

Lattice iCE Family Designing Tool “iCEcube2” Startup Manual

Sep-2013

Tecstar Company

Section 1 :

iCEcube2によるデザイン設計フロー

- スタートメニューより、プログラム > SiliconBlue iCEcube2 > iCEcube2 を起動します。
新規プロジェクトを作成するには、File > New Project を選択するか、Create a new Projectアイコンをクリックします。



新規プロジェクトの作成

任意のプロジェクト名を直接入力します。

任意のプロジェクト保存先フォルダを直接入力または右のブラウズアイコンからフォルダを指定します。
※新規プロジェクトを作成する場合はプロジェクト毎に新規プロジェクトフォルダを作成し、そのフォルダを保存先に指定することを推奨します。

※プロジェクト名、プロジェクトフォルダは半角英数のみを使用し、日本語やスペースは使用しないで下さい。

プルダウンメニューから設計するデバイスファミリー、デバイス型番、パッケージを選択します。

温度条件を選択します。

コア電圧マージンを選択します。±5%を選択して下さい。

各I/Oバンクの電源電圧を指定します。各I/Oバンクで使用するスタンダードに合わせて設定して下さい。

タイミング解析時にデバイススペックとしてBest, Typical, Worstのいずれのパラメータを算出するかを指定します。タイミング解析時に別途設定できます。

Start From Synthesis :
バンドル版Synplifyを使用して論理合成する段階から設計開始します。通常はこれを選択します。
Start From BackEnd :
有償版Synplify等で予め合成されたネットリストを用いて設計開始します。
IP Generation :
ユーザーオリジナルIPを生成する際に選択します。

設定完了後クリック

- ソースや制約ファイルが用意されている場合は、次のウィンドウでインポートすることができます。プロジェクト作成後にインポートすることも可能です。

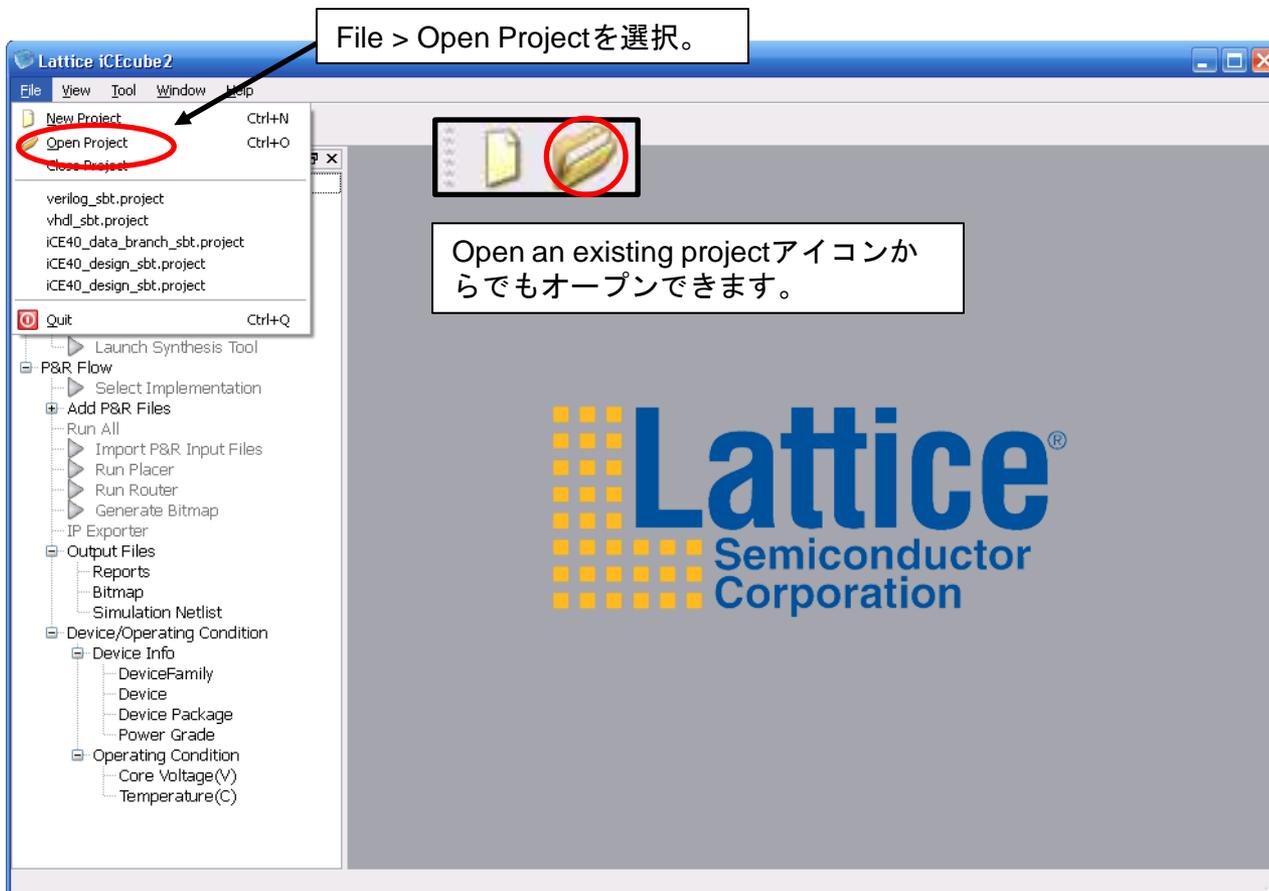
① Verilog, VHDL, EDIF, SDCファイル等を選択します。
(本資料ではプロジェクト作成後にVerilogファイルをインポートします。)

② クリックして右のウィンドウにインポート。

③ インポート完了後クリック。

以上新規プロジェクトが作成され、iCEcube2のメインウィンドウが表示されます。

- 既存のプロジェクトをオープンする場合は、File > Open Project より既存のプロジェクトファイル(.sbt.project)を選択します。Open an existing projectアイコンからでもオープンすることができます。



SiliconBlue iCEcube2 - [Output]

File View Tool Window Help

Project Name: test_design

Output

Current Implementation test_design_Implmnt its sbt path: C:/Projects/SiliconBlue/test_01/test_design/test_design_Implmnt\sbt C:\SbtTools\2011_12\synpro\bin\synplify_pro.exe C:\Projects\SiliconBlue\test_01\test_design\test_design_syn.prj
Current Implementation test_design_Implmnt its sbt path: C:/Projects/SiliconBlue/test_01/test_design/test_design_Implmnt\sbt

Project

- Synthesis Tool
 - Add Synthesis Files
 - Design Files
 - counter.v
 - Constraint Files
 - counter_syn.sdc
 - Launch Synthesis Tool
- P&R Flow
 - Select Implementation(test_design_Implmnt)
 - test_design.edf
 - test_design.scf
 - Add P&R Files
 - Run All
 - Import P&R Input Files
 - Run Placer
 - Run Router
 - Generate Bitmap
 - Output Files
 - Reports
 - counter_timing.rpt
 - placer.log
 - Bitmap
 - counter_bitmap.bin
 - counter_bitmap.hex
 - counter_bitmap_int.hex
 - Simulation Netlist
 - counter_sbt.sdf
 - counter_sbt.v
 - counter_sbt.vhd
 - counter_sbt_vital.sdf
 - Device/Operating Condition
 - Device Info
 - DeviceFamily ICE40
 - Device LP640
 - Device Package CM81
 - Power Grade
 - Operating Condition
 - Core Voltage(V) 1.14
 - Temperature(C) 70

インポート済みのソースファイル(Verilog, VHDL等)とSynopsys形式の制約ファイル(.sdc)が表示されます。

論理合成後のネットリスト(.edf)と.sdcが変換されたiCEcube2で使用される制約ファイル(.scf)が表示されます。

フィッティング後のレポートファイル(.rpt)とログ(.log)が表示されます。

生成されたビットマップファイル(書込みファイル)が表示されます。バイナリ(.bin)、HEXファイル(.hex)、インテルHEX(_int.hex)の3つの形式で生成されます。

Timing Simulation用の遅延データとソースファイルが表示されます。

設計中のデバイス情報が表示されます。設計中に変更可能です。

コア電圧、ジャンクション温度設定が表示されます。設計中に変更可能です。

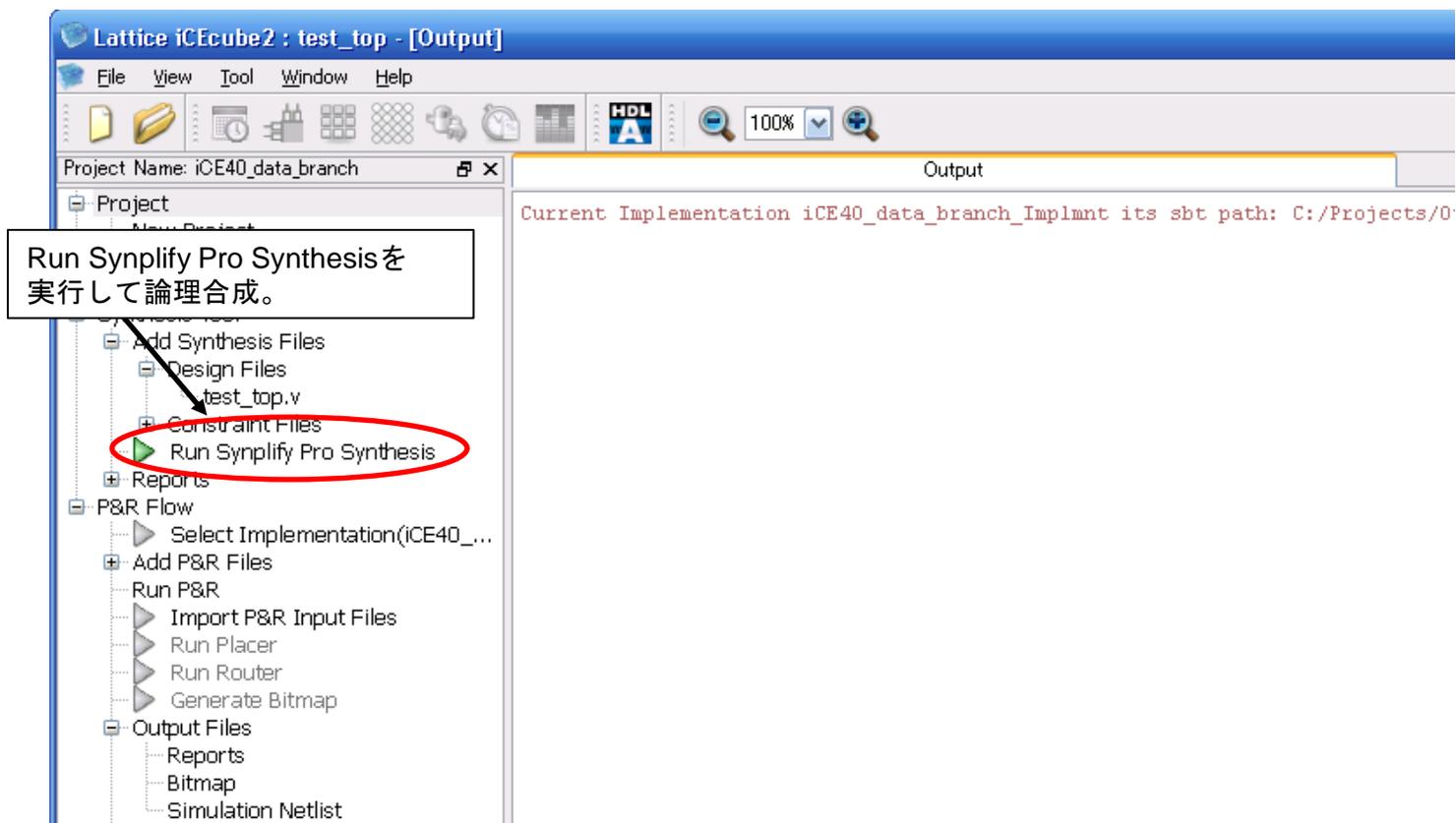
■ iCEcube2上では新規デザインソースの設計はできません。予めソースファイルを作成し、インポートする必要があります。

① Design Filesを右クリックし、Add Fileを選択します。

② 予め用意したソースファイルからインポートするファイルを選択し、Addボタン(">>")でFile to addボックスに移動させます。

③ 選択完了後、クリックします。

- ソースファイルがインポートされるとRun Synplify Pro Synthesisが実行可能になります。Run Synplify Pro Synthesisを実行すると、バックグラウンドでSynplify Proが起動し、論理合成を実行します。



(参考)作成したデザインの確認

- Synplify Proを単体起動して論理合成を行うと、Synplify Pro上のツールを用いて作成したデザインの確認が行えます。
(※このフローは確認が必要な場合のみ行います。必ず実施する必要はありません。)

① Tool -> Tool Options と選択。

② Synplify Pro タブにて Click "here" をクリックし、Synplify Pro を起動。

③ Run をクリックして論理合成を実行。

③ 0 errors であれば論理合成完了です。

Name	Size	Type	Modified
backup		Directory	18:53:27
corep		Directory	18:49:52
dm		Directory	18:49:53
sbt		Directory	18:54:10
synlog		Directory	18:50:05
synmp		Directory	21:18:24
synwork		Directory	21:15:54
AutoConstraint_counter...	186 bytes	sdc File	18:50:03
AutoConstraint_top.sdc	178 bytes	sdc File	21:15:57
rpt_counter_areasr	1 kB	areasr File	18:53:30
rpt_counter_areasr.htm	2 kB	html File	18:53:30
rpt_top_areasr	3 kB	areasr File	21:15:57
rpt_top_areasr.htm	3 kB	html File	21:15:57

