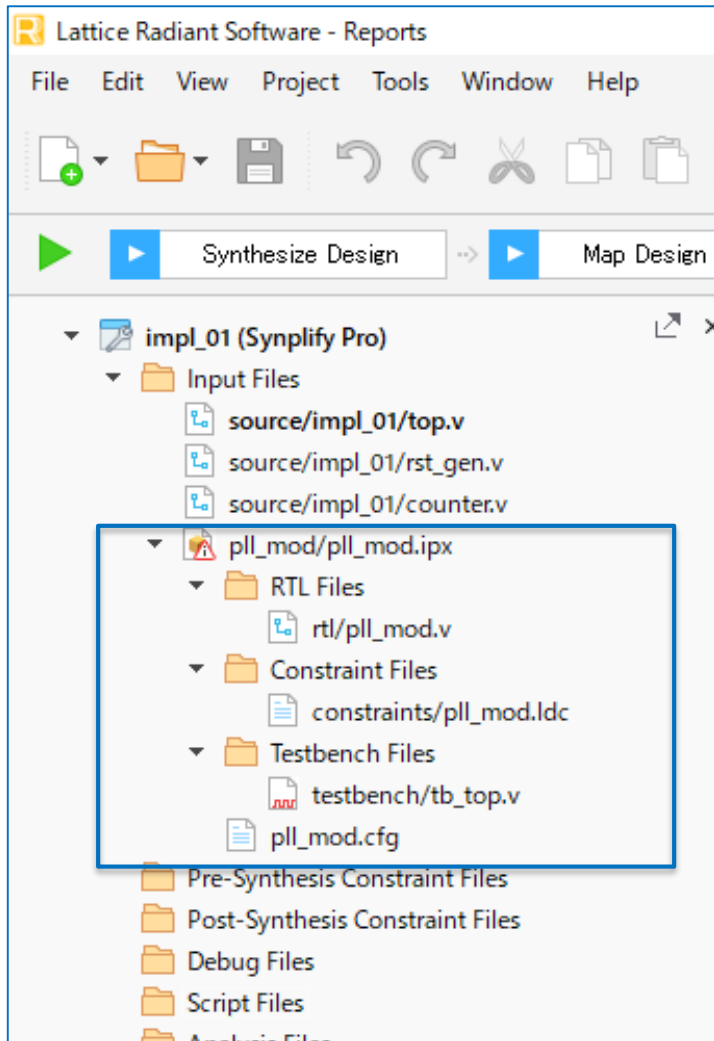


⑧ Finishをクリック



左図のInsert to Projectにチェックが入っている場合、生成したモジュールが自動でプロジェクトにインポートされます

(参考) 専用モジュールのファイル構成



xxx.ipx

→ 生成したモジュールの管理ファイルです。ダブルクリックするとモジュールの編集画面が開きます。このファイルがインポートされている場合のみ以下の構成ファイルがプロジェクトビューに同時表示されます

xxx.v

→ モジュールのソースファイルです。このソースファイルがipxファイルのいずれかがプロジェクトにインポートされている必要があります

xxx.Ldc

→ モジュールに付帯する制約ファイルです

xxx.tb_top.v

→ 生成したモジュールのシミュレーション用テストベンチファイルです

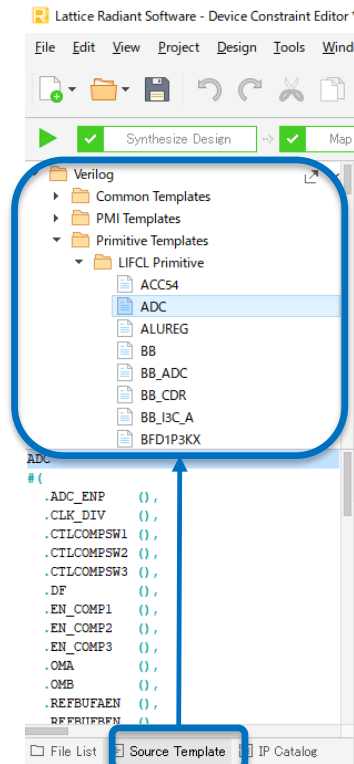
xxx.cfg

→ モジュールのコンフィグレーションファイルです。モジュール構成の概要を示しています

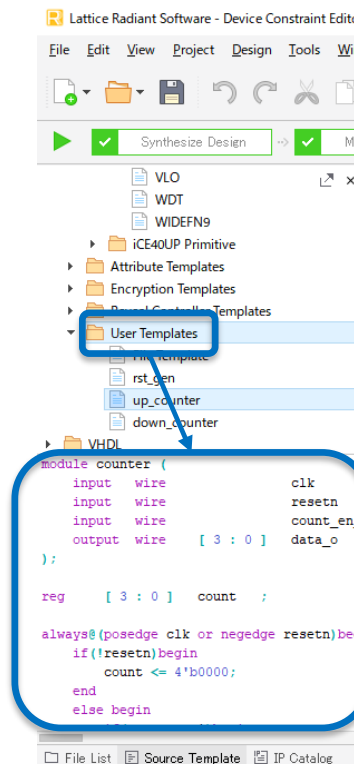
(参考) Source Template



- Project View下部のSource Templateを開くと各デバイスに用意されたコンポーネント（専用モジュールやFF、I/Oバッファ等）のプリミティブ記述を参照したり、ユーザーがよく使うソース記述をテンプレートとして記憶させることができます
- 主にユーザーのテンプレート記憶用途として使用して下さい。こちらに用意されているプリミティブ記述をソースファイルに記述して使用することは特定の場合を除き基本的に推奨致しません。必要な専用モジュールはIP Catalogから生成して使用して下さい。



Source Templateからプリミティブ記述を参照できますが、特定用途を除き、基本的に使用を推奨しておりません。専用マクロはIP Catalogから生成して下さい



User Templateに新規テンプレートを追加し、Copy & Pasteで使用することは問題ありません

デザインエントリー確認



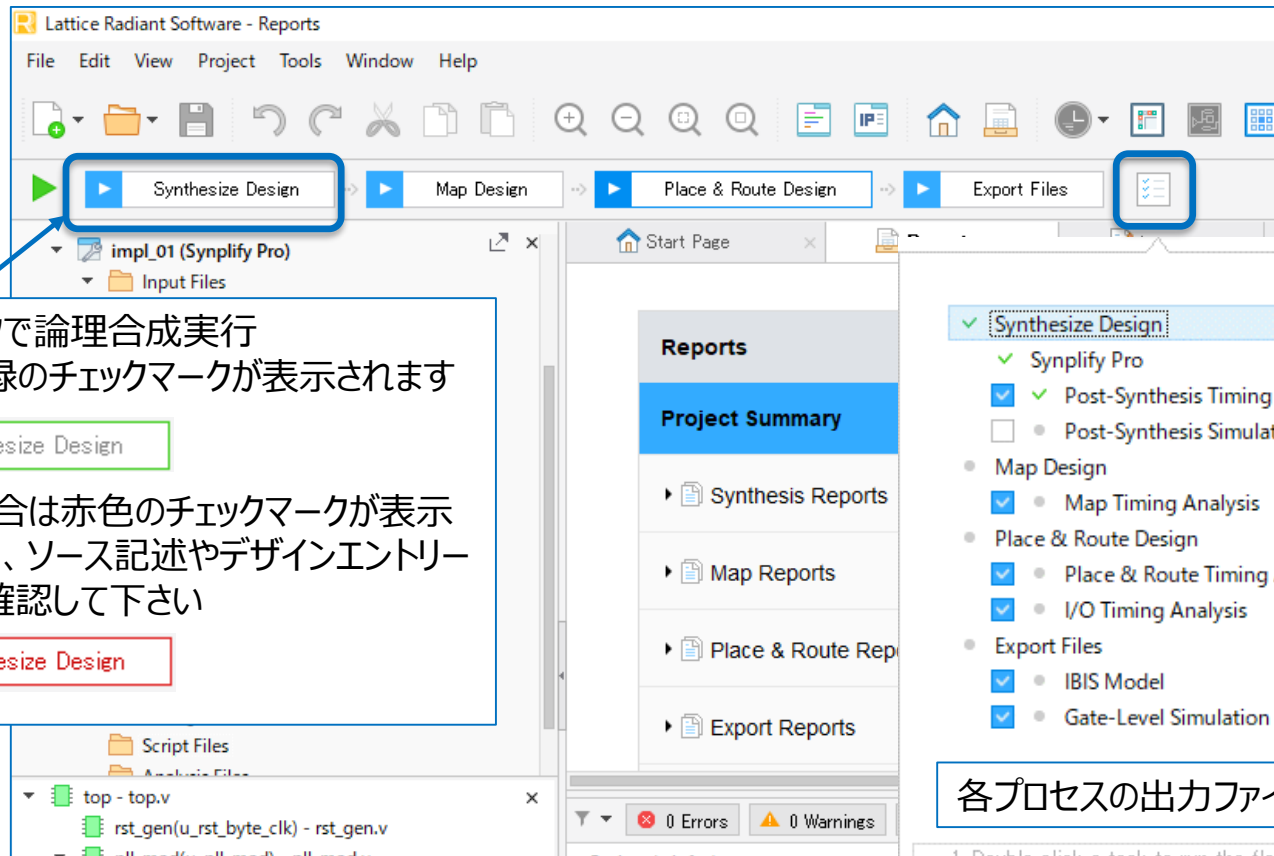
- デザインエントリーが正しく行われた場合、ツールがTopモジュールを自動認識し、太字で表示します
- 同時にProject View下部のモジュール階層構造が正しく表示されます

The screenshot shows the Lattice Radiant Software Reports window. The left pane displays a project tree for 'Radiant_Example'. The 'impl_01 (Synplify Pro)' folder is expanded, showing 'Input Files' and 'pll_mod/pll_mod.ipx'. The file 'source/impl_01/top.v' is highlighted with a blue box and a callout bubble that says 'Topモジュールが太字で表示されます' (Top module is displayed in bold). The right pane shows the 'Reports' section with a list of report categories: Synthesis Reports, Map Reports, Place & Route Reports, and Export Reports. The bottom pane shows the 'Project View' with a list of modules: 'top - top.v', 'rst_gen(u_rst_byte_clk) - rst_gen.v', 'pll_mod(u_pll_mod) - pll_mod.v', 'pll_mod_ipgen_lscpll(lscpll_inst) - pll_mod.v', and 'counter(u_counter) - counter.v'. A callout bubble points to this list with the text 'デザインの階層構造が表示されます' (Design hierarchy structure is displayed).

論理合成の実行



- Process ToolbarのSynthesize Designをクリックして論理合成を実行します
- 論理合成が成功すると緑のチェックマークが表示されます
- 各プロセス実行時に出力するファイルはExport Files右側のアイコンをクリックすることで設定可能です



ダブルクリックで論理合成実行
成功すると緑のチェックマークが表示されます



失敗した場合は赤色のチェックマークが表示
されますので、ソース記述やデザインエントリー
が正しいか確認して下さい




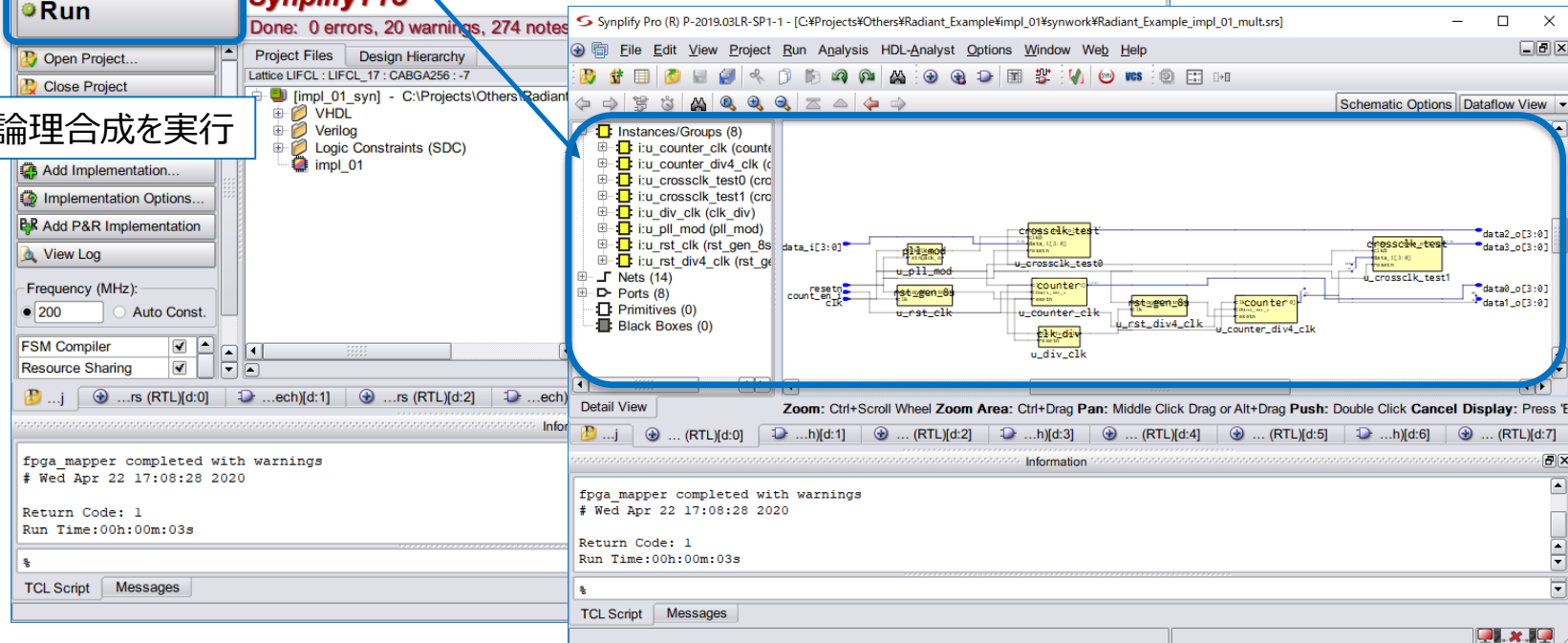
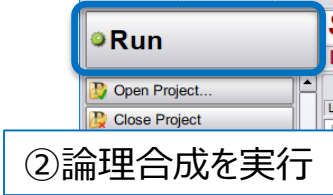
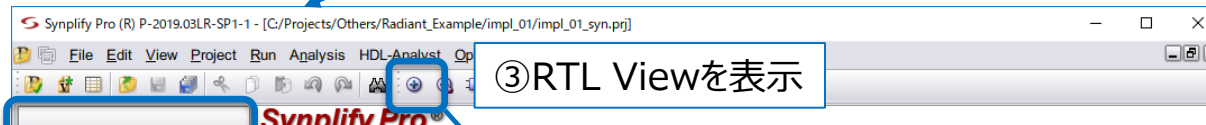
各プロセスの出力ファイルの設定が可能です

1. Double click a task to run the flow
2. Right click to show context menu

(参考) Synplify ProによるRTL Viewの確認




- Synplify Proに付属のRTL Viewで論理合成後の回路がシンボル接続で表示できます
- Process ToolbarのSynplify Proアイコン  をクリックして立ち上げ、論理合成を実行します
- Synplify Pro上のRTL Viewアイコンをクリックします

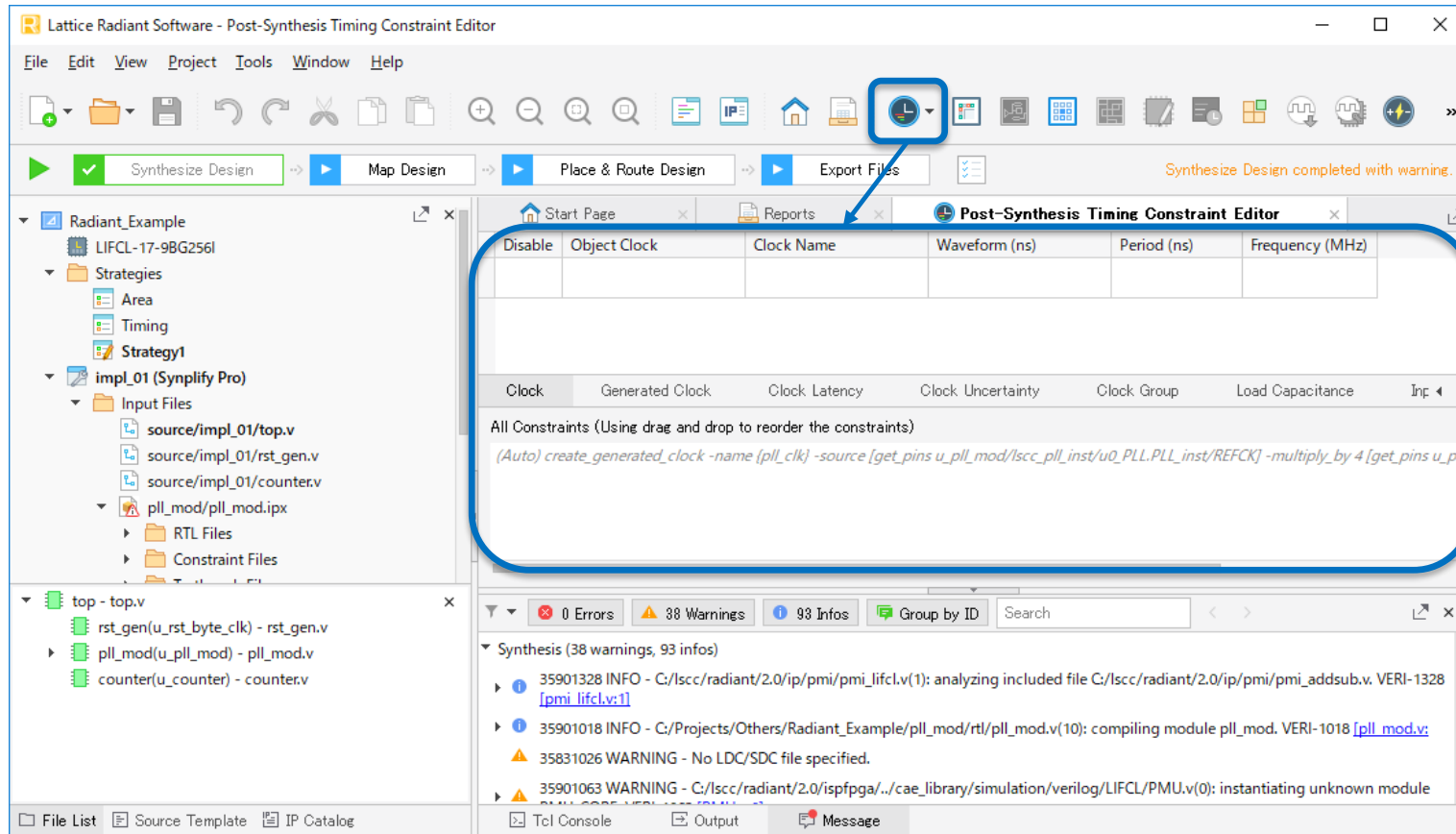


2. タイミング制約の設定



Timing Constraint Editor

- タイミング制約の設定はTiming Constraint Editorで行います
- Synthesize Designが完了している状態でTools > Timing Constraint Editor > Post Synthesis Timing Constraint Editorもしくは、ToolbarのTiming Constraint EditorアイコンからPost Synthesis Timing Constraint Editorを開きます



Fmax（最大動作周波数）制約

- Fmax制約の設定はClockタブもしくはGenerated Clockタブで行います
- Object Clockセルをダブルクリックすると、クロックタイプの選択画面を表示させることができます
- 下図を参照してクロックタイプを選択し、その他のパラメータを設定します
- 制約を削除する場合は右クリックでRemove Rowを選択するか、Disableにチェックを入れることで設定そのものは残したまま制約を外すことができます

Post-Synthesis Timing Constraint Editor

Disable	Object Clock	Clock Name	Waveform (ns)	Period (ns)	Frequency (MHz)
<input type="checkbox"/>	get_ports clk	clk	0;25	50.000	20

Object Edit

Object Type: CLOCKPORT

Available Objects

- clk

Object Type

- CLOCKPORT : ピンからの入力クロック信号
- CLOCKPIN : インスタンスまたはピンからのクロック信号
- CLOCKNET : 内部クロック信号

*ピン入力クロックに制約をかける場合は、CLOCKPINではなくCLOCKPORTを選択して下さい

Clock Name
クロック名。Object Clockを選択すると自動で入力されます

Waveform (ns)
クロックデューティを設定できます。例えば周期50nsのクロックに対し、0;25と設定した場合、0nsで立ち上がり、25nsで立ち下がる設定になり、デューティ比50:50の制約になります。空欄の場合、デューティ比は50:50になります

Period (ns) / Frequency (ns)
クロックの周期 / 周波数を入力します。どちらか片方を入力すると他方は自動入力されます



Fmax (最大動作周波数) 制約

- Generated Clockタブでは内部生成クロックに対して制約を設定できます
- Object Clockセルをダブルクリックすると、クロックタイプの選択画面を表示させることができます
- 内部PLLで生成するクロック等はここに自動で制約が入力されます

Disable	Object Clock	Source	Name	Divide Factor	Multiply Factor	Duty Cycle	Ec
	<code>get_pins u_pll_mod/lsccl_pll_inst/u0_PLL.PLL_inst/CLKOP</code>	<code>get_pins u_pll_mod/lsccl_pll_inst/u0_PLL.PLL_inst/REFCK</code>	<code>pll_clk</code>		4		
<input type="checkbox"/>	<code>get_nets div4_clk</code>	<code>get_ports clk</code>	<code>div4_clk</code>	4			