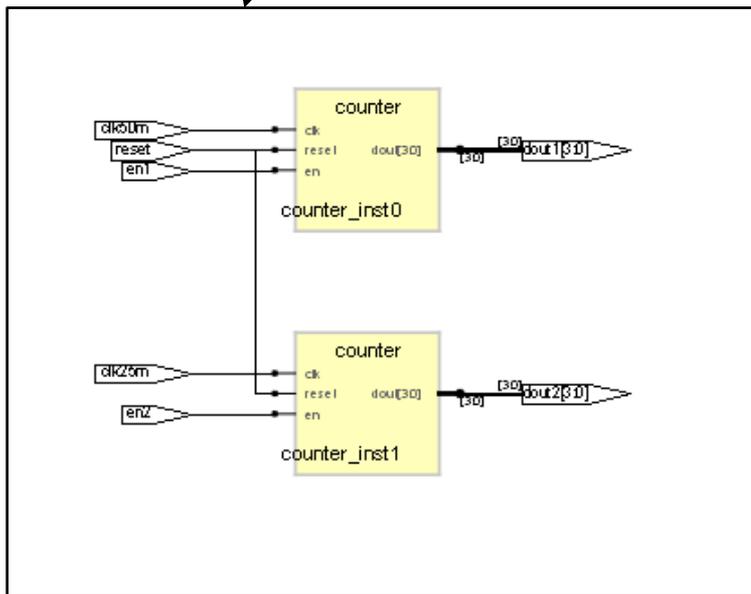
- 論理合成後、Design Hierarchyタブにてデザインの階層構造が確認できます。

Design Hierarchyタブに切り替えるとデザインの階層設計がツリー形式で確認できます。

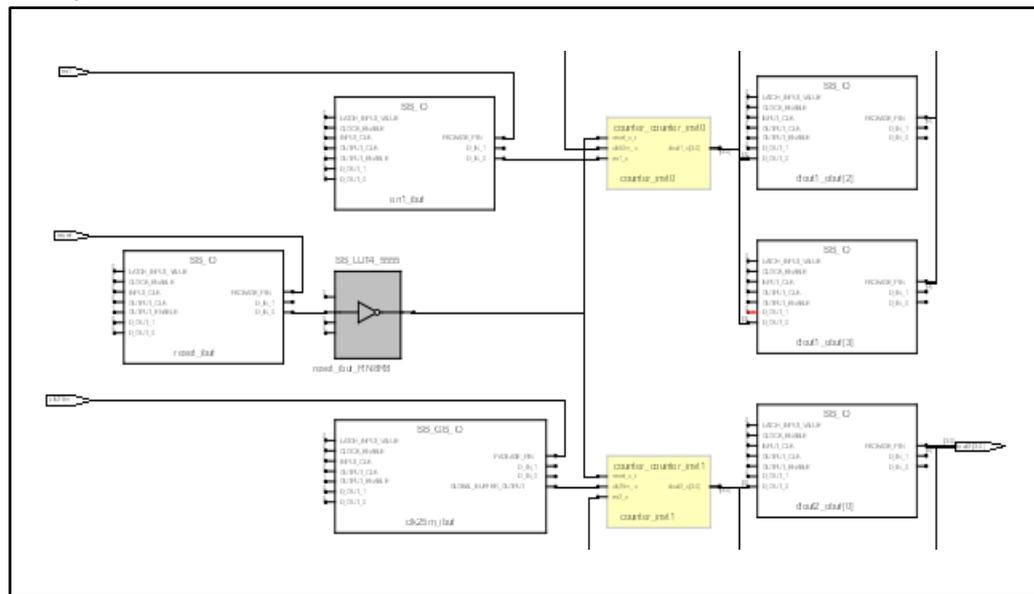
Instance (DesignBlock Name)	File Location
work.top	..\test_04\source\top.v
counter_inst1 (work.counter)	..\source\counter.v
sreg_inst (work.sreg)	..\source\sreg.v
counter_inst0 (work.counter)	..\source\counter.v
sreg_inst (work.sreg)	..\source\sreg.v

(参考)作成した設計の確認

- 論理合成後、Synplify Pro上のRTL Viewを実行すると論理展開されたブロック図が確認でき、Technology Viewを実行するとより詳細な内部構成図が確認できます。

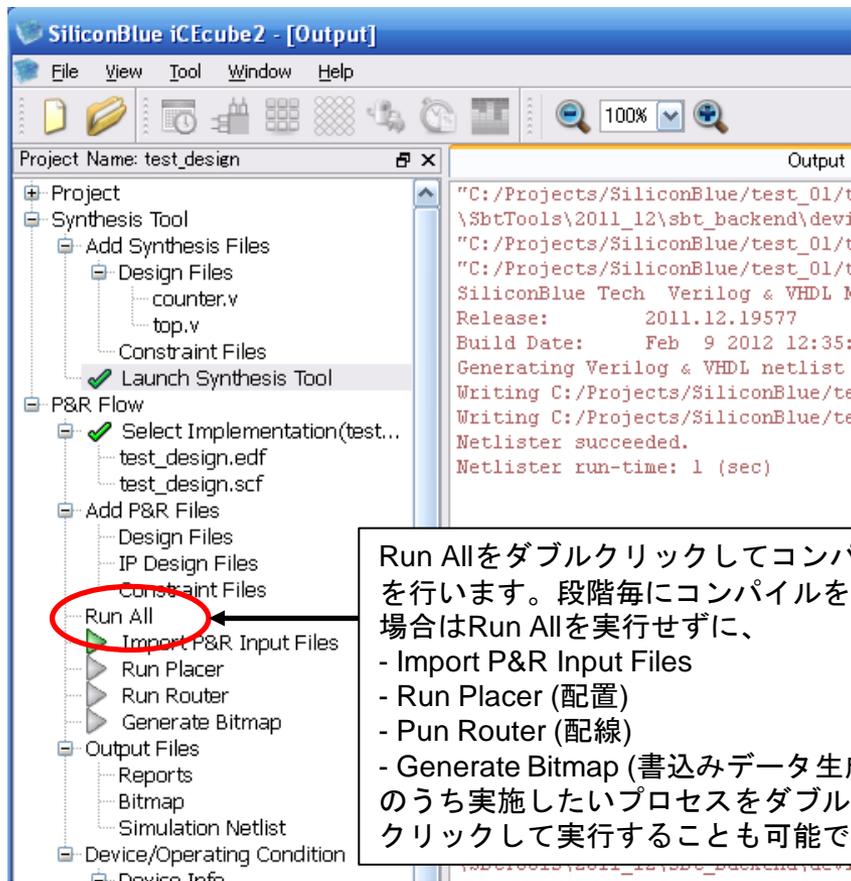


RTL View



Technology View

- 論理合成が完了すると、生成されたネットリスト(.edf) が自動でプロジェクトに追加されています。この状態でRun Allを実行し、コンパイルを行います。

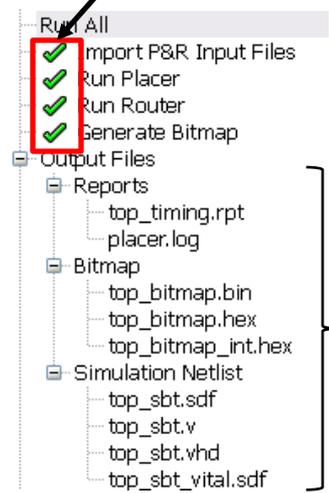


Run Allをダブルクリックしてコンパイルを行います。段階毎にコンパイルを行いたい場合はRun Allを実行せずに、

- Import P&R Input Files
- Run Placer (配置)
- Run Router (配線)
- Generate Bitmap (書込みデータ生成)

のうち実施したいプロセスをダブルクリックして実行することも可能です。

Run All実行後、全プロセスにチェックが付けばコンパイル完了です。



レポートファイル、ログファイル、ビットマップファイル3種、Timing Simulation用遅延ファイルが生成されます。

- iCEcube2からTiming Constraints Editorを起動します。P&R FlowでImport P&R Input Filesまで実行すると、Timing Constraints Editorが起動できるようになります。

The screenshot shows the Lattice iCEcube2 software interface. On the left, the 'P&R Flow' tree is visible, with 'Import P&R Input Files' highlighted by a red circle and an arrow pointing to a callout box. The main window shows the 'Timing Constraints Editor' with a table containing one row and a tab labeled 'untitled0.sdc' circled in red. A callout box points to this tab. Another callout box points to the 'Timing Constraints Editor' icon in the toolbar.

① Import P&R input Filesまで実行。

② Timing Constraints Editorを起動。

③ 制約設定タブが開きます。

Enabled	Source	Name	Period(ns)	Waveform(ns)
1				

- Clockタブにてクロック信号を設定します。ここで設定された信号がクロックとして認識され、Input DelayやOutput Delay制約でクロック信号として設定できるようになります。また、ここで設定しなかったクロックは、Setup/Holdタイミング解析時にもクロック信号として認識されず、誤ったクロック信号での解析がされるケースがありますので、デザイン上のクロック信号は必ずここで設定しておくことを推奨します。
また、最大動作周波数(Fmax)の設定もClocksタブにて同時に行います。

Project Name: iCE40_timing_constraint... [test_constraints.sdc]

Enabled	Source	Name	Period(ns)	Waveform(ns)
<input checked="" type="checkbox"/>	CLK	CLK	10	0 5
<input type="checkbox"/>				

① Clocksタブを選択。

② 設定を有効にする行にチェックを入れます。

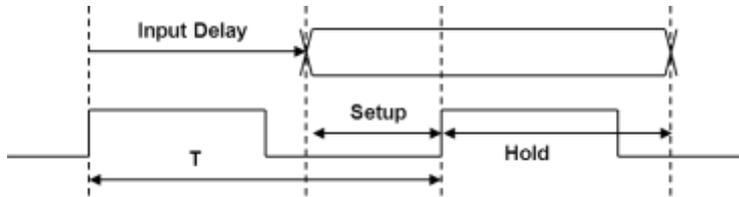
③ プルダウンメニューからクロック信号を選択。

④ 動作周波数制約としてクロック周期を設定します。

【オプション】
クロックの立ち上がりポイントと立下りポイントを設定します。この例では10nsの周期のクロックの0nsで立ち上がり、5nsで立ち下がる(Duty 50:50)設定になります。

Setup Time (Input Delay)制約の設定

- Setup Time制約はInput Delayタブで設定します。Input Delayとは下図のように入力データのクロックに対する遅延量の設定で、前段デバイスのClock to Output Timeに相当します。iCEcube2ではこの値から以下のようにSetup Timeを計算して制約に用います。



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

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

The screenshot shows the Lattice iCEcube2 software interface. The 'test_constraints.sdc*' file is open, and the 'Input Delay' tab is selected. The table below shows the configuration for four input signals.

Enabled	InputList	Clock	Delay Value(ns)	Clock Fall	Add Delay
<input checked="" type="checkbox"/>	DINO[0]	CLK	5	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	DIN1[2]	CLK	5	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	DINO[1]	CLK	5	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	DINO[2]	CLK	5	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>

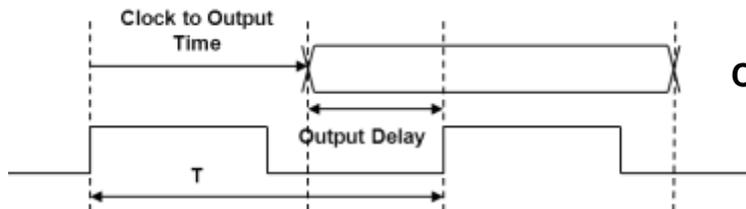
Annotations in the screenshot:

- ① Input Delayタブを選択。
- ② 設定を有効にする行にチェックを入れます。
- ③ プルダウンメニューからデータ信号を選択。
- ④ プルダウンメニューからクロック信号を選択。
※Clockタブで制約設定されているクロックのみ有効。
- ⑤ Input Delay量を入力。

【オプション】
立下りエッジからのDelayを設定する場合、Clock FallとAdd Delayにチェックを入れます。

Clock to Output Time (Output Delay)制約の設定

- Clock to Output Time制約はOutput Delayタブで設定します。Output Delayとは下図のように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、後段デバイスのSetup Timeに相当します。iCEcube2ではこの値から以下のようにClock to Output Timeを計算して制約に用います。



$$\text{Clock to Output Time} = T - \text{Output Delay}$$

②設定を有効にする行にチェックを入れます。

⑤Output Delay量を入力。

Enabled	OutputList	Clock	Delay Value(ns)	Clock Fall	Add Delay
<input checked="" type="checkbox"/>	DOUT0[0]	CLK	3	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	DOUT0[1]	CLK	3	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	DOUT0[2]	CLK	3	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	DOUT0[3]	CLK	3	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>

③プルダウンメニューからデータ信号を選択。

④プルダウンメニューからクロック信号を選択。
※Clockタブで制約設定されているクロックのみ有効。

【オプション】
立下りエッジからのDelayを設定する場合、Clock FallとAdd Delayにチェックを入れます。

①Output Delayタブを選択。

- Max Delay制約はMax-delayタブで設定します。Max Delay制約はあるピンから入力されたデータが別のピンから出力されるまでの遅延制約になります。