Generated Clock

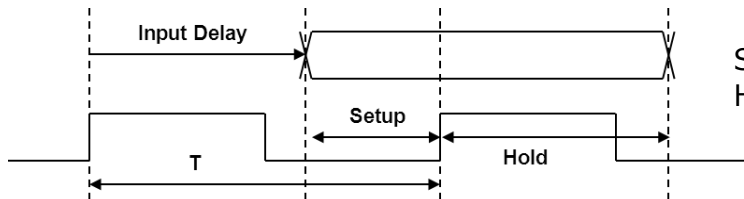
```
All Constraints (Using drag and drop to reorder the constraints)
(Auto) create_generated_clock -name {pll_clk} -source {get_pins u_pll_mod/lsccl_pll_inst/u0_PLL.PLL_inst/REFCK} -multiply_by
create_clock -name {clk} -period 50 -waveform {0 25} {get_ports clk}
create_generated_clock -name {div4_clk} -source {get_ports clk} -divide_by 4 {get_nets div4_clk}
set_false_path -from {get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}} -to {get_ports {data2_o[3] data2_o[2] data2_o[1] data2_o[0]}}
set_input_delay -clock {get_clocks clk} 5 {get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}}
set_output_delay -clock {get_clocks clk} 5 {get_ports {data_o[7] data_o[6] data_o[5] data_o[4] data_o[3] data_o[2] data_o[1] data_o[0]}}
set_load 3 {get_ports {data_o[7] data_o[6] data_o[5] data_o[4] data_o[3] data_o[2] data_o[1] data_o[0]}}
```

- Object Clock
制約対象の内部生成クロック信号を選択します
- Source
その内部生成クロックの元となるソースクロックを選択します
- Name
Object Clockを選択すると自動で入力されます
- Divide Factor / Multiply Factor
ソースクロックから見た分周率 or 低倍率を入力します
- Duty Cycle
Duty比を設定します。入力無しは50%設定です



Input Delayの設定

- Input Delayとは入力データのクロックに対する遅延設定で、クロック周期からSetup Timeを引いた値、つまり前段デバイスのClock to Output Timeに相当します
- 下図を参照して制約を与えるポートと基準クロック、パラメータを設定します



$$\text{Setup Time} = T - \text{Input Delay}$$

$$\text{Hold Time} = \text{Input Delay}$$

Post-Synthesis Timing Constraint Editor

File Edit Window Help

Disable	Constraint Type	Port	Clock	Clock Fall	Max	Min	Add Delay	Value (ns)
<input type="checkbox"/>	set_input_delay	get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}	get_clocks clk	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	5.000
<input type="checkbox"/>	set_input_delay	get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}	get_clocks clk	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	2.000

Clock
 Generated Clock
 Clock Latency
 Clock Uncertainty
 Clock Group
 Load Capacitance
 Input/Output Delay

All Constraints (Using drag and drop to reorder the constraints)

```

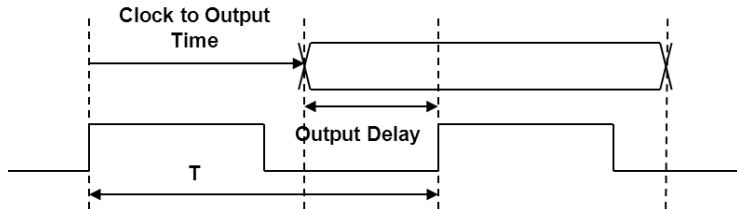
(Auto) create_generated_clock -name {pll_clk} -source [get_pins u_pll_mod/lsccl_pll_inst/u0.PLL.PLL_inst/REFCK] -multiply_by 4 [get_pins u_pll_mod/lsccl_pll_inst/create_clock -name {clk} -period 50 -waveform {0 25} [get_ports clk]
create_generated_clock -name {div4_clk} -source [get_ports clk] -divide_by 4 [get_nets div4_clk]
set_false_path -from [get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}] -to [get_ports {data2_o[3] data2_o[2] data2_o[1] data2_o[0]}]
set_input_delay -clock [get_clocks clk] -max 5 [get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}]
set_load 3 [get_ports {data0_o[7] data0_o[6] data0_o[5] data0_o[4] data0_o[3] data0_o[2] data0_o[1] data0_o[0]}]
set_input_delay -clock [get_clocks clk] -min 2 [get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}]
set_output_delay -clock [get_clocks clk] -max 10 [get_ports {data2_o[3] data2_o[2] data2_o[1] data2_o[0]}]
set_output_delay -clock [get_clocks clk] -min 5 [get_ports {data2_o[3] data2_o[2] data2_o[1] data2_o[0]}]

```

- Constraint Type
ここではInput Delay制約を選択します
- Port
制約対象となるデータ入力ポートを選択します
- Clock
制約の基準クロックを選択します
- Clock Fall
クロックの立下りエッジを基準として制約をかける場合にチェックを入れます
- Max / Min
制約が最大遅延 (Max) に対するものか、最小遅延 (Min) に対するものかを選択します
- Value
遅延量を設定します。デバイスのMin Setup Timeの値が $T - \text{Input Delay}$ (Max) より小さくならないように設定します。また、デバイスのMin Hold TimeがInput Delay (Min) より小さくならないように設定します
- Add Delay
同一データバスに複数の制約をかける際にチェックします。例えば同一のデータバスに同じクロックの立上がりと立下りの両方に対して制約をかける場合、Add Delayにチェックを入れないと前の制約が後に追加されている制約に上書きされます

Output Delayの設定

- Output Delayとは出力データをラッチするクロックエッジのデータに対する遅延の設定で、クロック周期からClock To Output Timeを引いた値、つまり後段デバイスのSetup Timeに相当します
- 下図を参照して制約を与えるポートと基準クロック、パラメータを設定します



Clock to Output Time = T - Output Delay

Post-Synthesis Timing Constraint Editor

File Edit Window Help

Disable	Constraint Type	Port	Clock	Clock Fall	Max	Min	Add Delay	Value (ns)
<input type="checkbox"/>	set_output_delay	get_ports {data2_o[3] data2_o[2] data2_o[1] data2_o[0]}	get_clocks clk	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	10.000
<input type="checkbox"/>	set_output_delay	get_ports {data2_o[3] data2_o[2] data2_o[1] data2_o[0]}	get_clocks clk	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	5.000

Clock
 Generated Clock
 Clock Latency
 Clock Uncertainty
 Clock Group
 Load Capacitance
 Input/Output Delay

All Constraints (Using drag and drop to reorder the constraints)

```

create_generated_clock -name {div4_clk} -source {get_ports clk} -divide_by 4 {get_nets div4_clk}
set_false_path -from {get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}} -to {get_ports {data2_o[3] data2_o[2] data2_o[1] data2_o[0]}}
set_input_delay -clock {get_clocks clk} -max 5 {get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}}
set_load 3 {get_ports {data0_o[7] data0_o[6] data0_o[5] data0_o[4] data0_o[3] data0_o[2] data0_o[1] data0_o[0]}}
set_input_delay -clock {get_clocks clk} -min 2 {get_ports {data_i[3] data_i[2] data_i[1] data_i[0]}}
set_output_delay -clock {get_clocks clk} -max 10 {get_ports {data2_o[3] data2_o[2] data2_o[1] data2_o[0]}}
set_output_delay -clock {get_clocks clk} -min 5 {get_ports {data2_o[3] data2_o[2] data2_o[1] data2_o[0]}}
set_clock_groups -group {get_clocks clk} -group {get_clocks div4_clk} -logically_exclusive
    
```

■ Constraint Type

ここではOutput Delay制約を選択します

■ Port

制約対象となるデータ出力ポートを選択します

■ Clock

制約の基準クロックを選択します

■ Clock Fall

クロックの立下りエッジを基準として制約をかける場合にチェックを入れます

■ Max / Min

制約が最大遅延 (Max) に対するものか、最小遅延 (Min) に対するものかを選択します

■ Value

遅延量を設定します。後段デバイスのMin Setup Timeの値よりもOutput Delay(Min)が小さくならないように設定します。また、後段デバイスのMin Hold TimeよりもT - Output Delay(Max)が小さくならないように設定します

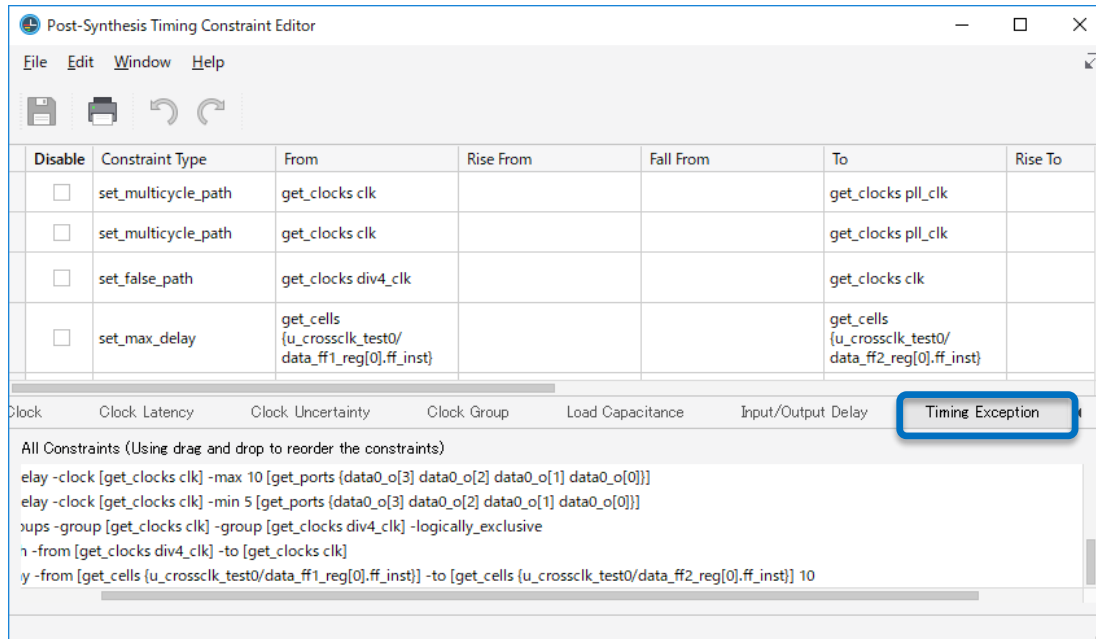
■ Add Delay

前ページを参照して下さい

その他のタイミング制約 (Min/Max Delay, Multicycle, False Path)



- 詳細なパス間の最大・最小遅延制約、マルチサイクル制約、False Path制約はTiming Exceptionタブから設定可能です
- 非同期クロックメインのデータ受け渡しの際のタイミング制約や、解析対象から外したいパス等はここから設定します



■ Constraint Type

- set_max_delay : 指定パス間の最大遅延制約設定
- set_min_delay : 指定パス間の最小遅延制約設定
- set_multicycle_path : マルチサイクルパス設定
- set_false_path : 解析対象外パスの設定

■ From / To

制約の起点と終点を設定します

■ Max / Min delay

Max Delay制約、Min Delay制約の制約値を設定します

■ Setup / Hold

マルチサイクル制約において、サイクル数Nを増すエッジをSetup側解析エッジにするかHold側解析エッジにするか選択します
(左図の解析エッジ参照) Setup側に制約をかけると、同時にHold側に自動で制約がかかり、タイミングエラーとなる場合があります。この場合Hold側にも個別に制約をかける必要があります

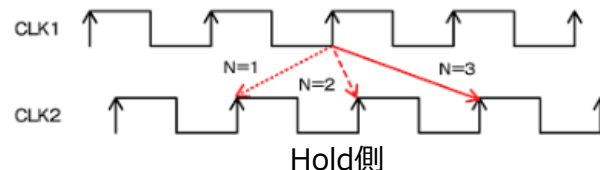
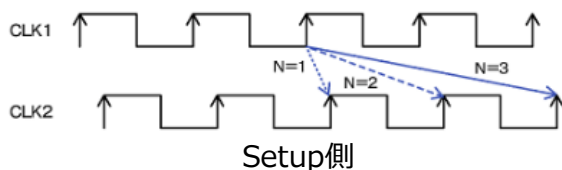
■ Start / End

マルチサイクル制約のサイクル数を送信側クロックでカウントする(start)か、受信側クロックでカウントするか(end)を選択します

■ Multiplier

マルチサイクル制約のサイクル数Nを設定します

*タイミング解析における解析エッジ





その他の制約(Load Capacitance)

- 出力ポートに対して、接続先の負荷容量の設定が可能です。出力タイミング解析時に負荷容量を加味した解析が行われるようになります
- タイミング解析の他に、SSOアナライザを使用した同時スイッチングノイズの影響度解析にもここで設定された負荷容量が使用されます

The screenshot shows the 'Post-Synthesis Timing Constraint Editor' window. A table lists constraints with columns for 'Disable', 'Port', and 'Load'. The first row is selected, showing a port list and a load value of 3.000. A blue box highlights the 'Port' and 'Load' columns with instructions. The 'Load Capacitance' tab is selected in the bottom navigation bar. The bottom pane shows a list of constraints with a scroll bar.

Disable	Port	Load
<input type="checkbox"/>	get_ports {data0_o[7] data0_o[6] data0_o[5] data0_o[4] data0_o[3] data0_o[2] data0_o[1] data0_o[0]}	3.000

- Port
負荷容量設定を与えるポートを選択します
- Load
負荷容量値 (pf) を設定します

Navigation tabs: Clock, Generated Clock, Clock Latency, Clock Uncertainty, Clock Group, **Load Capacitance**, Input/Output Delay, Timing Ex


All Constraints (Using drag and drop to reorder the constraints)

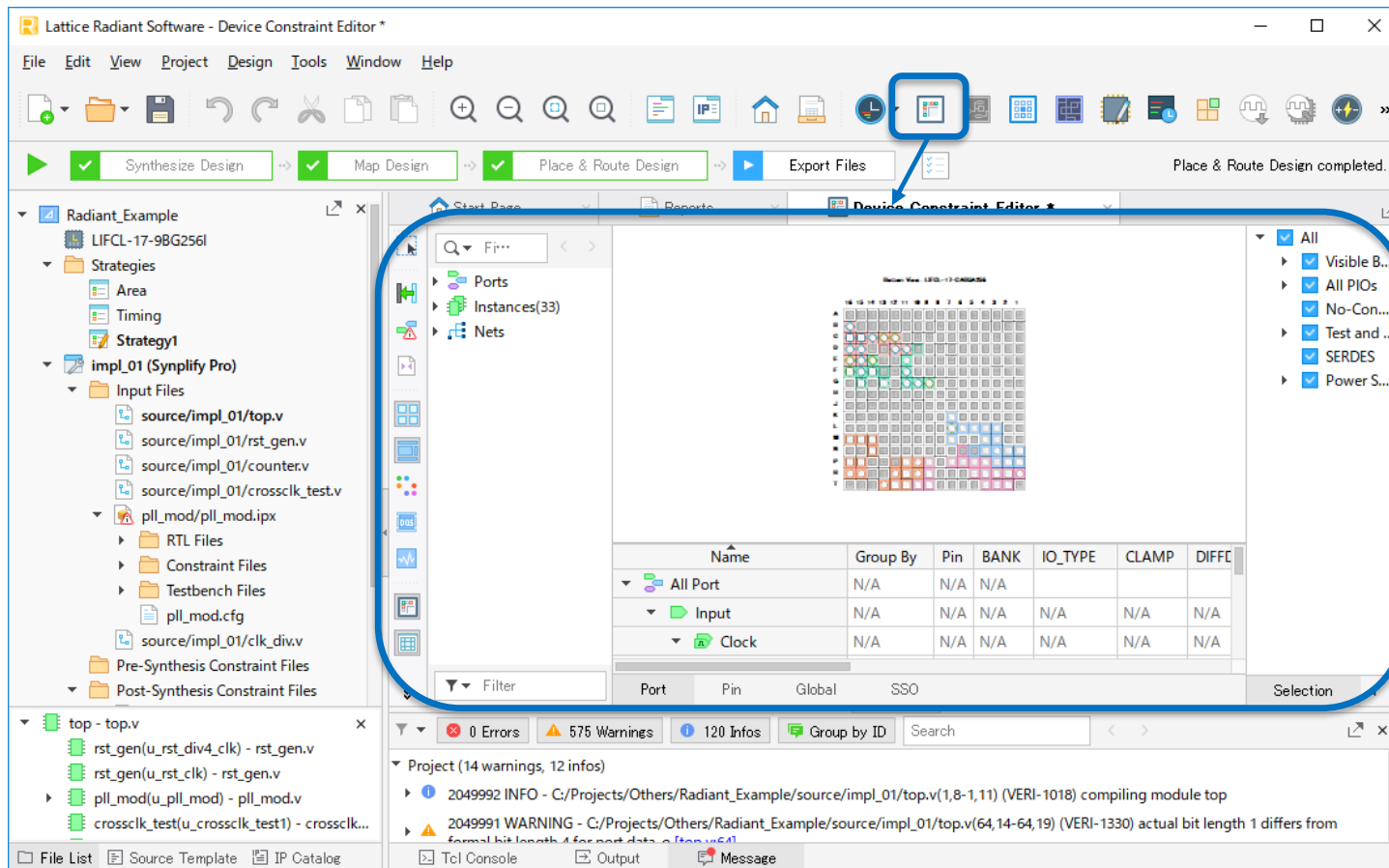
```
(Auto) create_generated_clock -name {pll_clk} -source [get_pins u_pll_mod/lsc_pll_inst/u0_PLL.PLL_inst/REFCK] -multiply_by 4 [get_pins u_pll_mod/lsc_pll_inst/u0_PLL.PLL_inst/REFCK]
create_clock -name {clk} -period 50 -waveform {0 25} [get_ports clk]
create_generated_clock -name {div4_clk} -source [get_ports clk] -divide_by 4 [get_nets div4_clk]
set_multicycle_path -setup -end -from [get_clocks clk] -to [get_clocks pll_clk] 2
set_multicycle_path -hold -end -from [get_clocks clk] -to [get_clocks pll_clk] 2
```


3. I/Oアサイメント、デバイス全般の設定

Device Constraint Editor



- 入出力信号のI/OへのアサインメントDevice Constraint Editorで行います
- Synthesize Designが完了している状態でTools > Device Constraint Editorもしくは、ToolbarのDevice Constraint Editorアイコンから開きます



Device Constraint Editor



- Device Constraint EditorのGUIは以下のようになっています

■ Netlist View
デザインに存在するポート、内部信号、内部コンポーネントの一覧です

■ Package View
デバイスパッケージをそのまま可視化したものです。Netlist Viewからドラッグ & ドロップでピンアサインできます

■ Device View
Package Viewに表示するピンの選択ができます

■ Spreadsheet View
表形式のピンアサイン含むI/O設定およびデバイス全般の設定ビューです

このアイコンで各ビューの表示、非表示を選択できます

Name	DRIVE	GLITCHFILTER	HYSTERESIS	OPENDRAIN	PULLMODE	SLEWRATE	TERMINATION
All Port							
Input	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Clock	N/A	N/A	N/A	N/A	N/A	N/A	N/A
clk	NA	ON	ON	OFF	DOWN	NA	OFF
count_en_i	NA	ON					
data_i[0]	NA	ON					
data_i[1]	NA	ON					
...	NA	ON					

Spreadsheet View



- Spreadsheet ViewのPortタブではピンアサインやI/O Type（電圧スタンダード）等を表形式のシートで細かく設定できます

Device Constraint Editor

File Edit View Design Window Help

Name	Group By	Pin	BANK	IO_TYPE	CLAMP	DIFFDRIVE	DIFFRESISTOR	DRIVE	GLITCHFILTER	HYSTERESIS	OPENDRAIN	PULLMODE	SLEWRATE	TERMINATION	VREF
All Port	N/A	N/A	N/A												
Input	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Clock	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
clk	N/A	L7	5	LVCMOS33	ON	NA	OFF	NA	ON	ON	OFF	DOWN	NA	OFF	
count_en_i	N/A	F16	1	LVCMOS33	ON	NA	OFF	NA	ON	ON	OFF	DOWN	NA	OFF	
data_i[0]	input_gr...	G9	1	LVCMOS33	ON	NA	OFF	NA	ON	ON	OFF	DOWN	NA	OFF	
data_i[1]	input_gr...	C13	0	LVCMOS33	ON	NA	OFF	NA	ON	ON	OFF	DOWN	NA	OFF	
data_i[2]	input_gr...	E16	0												
data_i[3]	input_gr...	E14	1												
resetn	N/A	C12	0												
Output	N/A	N/A	N/A												
data_o[0]	N/A	F15	1												
data_o[1]	N/A	G11	1												
data_o[2]	N/A	G13	1												
data_o[3]	N/A	F14	1												
data_1_o[0]	N/A	G15	1												
data_1_o[1]	N/A	C15	0												
data_1_o[2]	N/A	C14	0												
data_1_o[3]	N/A	D16	0												
data_2_o[0]	N/A	G10	1												
data_2_o[1]	N/A	D12	0												
data_2_o[2]	N/A	E15	0												
data_2_o[3]	N/A	F11	1												
data_3_o[0]	N/A	D11	0												

Port Pin Global SSO

各設定項目の内容は以下の通りです

*デバイス及びI/Oバンクや入出力設定によって設定可否が変わります

PIN : 配置したいピン番号を入力します

BANK : 配置したいI/Oバンクを入力します(PINを設定すると、こちらも自動的に設定されます)

IO_TYPE : I/Oスタンダードを設定します

CLAMP : クランプダイオードのON/OFF設定です

DIFFDRIVE : 差動出力のドライブ電流です

DIFFRESISTOR : 差動入力の内部終端の使用・不使用を選択します

DRIVE : ドライブ電流値を設定します。出力ピンにのみ設定可能です

HYSTERISYS : 入力ピンのヒステリシス設定です

OPENDRAIN : 出力ピンをオープンドレインにします（High出力がHi-z出力になります）

PULLMODE : 内部プルモードを設定します

SLEWRATE : 各デバイスに用意された出力スルーレートを設定できます。設定の差はIBISモデルで確認できます

VREF : 参照電圧が必要な入力の参照先設定。参照先はGlobal Preferenceで設定します

Spreadsheet View



- Spreadsheet ViewのPinタブではI/OのDual Functionを確認しながらピンアサインが可能です
- I/O TYPE等の細かい設定はPortタブで行う必要があります

The screenshot shows the Device Constraint Editor interface. The main window displays a table of pins and their configurations. A callout box points to the 'Signal Name' column, indicating a double-click action. An 'Assign Pins' dialog box is open, showing a list of pins and their assigned signals. A callout box points to the 'Signal Name' column in the dialog, indicating a drag-and-drop action. Another callout box points to the 'Assign' button, indicating a click action.

Pin	Pad Name	Dual Function	Polarity	BANK	BANK_VCC	IO_TYPE	Signal Name	Signal Type
Bank0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Diff_D16_D15	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
D15	FIO:PT63B	MD2/SD6/S2_OUT	Neg	0	Auto	LVCNOS33(LVCNOS33)	data0_o[0]...	(Output)
D16	FIO:PT63A	MISO/MD1/SD5/S2_IN/OSC_HI	Pos	0	Auto	(LVCNOS33)		
Diff_E15_E16	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
E15	FIO:PT67A	S7_IN	Pos	0	Auto	(LVCNOS33)	data2_o[2]	(Output)
E16	FIO:PT67B	S7_OUT						
Diff_D12_C13	N/A	N/A						
C13	FIO:PT65B	MCSNO/MSDO/S6_OUT						
D12	FIO:PT65A	MD3/SD7/S6_IN/OSC_BURST						
Diff_C15_C14	N/A	N/A						
C14	FIO:PT61B	MOSI/MD0/SD4/S1_OUT						
C15	FIO:PT61A	MCSN/S1_IN/PCLKT0_1						
Diff_C16_B16	N/A	N/A						
B16	FIO:PT59B	DONE/S0_OUT						
C16	FIO:PT59A	MCLK/S0_IN/PCLKT0_0						
Diff_D11_C12	N/A	N/A						
C12	FIO:PT57B	PROGRAMN						
D11	FIO:PT57A	INITN						
Bank1	N/A	N/A						
Diff_F16_G15	N/A	N/A						
F16	FIO:PR13A	PCLKT1_1						
G15	FIO:PR13B	PCLKT1_0/EIO						
Diff_F14_F15	N/A	N/A						

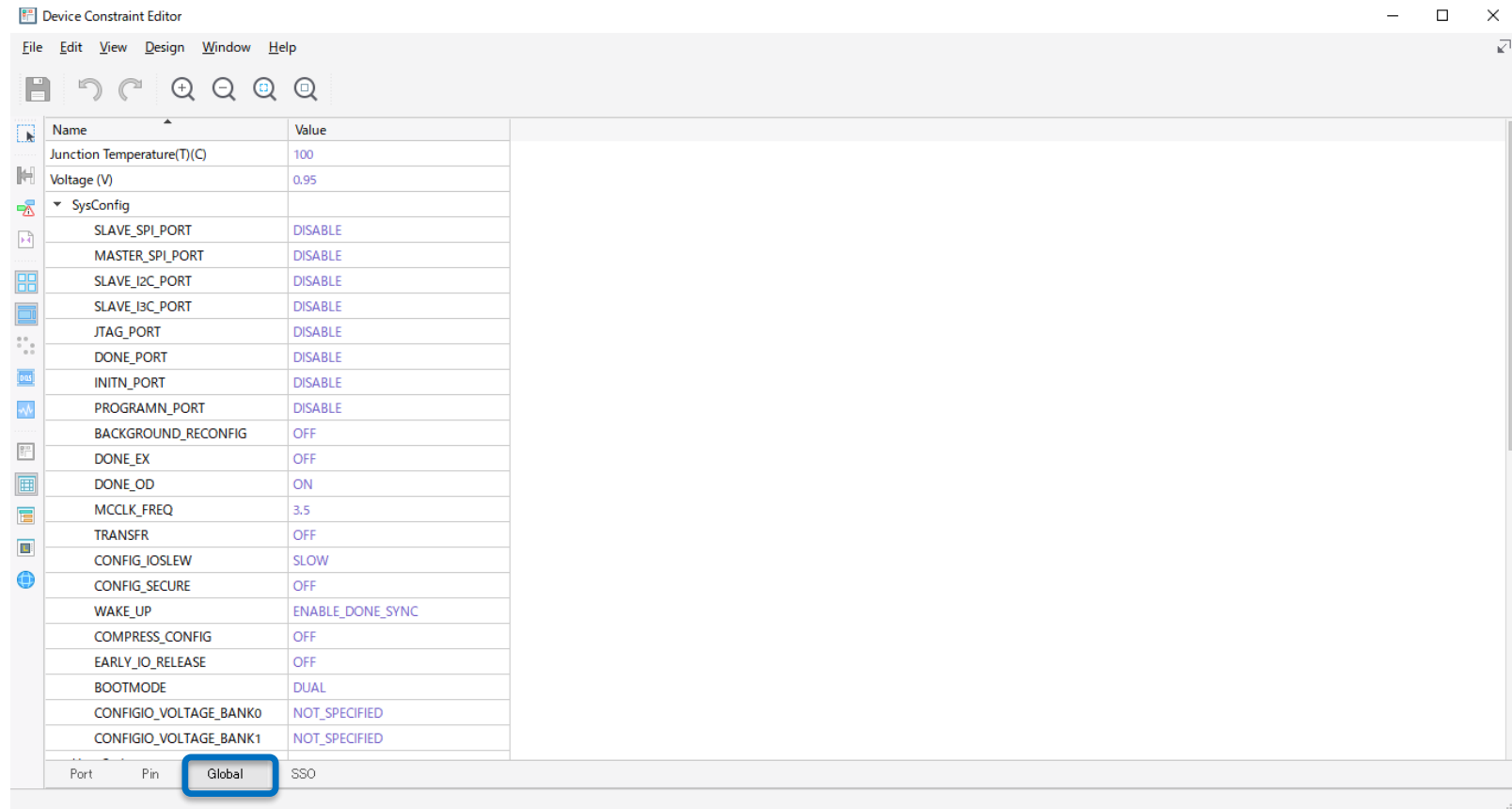
Annotations:

- ① Signal Nameの欄をダブルクリック
- ② アサインしたい信号をドラッグ & ドロップ
- ③ Assignをクリック

Spreadsheet View



- Spreadsheet ViewのGlobalタブではデバイスのコンフィグレーション関連ピンの有効・無効設定等の全般的な設定を行うことができます



Package View



- Package ViewではNetlist Viewに表示されているPortの信号をドラッグ&ドラッグによってグラフィカルにピンアサインが行えます
- ViewでTop View/Bottom Viewの切り替えや、差動ペアの表示も可能です

The screenshot shows the Device Constraint Editor interface. On the left, a tree view shows ports like 'data0_o[0] (F15)'. A blue box highlights this port, with a blue arrow pointing to a specific pin in the package grid. A text box next to the arrow says 'アサインしたいポートをドラッグ&ドロップ' (Drag & drop the port you want to assign). The package grid is titled 'Bottom View : LIFCL-17-CABGA256' and shows a grid of pins labeled A through T and 1 through 16. On the right, the 'View' menu is open, showing options like 'Zoom In', 'Zoom Out', 'Show Differential Pairs Ctrl+D', 'Show Default Value', 'Display IO Placement', 'Top View', and 'Bottom View'. A blue box highlights the 'Bottom View' option, with a text box below it saying 'Top View (デバイスを表側から見た配置)とBottom View (デバイスを裏側から見た配置)の切り替え' (Switching between Top View (device configuration seen from the front) and Bottom View (device configuration seen from the back)). Another blue box highlights the 'Show Differential Pairs Ctrl+D' option, with a text box above it saying '差動ペアの表示' (Display differential pairs).

(参考) SSO解析



- Spreadsheet ViewのSSOタブでSSO（同時スイッチング出力）の影響確認用の設定を行い、Package View上で影響度の確認を行うことができます
- 予めTiming Constraint EditorのLoad Capacitance設定で出力先の負荷容量の制約をかけておく必要があります

Name	Outload (pF)	Ground plane PCB noise(mV)	Switching ID	SSO Allowance(%)	Power plane PCB noise(mV)
All Port	N/A	0.00		100.00	-0.00
Output	N/A	N/A	N/A	N/A	
data0_o[0]	3.000000	50	group0	100.00	
data0_o[1]	3.000000	50	group0	100.00	
data0_o[2]	3.000000	50	group0	100.00	-50
data0_o[3]	3.000000	50	group0	100.00	-50
data1_o[0]	0.00	0.00		100.00	-0.00
data3_o[1]	0.00	0.00		100.00	-0.00
data3_o[2]	0.00	0.00		100.00	-0.00
data3_o[3]	0.00	0.00		100.00	-0.00

① Spreadsheet ViewのSSOタブで必要な設定を行います

Outload(pf) : 出力先の負荷容量制約。Timing Constraint Editorで設定
Ground plane PCB noise (mV) : PCB上でGNDプレーンに想定されるノイズ量
Switching ID : 同時スイッチングする出力グループに同じグループ名を設定
SSO Allowance (%) : デバイスの許容値の何%を超えるとエラー表示するか設定
Power plane PCB noise (mV) : PCB上で電源プレーンに想定されるノイズ量

Port Pin Global SSO

② Package Viewを表示し、Show SSO Viewを選択

Bottom View : LIFCL-17-CABGA256

SSO Analysis : Pin Results

	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
A																
B																
C																
D																
E																
F																
G																
H																
J																
K																
L																
M																
N																
P																
R																
T																

③ パッドにカーソルを合わせると解析結果が表示されます

Pin name:G13
Pad name:PR11B
Bank:1
Pdc locked:data0_o[2]
Differential pin:NEG
Complement pin:
Pin name:G11
Pad name:PR11A
SSO Status:
Pin Result:Passed
Pin SSO Drop: -56.99 mV, 4.38%
Pin SSO Bounce: 59.66 mV, 7.46%
Bank Result:Passed
Bank Based SSO: 59.66 mV
Dual Function:PCLKT1_2/EIO
IOBuf info:

4. デザインコンパイル～レポート確認

配置配線の実行



- Process ToolbarのPlace & Route Designをクリックして配置配線を実行します
- Map Design（論理回路の実リソースへのマッピング）から順番に行っても良いですが、Place & Route Designを実行すれば自動的にMap Designも実行されます
- Map Design, Place & Route Designが成功すると緑のチェックマークが表示されます

ダブルクリックで実行
成功すると緑のチェックマークが表示されます

失敗した場合は赤色のチェックマークが表示
されますので、リソース使用量や配線ルール
等に問題が無いか確認が必要です

Implementation Name:	impl_01	Performance Grad
Strategy Name:	Strategy1	Operating Condi
Part Number:	LIFCL-17-9BG2561	Synthesis:
Family:	LIFCL	Timing Errors:
Device:	LIFCL-17	Project Created:
Package:	CABGA256	Project Updated:
Project File:	C:/Projects/Others/Radiant_Example/Radiant_Examp	
Implementation Location:	C:/Projects/Others/Radiant_Example/impl_01	

70 Warnings 117 Infos Group by ID Search


1182001 INFO - SSO analyze finished successfully.

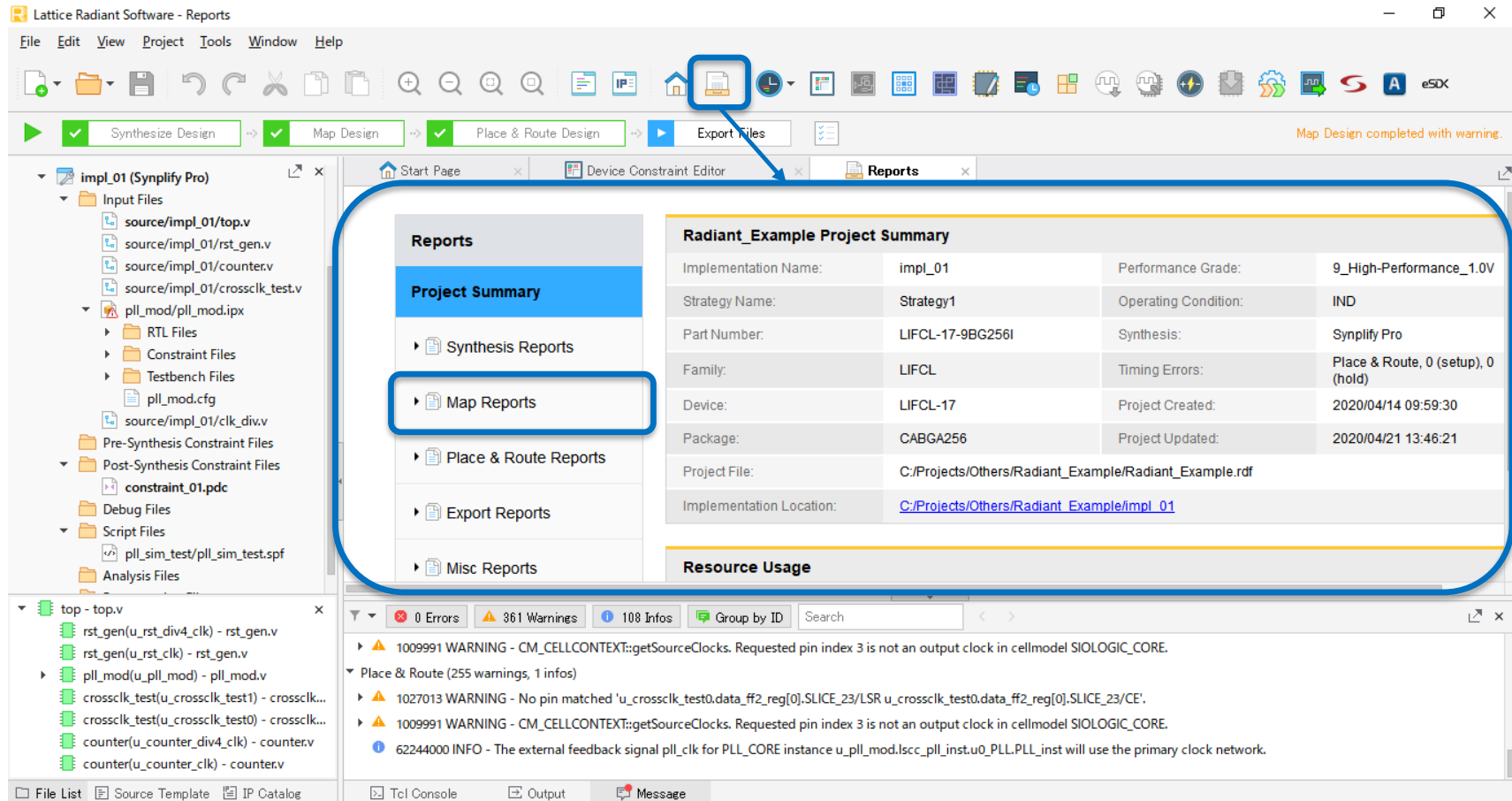
1027013 WARNING - No port matched 'data0_o[7] data0_o[6] data0_o[5] data0_o[4]'.

71003005 WARNING - The clock port [resetn] is assigned to a non clock dedicated pin [C12], which might affect the clock performance. Use dedicated clock resources for the port.

Map Reportの確認



- Map Reportではリソース使用状況の確認ができます
- Reportsアイコンをクリックし、レポート画面を表示します
- Reportsの項目でMap Reportsを選択します



The screenshot shows the Lattice Radiant Software Reports window. The Reports menu is open, and the 'Map Reports' option is highlighted. The Project Summary table is visible, showing details for the 'impl_01' project.

Radiant_Example Project Summary			
Implementation Name:	impl_01	Performance Grade:	9_High-Performance_1.0V
Strategy Name:	Strategy1	Operating Condition:	IND
Part Number:	LIFCL-17-9BG256I	Synthesis:	Synplify Pro
Family:	LIFCL	Timing Errors:	Place & Route, 0 (setup), 0 (hold)
Device:	LIFCL-17	Project Created:	2020/04/14 09:59:30
Package:	CABGA256	Project Updated:	2020/04/21 13:46:21
Project File:	C:/Projects/Others/Radiant_Example/Radiant_Example.rdf		
Implementation Location:	C:/Projects/Others/Radiant_Example/impl_01		

Resource Usage

0 Errors, 361 Warnings, 108 Infos

- 1009991 WARNING - CM_CELLCONTEXT::getSourceClocks. Requested pin index 3 is not an output clock in cellmodel SIOLOGIC_CORE.
- Place & Route (255 warnings, 1 infos)
- 1027013 WARNING - No pin matched 'u_crossclk_test0.data_ff2_reg[0].SLICE_23/LSR u_crossclk_test0.data_ff2_reg[0].SLICE_23/CE'.
- 1009991 WARNING - CM_CELLCONTEXT::getSourceClocks. Requested pin index 3 is not an output clock in cellmodel SIOLOGIC_CORE.
- 62244000 INFO - The external feedback signal pll_clk for PLL_CORE instance u_pll_mod.lsc_pll_inst.u0.PLL_inst will use the primary clock network.

Map (Map Report) の確認

- デバイスの基本的なリソース（LUT、SLICE、Register、EBR、DSP、PLL、OSC、I/O Logic等）の使用状況の確認はMapのDesign Summaryで行います
- 右上のContentsボタンにカーソルを合わせると、レポートコンテンツが表示されます
- 確認したいコンテンツをクリックすると、その記載箇所にジャンプします

The screenshot shows the 'Reports' window with the 'Map' report selected. The 'Design Summary' section is visible, listing various resources and their usage. A 'Contents' button is located in the top right corner of the report area. A 'Contents' panel is open, showing a list of report sections. Blue arrows indicate the flow from the 'Contents' button to the 'Design Summary' section in the report and to the 'Design Summary' item in the 'Contents' panel.

Design Summary

Number of registers:	39 out of 14037 (<1%)
Number of SLICE registers:	31 out of 13824 (<1%)
Number of PIO Input registers:	4 out of 7
Number of PIO Output registers:	4 out of 7
Number of PIO Tri-State registers:	0 out of 7
Number of LUT4s:	9 out of 13824 (<1%)
Number used as logic LUT4s:	
Number used as distributed RAM:	
Number used as ripple logic:	
Number of PIOs used:	23 out of 71 (32%)
Number of PIOs used for single ended IO:	2
Number of PIO pairs used for differential IO:	
Number allocated to regular speed PIOs:	22 out of 23
Number allocated to high speed PIOs:	1 out of 23
Number of Dedicated IO used for ADC/PCIE/DPHY:	0 out of 23
Number of IDDR/ODDR/TDDR functions used:	0 out of 23
Number of IOs using at least one DDR function:	0 (0%)
Number of Block RAMs:	0 out of 24 (0%)
Number of Large RAMs:	0 out of 5 (0%)
Number of Logical DSP Functions:	
Number of Pre-Adders (9+9):	0 out of 48 (0%)
Number of Multipliers (18x18):	0 out of 24 (0%)

Contents

- [Design Information](#)
- [Design Summary](#)
- [Design Errors/warnings](#)
- [IO \(PIO\) Attributes](#)
- [Removed logic](#)
- [PLL/DLL Summary](#)
- [ASIC Components](#)
- [GSR Usage](#)
- [Run Time and Memory Usage](#)

Place & Route Reportの確認



- Place & Route Reportではリソース使用状況の確認ができます
- Reportsの項目でPlace & Route Reportsを選択します

The screenshot shows the Lattice Radiant Software Reports window. The 'Reports' menu is open, and 'Place & Route Reports' is selected. The 'Radiant_Example Project Summary' table is visible, along with the 'Resource Usage' section.

Radiant_Example Project Summary			
Implementation Name:	impl_01	Performance Grade:	9_High-Performance_1.0V
Strategy Name:	Strategy1	Operating Condition:	IND
Part Number:	LIFCL-17-9BG256I	Synthesis:	Synplify Pro
Family:	LIFCL	Timing Errors:	Place & Route, 0 (setup), 0 (hold)
Device:	LIFCL-17	Project Created:	2020/04/14 09:59:30
Package:	CABGA256	Project Updated:	2020/04/21 13:46:21
Project File:	C:/Projects/Others/Radiant_Example/Radiant_Example.rdf		
Implementation Location:	C:/Projects/Others/Radiant_Example/impl_01		

Resource Usage	
0 Errors	361 Warnings
108 Infos	

Warnings:

- 1009991 WARNING - CM_CELLCONTEXT::getSourceClocks. Requested pin index 3 is not an output clock in cellmodel SIOLOGIC_CORE.
- 1027013 WARNING - No pin matched 'u_crossclk_test0.data_ff2_reg[0].SLICE_23/LSR u_crossclk_test0.data_ff2_reg[0].SLICE_23/CE'.
- 1009991 WARNING - CM_CELLCONTEXT::getSourceClocks. Requested pin index 3 is not an output clock in cellmodel SIOLOGIC_CORE.
- 62244000 INFO - The external feedback signal pll_clk for PLL_CORE instance u_pll_mod.lsccl_pll_inst.u0.PLL_inst will use the primary clock network.

Place & Route (Place & Route Report) の確認



- Place & Routeではデザインが使用しているリソースの使用率が確認できます
- リソースの使用率確認は基本的にはMap Reportで確認しますが、Place & RouteではMap Reportで確認できないクロック関連リソースの使用状況やクロック専用配線のアサイン状況が確認できます

The screenshot shows the 'Reports' window in a design tool. The left sidebar lists various report categories, with 'Place & Route' selected and highlighted in blue. The main content area displays the 'Place & Route' report, which includes a 'Clock Report' section. A blue arrow points from the 'Clock Report' link in the 'Contents' panel to the 'Clock Report' section in the main report. The 'Clock Report' section contains the following information:

Global Clock Resources:

CLK_PIN	: 1 out of 17 (5%)
PLL	: 1 out of 2 (50%)
DCS	: 0 out of 1 (0%)
DCC	: 0 out of 62 (0%)
ECLKDIV	: 0 out of 12 (0%)
PCLKDIV	: 0 out of 1 (0%)
OSC	: 0 out of 1 (0%)
DPHY	: 0 out of 2 (0%)

Global Clocks:

PRIMARY "pll_clk" from CLKOP on comp "u_pll_mod.lscg_pll_inst.u0_PLL.PLL_inst" on PLL site
PRIMARY "clk_c" from comp "clk" on CLK_PIN site "L7 (PB20A)", clk load = 23, ce load = 0,
PRIMARY "div4_clk" from Q0 on comp "u_div_clk.SLICE_17" on site "R20C37A", clk load = 9, c

PRIMARY : 3 out of 16 (18%)

Edge Clocks:

No edge clock selected.