① Output Delayタブを選択。

② 設定を有効にする行にチェックを入れます。

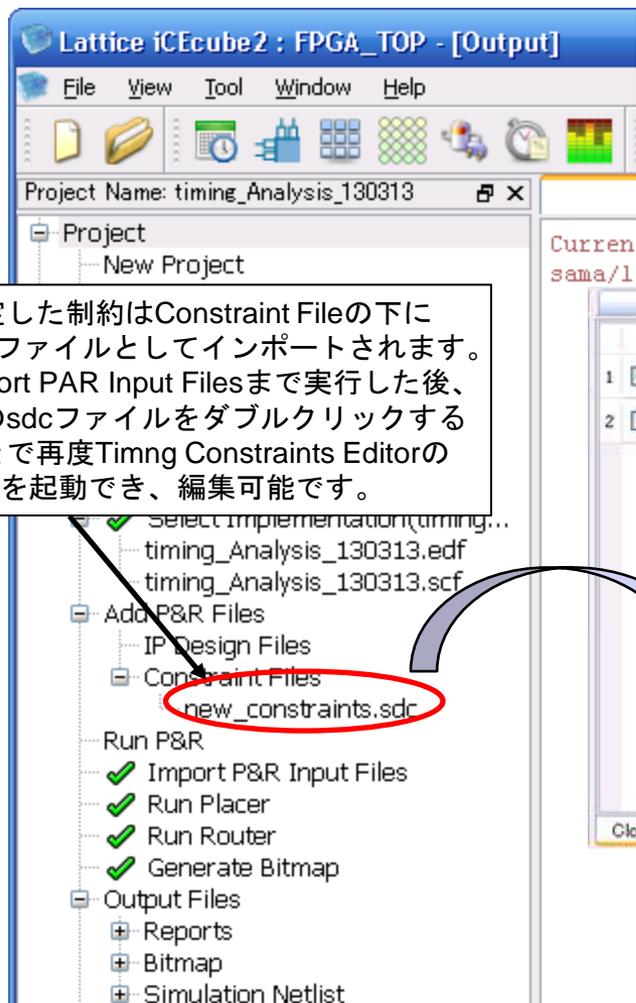
③ 遅延量を設定します。

④ プルダウンメニューからデータ入力ピンを選択。

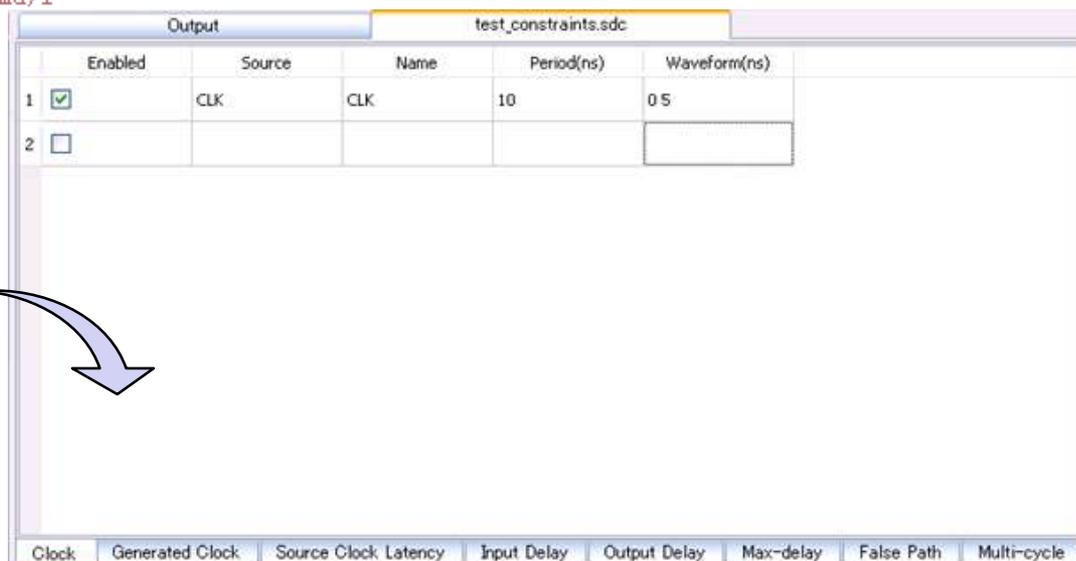
⑤ プルダウンメニューからデータ出力ピンを選択。

	Enabled	Delay Value(ns)	From	Rise From	Fall From	To	Rise To	Fall To	Through
1	<input checked="" type="checkbox"/>	7	THR_IN			THR_OUT			
2	<input type="checkbox"/>								

設定した制約はConstraint Fileの下に
sdcファイルとしてインポートされます。
Import PAR Input Filesまで実行した後、
このsdcファイルをダブルクリックする
ことで再度Timing Constraints Editorの
GUIを起動でき、編集可能です。



Current
sama/l



■ Pin Constraints Editorでピンアサイメントを行うことができます。

① Pin Constraints Editor
を起動。

【重要】
Lockedには必ずチェックを入れて下さい。チェックを入れていないとSaveを行ってもRun Allを実行すると設定が消えてしまいます。

② ピン番号、IOスタンダード、Pullモードを設定します。

Locked	Object List	Type	Pin Location	Bank	IO Standard	Pull Up
<input checked="" type="checkbox"/>	clk25m	Input	D3	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	clk50m	Input	E3	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	din1	Input	G1	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	din2	Input	E4	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	en1	Input	C2	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	en2	Input	E1	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	reset	Input	E2	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	dout1[0]	Output	B2	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	dout1[1]	Output	C1	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	dout1[2]	Output	B1	Left	SB_LVCMOS	Yes
<input checked="" type="checkbox"/>	dout1[3]	Output	C3	Left	SB_LVCMOS	Yes
<input type="checkbox"/>		Output	D2	Left	SB_LVCMOS	Yes
<input type="checkbox"/>		Output	F3	Left	SB_LVCMOS	Yes

EditorをSaveするとメッセージが出て制約ファイルがプロジェクトに追加されますので、追加後に再度Run Allを実施して下さい。再編集はPin Constraints Editorを再起動すれば実施できます。

■ Package ViewでGUIにてピンアサイメントを行うことができます。

① Package View
をクリック

② GUI上でドラッグ&ドロップによりピン配置
を行います。
【重要】
配置後は必ず右クリックのメニューからLock
を選択して下さい。LockせずにRun Allを実行
すると、設定が消えてしまいます。

Edit Pin Constraint
Merge/Unmerge FF
Lock
Unlock

IOスタンダードやPullモードを設定する
には、GUIまたはPortリストで信号を
右クリックし、Edit Pin Constraintを選択
して以下のWindowを開き、設定します。

Pin Constraint

Port Name: en1
IO Name: en1_ibuf
Pin Location: C2
IO Standard: SB_LVCMOS
Pull Up: Yes

OK Cancel

Package ViewをSaveするとメッセージが
出て制約ファイルがプロジェクトに追加さ
れますので、追加後に再度Run Allを実施
して下さい。
再編集はPackage Viewを再起動すれば実
施できます。

iCE40LP640 - CM81

- タイミング制約に対して、コンパイル後の結果がどのようになっているかレポートファイル(.rpt)またはTiming Analyzerにて確認が行えます。ここではTiming Analyzerによる確認方法を説明します。

Timing Analyzerを起動。

Timing Analyzerタブが開きます。

The screenshot shows the Timing Analyzer window with the following data:

Clock Name	Worst Slack(ps)	FMAX(MHZ)	Target Frequency(MHZ)	Failing Path #
1 clk25m	22752	57.98	25	0
2 clk50m	2752	57.98	50	0

Critical Path(1)

Start Point	End Point	Slack	Delay	Skew	Launch Clock	Capture Clock
1 counter_inst1.co...	dout2[0]	22752	5007	-1877	clk25m:R	clk25m:R

■最大動作周波数の確認はClock Summaryにて行います。

①Clock Summaryを選択。

③詳細表示ボタンをクリックすると、Detailウィンドウが展開し、ワーストパスの詳細が確認できます。

②制約設定されているクロックに対して、コンパイル結果が表示されます。任意のクロックをクリックすると、下にそのクロックドメイン内でのワーストパスが表示されます。

②上に表示されているリストのうち、クリックしたクロックドメイン内のワーストパスが表示されます。

タイミング結果 制約値

Clock Name	Worst Slack(ps)	FMAX(MHZ)	Target Frequency(MHZ)	Failing Path #
1 clk25m	33085	144.61	25	0
2 clk50m	13131	145.57	50	0

Critical Path(1)

Start Point	End Point	Slack	Delay	Skew	Launch Clock	Capture Clock
1 counter_inst1.co...	counter_inst1.c...	33085	6111	0	clk25m:R	clk25m:R

Path Detail

```

Start      counter_inst1.count_0_IC_1_2_0/icout
End        counter_inst1.count_31_IC_1_5_7/in3
Reference  clk25m
Setup Constraint 40000(p)
Path Slack  33085(p)

Capture Clock Arrival Time(clk25m:P#2)  40000
+ Capture Clock Source Latency           0
+ Capture Clock Path Delay              1877
- Setup Time                             -440

-----
End-of-path required time (ps)           41436

Launch Clock Arrival Time(clk25m:P#1)    0
+ Launch Clock Source Latency            0
+ Launch Clock Path Delay                1877
+ Clock To 0                             364
    
```

Data Path Clock Paths

Pin Name	Model No.	AT
counter_inst1.count_0_IC_1_2...	LogicCell40...	364 2241
Routing Delay		834
counter_inst1.count_0_IC_1_2...	LogicCell40...	0 3075
counter_inst1.count_0_IC_1_2...	LogicCell40...	409 3484
counter_inst1.count_1_IC_1_2...	LogicCell40...	0 3484
counter_inst1.count_1_IC_1_2...	LogicCell40...	136 3621
counter_inst1.count_2_IC_1_2...	LogicCell40...	0 3621
counter_inst1.count_2_IC_1_2...	LogicCell40...	136 3757
counter_inst1.count_3_IC_1_2...	LogicCell40...	0 3757
counter_inst1.count_3_IC_1_2...	LogicCell40...	136 3894
counter_inst1.count_4_IC_1_2...	LogicCell40...	0 3894
counter_inst1.count_4_IC_1_2...	LogicCell40...	136 4030

※Slack = データ遅延要求(Required Time) - 実際のワーストデータ遅延(Arrival Time)
 この値がプラスであれば制約をミートしています。また値が大きい程、制約に対して
 タイミングマージンが多いことを意味します。

■ Path Detailの見方は以下の通りです。

	Clock Name	Worst Slack(ps)	FMAX(MHZ)
1	clk25m	33085	144.61

```

Start      counter_inst1.count_0_LC_1_2_0/lcout
End        counter_inst1.count_31_LC_1_5_7/in3
Reference  clk25m
Setup Constraint 40000(p)
Path Slack  33085(p)

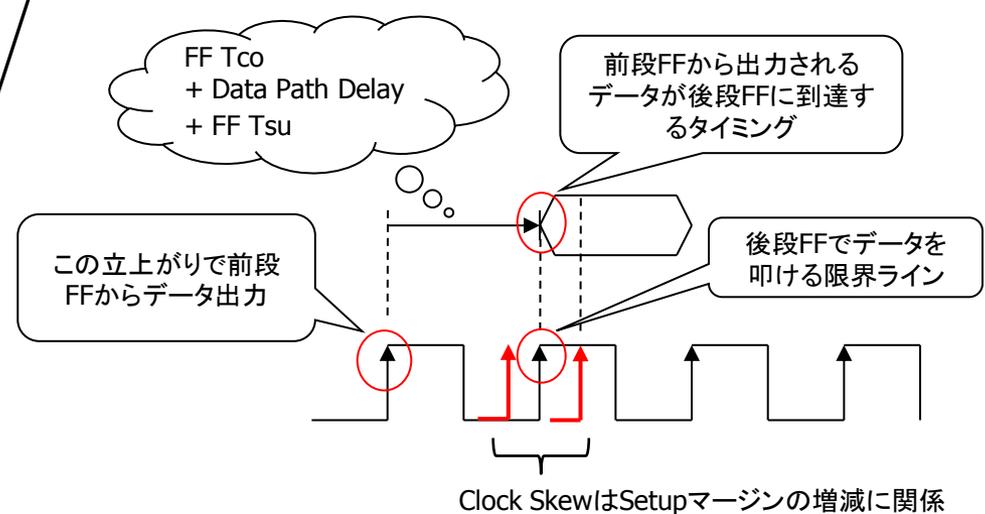
-----
Capture Clock Arrival Time(clk25m:R#2)      40000
+ Capture Clock Source Latency              0
+ Capture Clock Path Delay                 1877
- Setup Time                               -440
-----
End-of-path required time (ps)              41436

Launch Clock Arrival Time(clk25m:R#1)       0
+ Launch Clock Source Latency              0
+ Launch Clock Path Delay                 1877
+ Clock To Q                               364
+ Data Path Delay                         6111
-----
End-of-path arrival time (ps)              8352
Data path delay consists of logic delay (31 level(s)) 4913
ps, and routing delay 1197 ps.
    
```

ワーストパスの起点(前段FF)と終点(後段FF)の情報

後段FFまでのクロック遅延と、後段FFのSetup Time

前段FFまでのクロック遅延、前段FFのTco、データパス遅延



[Fmax 計算式]

$$1 / \{FF_Tco + Routing + FF_Tsu - (後段Clock Path Delay - 前段Clock Path Delay)\}$$

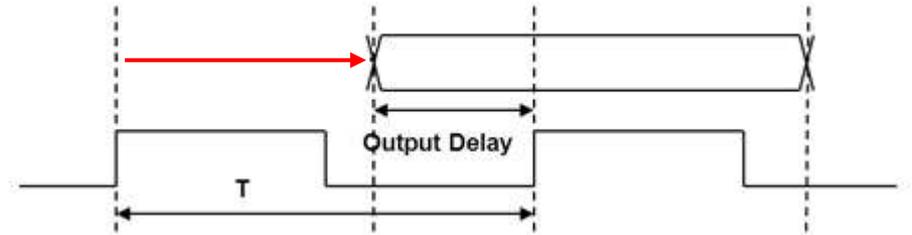
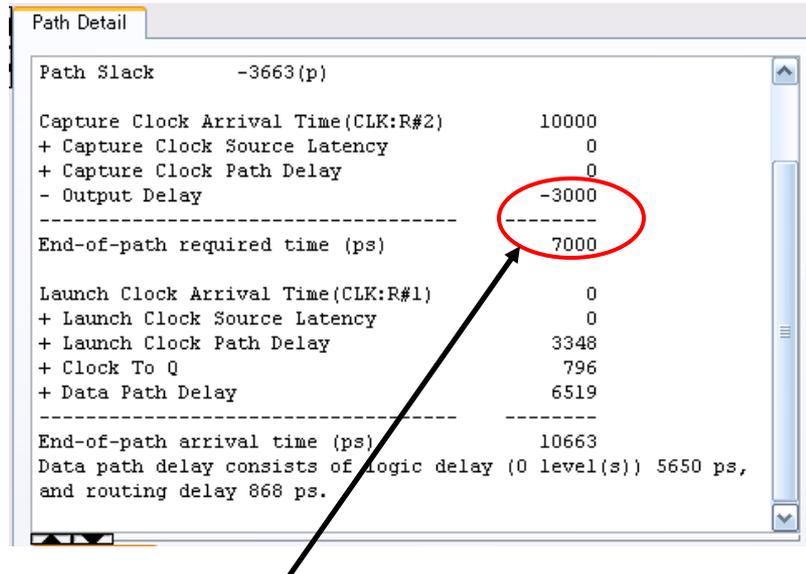
レポート内容を計算式にあてはめると、

$$364 + 6111 + 440 - (1877 - 1877) = 6915 \text{ ps} = 6.915 \text{ ns}$$

$$1 / 6.915 \text{ ns} \doteq 144.61 \text{ MHz}$$

最大動作周波数(Fmax)の確認

- Output Delay制約がかかっており、最終段レジスタから出力ピンまでの遅延がワーストパスになる場合、クロック周期からOutput Delay設定値を引いた遅延量が最大動作周波数制約となり、結果に反映されます。



許容される遅延量

= クロック周波数 - 次段デバイスのSetup Time (Output Delay)

最大動作周波数制約100MHz(10ns)で、Output Delay制約3nsをかけている場合、最終段レジスタからピンまでの遅延が $10 - 3 = 7\text{ns}$ 以上発生すると次段デバイスの初段レジスタでデータをラッチできないため、タイミング違反となります。

例えば最大動作周波数周波数制約100MHzでOutput Delay制約3nsの場合、100MHzクロックの周期10nsから次段デバイスのSetup Time 3nsを引いた7nsと最終段レジスタからピンまでの遅延が評価されます。

Setup Time / Hold Timeの確認

■ Setup Time / Hold TimeはDatasheetタブにて確認できます。

① Datasheetタブを選択。

② Input Pad to FFを選択して、Setup Time / Hold Timeの確認したい方のタブを選択します。

Timing Cornerをクリックすると、PVTバラつきがBest/Typical/Worstのそれぞれのケースを選択できます。Setup Time / Hold TimeはBest / Worstのどちらでワースト値になるかケースバイケースなので両方とも確認するようにしてください。

The screenshot shows the 'Timing Corner' settings dialog box. The 'Perform timing analysis based on' section has three radio buttons: 'Best' (selected), 'Typical', and 'Worst'. The 'Reset to Project Setting' button is at the bottom.

Data Port	Clock Port	Delay(ps)	Clock Reference:Phase
1 din1	clk25m	1865	clk25m:R
2 din2	clk25m	1653	clk25m:R
3 en1	clk50m	1971	clk50m:R
4 en2	clk25m	1971	clk25m:R
5 reset	clk50m	5456	clk50m:R
6 reset	clk25m	5456	clk25m:R

タイミング結果

Clock to Output Delay (Tco)の確認

■ Clock to Output Delay (Tco)はDatasheetタブにて確認できます。

① Datasheetタブを選択。

② FF to Output Padを選択して、Min Tco/ Max Tcoの確認したい方のタブを選択します。

Timing Cornerをクリックすると、PVTバラつきがBest/Typical/Worstのそれぞれのケースを選択できます。TcoはBest / Worstのどちらでワースト値になるかケースバイケースなので両方とも確認するようにしてください。

The screenshot shows the software interface for timing analysis. The 'Datasheet' tab is selected, and the 'Timing Corner' button is highlighted. The 'FF to Output Pad' section is active, with 'Min Clock to Out' selected. A table displays the timing results for various data ports. The 'Timing Corner' dialog is open, showing device information (iCE40 LP640) and operating conditions (Commercial, 0V, 25V, 70V). The 'Perform timing analysis based on' section is set to 'Best'.

	Data Port	Clock Port	Delay(ps)	Clock Reference:Phase
1	dout1[0]	clk50m	7354	clk50m:R
2	dout1[1]	clk50m	7369	clk50m:R
3	dout1[2]	clk50m	6777	clk50m:R
4	dout1[3]	clk50m	6656	clk50m:R
5	dout2[0]	clk25m	7050	clk25m:R
6	dout2[1]	clk25m	5959	clk25m:R
7	dout2[2]	clk25m	5959	clk25m:R
8	dout2[3]	clk25m	7050	clk25m:R
9	sample_out	clk50m	6459	clk50m:R

タイミング結果

■ピン間のMax Delay/Min DelayはDatasheetタブにて確認できます。

①Datasheetタブを選択。

②Pad to Padを選択してMin Delay/ Max Delayの確認したい方のタブを選択します。

Timing Cornerをクリックすると、PVTバラつきがBest/Typical/Worstのそれぞれのケースを選択できます。PVTバラつきを考慮して確認を行うためにも、Best/Worstの両方のケースでの確認をお勧めします。

	Port(Input)	Port(Output)	Delay(ps)
1	din1	through_out1	6944
2	din2	through_out2	5974

タイミング結果

■ Multi Cycle Pathの確認はAnalyze Pathsタブで該当パスをサーチして確認します。

① Analyze Pathを選択。

② By Slackを選択。

③ Clock to Clockタブで制約をかけたクロックドメインを選択。

④ 立上り/立下りを選択します。

⑤ Searchをクリックして Path Summaryウィンドウに該当パスを表示します。

Path Summaryウィンドウで該当パスを選択し、詳細表示ボタンをクリックしてPath Detailを表示し、タイミングを確認します。

タイミング結果を確認

Start Point	End Point	Slack	Delay	Skew	Launch Clock	Capture Clock	
1	sample_reg1_IC_...	sample_reg2_L...	37862	1334	0	clk25m:R	clk50m:R

```

Path Detail
Start      sample_reg1_IC_2_3_6/icout
End        sample_reg2_IC_2_1_7/in3
Preference
Setup Constraint 40000(p)
Path Slack   37862(p)
+ Capture Clock Source Latency      0
+ Capture Clock Path Delay          1877
+ Maximum Delay                     40000
- Setup Time                        -440
-----
End-of-path required time (ps)      41436
+ Launch Clock Source Latency        0
+ Launch Clock Path Delay            1877
+ Clock To 0                         364
    
```

Pin Name	Model No.	AT
sample_reg1_IC_2_3_6/icout	LogicCell40...	364 3241
Routing Delay	1...	
sample_reg2_IC_2_1_7/in3	LogicCell40...	0 3575

- Analyze Pathで該当パスをサーチする方法以外に、Clock Relationship Summaryにて制約設定を行ったクロックドメインの項目を探し、Path Detailにて確認することもできます。

①Clock Relationship Summaryを選択。

Save Summary

	Launch Clock	Capture Clock	Constraint	Slack
1	clk25m:R	clk25m:R	40000	33085
2	clk25m:R	clk25m:F	No Path	No Path
3	clk25m:F	clk25m:R	No Path	No Path
4	clk25m:F	clk25m:F	No Path	No Path
5	clk25m:R	clk50m:R	40000	37862
6	clk25m:R	clk50m:F	False Path	False Path
7	clk2	clk25m:R	False Path	False Path
8	clk2	clk25m:F	False Path	False Path
9	clk50m:R	clk25m:R	False Path	False Path
10	clk50m:R	clk25m:F	False Path	False Path
11	clk50m:F	clk25m:R	False Path	False Path

Path Detail

```

Start      sample_reg1_IC_2_3_6/lcout
End        sample_reg2_IC_2_1_7/in3
Reference
Setup Constraint 40000(p)
Path Slack   37862(p)

+ Capture Clock Source Latency           0
+ Capture Clock Path Delay               1877
+ Maximum Delay                           40000
- Setup Time                             -440
-----
End-of-path required time (ps)           41436

+ Launch Clock Source Latency             0
+ Launch Clock Path Delay                1877
+ Clock To Q                             364
+ Data Path Delay                        1334
-----
End-of-path arrival time (ps)            3575
Data path delay consists of logic delay (0 level(s)) 0 ps, and
routing delay 1334 ps.
    
```

タイミング結果を確認

②制約設定を行ったクロックドメインの項目を探し、Path Detailを表示。

■消費電力の見積もりはPower Estimatorで行います。

①Power Estimatorを起動します。

The screenshot shows the Power Estimator window with the following data:

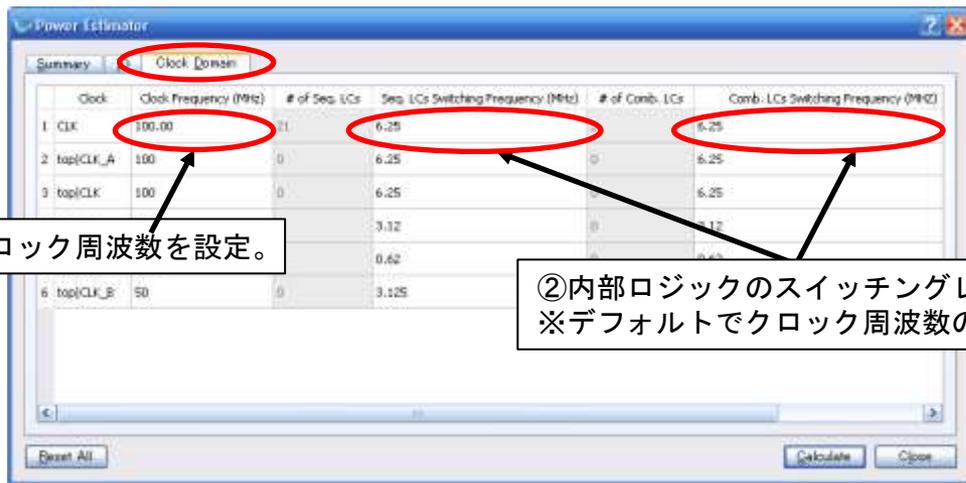
Category	Value
Core Vdd(V)	1.2
IO Voltage	3.3
Left Bank IO Voltage(V)	3.3
Right Bank IO Voltage(V)	3.3
Top Bank IO Voltage(V)	3.3
Bottom Bank IO Voltage(V)	3.3
Dynamic Power Breakdown	
Core Power (mW)	26.4142
IO Power (mW)	22.7404
Power Consumption	
Static Power (mW)	0.161823
Dynamic Power (mW)	49.1546
Total Power (mW)	49.3164
Process	Worst Case
Temperature (°C)	50.00

②Core Voltage, IO Voltageを設定します。

③解析デバイス条件をTypical/Worstから選択します。

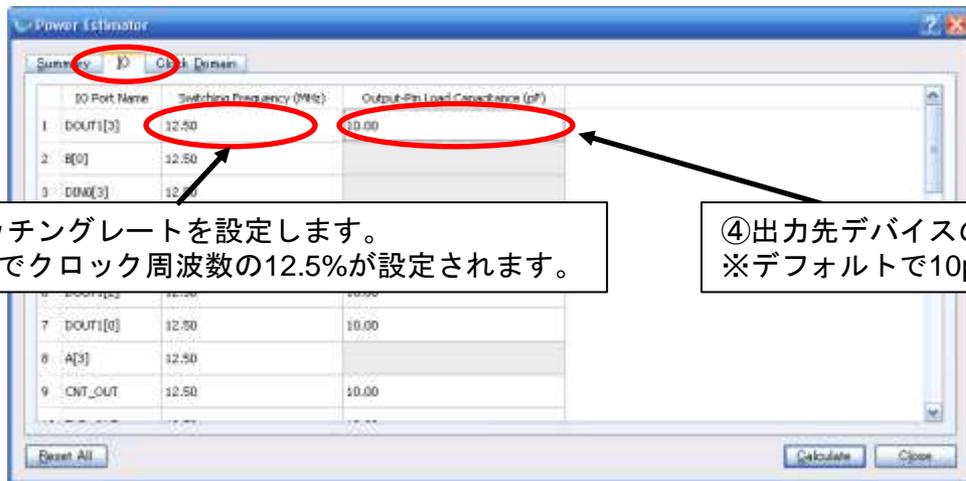
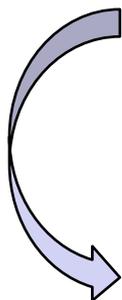
④ジャンクション温度 (T_j) を設定します。

■動作クロック周波数と、内部ロジックセルのスイッチングレート、I/Oスイッチングレートを入力します。



①動作クロック周波数を設定。

②内部ロジックのスイッチングレートを設定します。
※デフォルトでクロック周波数の6.25%が設定されます。



③I/Oのスイッチングレートを設定します。
※デフォルトでクロック周波数の12.5%が設定されます。

④出力先デバイスの負荷容量を設定します。
※デフォルトで10pfが設定されています。

- 設定完了後、Calculateをクリックして消費電力見積もり結果を表示します。

The screenshot shows the SiliconBlue iCEcube2 software interface. The main window displays the project name 'YUV422_to_RGB888' and the output path. A 'Power Estimator' dialog box is open, showing the following data:

Dynamic Power Breakdown	
Core Power (mW):	26.4142
IO Power (mW):	39.6228
Power Consumption	
Static Power (mW):	0.803916
Dynamic Power (mW):	66.037
Total Power (mW):	66.8409

Additional settings in the dialog include Core Vdd (1.2V), IO Voltage (3.3V), and Temperature (50.00°C). The 'Calculate' button is circled in red, and a callout box points to it with the text '①Calculateをクリック。'. Another callout box points to the power results table with the text '②消費電力見積もり結果が表示されます。'.