Signal/Pad (Place & Route Report) の確認

- Signal/Padでは配置配線後の実際のピン配置結果、Buffer Typeやその他I/O設定の反映結果が確認できます

Reports

File Edit View Window

Reports

- Project Summary
- Synthesis Reports
- Map Reports
- Place & Route Reports
 - Place & Route
 - Signal/Pad**
 - Place & Route Timing Analysis
 - I/O Timing Analysis

Signal/Pad

Wed Apr 22 14:16:05 2020

Pinout by Port Name:

Port Name	Pin/Bank	Buffer Type	Site	Properties
clk	L7/5	LVC MOS18H_IN	PB20A	SLEW:NA PULL:DOWN
count_en_i	F16/1	LVC MOS33_IN	PR13A	SLEW:NA PULL:DOWN
data0_o[0]	F15/1	LVC MOS33_OUT	PR9B	DRIVE:8mA SLEW:SLOW
data0_o[1]	G11/1	LVC MOS33_OUT	PR11A	DRIVE:8mA SLEW:SLOW
data0_o[2]	G13/1	LVC MOS33_OUT	PR11B	DRIVE:8mA SLEW:SLOW
data0_o[3]	F14/1	LVC MOS33_OUT	PR9A	DRIVE:8mA SLEW:SLOW
data1_o[0]	G15/1	LVC MOS33_OUT	PR13B	DRIVE:8mA SLEW:SLOW
data1_o[1]	C15/0	LVC MOS33_OUT	PT61A	DRIVE:8mA SLEW:SLOW
data1_o[2]	C14/0	LVC MOS33_OUT	PT61B	DRIVE:8mA SLEW:SLOW
data1_o[3]	D16/0	LVC MOS33_OUT	PT63A	DRIVE:8mA SLEW:SLOW
data2_o[0]	G10/1	LVC MOS33_OUT	PR7B	DRIVE:8mA SLEW:SLOW
data2_o[1]	D12/0	LVC MOS33_OUT	PT65A	DRIVE:8mA SLEW:SLOW
data2_o[2]	E15/0	LVC MOS33_OUT	PT67A	DRIVE:8mA SLEW:SLOW
data2_o[3]	F11/1	LVC MOS33_OUT	PR5A	DRIVE:8mA SLEW:SLOW
data3_o[0]	D11/0	LVC MOS33_OUT	PT57A	DRIVE:8mA SLEW:SLOW
data3_o[1]	D15/0	LVC MOS33_OUT	PT63B	DRIVE:8mA SLEW:SLOW
data3_o[2]	B16/0	LVC MOS33_OUT	PT59B	DRIVE:8mA SLEW:SLOW

Place & Route Timing Analysis (Place & Route Report) の確認



- Place & Route Timing Analysisでは配置配線後の静的タイミング解析（STA）結果を確認できます
- 右上のContentsボタンから確認したいレポートを選択します

Place & Route Timing Analysis (Place & Route Report) の確認



- 以下はCLOCK SUMMARY (Fmax解析結果) の表示例です

The screenshot shows the 'Place & Route Timing Analysis' report. The left sidebar contains a 'Reports' menu with 'Place & Route Timing Analysis' selected. The main content area displays the following sections:

2.2 Clock "clk"
create_clock -name {clk} -period 50 -waveform {0 25} [get_ports clk]
Single Clock Domain ← 同一クロックドメイン内でのFmax解析結果

Clock clk	Period	Frequency
From clk	50.000 ns	20.000 MHz
Actual (all paths)	21.497 ns	46.518 MHz

Minimum Pulse Width Period
clk_pad.bb_inst/B (MPW) (50% duty cycle) | 5.000 ns | 200.000 MHz

制約値(Target)と実際(Actual)の結果

Clock Domain Crossing ← クロックドメインを跨ぐ (非同期パス) Fmax解析結果

Clock clk	Worst Time Between Edges	Comment
From pll_clk	---	No path
From div4_clk	---	False path

パスが存在しない場合 "No path", False Path制約がかかっている場合 "False path", パスが存在している場合はSlackが表示されます

2.3 Clock "div4_clk"
create_generated_clock -name {div4_clk} -source [get_ports c:
Single Clock Domain

Clock div4_clk	Period	Frequency
From div4_clk	200.000 ns	5.000 MHz
Actual (all paths)	2.000 ns	500.000 MHz

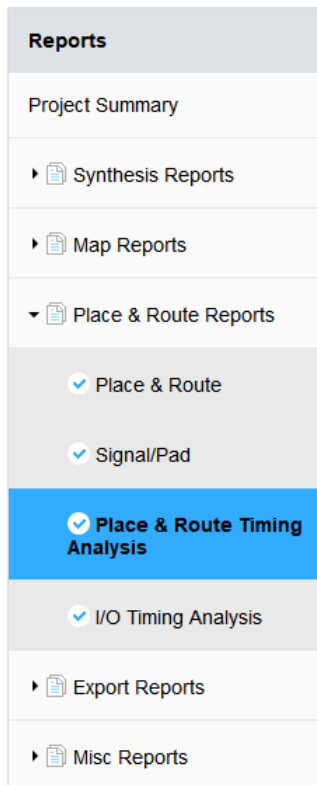
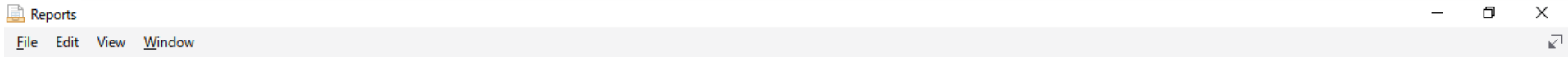
Contents

- [Timing Report](#)
- [1 DESIGN CHECKING](#)
 - [1.1 SDC Constraints](#)
 - [1.2 Combinational Loop](#)
- [2 CLOCK SUMMARY](#)
- [3 TIMING ANALYSIS SUMMARY](#)
 - [3.1 Overall \(Setup and Hold\)](#)
 - [3.2 Setup Summary Report](#)
 - [3.3 Hold Summary Report](#)
 - [3.4 Unconstrained Report](#)
- [4 DETAILED REPORT](#)
 - [4.1 Setup Detailed Report](#)
 - [4.2 Hold Detailed Report](#)

Place & Route Timing Analysis (Place & Route Report) の確認



- 以下はTIMING ANALYSIS SUMMARYの "3.1 Overall (Setup and Hold)" の表示例です



Place & Route Timing Analysis

3 TIMING ANALYSIS SUMMARY

3.1 Overall (Setup and Hold)

3.1.1 Constraint Coverage

Constraint Coverage: 84.9315% ← タイミング解析のカバレッジ (解析率)

3.1.2 Timing Errors

Timing Errors: 0 endpoints (setup), 3 endpoints (hold) ← Setup側、Hold側両方の合計エラー数

3.1.3 Total Timing Score

Total Negative Slack: 0.000 ns (setup), 71.733 ns (hold) ← Setup側、Hold側両方のNegative Slackの合計

3.2 Setup Summary Report

Listing 10 End Points	Slack
u_crossclk_test0/data_ff2_reg[0].ff_inst/DF	7.481 ns
data2_o_pad[0].bb_inst_IOL/TXDATA0	11.052 ns
data2_o_pad[1].bb_inst_IOL/TXDATA0	11.052 ns
data2_o_pad[2].bb_inst_IOL/TXDATA0	11.052 ns
data2_o_pad[3].bb_inst_IOL/TXDATA0	11.052 ns
data0_o[0]	28.503 ns
data0_o[3]	28.804 ns
data0_o[1]	28.812 ns
data0_o[2]	28.983 ns
u_crossclk_test0/data_ff2_reg[3].ff_inst/DF	34.952 ns

Contents

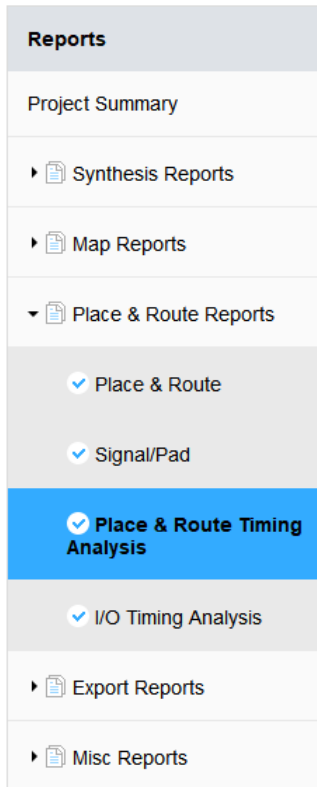
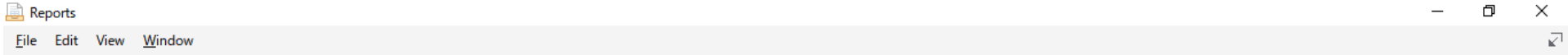
- [Timing Report](#)
- [1 DESIGN CHECKING](#)
 - [1.1 SDC Constraints](#)
 - [1.2 Combinational Loop](#)
- [2 CLOCK SUMMARY](#)
- [3 TIMING ANALYSIS SUMMARY](#)
 - [3.1 Overall \(Setup and Hold\)](#)
 - [3.2 Setup Summary Report](#)
 - [3.3 Hold Summary Report](#)
 - [3.4 Unconstrained Report](#)
- [4 DETAILED REPORT](#)
 - [4.1 Setup Detailed Report](#)
 - [4.2 Hold Detailed Report](#)

*Slack = 制約値に対するマージン (ns)
Negative Slack (値がマイナス表示のもの) は
タイミングマージンがマイナスであることを示しています

Place & Route Timing Analysis (Place & Route Report) の確認



- 以下はTIMING ANALYSIS SUMMARYの“3.2 Setup Summary Report”と“3.3 Hold Summary Report”の表示例です



Place & Route Timing Analysis

3.2 Setup Summary Report

Listing 10 End Points	Slack
u_crossclk_test0/data_ff2_reg[0].ff_inst/DF	7.481 ns
data2_o_pad[0].bb_inst_IOL/TXDATA0	11.052 ns
data2_o_pad[1].bb_inst_IOL/TXDATA0	11.052 ns
data2_o_pad[2].bb_inst_IOL/TXDATA0	11.052 ns
data2_o_pad[3].bb_inst_IOL/TXDATA0	11.052 ns
data0_o[0]	28.503 ns
data0_o[3]	28.804 ns
data0_o[1]	28.812 ns
data0_o[2]	28.983 ns
u_crossclk_test0/data_ff2_reg[3].ff_inst/DF	34.952 ns

Setup # of endpoints with negative slack: | 0

3.3 Hold Summary Report

Listing 10 End Points	Slack
u_crossclk_test0/data_ff2_reg[2].ff_inst/DF	-23.916 ns
u_crossclk_test0/data_ff2_reg[1].ff_inst/DF	-23.916 ns
u_crossclk_test0/data_ff2_reg[3].ff_inst/DF	-23.901 ns
u_rst_clk/rst_ff[7].ff_inst/DF	0.072 ns
u_rst_clk/rst_ff_reg[6].ff_inst/DF	0.072 ns
u_rst_clk/rst_ff_reg[3].ff_inst/DF	0.072 ns
u_rst_clk/rst_ff_reg[2].ff_inst/DF	0.072 ns
u_rst_clk/rst_ff_reg[4].ff_inst/DF	0.082 ns
u_rst_clk/rst_ff_reg[5].ff_inst/DF	0.087 ns
u_rst_clk/rst_ff_reg[1].ff_inst/DF	0.090 ns

Hold # of endpoints with negative slack: | 3

Setup側の解析に対して、Slackが最小のパス（マージンが最小のパス）からワースト10までが表示されます

Hold側の解析に対して、Slackが最小のパス（マージンが最小のパス）からワースト10までが表示されます

← Negative Slack (タイミングエラー) パスの合計

← Negative Slack (タイミングエラー) パスの合計

Contents

- [Timing Report](#)
- [1 DESIGN CHECKING](#)
 - [1.1 SDC Constraints](#)
 - [1.2 Combinational Loop](#)
- [2 CLOCK SUMMARY](#)
- [3 TIMING ANALYSIS SUMMARY](#)
 - [3.1 Overall \(Setup and Hold\)](#)
 - [3.2 Setup Summary Report](#)
 - [3.3 Hold Summary Report](#)
 - [3.4 Unconstrained Report](#)
- [4 DETAILED REPORT](#)
 - [4.1 Setup Detailed Report](#)
 - [4.2 Hold Detailed Report](#)

Place & Route Timing Analysis (Place & Route Report) の確認



- 以下はTIMING ANALYSIS SUMMARYの“3.4 Unconstrained Report”の表示例です

Reports

File Edit View Window

Reports

- Project Summary
- Synthesis Reports
- Map Reports
- Place & Route Reports
 - Place & Route
 - Signal/Pad
 - Place & Route Timing Analysis**
 - I/O Timing Analysis
- Export Reports
- Misc Reports

Place & Route Timing Analysis

3.4 Unconstrained Report

制約がかかっておらずタイミング解析されていないパスが表示されます

3.4.1 Unconstrained Start/End Points

Clocked but unconstrained timing start points

Listing 4 Start Points	Type
data2_o_pad[3].bb_inst_IOL/DOUT	No required time
data2_o_pad[2].bb_inst_IOL/DOUT	No required time
data2_o_pad[1].bb_inst_IOL/DOUT	No required time
data2_o_pad[0].bb_inst_IOL/DOUT	No required time

Number of unconstrained timing start points: 4

Clocked but unconstrained timing end points

Listing 3 End Points	Type
u_counter_clk/count[1].ff_inst/CE	No arrival time
u_counter_clk/count[2].ff_inst/CE	No arrival time
u_counter_clk/count[3].ff_inst/CE	No arrival time

Number of unconstrained timing end points: 3

3.4.2 Start/End Points Without Timing Constraints

Contents

- Timing Report
- 1 DESIGN CHECKING
 - 1.1 SDC Constraints
 - 1.2 Combinational Loop
- 2 CLOCK SUMMARY
- 3 TIMING ANALYSIS SUMMARY
 - 3.1 Overall (Setup and Hold)
 - 3.2 Setup Summary Report
 - 3.3 Hold Summary Report
 - 3.4 Unconstrained Report**
- 4 DETAILED REPORT
 - 4.1 Setup Detailed Report
 - 4.2 Hold Detailed Report

Place & Route Timing Analysis (Place & Route Report) の確認



- 以下はTIMING ANALYSIS SUMMARYの "4 DETAILED REPORT" の表示例です
- この例では同期クロック間でのFmax解析の詳細を示しています



Reports

Project Summary

- Synthesis Reports
- Map Reports
- Place & Route Reports
 - Place & Route
 - Signal/Pad
 - Place & Route Timing Analysis**
 - I/O Timing Analysis
- Export Reports
- Misc Reports

Place & Route Timing Analysis

```

++++ Path 8 ++++++
Path Begin      : u_rst_clk/rst_ff[7].ff_inst/Q (SLICE_R20C40A)
Path End       : u_counter_clk/count[3].ff_inst/LSR (SLICE_R11C73B)
Source Clock   : clk
Destination Clock: clk
Logic Level    : 1
Delay Ratio    : 83.0% (route), 17.0% (logic)
Clock Skew     : -0.177 ns
Setup Constraint: 10.000 ns
Path Slack     : 7.443 ns (Passed)
    
```

解析パスの起点と終点
 起点のクロックソースと終点のクロックソース (同じであれば同期パス)
 ← 設定されているタイミング制約

Shown in: Floor Planner Physical View

Name	Cell/Site Name	Delay Name	Delay	Arrival Time	Fanout
clk	top	CLOCK LATENCY	0.000	0.000	1
clk	top	NET DELAY	0.000	0.000	1
clk_pad.bb_inst/B->clk_pad.bb_inst/O	SEIO18_CORE_L7	PADI_DEL	0.562	0.562	23
u_rst_clk/clk_c	SEIO18_CORE_L7	NET DELAY	1.837	2.399	1

解析起点までのクロック遅延詳細

Shown in: Floor Planner Physical View

```

u_rst_clk/rst_ff[7].ff_inst/CLK->u_rst_clk/rst_ff[7].ff_inst/Q
SLICE_R20C40A  REG_DEL  0.333  2.732  6
u_rst_clk/rstn_clk ( LSR )
SLICE_R20C40A  NET DELAY  1.623  4.355  1
    
```

データバス間遅延詳細

Shown in: Floor Planner Physical View

clk	top	CONSTRAINT	0.000	10.000	1
clk	top	CLOCK LATENCY	0.000	10.000	1
clk	top	NET DELAY	0.000	10.000	1
clk_pad.bb_inst/B->clk_pad.bb_inst/O	SEIO18_CORE_L7	PADI_DEL	0.562	10.562	23

解析終点までのクロック遅延詳細

Contents

- Timing Report
- 1 DESIGN CHECKING
 - 1.1 SDC Constraints
 - 1.2 Combinational Loop
- 2 CLOCK SUMMARY
- 3 TIMING ANALYSIS SUMMARY
 - 3.1 Overall (Setup and Hold)
 - 3.2 Setup Summary Report
 - 3.3 Hold Summary Report
- 4 DETAILED REPORT**
 - 4.1 Setup Detailed Report
 - 4.2 Hold Detailed Report

I/O Timing Analysis (Place & Route Report) の確認



- I/O Timing Analysisでは配置配線後のI/Oタイミング解析（STA）結果を確認できます
- 入力ポートのワーストSetup/Hold Timeと解析クロックエッジ、出力ポートのワーストTcoと解析クロックエッジを以下のように確認できます

The screenshot shows a software window titled "Reports" with a menu bar (File, Edit, View, Window). The left sidebar lists report categories: Project Summary, Synthesis Reports, Map Reports, Place & Route Reports, and Misc Reports. Under "Place & Route Reports", "I/O Timing Analysis" is selected and highlighted with a blue box. The main content area displays the "I/O Timing Analysis" report, which includes a Performance Grade Translation Table, FPGA Input Port Results, and FPGA Output Port Results for Performance Grade: 9_High-Performance_1.0V.

```
I/O Timing Report
=====
Performance Grade Translation Table
-----
Code          | Performance Grade
-----
9_High-Performance_1.0V| 9_High-Performance_1.0V
-----

FPGA Input Port Results for Performance Grade: 9_High-Performance_1.0V
-----
Port Name| Setup | Grade |Edge| Hold | Grade |edge|Clock Port
-----
data_i[1]|-0.517 ns | 9_High-Performance_1.0V| R |1.426 ns | 9_High-Performance_1.0V| R |clk
data_i[2]|-0.517 ns | 9_High-Performance_1.0V| R |1.426 ns | 9_High-Performance_1.0V| R |clk
data_i[0]|-0.551 ns | 9_High-Performance_1.0V| R |1.460 ns | 9_High-Performance_1.0V| R |clk
data_i[3]|-0.551 ns | 9_High-Performance_1.0V| R |1.460 ns | 9_High-Performance_1.0V| R |clk
-----

FPGA Output Port Results for Performance Grade: 9_High-Performance_1.0V
-----
Port Name |Clock To Out (MAX)| Grade |Edge|Clock To Out (MIN)| Grade |edge|Clock Port
-----
data0_o[0]| 11.497 ns | 9_High-Performance_1.0V| R | 10.888 ns | 9_High-Performance_1.0V| R |clk
data0_o[3]| 11.196 ns | 9_High-Performance_1.0V| R | 10.677 ns | 9_High-Performance_1.0V| R |clk
data0_o[1]| 11.188 ns | 9_High-Performance_1.0V| R | 10.668 ns | 9_High-Performance_1.0V| R |clk
data0_o[2]| 11.017 ns | 9_High-Performance_1.0V| R | 10.553 ns | 9_High-Performance_1.0V| R |clk
-----
```

(参考) Strategy設定



- タイミングエラーが出た場合の対処として、Strategy(コンパイルオプション)を変更することができます
- Strategyを変更してPlace & Route Designを実行することで配置配線結果に変化を与えます
- デフォルトで“Area”, “Timing”, “Strategy1”が用意されており、Strategy1がアクティブになっています
- Project ViewのStrategiesの下にあるStrategy1をダブルクリックするか右クリックからEditを選択すると設定画面が開きます
- 配置配線時に与えるコンパイルオプションはPlace & Route Designを選択します

Strategyをダブルクリックか右クリックでEditを選択

Name	Type	Value
Command Line Options	Text	
Disable Auto Hold Timing Correction	T/F	<input type="checkbox"/>
Disable Timing Driven	T/F	<input type="checkbox"/>
Multi-Tasking Node List	File	
Number of Host Machine Cores	Num	1
Pack Logic Block Utility [blank or 0 to 100]	Num	
Path-based Placement	List	Off
Placement Iteration Start Point	Num	1
Placement Iterations [0-100]	Num	1
Placement Save Best Run [1-100]	Num	1
Prioritize Hold Correction Over Setup Performance	T/F	<input type="checkbox"/>
Run Placement Only	T/F	<input type="checkbox"/>
Set Speed Grade for Hold Optimization	List	Default
Set Speed Grade for Setup Optimization	List	Default
Stop Once Timing is Met	T/F	<input type="checkbox"/>

各種設定の詳細はHelpから確認できます

(参考) Strategyの追加



- Strategiesを右クリック > Add > New Strategyで新しいStrategyを作成することができます
- 設定の異なるStrategyを複数用意し、切り替えてコンパイル結果を比較することができます

① Strategiesを右クリック

② Add > New Strategyを選択

③ Strategy ID (Projectに表示されるStrategy名)を入力
File NameはStrategy IDがコピーされます

④ OKをクリックするとProjectにインポートされます

(参考) Strategyの追加



- アクティブなStrategyを切り替えるにはStrategy名を右クリックし、“Set as Active Strategy”を選択します

The screenshot shows the Lattice Radiant Software interface with the 'Reports' window open. The 'Radiant_Example Project Summary' table is visible, showing the current strategy is 'Strategy1'. A context menu is open over the 'Strategies' folder in the project tree, with 'Strategy2' selected. The 'Set as Active Strategy' option is highlighted in the menu. A blue arrow points from this option to 'Strategy2' in the project tree, which is now bolded. A second blue arrow points from the bolded 'Strategy2' to the 'Set as Active Strategy' option in the menu.