①Calculateをクリック。

■オンボードにコンフィグレーション用のSPIフラッシュを用意している場合、以下のフローで書き込みを行います。

① Programmerを起動。

② iCE Cableを選択。

③ 使用するROMを選択。
※選択可能なROMは動作確認が取れている以下の5種類です。
M25P20 (ST-Micro製)
M25P16 (ST-Micro製)
M25P32 (ST-Micro製)
M25P80 (ST-Micro製)
PMOD (3rdベンダー製 SPI Flash ROMモジュール)

④ Image Typeを選択します。
■Single Image
⇒ SPI FlashにConfig Dataを1種類のみ書き込みます。
■Multiple Image
⇒ SPI FlashにConfig Dataを複数(最大4つ)書き込みます。
どのデータでコンフィグするかはツールでスタティックに
選択する方法と、専用ピン(CBSEL[1:0])からダイナミック
に選択する方法があります。

⑤ Image File Settingsから書き込みデータ
に関する設定を入力します。
詳細は次ページ以降で説明します。

■ Image TypeでSingle Imageを選択している場合の操作は以下の通りです。

The image shows two software windows. The 'Programmer' window on the left has 'iCE Cable' selected under 'Programming Hardware', 'External SPI Serial Flash PROM' selected under 'Programming Target', and 'Single Image' selected under 'Image Type'. The 'Image File Settings' button is circled in red. The 'Execute' button is also circled in red. The 'ColdBoot/WarmBoot Setup' window on the right shows 'Raw Hexadecimal' selected under 'Format' and 'blink40_demo_bitmap.hex' entered in the 'Config. File' field. The 'OK' button is circled in red. Blue arrows indicate the flow from the 'Execute' button in the Programmer window to the 'ColdBoot/WarmBoot Setup' dialog, and from the 'OK' button back to the Programmer window.

①プロジェクト内で既にConfig Fileが生成されている場合はデフォルトでそのデータが設定されています。別のデータを選択する場合はブラウザボタンからファイル(.hex)を選択しなおします。

②ファイルフォーマットはRaw HexadecimalのままでOKです。

③OKをクリックし、設定を完了。

④Executeをクリックすると書き込みが実行されます。

■ Image TypeでMultiple Imageを選択している場合の操作は以下の通りです。

The image shows two software windows: 'Programmer' on the left and 'ColdBoot/WarmBoot Setup' on the right. The 'Programmer' window has 'iCE Cable' selected under 'Programming Hardware' and 'Multiple Images' selected under 'Image Type'. The 'Execute' button is circled in red. The 'ColdBoot/WarmBoot Setup' window shows 'Enable Cold Boot' checked, and four 'Image' sections (Image 0 to Image 3) are highlighted with a red box. Each image section has fields for 'Start Address' and 'Config. File'. The 'Format' dropdown is set to 'Raw Hexadecimal'. The 'OK' button at the bottom is also circled in red.

① Enable Cold Bootにチェックを入れると、専用ピンからダイナミックにコンフィグデータを選択する設定になります。チェックを外した場合は下のボックスで0-3番目のどのデータでコンフィグするかを選択します。

② Config Dataを4つまで選択します(データは予め用意しておく必要があります)。ブラウザボタンからデータを選択して下さい。各データを書き込むStart Addressの設定はデフォルトのまま構いません。
※SPI Flashの容量を超える分のデータ数をセットした場合、そのデータでのコンフィグはできません。

③ OKをクリックし、設定を完了。

④ Executeをクリックすると書き込みが実行されます。

Section 2 : 専用モジュールの作成

- RAMモジュール等の生成はSynplify PROにて行います。論理合成時の手順でSynplify PROを起動し、SYNCoreを立ち上げて、作成したいモジュールを選択し、パラメータを設定して生成します。

① SYNCoreを起動。

② 生成したいRAMモジュールを選択。

③ 各種パラメータを設定。

④ Generateをクリック。

- PLLの生成は PLL Module Generatorにて行います。

① Tool > Configure PLL Moduleから PLL Module Generatorを起動。

② 新規作成の場合はCreate a new PLL configurationを選択し、PLLモジュール名を入力します。既存のPLLモジュールの設定変更を行う場合はModify an existing PLL configurationを選択し、PLLソースを選択します。

③ OKをクリックすると次ページ以降に記載のPLL設定画面が開きます。

PLL Type

Select the number of global networks to be driven by the PLL output: 1

How will the PLL Source Clock be driven ?

- General Purpose IO Pad or Core Logic
- Dedicated Clock Pad (Single Ended)
 - The PLL source clock will be used on chip without frequency/phase/delay adjustments
- Dedicated Clock Pads (TMDS)

PLL Operation Mode

■General Purpose IO Pad or Core Logic
 ⇒ PLL入力クロックが汎用I/Oからの入力クロック、または内部ロジックで生成したゲートクロックの場合に選択します。

■Dedicated Clock Pad (Single Ended)
 ⇒ PLL入力クロックがクロック入力専用ピンからの入力クロックの場合に選択します。こちらを選択した場合、入力クロックを分岐させてFPGA内部で使用することはできません。そのため、入力クロックをFPGA内で使用したい場合はGlobal Clock Networkを2本ドライブする設定にし、下のチェックボックスにチェックを入れることで、Port Aより入力クロックがそのまま出力されるので、このクロックを使用します。

■Dedicated Clock Pad (TMDS)
 ⇒ PLL入力クロックがクロック入力専用ピンから作動信号で入力される場合に選択します。

No

Fine delay adjustment setting (Enter a value in the range 0 - 15):

< Back Next > Finish Cancel

PLL出力を何ポートGlobal Clock Networkに接続するか選択します。(1 or 2)

■General Purpose IO Pad or Core Logic
 ⇒ PLL入力クロックが汎用I/Oからの入力クロック、または内部ロジックで生成したゲートクロックの場合に選択します。

■Dedicated Clock Pad (Single Ended)
 ⇒ PLL入力クロックがクロック入力専用ピンからの入力クロックの場合に選択します。こちらを選択した場合、入力クロックを分岐させてFPGA内部で使用することはできません。そのため、入力クロックをFPGA内で使用したい場合はGlobal Clock Networkを2本ドライブする設定にし、下のチェックボックスにチェックを入れることで、Port Aより入力クロックがそのまま出力されるので、このクロックを使用します。

■Dedicated Clock Pad (TMDS)
 ⇒ PLL入力クロックがクロック入力専用ピンから作動信号で入力される場合に選択します。

PLL出力クロックの生成方法についての設定です。

- Using a feedback path internal to the PLL (PLL内部フィードバックパス使用)
 - No Compensation
⇒ 周波数設定のみ有効。
 - Delay Compensation using only the Fine Delay Adjustment Block
⇒ 周波数設定と遅延微調整が有効。
 - Delay Compensation using the Phase Shifter and the Fine Delay Adjustment Block
⇒ 周波数設定、遅延微調整、位相シフト機能が有効
- Using a feedback path external to the PLL (PLLの外からのフィードバックを使用)

PLL Module Generator

PLL Type

Select the number of global PLLs

How will the PLL Source Clock be generated?

General Purpose Internal PLL

Dedicated Clock Pads (CMOS)

The PLL source clock is provided by the user

Dedicated Clock Pads (TIMDS)

PLL Operation Modes

How will the PLL output be generated ?

Using a feedback path internal to the PLL

- No Compensation mode
- Delay Compensation using only the Fine Delay Adjustment Block
- Delay Compensation using the Phase Shifter and the Fine Delay Adjustment Block (Recommended mode for applications like LVDS Display Panel and DDR)

Using a feedback path external to the PLL

The external feedback path will include a divider implemented by the user in logic, with default divide-by factor of 1

Set divide-by factor to

Fine Delay Adjustment Settings

Do you want to dynamically control the delay of the Fine Delay Adjustment Block ?

Yes

No

Fine delay adjustment setting (Enter a value in the range 0 - 15):

< Back Next > Finish Cancel

PLL Module Generator

PLL Type

Select the number of global networks to be driven by the PLL outputs:

How will the PLL Source Clock be driven ?

General Purpose IO Pad or Core Logic

Dedicated Clock Pad (Single Ended)

The PLL source clock will be used on chip without frequency/phase/delay adjustments

Dedicated Clock Pads (TMDS)

PLL Operation Modes

How will the PLL output

Using a feedback

No Compens

Delay Comp

Delay Comp (Recommend)

Using a feedback

The external feedback path will include a divider implemented by the user in logic, with default divide-by factor of 1

Set divide-by factor to

Fine Delay Adjustment Settings

Do you want to dynamically control the delay of the Fine Delay Adjustment Block ?

Yes

No

Fine delay adjustment setting (Enter a value in the range 0 - 15):

遅延微調整機能のコントロール方法をダイナミックかスタティックか選択します。遅延微調整機能が有効なPLLモードの場合にのみ設定できます。

■Yes
⇒ デバイス動作中にダイナミックに遅延調整を行います。遅延調整用のポートがPLLに追加されます。

■No
⇒ 本設定画面で設定した遅延設定値で固定します。設定値は0~15で選択でき、 $(n+1) \times 165\text{ps(TYP)}$ で遅延します。



PLL Module Generator

Phase Shift Specification

Specify the phase shift for the PLL output: 90deg

Additional Delay Settings

PLL位相シフト量の設定です。

- 1ポートGlobal Clock Networkに接続されるPLLタイプ
- 2ポートGlobal Clock Networkに接続される設定で、入力クロックを遅延調整、位相シフト無効で出力するチェックボックス（P32参照）にチェックが入っているタイプ

に対して、0deg or 90deg から選びます。位相シフトはPort Bに反映され、Port Aに対して0degまたは90degシフトします。

No

Target Application

LVDS Display Panel

The frequency on Port B of the PLL can be 7x or 3.5x the frequency on Port A.

Specify the frequency ratio of the clock on Port B w.r.t. to the clock generated on Port A:

DDR Application

上記以外のPLLタイプの場合、位相シフト設定がターゲットアプリケーション設定に変化します。

- LVDS Display Panel
 - ⇒ x7、x3.5通倍モードが使用できるようになります。Port Bの周波数をPort Aの周波数のx7 または x3.5のいずれにするか選択します。
- DDR Application
 - ⇒ Port Bの位相をPort Aに対して0degまたは90degシフトのいずれにするか 選択します。

PLL Module Generator

Phase Shift Specification

Specify the phase shift for the PLL output: 90deg

Additional Delay Settings

Do you wish to specify additional delay on the PLL outputs ?

Yes

Do you want to dynamically control the delay of this Additional Delay Adjustment Block ?

Yes

No

Fine Delay Adjustment Block setting (Enter a value in the range 0 - 15):

No

< Back Next > Finish Cancel

Port Aに対する追加遅延微調整の設定をします。P34に記載の遅延調整に追加でPort A
 に対してのみ、さらに $(n+1) \times 165\text{ps(TYP)}$ の遅延を加えます。Port A設定が不可のPLL
 タイプ（入力クロックを遅延調整、位相シフト無効で出力するタイプ）に関してはこの
 設定はできません。

PLL Module Generator

PLL Input/Output Frequency

Input frequency (Mhz): 50

Output frequency on port A (Mhz): 100

Others

Create a LOCK output port

Create a BYPASS port that will bypass the PLL reference clock to the PLL output port
(Note that the PLL requires re-locking when the BYPASS signal is de-asserted, for all modes other than the "No Compensation mode")

Low Power Mode

Enable latching of PLL output clock (iCEGate) on port A
(Note that the PLL requires re-locking after the latch signal is de-asserted, when the feedback path is external to the PLL)

Enable latching of PLL output clock (iCEGate) on port B

< Back Next > Finish Cancel

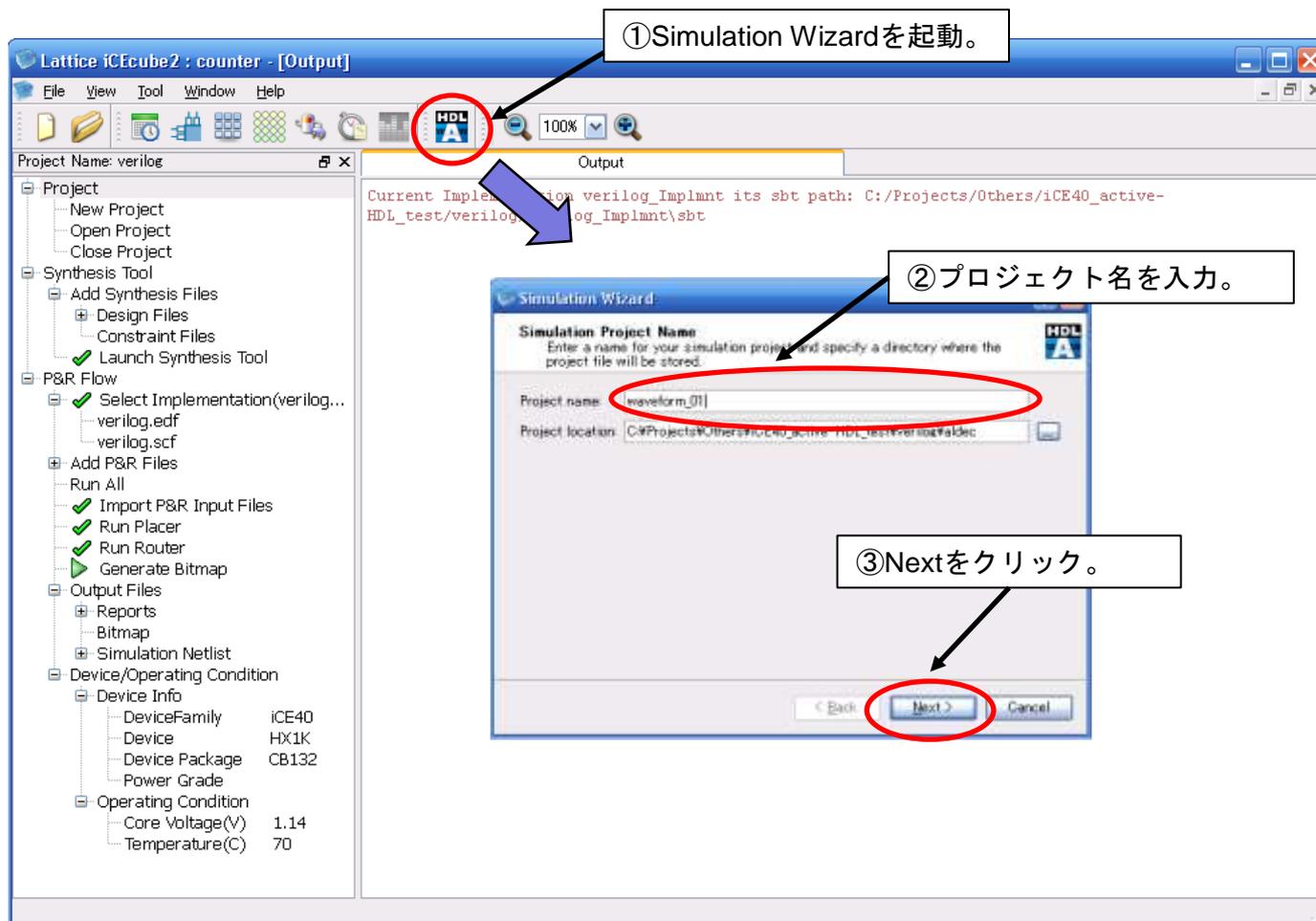
■PLL Input/Output Frequency
⇒ PLL入力クロック周波数と出力クロック周波数を設定します。

■Others

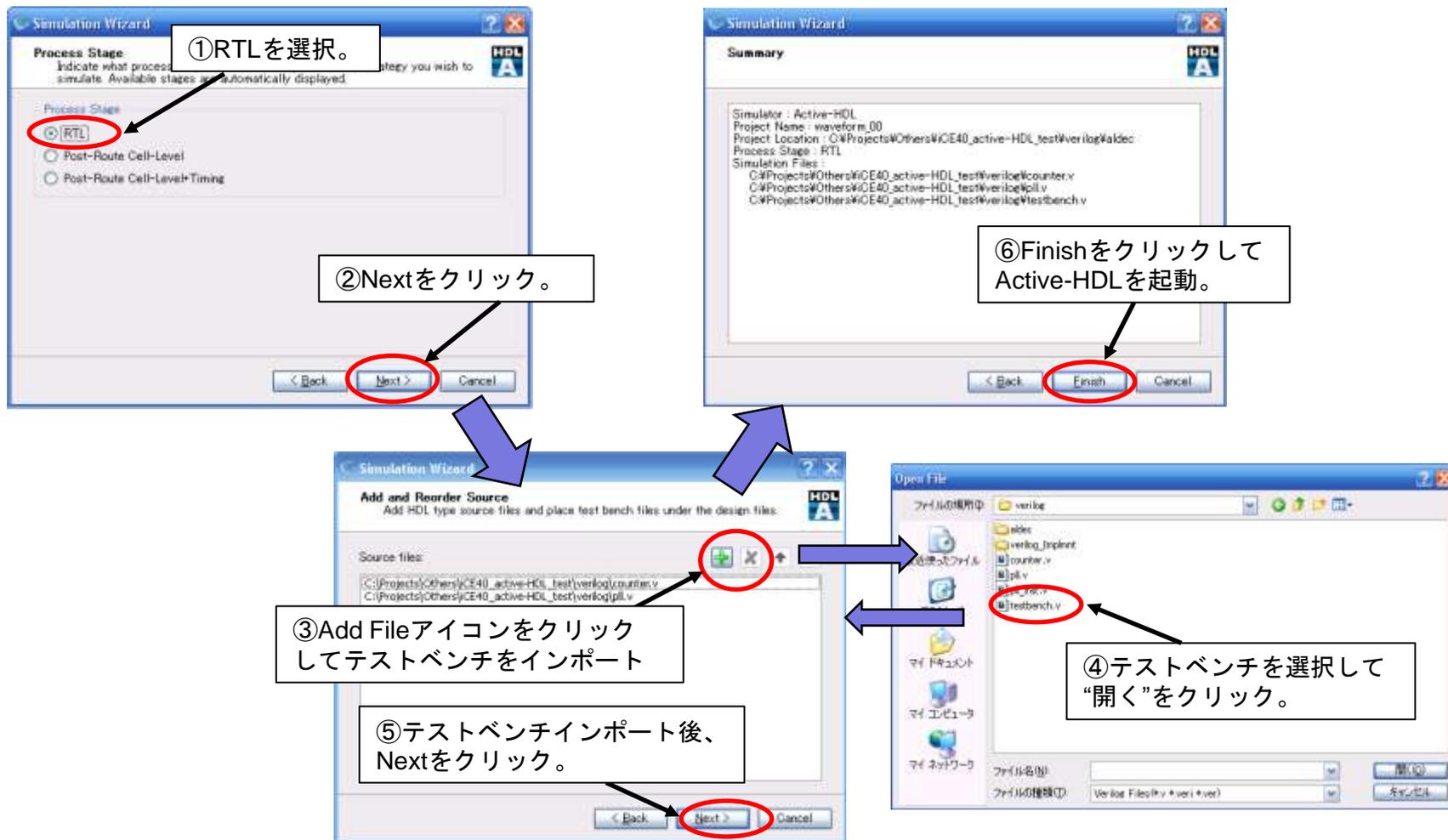
- Create a LOCK output port
⇒PLLのロックステータス出力ポートを追加します。
- Create a BYPASS port that will bypass the PLL reference clock to the PLL output port
⇒ PLLをパワーダウンモードにし、入力クロックをそのまま出力クロックに出力するBypass制御ポートを追加します。
- Low Power Mode
⇒ iCEGateの制御ポートの有無を設定します。iCEGate有効時、出力クロックは前値保持し、トグルを抑えることで電力消費を抑えます。

Section 3 : 論理シミュレーション

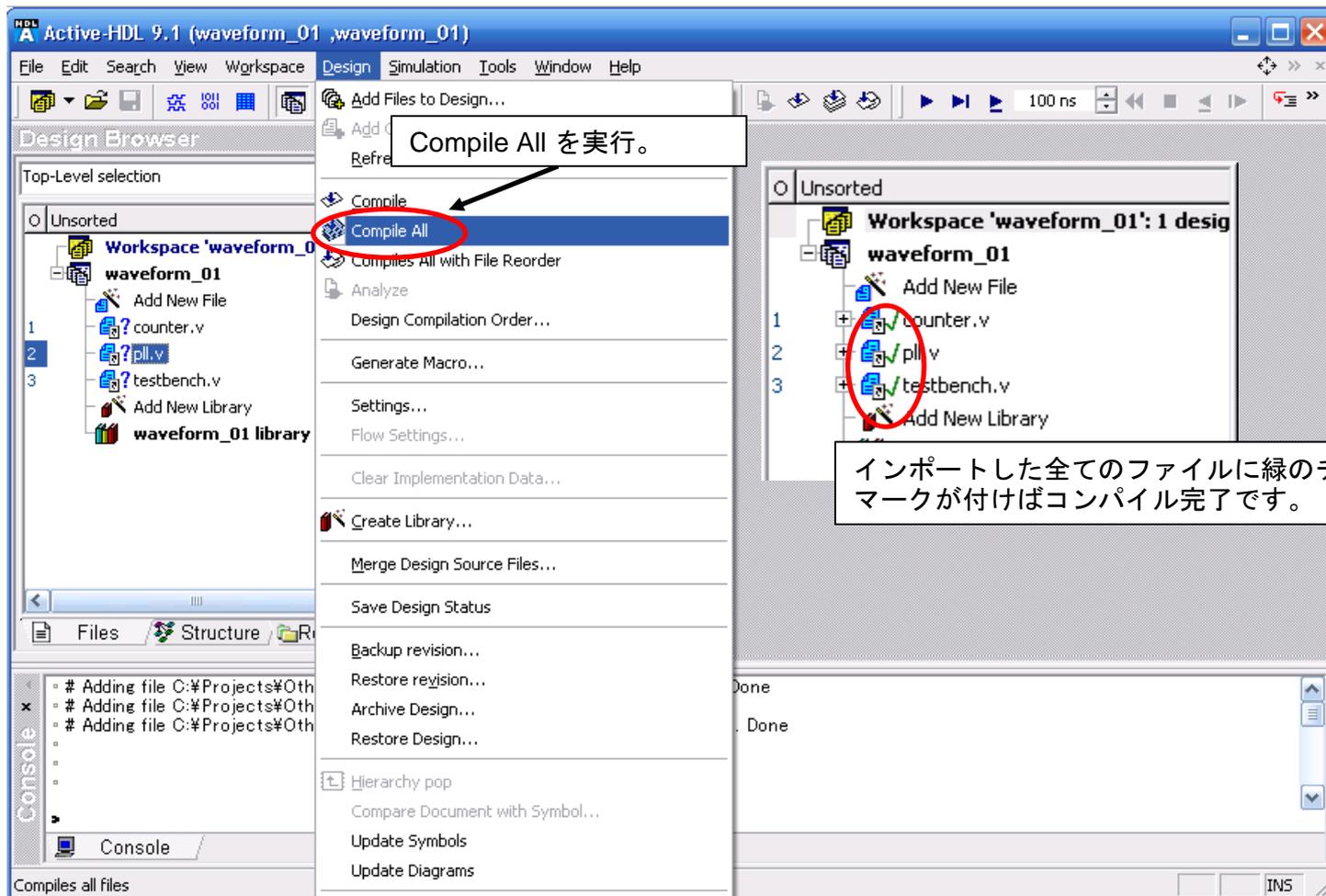
- Lattice社ではシミュレーションツールとしてActive-HDLがバンドルされています。
iCEcube2上からSimulation Wizardを使用してActive-HDLプロジェクトを生成します。