① Strategy名を右クリック

② Set as Active Strategyを選択

アクティブになったStrategyは太字で表示されます

(参考) Run Manager



- 複数のImplementation (サブProject) を作成し、それぞれに異なるStrategyや制約を設定し、同時にコンパイルを回すことができます
- まず、File > New > Implementationから新しいImplementationを作成します

① Implementation名を入力
Directory名はImplementation名がコピーされます

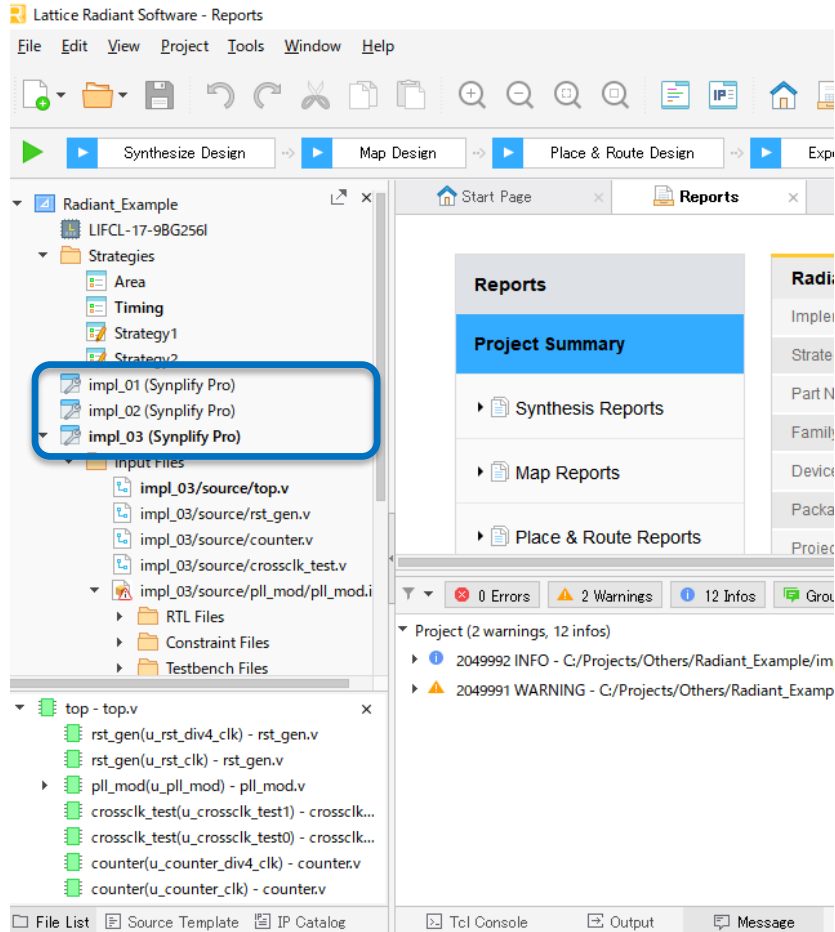
② Add Sourceでソースコード、制約ファイル等を追加します (Implementation作成後でも追加可能)

③ OKをクリック

(参考) Run Manager




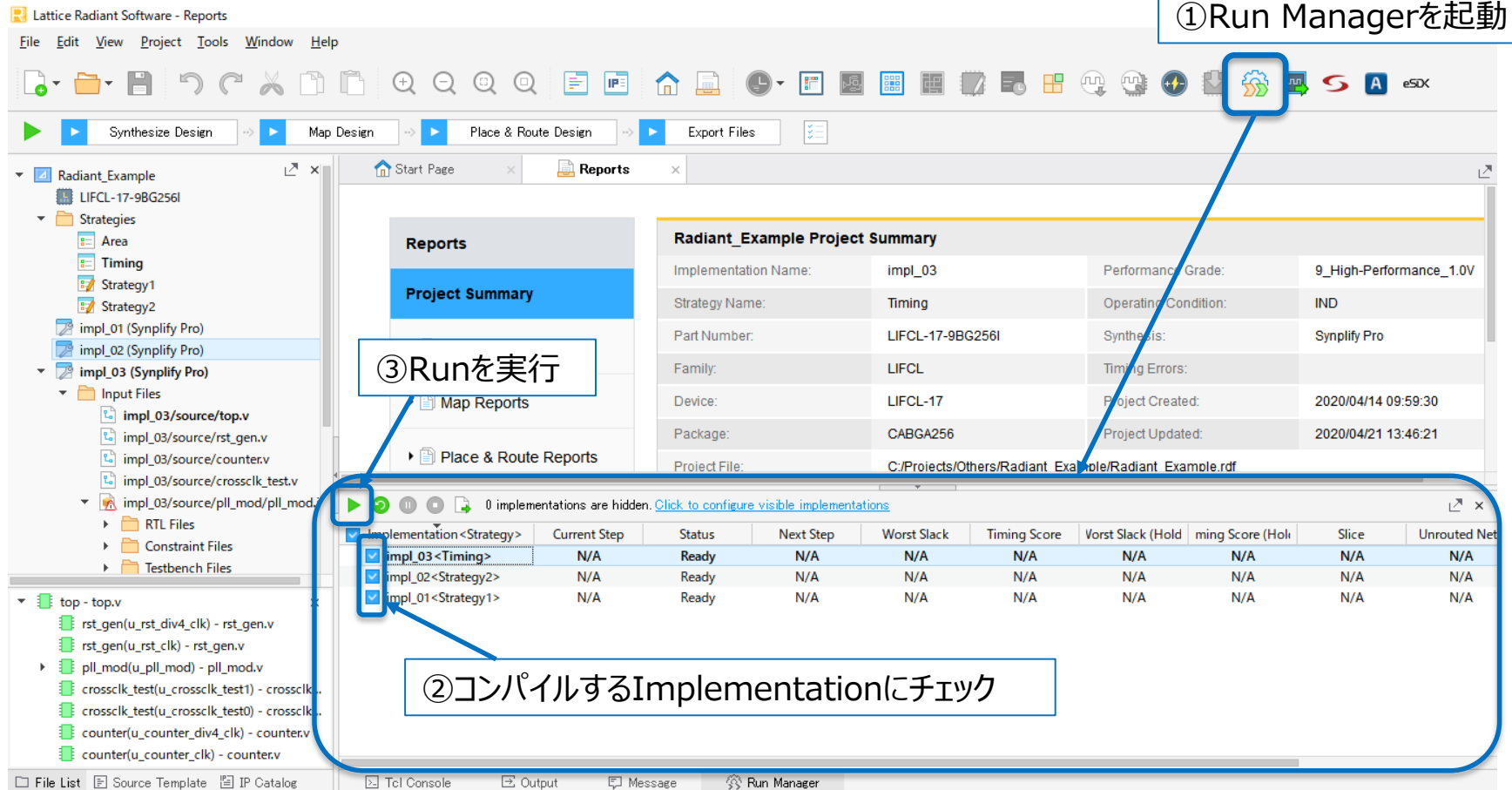
- 作成したImplementationがProject Viewにインポートされます
- アクティブなImplementationを切り替えてImplementation毎にStrategyやソースコード、制約設定を変更することができます



(参考) Run Manager



- Tools > Run Manager またはToolbarのRun Managerアイコン  からRun Managerを起動します
- コンパイルを実行するImplementationにチェックを入れ、Runボタンをクリックします
- レポートはImplementationを切り替えて確認します



① Run Managerを起動

② コンパイルするImplementationにチェック

③ Runを実行

Radiant_Example Project Summary

Implementation Name:	impl_03	Performance Grade:	9_High-Performance_1.0V
Strategy Name:	Timing	Operating Condition:	IND
Part Number:	LIFCL-17-9BG256I	Synthesis:	Synplify Pro
Family:	LIFCL	Timing Errors:	
Device:	LIFCL-17	Project Created:	2020/04/14 09:59:30
Package:	CABGA256	Project Updated:	2020/04/21 13:46:21
Project File:	C:/Projects/Others/Radiant_Example/Radiant_Example.rdf		


0 implementations are hidden. [Click to configure visible implementations](#)

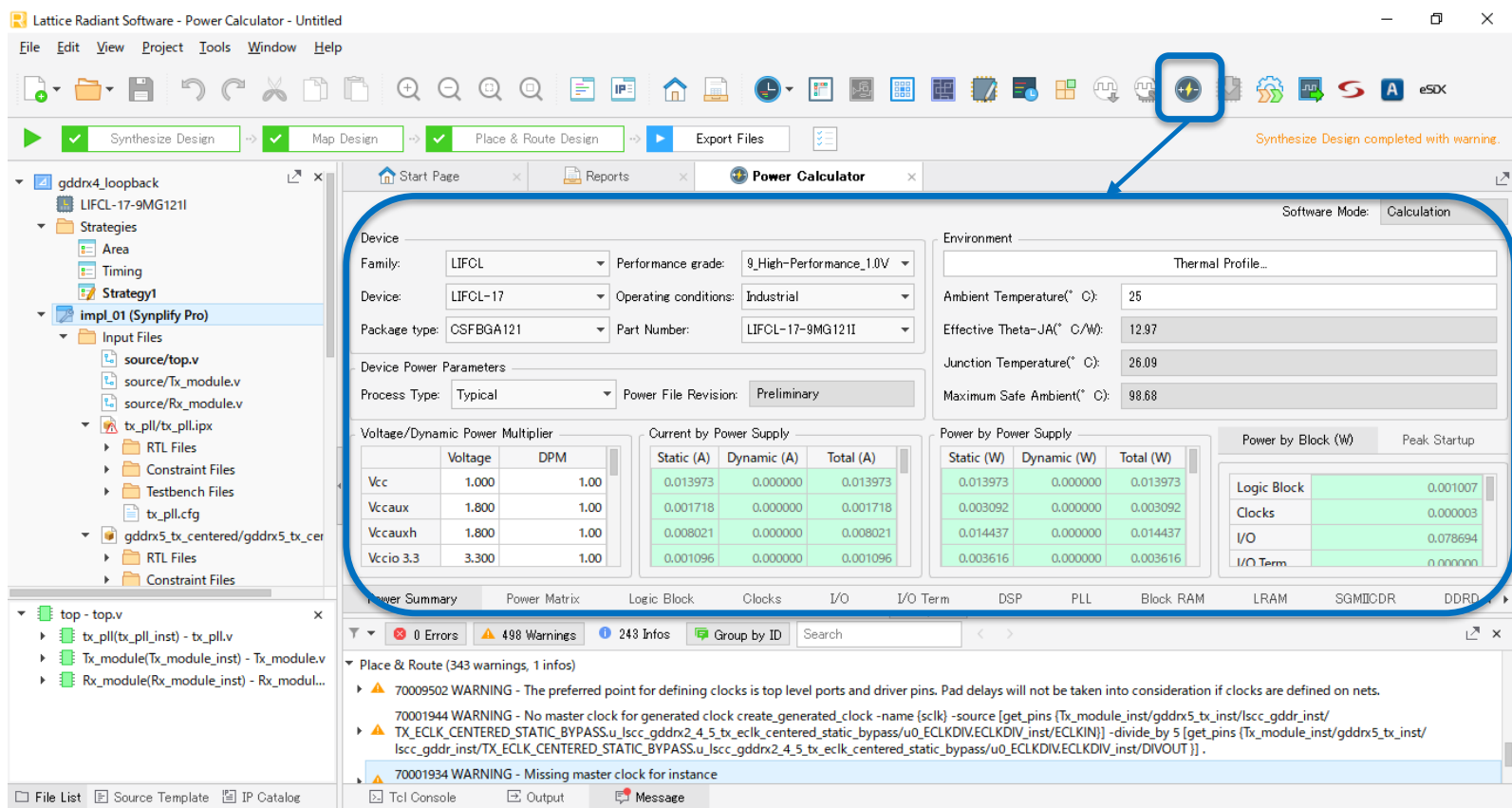
Implementation <Strategy>	Current Step	Status	Next Step	Worst Slack	Timing Score	Worst Slack (Hold)	Timing Score (Hold)	Slice	Unrouted Net
<input checked="" type="checkbox"/> impl_03 <Timing>	N/A	Ready	N/A	N/A	N/A	N/A	N/A	N/A	N/A
<input checked="" type="checkbox"/> impl_02 <Strategy2>	N/A	Ready	N/A	N/A	N/A	N/A	N/A	N/A	N/A
<input checked="" type="checkbox"/> impl_01 <Strategy1>	N/A	Ready	N/A	N/A	N/A	N/A	N/A	N/A	N/A

5. 消費電力の見積もり



Power Calculator

- 消費電力の見積もりはPower Calculatorを使用します
- Power Calculatorには実際のデザインコンパイル結果を用いるCalculationモードと、使用リソースを概算で入力して見積もるEstimationモードが用意されていますが、ここではCalculationモードについて説明します
- Tool > Power Calculator もしくはToolbarのPower Calculatorアイコン  から起動します



Power Calculator



- Power CalculatorのGUIは以下のようになっています

■ Device

見積もり対象デバイスの情報、設定入力

■ Environment

周囲温度、実装条件等の入力及びジャンクション温度、最大周囲温度等の表示

■ 見積もりモード表示

Software Mode: Calculation

Device

Family: LIFCL Performance grade: 9_High-Performance_1.0V

Device: LIFCL-17 Operating conditions: Industrial

Package type: CSFBGA121 Part Number: LIFCL-17-9MG121I

Device Power Parameters

Process Type: Typical Power File Revision: Preliminary

Environment

Thermal Profile...

Ambient Temperature(* C): 25

Effective Theta-JA(* C/W): 12.97

Junction Temperature(* C): 26.37

Maximum Safe Ambient(* C): 98.39

Voltage/Dynamic Power Multiplier			Current by Power Supply			Power by Power Supply			Power by Block (W) / Peak Startup	
	Voltage	DPM	Static (A)	Dynamic (A)	Total (A)	Static (W)	Dynamic (W)	Total (W)		Peak Startup
Vcc	1.000	1.00	0.013990	0.002452	0.016442	0.013990	0.002452	0.016442	Clocks	0.000580
Vccaux	1.800	1.00	0.001720	0.000908	0.002628	0.003097	0.001634	0.004731	I/O	0.100009
Vccauxh	1.800	1.00	0.008021	0.001352	0.009373	0.014437	0.002434	0.016871	I/O Term	0.000000
Vccio 3.3	3.300	1.00	0.001096	0.003909	0.005005	0.003617	0.012898	0.016515	DSP	0.000087
Vccio 2.5	2.500	1.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	PLL	0.000621
Vccio 1.8	1.800	1.00	0.026784	0.001499	0.028283	0.048211	0.002698	0.050910	Block RAM	0.000035
Vccio 1.35	1.350	1.00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000		
			0.000000	0.000000	0.000000	0.000000	0.000000	0.000000		
			0.000000	0.000000	0.000000	0.000000	0.000000	0.000000		
			0.000000	0.000000	0.000000	0.000000	0.000000	0.000000		
			0.000013	0.000000	0.000013	0.000013	0.000000	0.000013		
Vccadphy	1.800	1.00	0.000232	0.000000	0.000232	0.000417	0.000000	0.000417	MIPIDPHY	0.000441
Vccplldphy	1.000	1.00	0.000010	0.000000	0.000010	0.000010	0.000000	0.000010	ADC	0.000069
			0.051866	0.010120	0.061986	0.083792	0.022117	0.105909	Misc	0.002534
									Total	0.105942

■ Voltage/Dynamic Power Multiplier

電源電圧設定とマージン設定

■ Power by Block (W) / Peak Startup

ブロック毎の消費電力 / 起動時ピーク電流

■ Current by Power Supply

電源電圧毎の消費電流

■ Power by Power Supply

電源電圧毎の消費電力

Power Calculator



- 以下のフローで動作周波数設定を行います

① Edit > Frequency Settingを選択

② 動作周波数を設定

③ OKをクリック

Voltage/Dynamic Power Multiplier		
	Voltage	DPM
Vcc	1.000	1.00
Vccaux	1.800	1.00
Vccauxh	1.800	1.00
Vccio 3.3	3.300	1.00
Vccio 2.5	2.500	1.00
Vccio 1.8	1.800	1.00
Vccio 1.35	1.350	1.00
Vccio 1.5	1.500	1.00
Vccio 1.2	1.200	1.00
Vccio 1.0	1.000	1.00
Vccdphy	1.000	1.00
Vccadphy	1.800	1.00
Vccplldphy	1.000	1.00

Current by Power Supply	
Static (A)	Dynamic (A)
0.013973	0.000000
0.001718	0.000000
0.008021	0.000000
0.001096	0.000000

Power by Block (W)	
Logic Block	Peak Startup
3092	0.001007
4437	0.000003
3616	0.078694
	0.000000
	0.000086
	0.000620
	0.000035
	0.000016
	0.000036
	0.000224
	0.000002
	0.000052
	0.000440
	0.000069
	0.002527

■ Frequency Default

全てのクロックに一律で設定した周波数を与えます

■ Use Frequency TWR

Place & Routeまで完了している場合に使用します（推奨）

- Minimum of Constraint and Timing
制約設定と実際の最大動作周波数のうち小さいものを適用（推奨）
- Always use Constraint
制約設定で与えた周波数を適用
- Always use Timing
最大動作周波数を適用

Power Calculator



- 以下のフローでActivity Factorを設定します

① Edit > Activity Factor Settingを選択

② Activity Factorを設定

③ OKをクリック

*Activity Factor = 内部ロジックのスイッチングレート
内部ロジックがクロックエッジに対してどの程度の頻度でスイッチするかを示すレート
毎エッジ遷移する場合を100%と考えます。デフォルトで10%が入力されています

Voltage/Dynamic Power Multiplier		
	Voltage	DPM
Vcc	1.000	1.00
Vccaux	1.800	1.00
Vccauxh	1.800	1.00
Vccio 3.3	3.300	1.00
Vccio 2.5	2.500	1.00
Vccio 1.8	1.800	1.00
Vccio 1.35	1.350	1.00
Vccio 1.5	1.500	1.00
Vccio 1.2	1.200	1.00
Vccio 1.0	1.000	1.00
Vccdphy	1.000	1.00
Vccadphy	1.800	1.00
Vccplldphy	1.000	1.00

Current by Power Supply					
	Static (A)	Dynamic (A)	Total (A)	Static (W)	Dynamic (W)
	0.013973	0.000000	0.013973	0.013973	0.000000
	0.001718	0.000000	0.001718	0.003092	0.000000
	0.008021	0.000000	0.008021	0.014437	0.000000
	0.001096	0.000000	0.001096	0.003616	0.000000

Power by Block (W)	
Block	Power (W)
Logic Block	0.001007
Clocks	0.000003
I/O	0.078694
I/O Term	0.000000
	0.000086
	0.000620
	0.000035
	0.000016
SGMIICDR	0.000036
DDRDLL	0.000224
DLLDEL	0.000002
DQS	0.000052
MIPIDPHY	0.000440
ADC	0.000069
Misc	0.002527

Power Calculator



- 以下のフローで周囲環境を設定します

① Thermal Profileをクリック

② Boardサイズを選択

③ Heat Sinkの有無、Airflowの強度を選択

④ OKをクリック

⑤ デバイス周囲温度を設定

Environment

Thermal Profile...