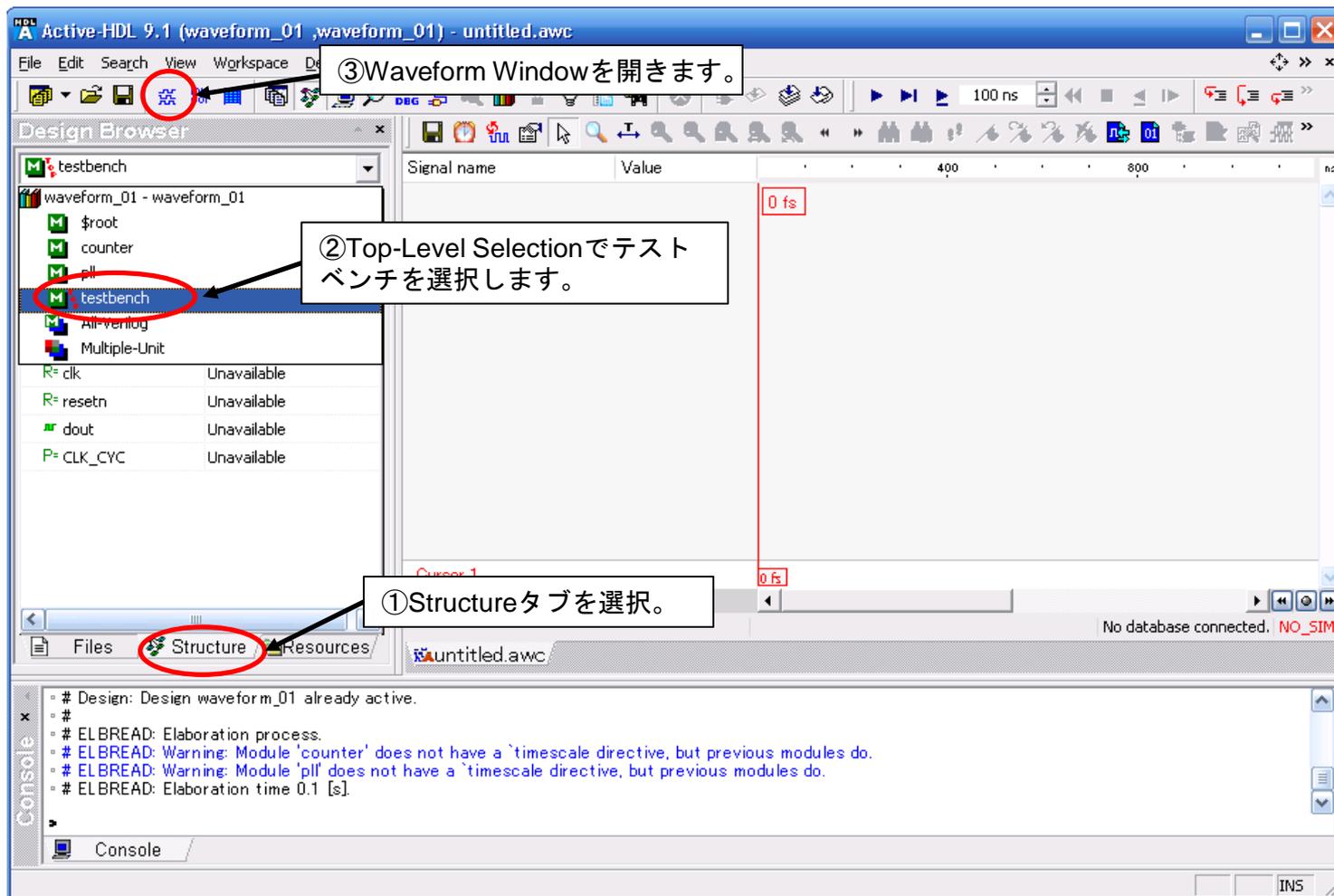
■ 以下の手順でActive-HDLプロジェクトを作成します。



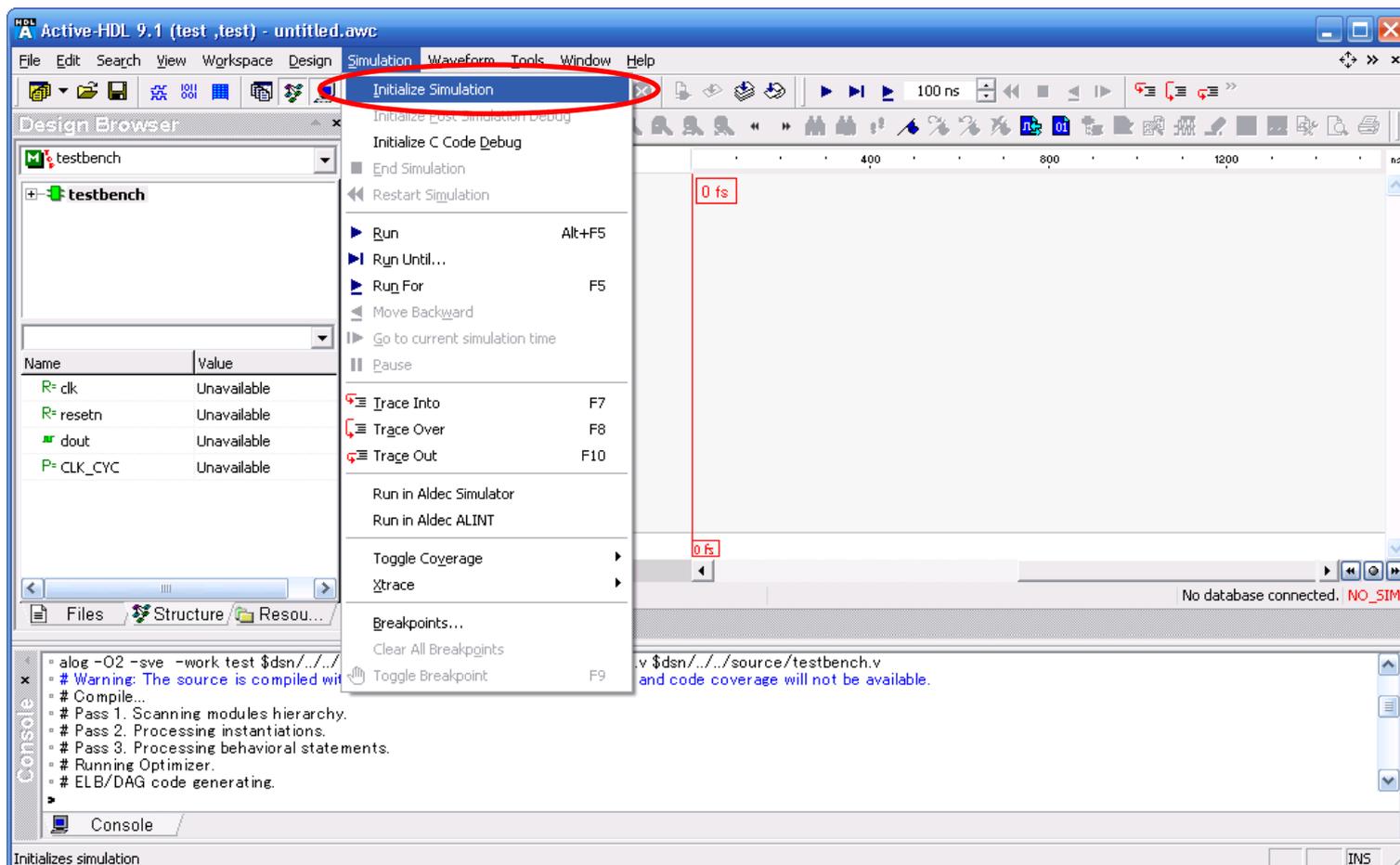
- Active-HDL起動後、Design > Compile Allでコンパイルを実行します。



- シミュレーショントップモジュールを選択し、Waveform Windowを開きます。



- Simulation > Initialize Simulation を実行します。



- 観測する信号をWaveform Windowにドラッグ&ドロップし、シミュレーションを実行します。

① 観測する信号が含まれるモジュールをハイライトします。

② 観測する信号をWaveform Windowにドラッグ&ドロップします。

③ Runアイコンでシミュレーションを開始し、Stopアイコンで停止します。

Signal name	Value
- clk	x
- resetn	x
- count	x
- pll_clk	x

```

# Selected Top-Level testbench (testbench)
# KERNEL: Signal '/testbench/UUT/clk' has already been traced
# KERNEL: Signal '/testbench/UUT/resetn' has already been traced
# KERNEL: Signal '/testbench/UUT/dout' has already been traced
# KERNEL: Signal '/testbench/UUT/count' has already been traced
# KERNEL: Signal '/testbench/UUT/pll_clk' has already been traced
    
```

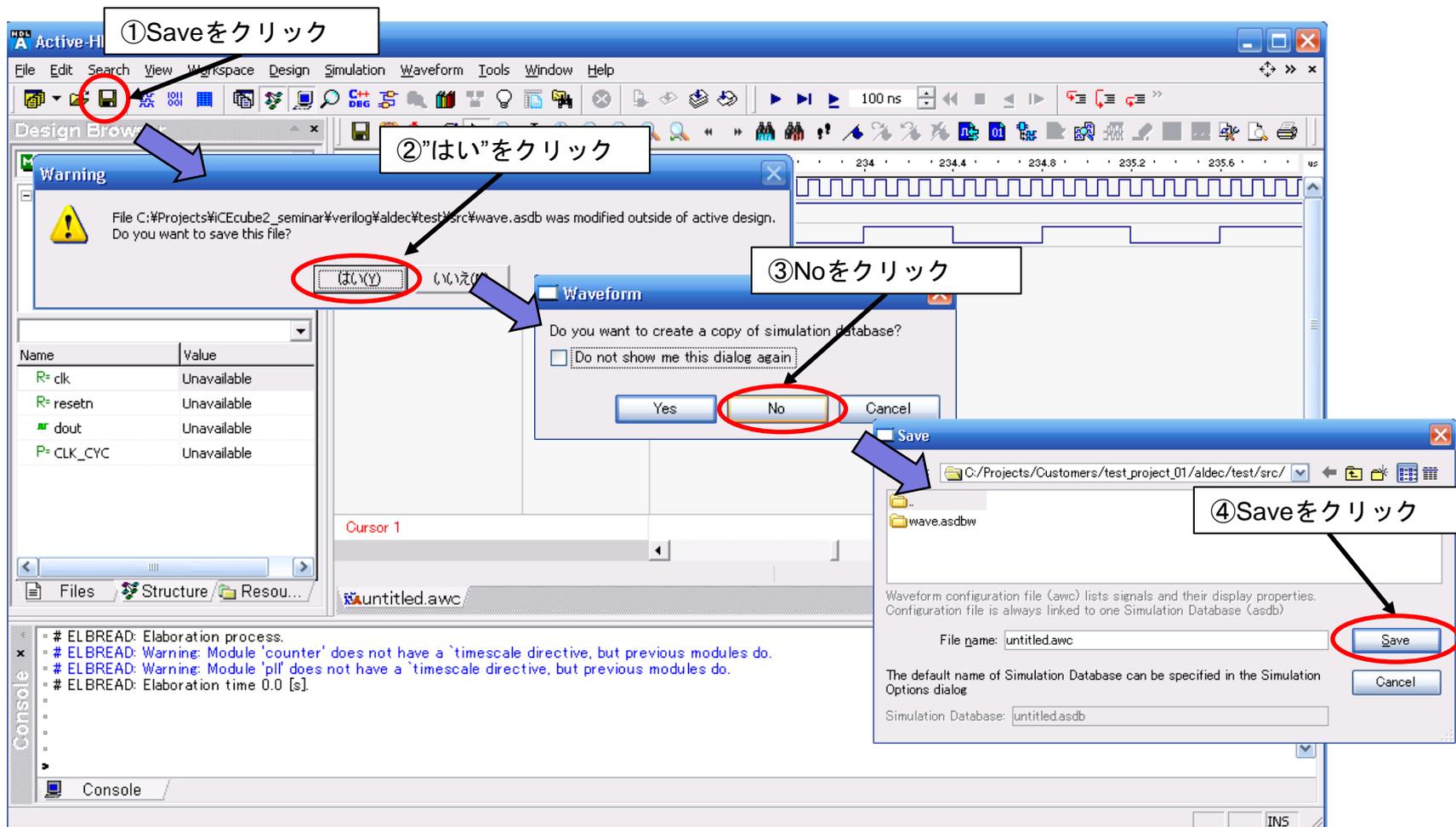
任意拡大ツールで拡大したい場所を選択して拡大できます。

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。

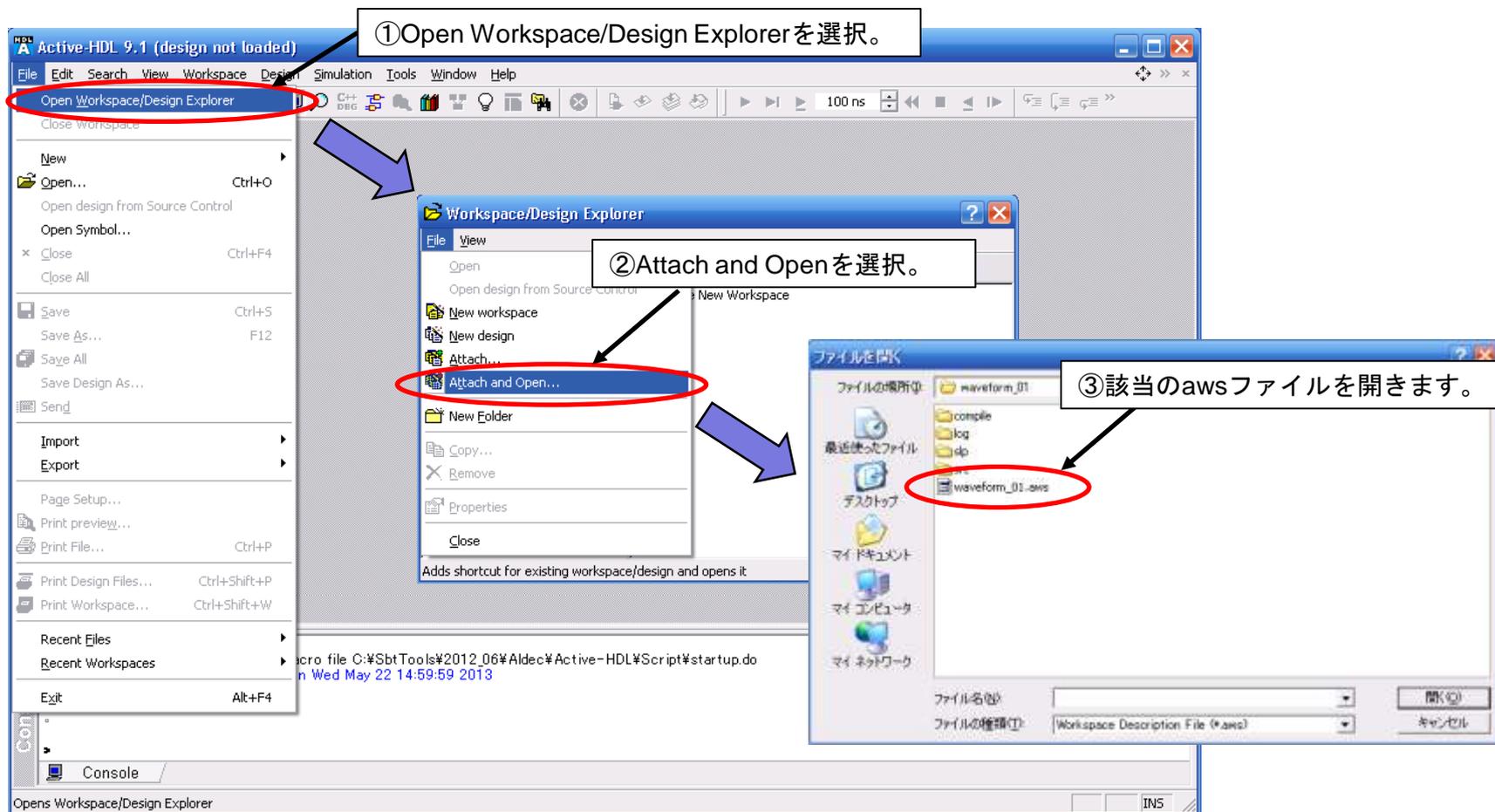
右クリックから"Add Cursor"でカーソルを追加し、カーソル間の時間を確認することができます。

Name	Value
R= clk	0
R= resetn	1
R= en	1
dout	9 to A
dout(3)	E
dout(2)	F
dout(1)	0
dout(0)	1