Ambient Temperature(* C): 25

Effective Theta-JA(* C/W): 12.97

Junction Temperature(* C): 26.09

Maximum Safe Ambient(* C): 98.68

Power by Power Supply

Power by Block (W)

Peak Startup

Block	Power (W)
Block	0.001007
Clocks	0.000003
I/O	0.078694
PLL	0.000620
Block RAM	0.000035
LRAM	0.000016
SGMIICDR	0.000036
DDRDLL	0.000224
DLLDEL	0.000002
DQS	0.000052
MIPIDPHY	0.000440
ADC	0.000069
Misc	0.002527

Power Calculator



- デバイスプロセス条件のTypical / Worstを設定し、見積もり結果を確認します

① Typical or Worstを選択

ジャンクション温度の見積もり結果
最大周囲温度見積もり結果

電源毎の消費電流見積もり結果

電源毎の消費電力見積もり結果

トータル消費電流見積もり結果

トータル消費電力見積もり結果

	Voltage	DPM
Vcc	1.000	1.00
Vccaux	1.800	1.00
Vccauxh	1.800	1.00
Vccio 3.3	3.300	1.00
Vccio 2.5	2.500	1.00
Vccio 1.8	1.800	1.00
Vccio 1.35	1.350	1.00
Vccio 1.5	1.500	1.00
Vccio 1.2	1.200	1.00
Vccadphy	1.800	1.00
Vccplldphy	1.000	1.00

	Static (A)	Dynamic (A)	Total (A)
Vcc	0.013973	0.000000	0.013973
Vccaux	0.001718	0.000000	0.001718
Vccauxh	0.008021	0.000000	0.008021
Vccio 3.3	0.001096	0.000000	0.001096
Vccio 2.5	0.000000	0.000000	0.000000
Vccio 1.8	0.026788	0.000000	0.026788
Vccio 1.35	0.000000	0.000000	0.000000
Vccio 1.5	0.000000	0.000000	0.000000
Vccio 1.2	0.000000	0.000000	0.000000
Vccadphy	0.000000	0.000000	0.000000
Vccplldphy	0.000013	0.000000	0.000013
	0.000232	0.000000	0.000232
	0.000010	0.000000	0.000010
Total	0.051850	0.000000	0.051850


	Static (W)	Dynamic (W)	Total (W)
Vcc	0.013973	0.000000	0.013973
Vccaux	0.003092	0.000000	0.003092
Vccauxh	0.014437	0.000000	0.014437
Vccio 3.3	0.003616	0.000000	0.003616
Vccio 2.5	0.000000	0.000000	0.000000
Vccio 1.8	0.048219	0.000000	0.048219
Vccio 1.35	0.000000	0.000000	0.000000
Vccio 1.5	0.000000	0.000000	0.000000
Vccio 1.2	0.000000	0.000000	0.000000
Vccadphy	0.000000	0.000000	0.000000
Vccplldphy	0.000013	0.000000	0.000013
	0.000417	0.000000	0.000417
	0.000010	0.000000	0.000010
Total	0.083777	0.000000	0.083777

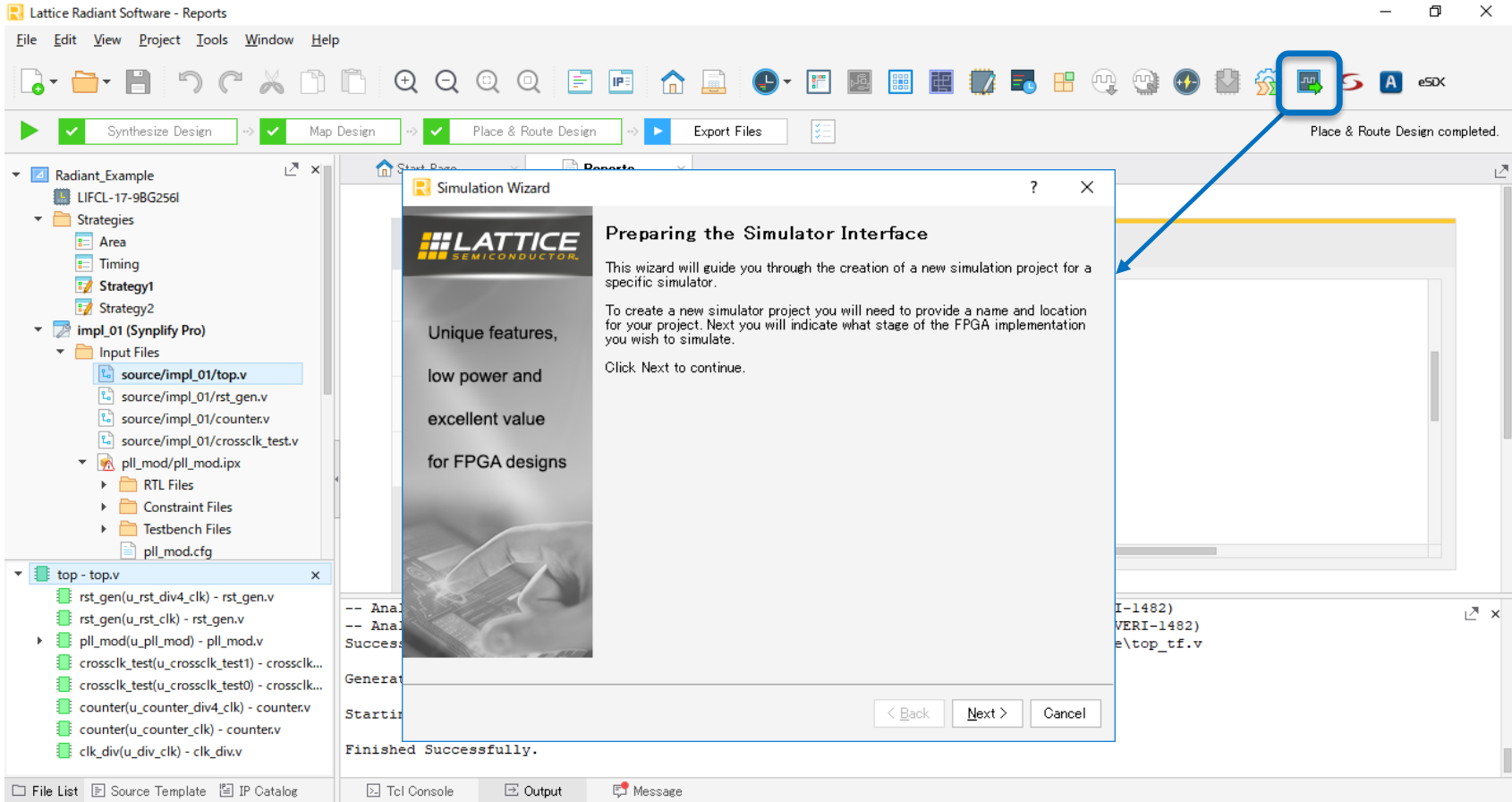
Logic Block	Peak Startup
Logic Block	0.001007
Clocks	0.000003
I/O	0.078694
I/O Term	0.000000
DSP	0.000086
PLL	0.000620
Block RAM	0.000035
LRAM	0.000016
DLDEE	0.000002
DQS	0.000052
MIPIDPHY	0.000440
ADC	0.000069
Misc	0.002527

6. Functionシミュレーション



Simulation Wizardの実行

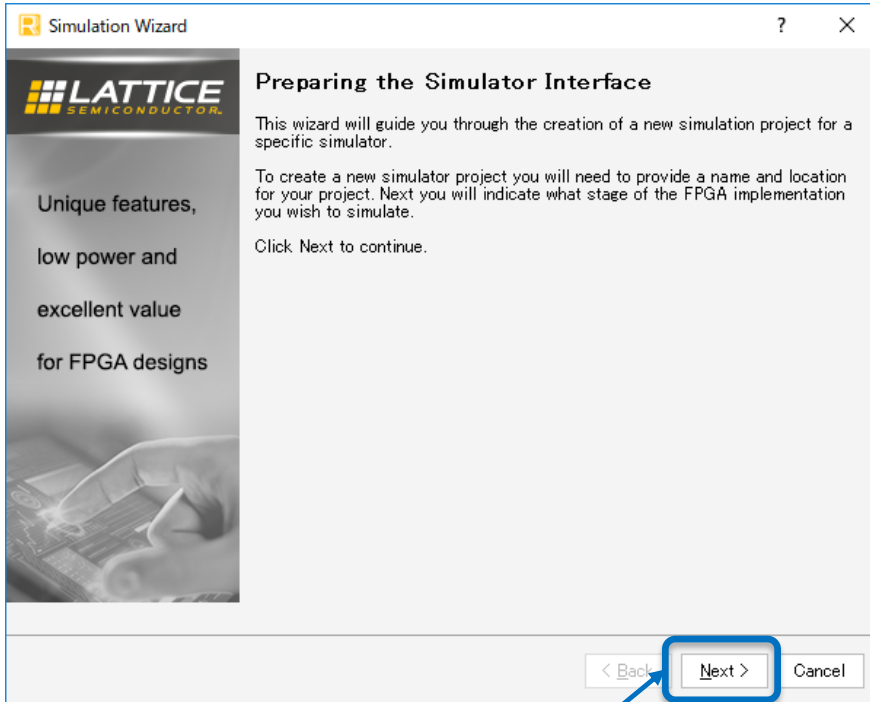
- RadiantではActive-HDLでのシミュレーションをSimulation Wizardを使用して実行することができます
- Tool > Simulation Wizard またはToolbarからSimulation WizardアイコンをクリックしてSimulation Wizardを起動します



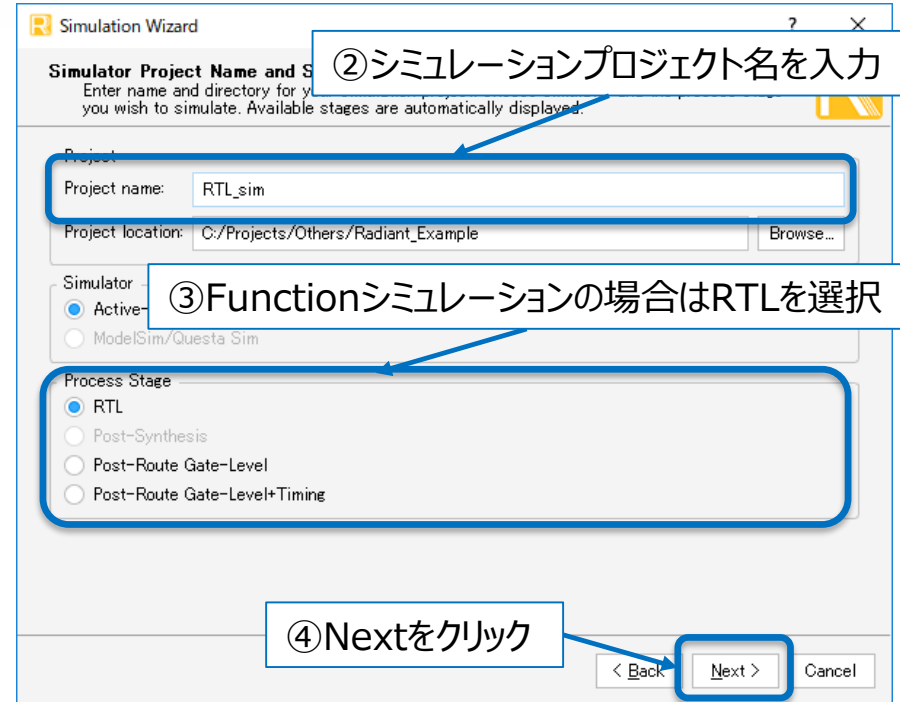
Simulation Projectの作成



- 以下のフローに従い、シミュレーションプロジェクトを作成します



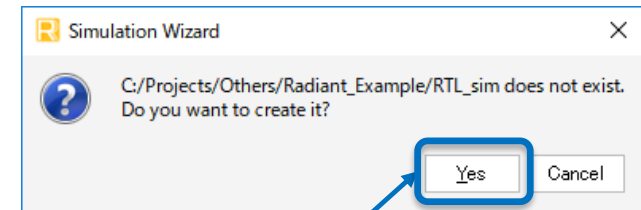
①Nextをクリック



②シミュレーションプロジェクト名を入力

③Functionシミュレーションの場合はRTLを選択

④Nextをクリック

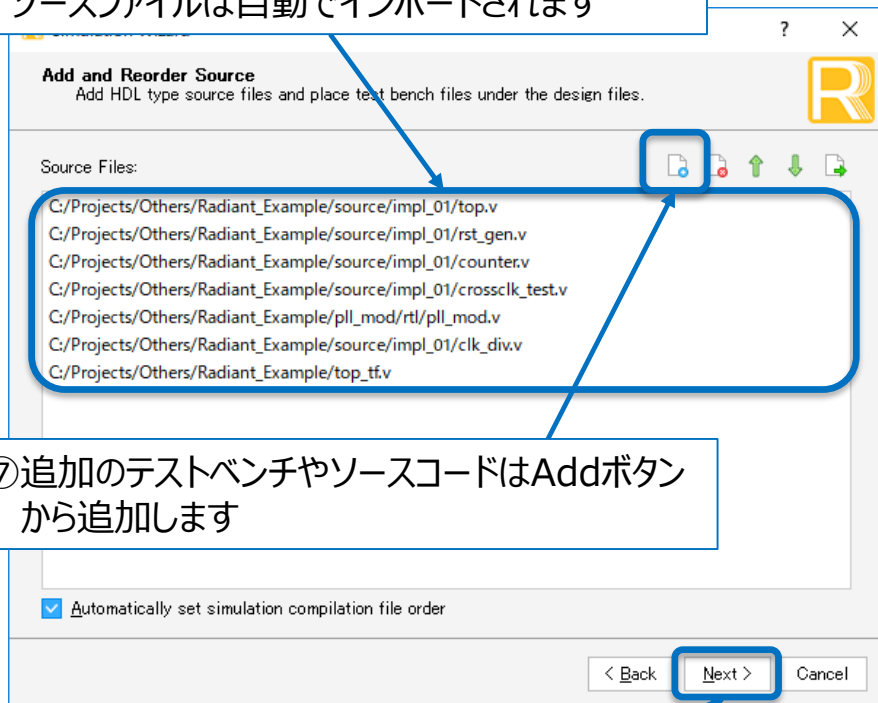


⑤Yesをクリック

Simulation Projectの作成

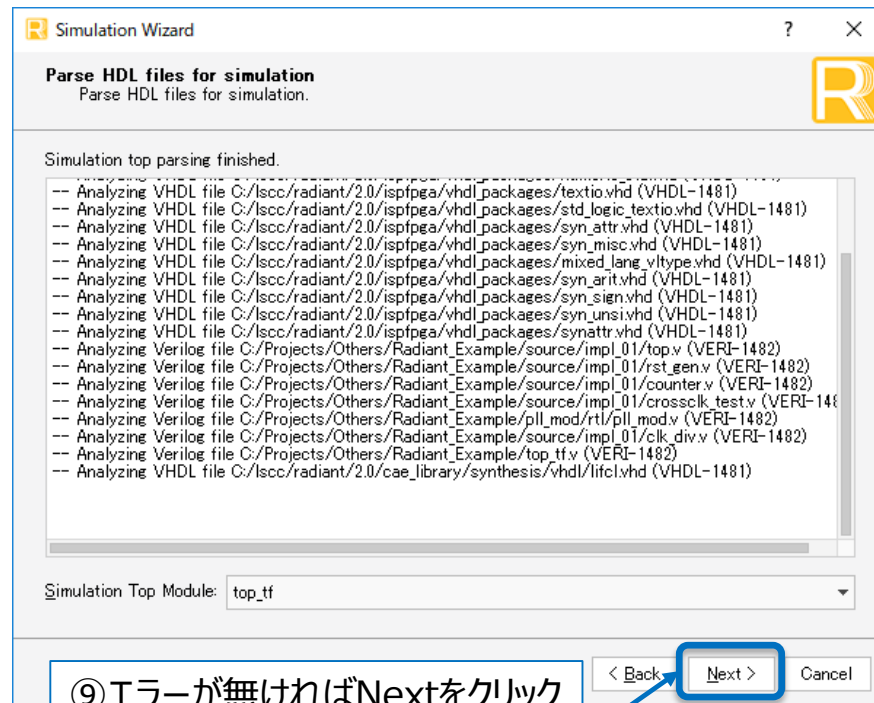


⑥ Radiantのプロジェクトにインポートされているソースファイルは自動でインポートされます



⑦ 追加のテストベンチやソースコードはAddボタンから追加します

⑧ Nextをクリック

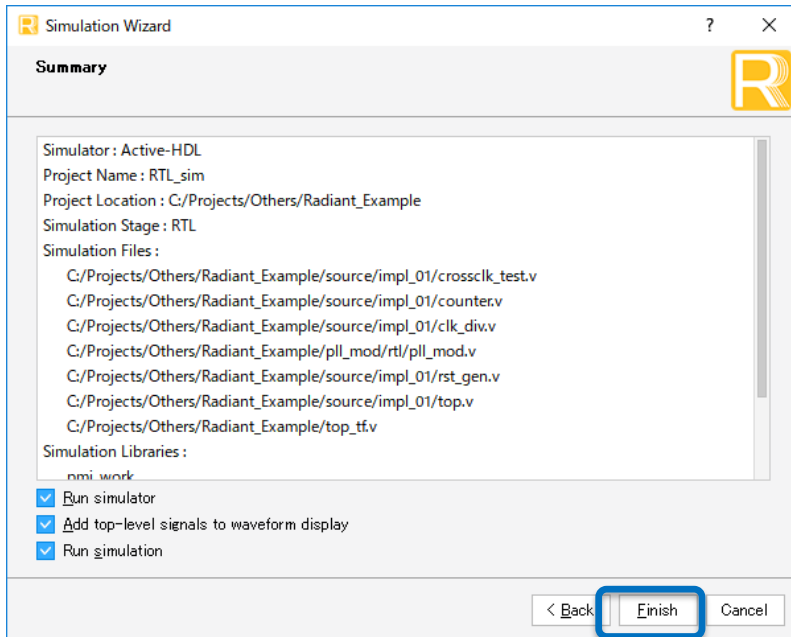


⑨ エラーが無ければNextをクリック
エラーがある場合はソースコードを修正して下さい

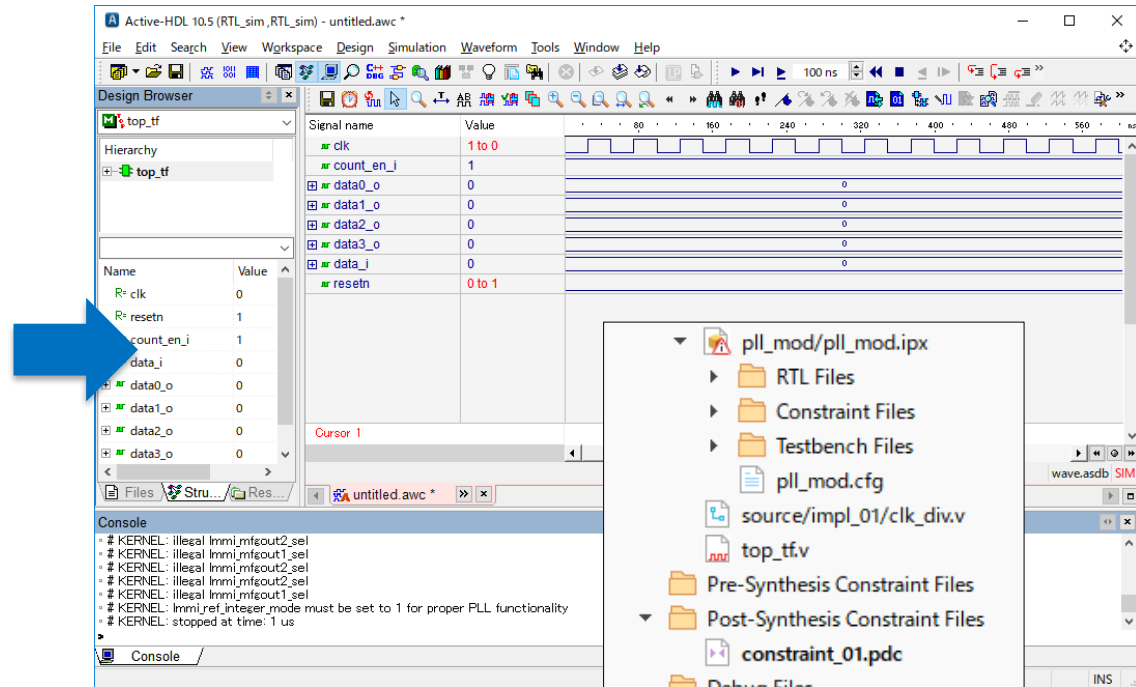
Simulation Projectの作成



- FinishをクリックするとActive-HDLが立ち上がります
- 作成したシミュレーションプロジェクトはRadiantのProject View内のScript Fileにインポートされ、このファイルをダブルクリックすることでいつでもシミュレーションプロジェクトを起動できます



⑩Finishをクリック



シミュレーション起動ファイル(.spf)がインポートされます

Active-HDLでの波形確認



① 確認したい信号が存在する
インスタンスを選択

Signal name	Value
clk	1
count_en_i	1
data0_o	6
data1_o	A
data1_o[3]	1
data1_o[2]	0
data1_o[1]	1
data1_o[0]	0
data2_o	0
data3_o	0
data_i	0
resetn	1
clk_i_i	1
rstn_i	1
clkop_o	1 to 0
lock_o	1

Console

```
# KERNEL: illegal Immi_mfgout1_sel
# KERNEL: illegal Immi_mfgout1_sel
# KERNEL: illegal Immi_mfgout1_sel
# KERNEL: illegal Immi_mfgout1_sel
# KERNEL: Immi_ref_integer_mode must be set to 1 for proper PLL functionality
# KERNEL: stopped at time: 392093750 ps
run
```

③ Runボタンをクリックして
Simulation実行

④ Simulation停止は
Breakをクリック

② インスタンス内の信号が表示されるので
確認したい信号をWaveformウィンドウに
ドラッグ&ドロップ

Active-HDLでの波形確認



波形拡大、縮小、Zoom to Fit (波形全体表示) ツールで希望の倍率に変更できます

The screenshot shows the Active-HDL 10.5 interface. At the top, the menu bar includes File, Edit, Search, View, Workspace, Design, Simulation, Waveform, Tools, Window, and Help. Below the menu is a toolbar with icons for zooming and other functions. The Design Browser on the left shows a hierarchy for 'top_tf'. The main area displays a waveform for 'data0_o[3]' and 'data0_o[2]'. A zoomed-in view of a portion of the waveform is shown in a separate window, with a text box indicating that the zoom tool can be used to expand any selected area. A context menu is open over the waveform, showing options like 'Add Cursor', 'Remove Cursor', and 'Measurement'. A text box explains that a cursor can be added via a right-click to check the time between cursors. The console window at the bottom is also visible.

任意拡大ツールで拡大したい場所を選択して拡大できます

- Add Cursor
- Remove Cursor
- Measurement
 - Find Signal (Ctrl+T)
 - Find Value (Ctrl+F)
 - Go To Time (Ctrl+G)
- Browse by
 - Zoom Mode (Ctrl+Shift+Z)
 - Measurement Mode (Ctrl+Shift+Y)

右クリックから“Add Cursor”でカーソルを追加し、カーソル間の時間を確認することができます

8. デザイン書き込み

デザイン書き込み

- RADIANTでは、コンフィグレーションデータの生成から書き込みまで同ツール内で行うことが可能です
- ここでは、コンフィグレーションデータの生成と書き込みフローについて説明します

Process ToolbarのExport Filesをダブルクリックして、コンフィグレーションデータ(.bit)を生成します。コンフィグレーションデータ(.bit)はプロジェクトフォルダ内のインプリメンテーションフォルダ（デフォルトは"impl_1"）の中に生成されます

マークに表示となるとコンフィグレーションデータの生成が完了します

```
module top
(
  RST_n,
  SOC,
  ANALOG_IN_P,
  ANALOG_IN_N,
  OUT_DATA,
  LED_PORT,
  adc_clk
);

input RST_n;
input SOC;
input ANALOG_IN_P;
input ANALOG_IN_N;
output [11:0] OUT_DATA;
output LED_PORT;
output adc_clk;

`define IDLE 3'b000
`define CAL 3'b001
`define W_SOC 3'b010
`define SOC 3'b011
`define CONV 3'b100

reg [15:0] counter;
reg [21:0] test_LED;
reg [2:0] soc_r;
reg soc_flag;
reg [3:0] state_cnt;
reg [2:0] adc_state;
reg cal_i;
reg soc_i;

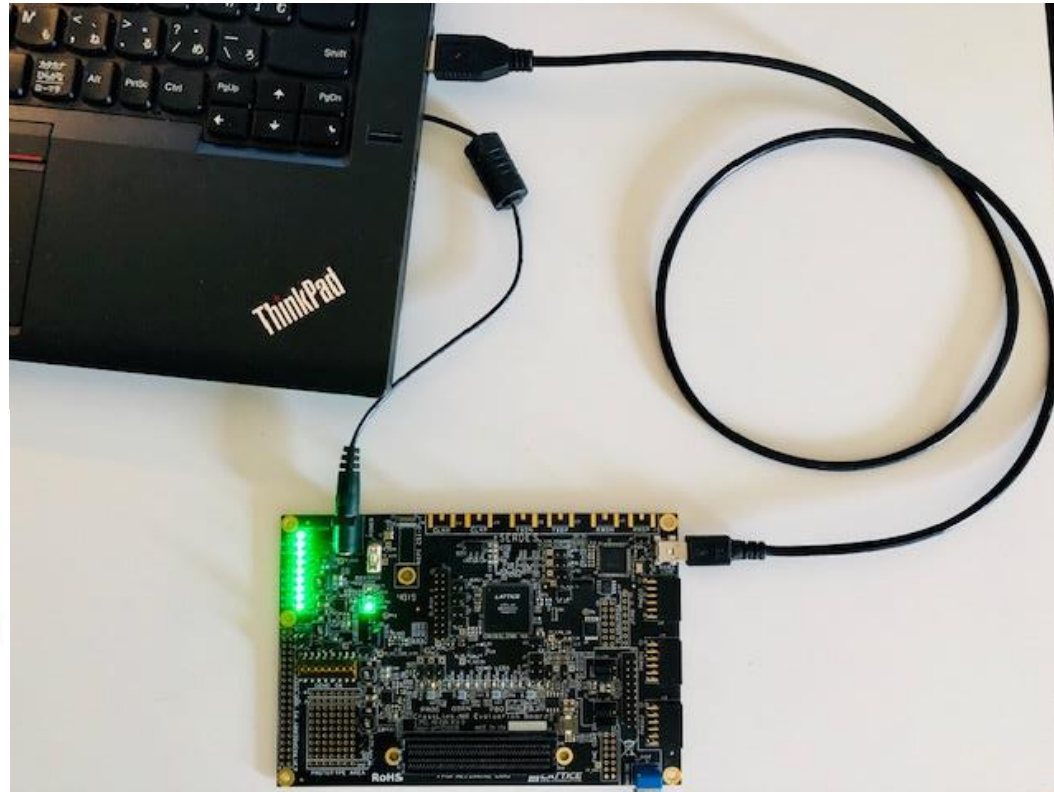
wire soc_clk;
```

STA Runtime and Peak Memory Usage :
Total CPU Time: 4 secs
Total REAL Time: 4 secs
Peak Memory Usage: 350 MB
Done: completed successfully

デザイン書き込み




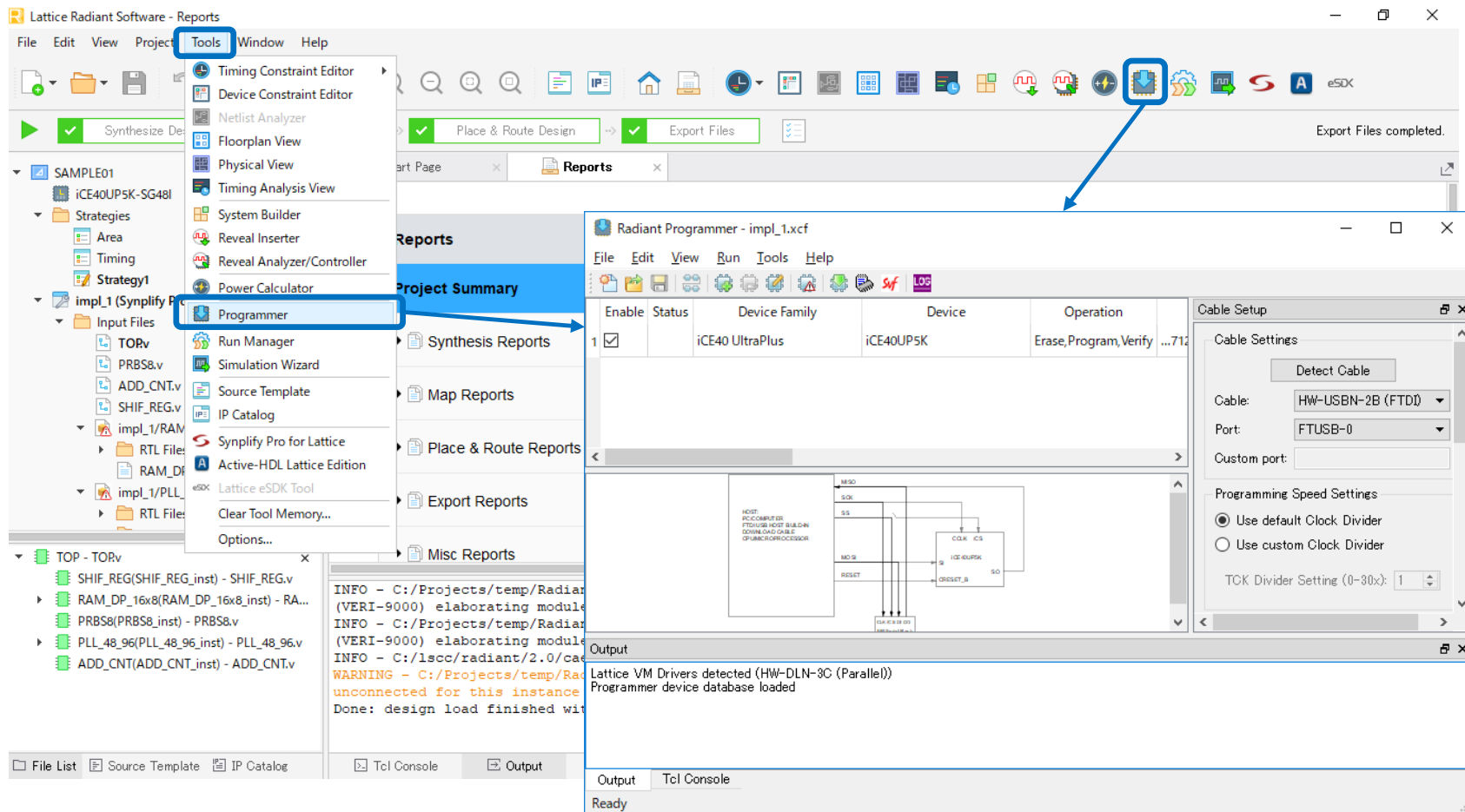
- PCとデバイスが実装されたボードを接続します
- ここでは、Lattice評価ボードを使用した場合を紹介していますが、お客様ボードでの書き込みフローも操作方法は共通となります



お客様ボードの場合、USBダウンロードケーブル
HW-USBN-2B（JTAG/SPI/I2C対応）を使用します

デザイン書き込み

- Tools > Programmer もしくはToolbarからProgrammer  アイコンをクリックし、Programmerを起動します



The screenshot shows the Lattice Radiant Software interface. The 'Tools' menu is open, and the 'Programmer' option is highlighted. A blue arrow points from the 'Programmer' option in the menu to the 'Programmer' icon in the toolbar. Another blue arrow points from the 'Programmer' icon in the toolbar to the 'Radiant Programmer - impl_1.xcf' window. The 'Radiant Programmer' window is open, showing a table with columns: Enable, Status, Device Family, Device, and Operation. The table contains one row with '1' in the Enable column, a checked box in the Status column, 'iCE40 UltraPlus' in the Device Family column, 'iCE40UP5K' in the Device column, and 'Erase,Program,Verify ...712' in the Operation column. The 'Cable Setup' panel is also visible, showing 'Cable: HW-USB-2B (FTDI)' and 'Port: FTUSB-0'. The 'Output' panel shows the following text:

```
INFO - C:/Projects/temp/Radiant (VERI-9000) elaborating module
INFO - C:/Projects/temp/Radiant (VERI-9000) elaborating module
INFO - C:/lssc/radiant/2.0/cad
WARNING - C:/Projects/temp/Radiant (VERI-9000) elaborating module
Warning: Lattice VM Drivers detected (HW-DLN-3C (Parallel))
Programmer device database loaded
Done: design load finished with
```

デザイン書き込み



Programmerを起動すると自動で接続されているデバイスがスキャンされますが、正しくデバイスのスキャンが完了していない場合は、“Scan Device”をクリックします

Radiant Programmer Impl_1_1.xcf *

File Edit View Run Tools Help

Enable	Status	Device Family	Device	Operation	File Name
1	<input checked="" type="checkbox"/>	LIFCL	LIFCL-40	Fast Configuration	...rossLink_NX_WG/impl_1/crossLink_NX_WG_impl_1.bit

デバイスのスキャンが正常に完了すると、“Device Family”、“Device”に接続されているデバイス型番が反映されます

デバイスへの書き込みオペレーションを設定します。書き込みオペレーション詳細については、次ページに記載しています

デザイン書き込み



Radiant Programmer - impl_1_1.xcf *

File Edit View Run Tools Help

Enable	Status	Device Family	Device	Operation	File Name
1	<input checked="" type="checkbox"/>	LIFCL	LIFCL-40	Fast Configuration	...rossLink_NX_WG/impl_1/crossLink_NX_WG_impl_1.bit

書き込みオペレーションの詳細設定は、青枠部分をダブルクリックします

LIFCL - LIFCL-40 - Device Properties

General Device Information

Device Operation

Target Memory: External SPI Flash Memory (SPI FLASH)

Port Interface: JTAG2SPI

Access Mode: Direct Programming

Operation: Erase, Program, Verify

Programming Options

Programming file: D:\test\impl_1\crosslink_nx_ADC_test_impl_1.bit

SPI Flash Options

Family: SPI Serial Flash

Vendor: Macronix

Device: MX25L12833F

Package: 8-pin SOP

SPI Programming

Data file size (Bytes): 808978 Load from File

Start address (Hex): 0x00000000

End address (Hex): 0x000C0000

Erase SPI part on programming error

Secure SPI flash golden pattern sectors

OK Cancel

コンフィグレーションメモリの書き込み先を選択します。外部 SPI-FLASH、内部NVCM、SRAMから選択可能です

書き込む際のI/Fを選択します。JTAG、又はSPIから選択可能です。デバイスとケーブルを接続しているI/Fに合わせて設定してください

アクセスモードを選択します。通常のコンフィグレーションデータ書き込みであれば、“Direct Programming”を選択します

書き込み時のオペレーション、その他実施したいオペレーションメニューから選択します。通常、外部SPI-FLASHに書き込む際は、“Erase Program Verify”を選択します

書き込むコンフィグレーションデータを選択します

SPI-FLASHへ書き込む際は、型番に合わせて各種設定します。設定が誤っていると書き込みエラーが発生します

すべての設定が完了したらOKをクリックします

デザイン書き込み

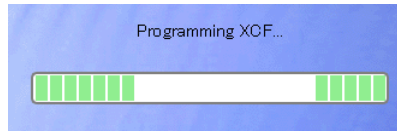


Radiant Programmer - impl_1.xcf *

File Edit View Run Tools Help

書き込みは“Program Device”をクリックして開始します

Enable	Status	Device Family	Device	Operation
1 <input checked="" type="checkbox"/>		LIFCL	LIFCL-40	Erase,Program,Verify



書き込みには数十秒ほどかかります。

Radiant Programmer - impl_1.xcf *

File Edit View Run Tools Help

Enable	Status	Device Family	Device	Operation
1 <input checked="" type="checkbox"/>	PASS	LIFCL	LIFCL-40	Erase,Program,Verify

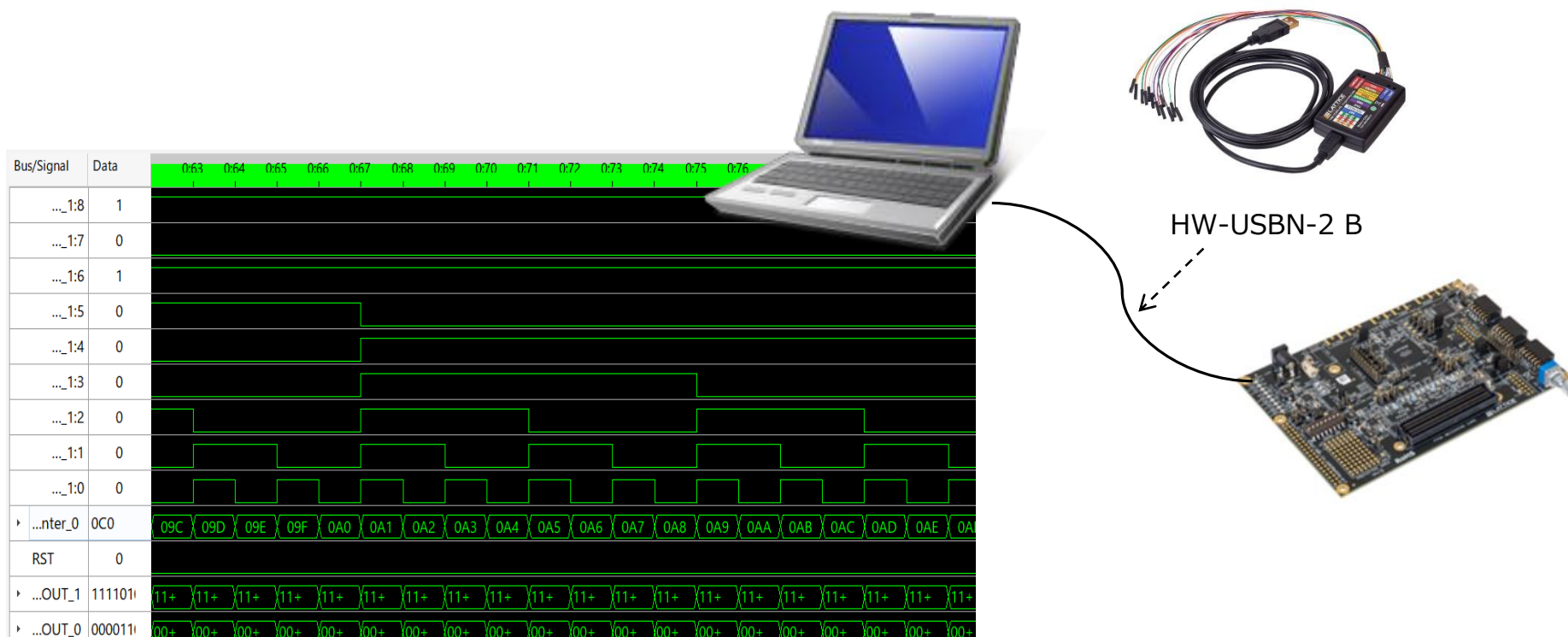
書き込みが正常に完了すると、“PASS”と表示されます

7. 実機上での内部波形観測


実機上での内部波形観測

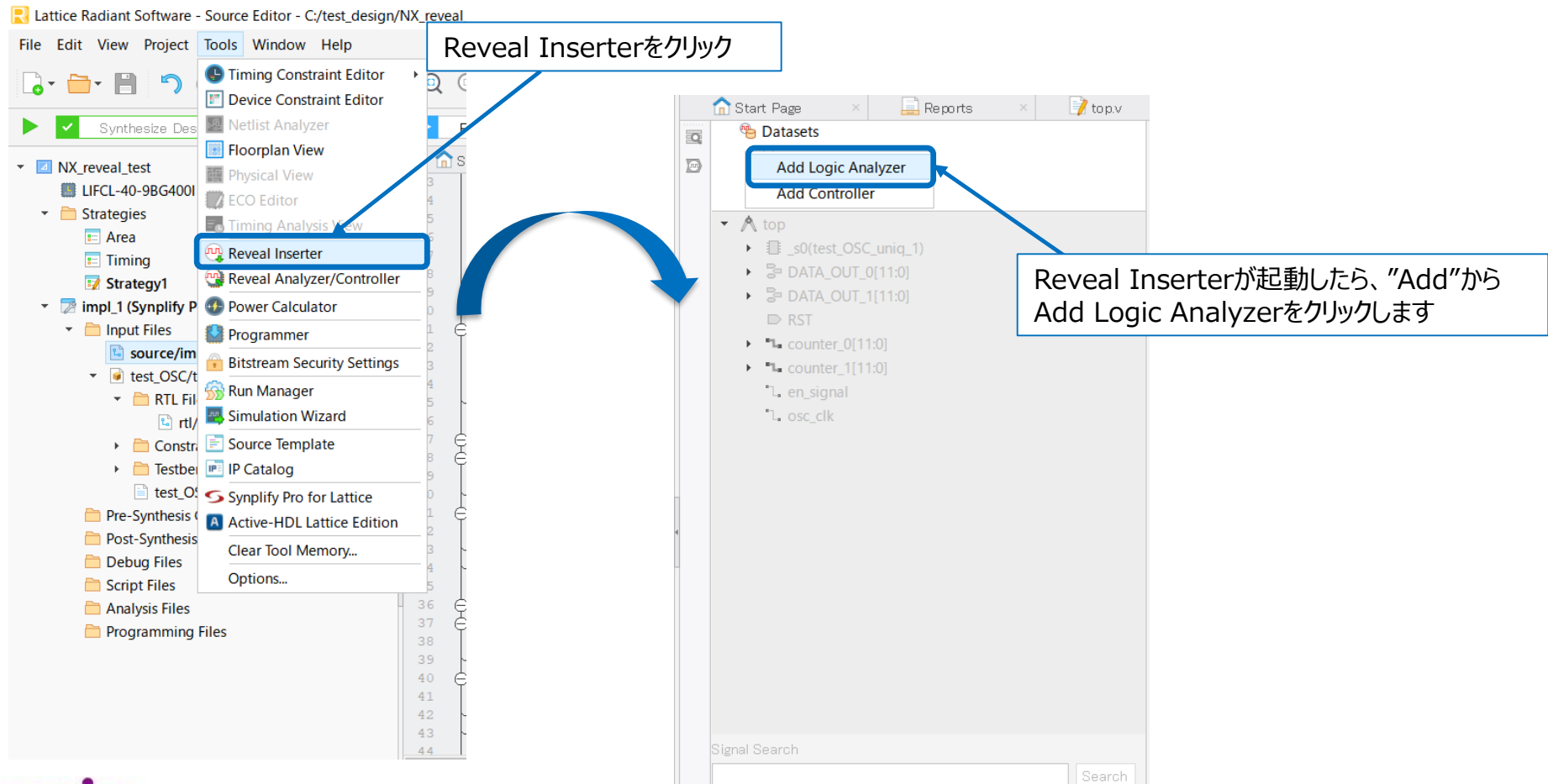


- RADIANTでは、デバイス内部の実機波形を観測できるデバッグ用機能、Revealが用意されています
- Revealを使用した実機上での波形観測は、下図のようにPCと、デバイスが実装されたボードを接続します
- ここでは、Revealを使用した内部波形観測フローを紹介します



実機上での内部波形観測

- Revealを使用するには、デバイス内部にRevealコアを生成 & 実装する必要がある為、Reveal InserterでRevealコアを生成します
- Tools > Reveal Inserter もしくはToolbarからReveal Inserterアイコン  をクリックします



The screenshot shows the Lattice Radiant Software Source Editor interface. The 'Tools' menu is open, and the 'Reveal Inserter' option is highlighted with a blue box and a callout that says 'Reveal Inserterをクリック'. A large blue arrow points from this menu item to the 'Datasets' window on the right. In the 'Datasets' window, the 'Add Logic Analyzer' option is highlighted with a blue box and a callout that says 'Reveal Inserterが起動したら、“Add”から Add Logic Analyzerをクリックします'. The 'Datasets' window shows a tree view of the design hierarchy, including 'top', '_s0(test_OSC_uniq_1)', 'DATA_OUT_0[11:0]', 'DATA_OUT_1[11:0]', 'RST', 'counter_0[11:0]', 'counter_1[11:0]', 'en_signal', and 'osc_clk'. The 'Signal Search' field is visible at the bottom of the 'Datasets' window.

実機上での内部波形観測



観測可能な信号がリストで表示されます。これらの信号から観測した信号を選択し、Traceの下にドラッグ&ドロップします

The screenshot shows the Reveal Inserter interface with several components highlighted by blue boxes and arrows:

- Left Panel (Datasets):** A list of signals including `DATA_OUT_0[11:0]@Tc`, `DATA_OUT_1[11:0]@Tc`, `RST@Tc`, `counter_0[11:0]@Tc`, `counter_1[11:0]@Tc`, `en_signal@Tc`, and `osc_clk@C`.
- Trace Panel:** A list of signals including `DATA_OUT_0`, `DATA_OUT_1`, `RST`, `counter_0`, `counter_1`, and `en_signal`.
- Configuration Panel:** Settings for the trace, including:
 - Sample Clock:** Set to `osc_clk`.
 - Buffer Depth:** Set to `256`.
 - Implementation:** Set to `EBR`.
 - Data Capture Mode:** Set to `Single Trigger Capture`.

サンプリングクロックを選択します。必ず10MHz以上のクロックを選択してください

内部信号をバッファする為のメモリタイプを選択します

Buffer Depthを決定します。Buffer Depthの値を高くすることによって、観測したい信号の数を増やすことが可能ですが、実デザインで使用しているEBR、又はDistributed RAM容量によってRevealで使用可能な上限値が決まります

実機上での内部波形観測



下タブをTrigger Signal Setupに切り替えます

The screenshot shows the Reveal Inserter software interface. On the left, a tree view shows the project structure under 'top', including components like '_s0(test_OSC_uniq_1)', 'DATA_OUT_0[11:0]@Tc', 'DATA_OUT_1[11:0]@Tc', 'RST@Tc', 'counter_0[11:0]@Tc', 'counter_1[11:0]@Tc', 'en_signal@Tc,Tg', and 'osc_clk@C'. The 'en_signal@Tc,Tg' component is selected. The main window displays the 'Trigger Unit' configuration. A table lists the trigger units:

Name	Signals (MSB:LSB)	Operator	Radix	Value
1 TU1	en_signal	==	Bin	1

Below the table are 'Add' and 'Remove' buttons. The 'Default Trigger Rac' is set to 'Bin'. Below the table is the 'Trigger Expression' section with a table:

Name	Expression	RAM Type	Sequence Depth	Max Sequence Depth	Max Event Counter
------	------------	----------	----------------	--------------------	-------------------

Below this table are 'Add' and 'Remove' buttons. The 'Event Counter' section has 'Enable final trigger counter' checked and 'Event Counter Val' set to 8. The 'Trigger Out' section has 'Enable Trigger Out' checked, 'Net' set to 'NET', and 'reveal_debug_top_LA0_net' selected. 'Polarity' is set to 'Active High' and 'Minimum pulse width' is 0. At the bottom, the 'Trace Signal Setup' tab is active, and the 'Trigger Signal Setup' sub-tab is selected. A blue box highlights the 'Trigger Signal Setup' sub-tab. A blue arrow points from the text box above to this sub-tab.