- シミュレーション完了後、プロジェクトを保存して終了します。



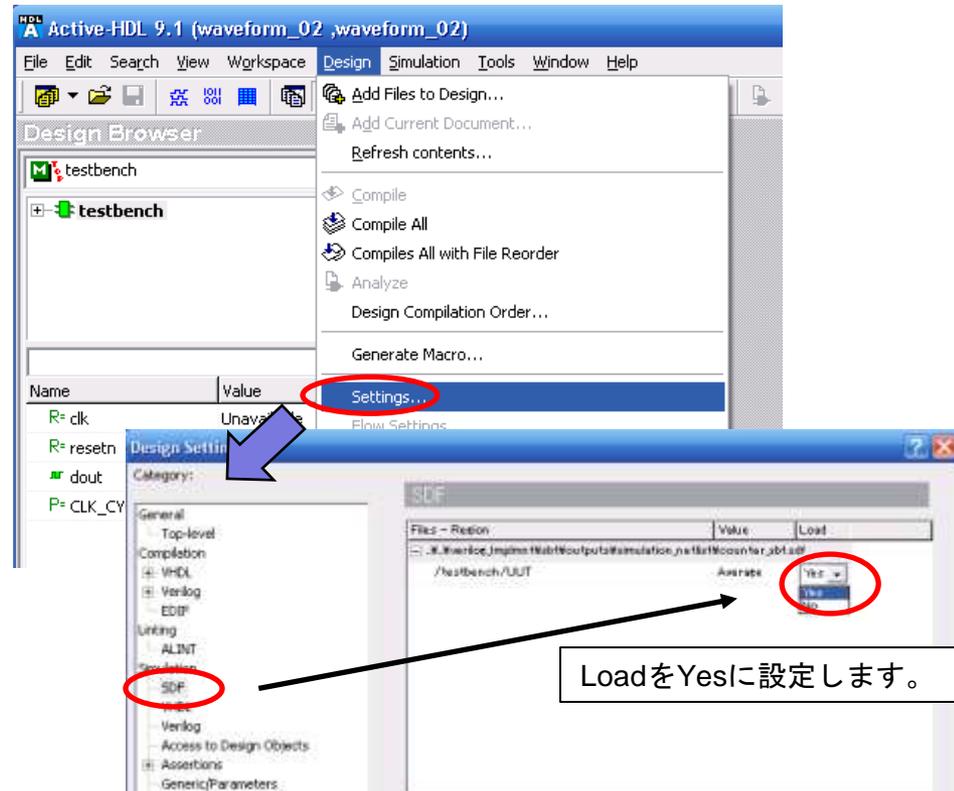
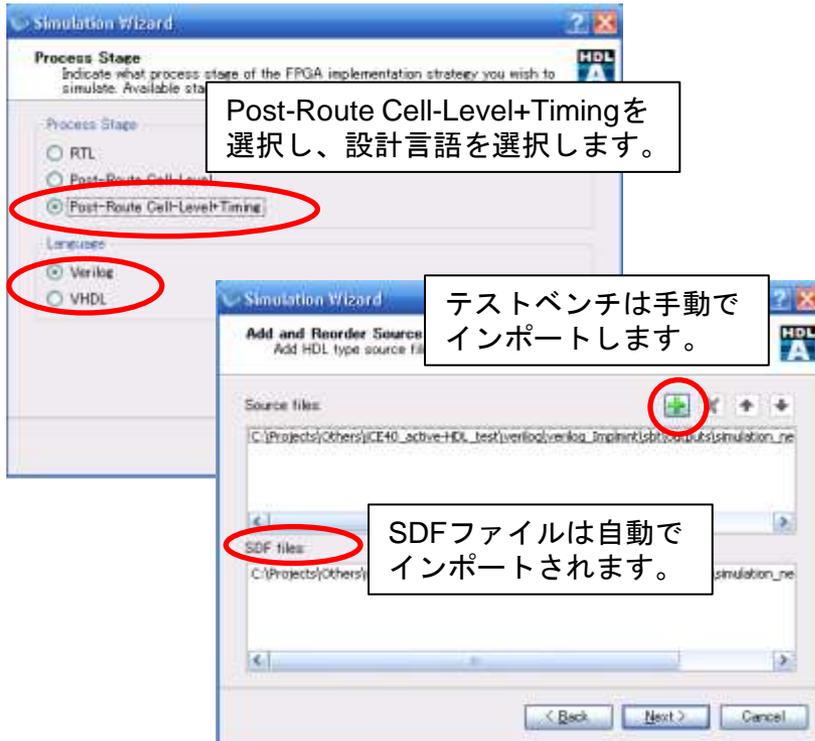
- 次回以降再度プロジェクトを開く場合は、スタートメニューより、プログラム > Lattice iCEcube2 201x_xx > avhdl を起動し、File > Open Workspace/Design Explorer からプロジェクトを開きます。



- Timing Simulationも基本操作はFunction Simulationの場合と同じですが、Simulation Wizardにて“Post-Route Cell-Level+Timing”を選択する箇所と、Active-HDL上で遅延情報の有効化を行う箇所が異なります。

P46に記載のフロー中で“Post-Route Cell-Level+Timing”を選択します。ソースファイルのインポート画面にて自動で遅延情報ファイル（SDF）がインポートされます。
なお、ここでもテストベンチは手動でインポートして下さい。

Active-HDL起動後は、All Compileを実行し、Top-Level Selectionでテストベンチ選択後、Design > Setting > SDF と進み、遅延情報の有効化を行います。



※Model SimはLattice社から無償提供されておりません。予め別途Mentor Graphics社より購入して下さい。また、PLLのライブラリはVerilog用のみ用意されておりますので、VHDLでシミュレーションを行う場合はMixed Signal対応バージョンのModel Simをご用意下さい。

- シミュレーション開始前に、Synplify PROでFunction Simulation Model (vmファイル、vhmファイル)を生成する必要があります。Synplify PROを起動し、以下の手順でを生成します。
vmファイルはVerilogソースでのシミュレーション、vhmファイルはVHDLソースでのシミュレーションに使用します。

※本マニュアルではVerilogソースでのシミュレーションについて説明を行います。

①Synplify PROを起動。

②Implementation Optionsを起動。

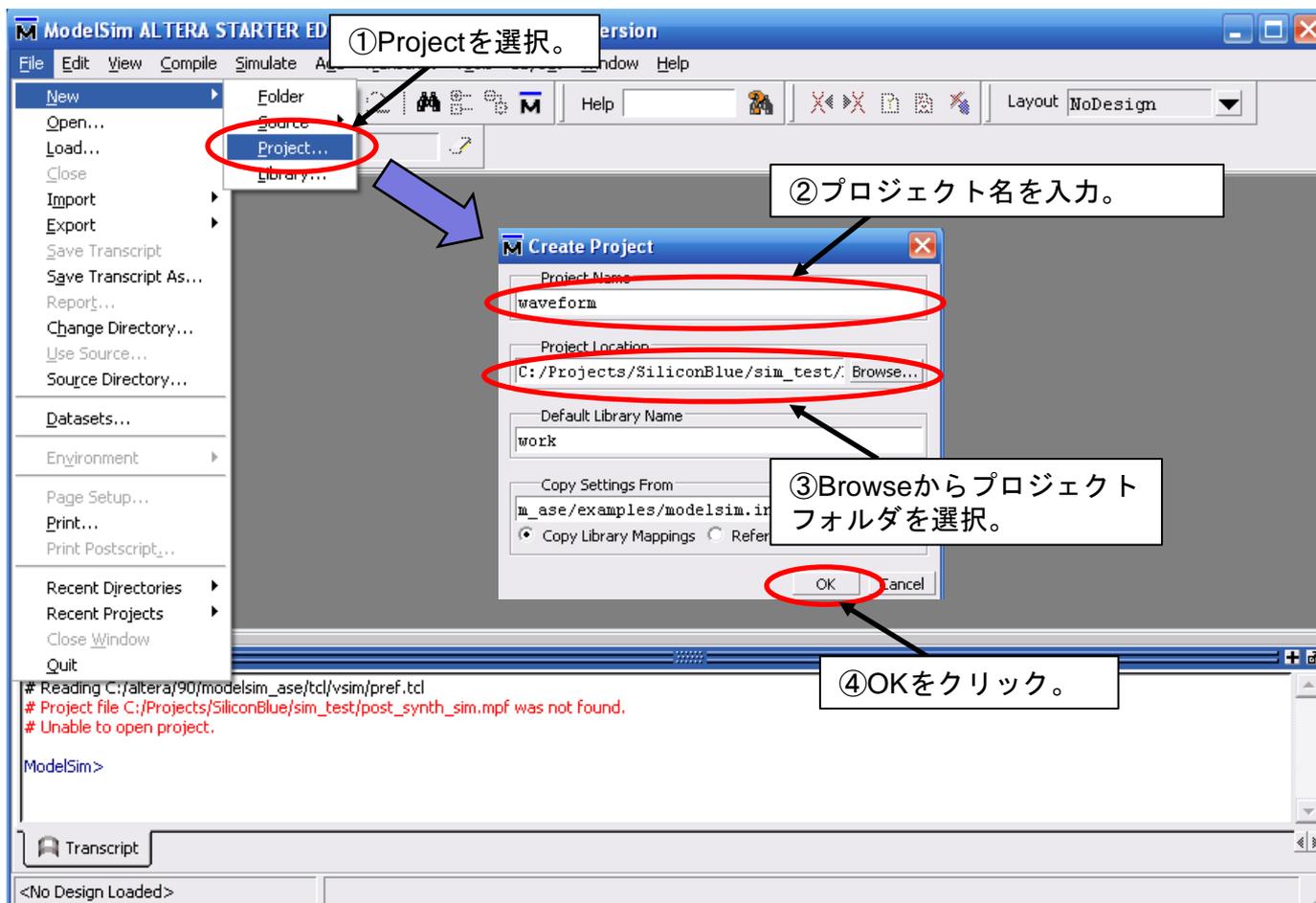
③Implementation Result タブを選択

④"Write Mapped Verilog Netlist"と "Write Mapped VHDL Netlist"に チェックを入れる。

⑤OKをクリックし、 Synplify PRO トップ画面 でRUNを実行する。

vmファイル、vhmファイルはプロジェクトフォルダ内の <プロジェクト名>_Inplmnt/ に生成されます。

- Model Simを起動し、File > New > Project から新規プロジェクトを作成します。



■ テストベンチ、iCEライブラリ、シミュレーション用モジュール(vm、vhmファイル)をインポートします。

① Add Existing Fileをクリック。

② Browseをクリックし、ファイル選択ウィンドウを開きます。

③ テストベンチ、iCEライブラリ、vmファイルをインポートします。

④ インポート完了後、Closeをクリック。

※iCEライブラリは以下のフォルダに用意されています。
Verilog用 : <iCEcube2インストールディレクトリ>/Verilog
VHDL用 : <iCEcube2インストールディレクトリ>/VHDL
※vm、vhmファイルの生成方法、生成場所に関してはP53を参照して下さい。
※PLLを使用しているデザインの場合、追加でPLLライブラリ "ABIPTBS8.v", "ABIWTCZ4.v"をインポートする必要があります。
<iCEcube2インストールディレクトリ>/Verilog
に用意されています。

※All Filesを選択しないとvm、vhmファイルが表示されないのをご注意下さい。

- ファイルインポート後、Compile > Compile All を実行します。

The screenshot shows the ModelSim interface. The 'Compile' menu is open, and 'Compile All' is highlighted with a red circle. A callout box points to this menu item with the text 'Compile All を実行。' (Execute Compile All).

Another callout box points to the 'Status' column of the workspace table, containing the text '全てのファイルに緑のチェックマークが付けばコンパイル完了です。' (When all files have a green checkmark, compilation is complete).

The workspace table shows the following files and their compilation status:

Name	Status	Type	Ord
counter_tb.v	✓	Verilog	0
sb_ice_syn.v	✓	Verilog	1
sim_test.v	✓	Verilog	2

The 'Status' column for all three files contains a green checkmark, which is circled in red. The transcript window at the bottom shows the following output:

```

ModelSim>
# Reading C:/altera/90/modelsim_ase/tcl/vsim/pref.tcl
# Project file C:/Projects/SiliconBlue/sim_test/post_synth_sim.mpf was not found.
# Unable to open project.
# reading C:\altera\90\modelsim_ase\win32aloem\..\modelsim.ini
# Loading project waveform
ModelSim>
    
```

- Simulate > Start Simulationを実行し、テストベンチを選択します。

① Start Simulationを実行。

Name	Type	Path
work	Library	C:\%Projects%\Customers\%Panasonic%\PF5
counter	Module	C:/Projects/SiliconBlue/sim_test/sim_sou
VCC	Module	C:/Projects/SiliconBlue/sim_test/sim_sou
Delay4Buf	Module	C:/Projects/SiliconBlue/sim_test/sim_sou
FineDlyAdj	Module	C:/Projects/SiliconBlue/sim_test/sim_sou
GND	Module	C:/Projects/SiliconBlue/sim_test/sim_sou
mask_decoder	Module	C:/Projects/SiliconBlue/sim_test/sim_sou
mux4to1	Module	C:/Projects/SiliconBlue/sim_test/sim_sou
counter_tb	Module	C:/Projects/SiliconBlue/sim_test/sim_sou

② テストベンチを選択。

```

Transcript
# reading C:\altera\90\modelsim_ase\win32\altem\..\modelsim.ini
# Loading project waveform
# Compile of counter_tb.v was successful.
# Compile of sb_ice_syn.v was successful.
# Compile of sim_test.vm was successful.
# 3 compiles, 0 failed with no errors.
# Load canceled
ModelSim>
    
```

■ 観測する信号をWaveウィンドウにドラッグ&ドロップし、シミュレーションを実行します。

① 観測する信号が含まれるモジュールをハイライトします。

② Objectsウィンドウから観測する信号をWave Windowにドラッグ&ドロップします。

③ Run、Run-All、Breakを使用してシミュレーションを実行します。

Transcript

```
# .main_pane.mdi.interior.cs.vm.paneset.cli_0.wf.clip.cs.pw.wf
VSIM 3> run -all
# ** Note: $finish : C:/Projects/SiliconBlue/sim_test/sim_source/counter_tb.v(42)
# Time: 7 us Iteration: 0 Instance: /counter_tb
# 1
# Break in Module counter_tb at C:/Projects/SiliconBlue/sim_test/sim_source/counter_tb.v line 42
```

Project : waveform Now: 7 us Delta: 0 sim:/counter_tb

※各種ウィンドウが表示されていない場合は、Viewから開くことができます。

Date	Revision	Old-page	New-Page	Change information	Updated by
2012/5/10	1.0			First Revision	W.Nakatsuka
2012/6/14	2.0	P29	P29 - P31	Config Data書き込み設定方法を追加・修正	W.Nakatsuka
2012/6/18	2.1	P35	P35	PLL生成画面にてClock Pad選択時の説明を修正	W.Nakatsuka
2012/6/25	2.2	P46, P52	P46, P52	PLLシミュレーションライブラリの記載を修正 (ABIWTCZ4.vを追加)	W.Nakatsuka
2012/7/6	2.3	P28	P28 - P30	消費電力見積もりについての説明を追加。	W.Nakatsuka
2012/8/23	3.0	P36-P39	P36-P49	PLL Generatorの改版により、記載を修正。	W.Nakatsuka
2012/8/23	3.0		P44-P52	iCEcube2のActive-HDLバンドル化につきSimulationフロー全般を修正。	W.Nakatsuka
2012/12/27	3.1	P48	P48	Initialize Simulationによるシミュレーション開始に修正	W.Nakatsuka
2012/12/27	3.1		P51	Active-HDLプロジェクトの保存方法を追加	W.Nakatsuka
2013/4/20	4.0	P13-P16	P15-P18	タイミング制約設定の方法を修正	W.Nakatsuka
2013/4/20	4.0	P28-P30	P30-32	消費電力見積もりについてProcess条件とTj設定について追加	W.Nakatsuka
2013/9/9	5.0	P9-P10	P9-P10	SynpliFy Pro/バックグラウンド実行化によりマニュアル修正	W.Nakatsuka
2013/9/9	5.0		P19	Timing Constraints Editorの再起動方法を追加	W.Nakatsuka