Trigger Unitでは、トリガー信号と条件を設定します。Addボタンをクリックして、トリガー信号と条件、信号観測時の表示方法をBin/Oct/Dec/Hexから設定します。この例では、"en_signal"がHighになったらトリガーがかかる設定となっています

実機上での内部波形観測



Trigger Expressionでは、Trigger unitで設定した複数のトリガーを組み合わせることで論理を組み込むことが可能です。この例では、“en_signal”がHighかつ、“RST”がLowの時にトリガーがかかる設定となっています。論理はExpressionのセルに直接記述します。論理の記述方法は次ページを参照ください。尚、Trigger Expressionでは、論理の組み合わせが不要な場合もExpressionのセルにTrigger Unitで設定したトリガー条件を入力する必要があります。この例では、TU1又はTU2のいずれかを記述します。

The screenshot shows the Reveal Inserter software interface. On the left is a tree view of datasets, with 'top_LA0' selected. The main window is divided into two sections: 'Trigger Unit' and 'Trigger Expression'.

Trigger Unit Configuration:

Name	Signals (MSB:LSB)	Operator	Radix	Value
1 TU1	en_signal	==	Bin	1
2 TU2	RST	==	Bin	0

Trigger Expression Configuration:

Name	Expression	RAM Type	Sequence Depth	Max Sequence Depth	Max Event Counter
1 TE1	TU1 & TU2	3 Slices	1	2	1

Additional settings include: Event Counter (Enable final trigger counter: , Event Counter Val: 8), Trigger Out (Enable Trigger Out: , Net: NET, reveal_debug_top_LA0_net), and Polarity (Active High, Minimum pulse width: 0).

演算子	演算子の意味
&	前後のTUの and 論理
	前後のTUの or 論理
^	前後のTUの xor 論理
!	後に続くTUの not 論理
()	括弧内の論理式を優先
next	前のTU成立後、次のサンプルで後のTUが成立
then	前のTU成立後、後のTUが成立
#	前に指定されたTUが、後に指定された回数成立
##	前に指定されたTUが、後指定された回数連続して成立

■ Trigger Expression記述例

TU1 & TU2 : TU1とTU2が同時にTrueの場合にトリガ生成
TU1 | TU2 : TU1もしくはTU2がTrueの場合にトリガ生成
!TU3 : TU3がTrueでない場合にトリガ生成
TU1 & !TU4 : TU1がTrueかつTU4がTrueでない場合にトリガ生成
TU3 ^ TU1 : TU3もしくはTU1のどちらか一方がTrueの場合にトリガ生成
TU1 next TU2 : TU1がTrueになった次のサンプリングでTU2がTrueの場合にトリガ生成
TU1 then TU2 : TU1がTrueになった次のサンプリング以降にTU2がTrueの場合にトリガ生成
TU5 #2 : TU5が2回Trueになった場合にトリガ生成
TU5 ##2 : TU5が2回連続してTrueになった場合にトリガ生成

* #および##演算子で繰り返し回数を指定する場合は“Max Event Counter”セルの設定が必要

* thenおよびnext演算子でシーケンシャル・トリガを生成する場合は“Max Sequence Depth”セルの設定が必要

実機上での内部波形観測



すべての設定が完了したら、Design Rule Checkアイコンをクリックします

Name	Signals (MSB:LSB)	Operator	Radix	Value
1 TU1	en_signal	==	Bin	1

Name	Expression	RAM Type	Sequence Depth	Max Sequence Depth	Max Event Counter
1 TE1	TU1	3 Slices	1	2	1

```
Starting: "rv1_del_tu TU2"  
Checking design rules ...  
INFO - The number of EBRs needed  
INFO - The number of DistRAM (logic/ROM/RAM) slices needed is 3.  
Design Rule Check PASSED.
```

すべての設定において問題が無ければ、
"Design Rule Check PASSED"と表示されます

実機上での内部波形観測



Insert Debugアイコンをクリックします

ポップアップが表示されたら、例の様に2つのチェックが入っているを確認らして、OKをクリックします

Revealコアの名称を記述し、保存ボタンをクリックします

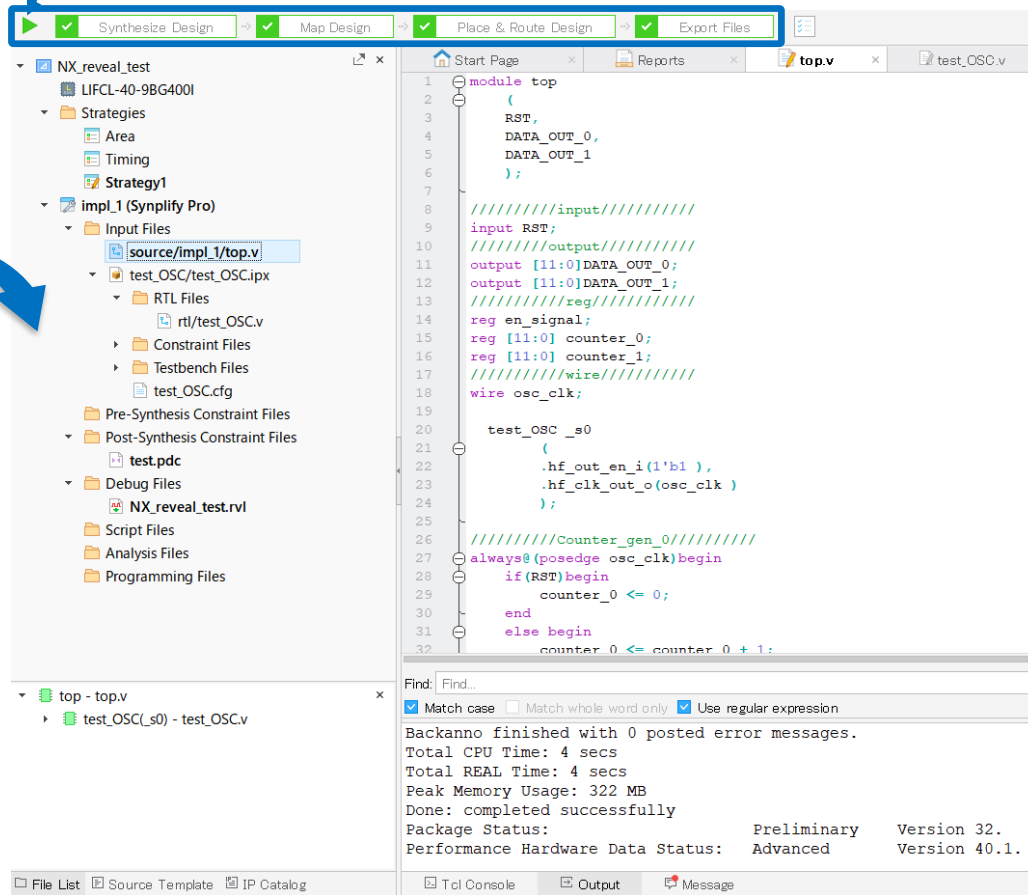
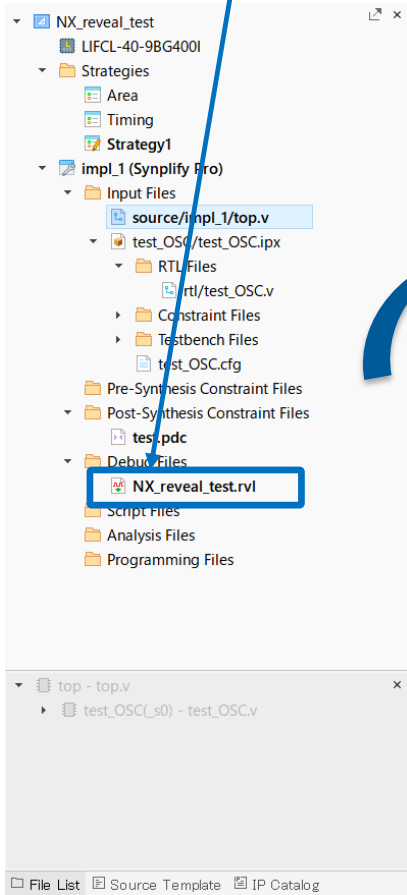
ファイル名(N): NX_reveal_test.rvl
ファイルの種類(T): Reveal Project (*.rvl)

実機上での内部波形観測



File ListタブにRevealコアファイル(.rvl)が追加されている事を確認します

Export Filesプロセスまで完了させ、Revealコアが実装されたコンフィグレーションデータを生成します



実機上での内部波形観測



Revealコアが実装されたコンフィグレーションデータをProgrammerを使用して書き込みます


The screenshot shows the Radiant Programmer software interface. The title bar reads "Radiant Programmer - impl_1.xcf *". The menu bar includes "File", "Edit", "View", "Run", "Tools", and "Help". Below the menu bar is a toolbar with various icons. The main area contains a table with the following data:

Enable	Status	Device Family	Device	Operation	File Name
1	<input checked="" type="checkbox"/>	LIFCL	LIFCL-40	Erase,Program,Verify	...test/NX_reveal_test/impl_1/NX_reveal_test_impl_1.bit

Below the table, there is a blue box with the text "Programming XCF..." and a progress bar with several green segments, indicating the progress of the programming operation.

実機上での内部波形観測

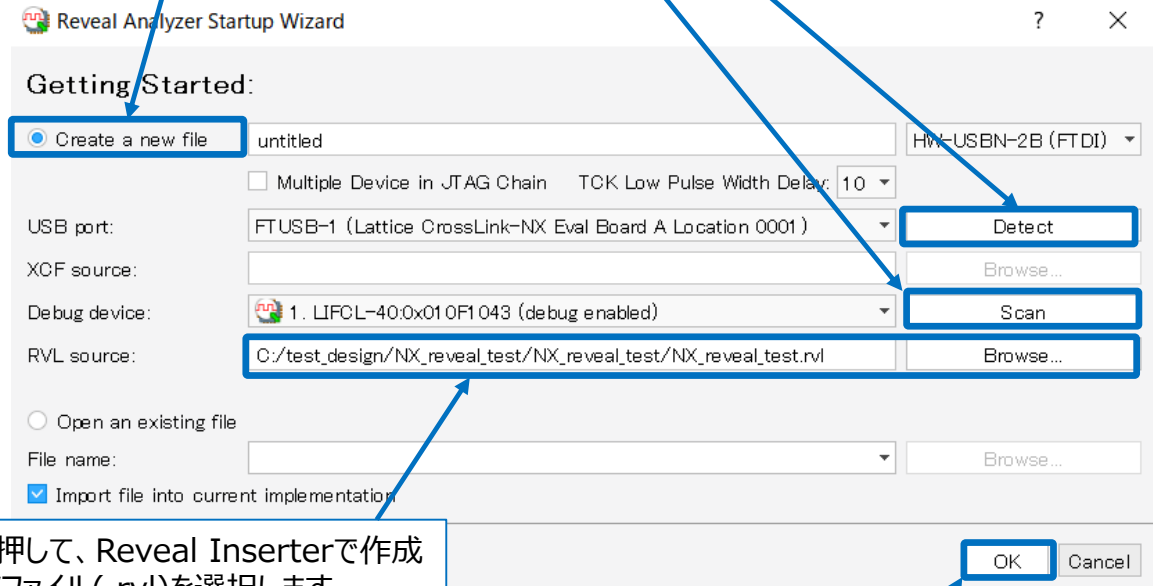
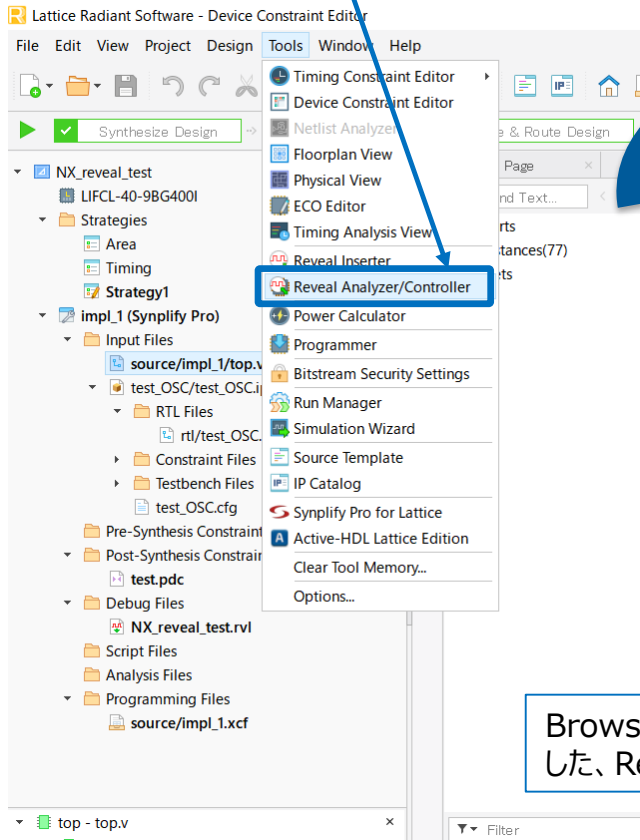


書き込みが完了したら、Tools > Reveal Analyzer/Controller もしくはToolbarから Reveal Analyzer/Controllerアイコンをクリックします

ポップアップ画面が出てきたら、Create a new file にチェックを入れます。以降、デバイスとPCは接続した状態で作業を行う必要があります

Detectボタンをクリックし、PCと接続されているケーブルを認証させます

Scanボタンを押して対象デバイスを認識させます




Browseボタンを押して、Reveal Inserterで作成した、Revealコアファイル(.rvl)を選択します

上記設定が完了したらOKボタンをクリックします

実機上での内部波形観測



Reveal Analyzerが起動したら、Ready表示になっていることを確認して、 アイコンをクリックします

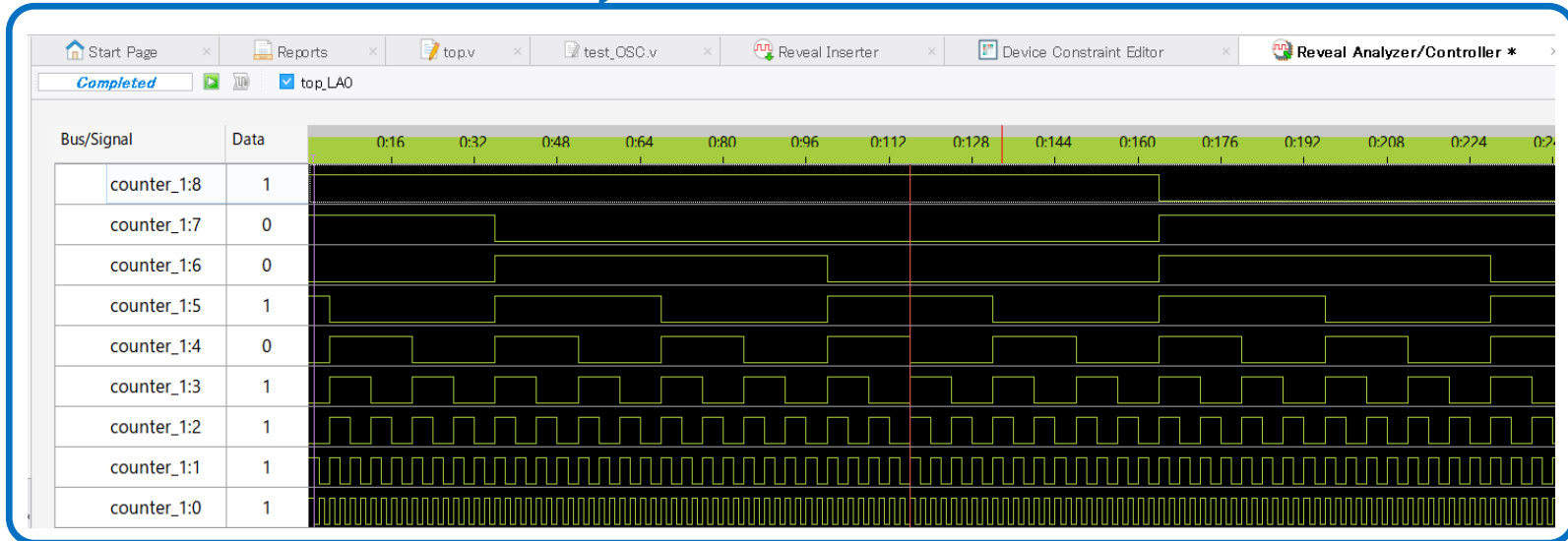
The screenshot shows the Reveal Analyzer interface. The 'Trigger Unit' section contains a table with the following data:

Name	Signals (MSB:LSB)	Operator	Radix	Value
TU1	en_signal	==	Bin	1

The 'Trigger Expression' section contains a table with the following data:

Name	Expression	Sequence Depth	Max Sequence
<input checked="" type="checkbox"/> TE1	TU1	1	2

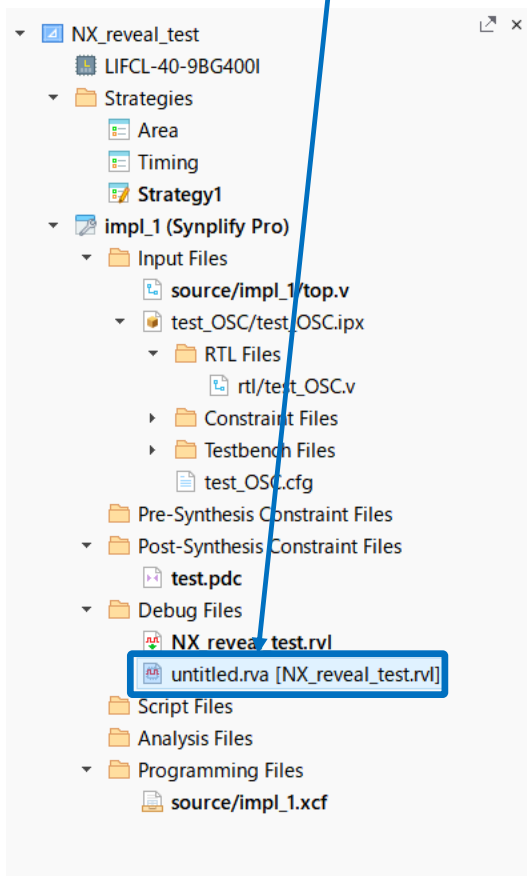
トリガーがかかると、下図のように内部信号が表示されます



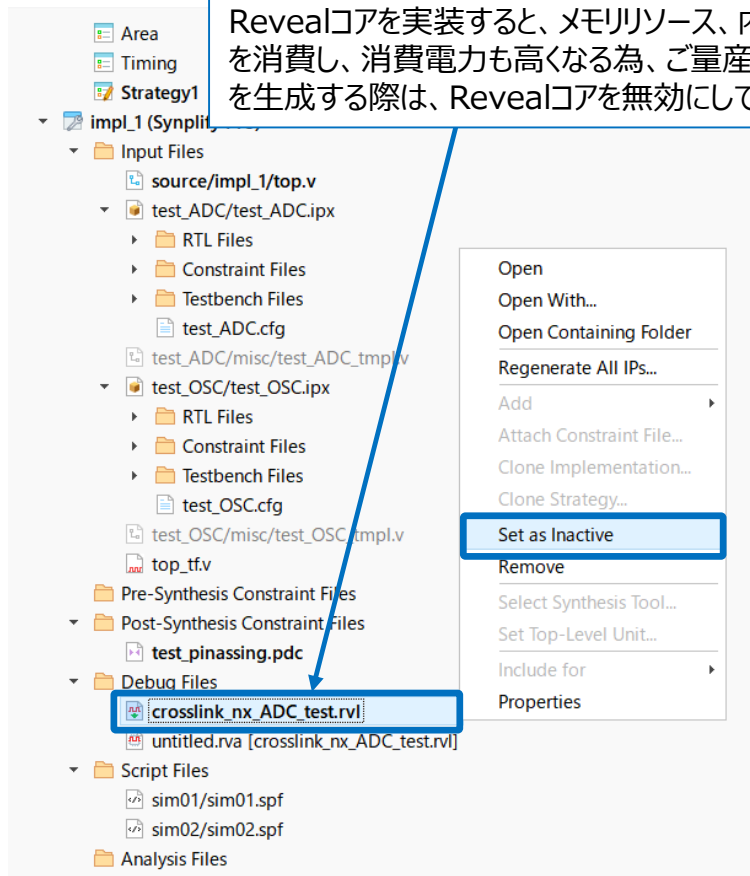
実機上での内部波形観測



初回移行、再度Reveal Analyzerを起動するときは、File ListのReveal Analyzer起動ファイル(.rva)をダブルクリックするだけで、起動させることが可能です



Revealコアは、File ListのRevealコアファイル(.rvl)を右クリック後、Set as Inactiveを選択することによって無効にすることが可能です。Revealコアを実装すると、メモリリソース、内部ロジック、配線、リソースを消費し、消費電力も高くなる為、ご量産のコンフィグレーションデータを生成する際は、Revealコアを無効にして頂く事をお勧め致します



Revision History



Date	Revision	Page	Change Information	Updated by
2020/04/24	1.0		First Revision	W. Nakatsuka
2020/04/30	1.1	P74~P94	デザイン書き込み方法、内部波形観測方法を追加	K. Ishigaki
2021/08/02	1.2	P19	誤記修正	H. Nogawa