

TestBencher Pro



タイミングダイアグラム入力による、テストベンチ生成ツール
対応言語：VHDL、Verilog、OpenVera、e、
SystemC、及び、C++ (TestBuilder)



InterLink



予定

- [] SynaptiCAD社製品群の中での、TestBencher Proの位置づけ
- [] タイミングダイアグラム記述方法
- [] Setup、Holdパラメータによるタイミング解析例波形の例
- [] 波形記述例
- [] MS-Wordへの波形貼り付け
- [] TestBencher Pro 製品概要
- [] コード生成アーキテクチャ
- [] TestBencherデザインフロー
- [] 操作デモ - SRAMテスト -
- [] e テンポラル記述 - 出力例
- [] TestBencherデザインフロー : e
- [] TestBencherデザインフロー : OpenVera
- [] TestBencherデザインフロー : OpenVera、SystemC
- [] TestBencherデザインフロー : VHDL、Verilog、TestBuilder



InterLink



Synapt iCAD社製品群の中での、 TestBencher Proの位置づけ

TestBencher Proは
製品群の最上位製品
→ (OLE機能を除く)
下位製品の全ての
機能を含む

TestBencher Pro						
VeriLogger Pro						
DataSheet Pro						
WaveFormer Pro						
TimingDiagrammer Pro						
GigaWave Viewer						
Feature Sets						
Test Bench Generation	-	-	-	-	-	All
Simulation Features	-	-	Limited	Limited	All	All
Waveform Stimulus	-	-	Limited	Limited	All	All
Waveform Translation	-	-	All	All	All	All
Test Equipment	-	-	All	All	All	All
Timing Analysis	-	All	All	All	All	All
Documentation Features	-	Limited	Limited	All	Limited	Limited
Optional Features						
OLE	-	Option	Option	Included	Option	Option
GigaWave	Included	Option	Option	Option	Option	Option
Waveform Comparison	-	Option	Option	Option	Option	Option
G Series	-	Option	Option	Option	Option	Option
MultiWindow	Option	Option	Option	Included	Option	Included



InterLink



Synapt iCAD社製品群の中での、 TestBench Proの位置づけ（続き）

TestBench Pro :

- グラフィカルな入力によるテストベンチ生成ツールです。
- VeriLogger Proの機能を有します。
- 言語に依存しないタイミングダイアグラムから、リアクティブ（自身で応答可能）なテストベンチやバスファンクショナルモデルを生成します。
- サポートしている言語 : VHDL、Verilog、OpenVera、e、SystemC、およびC++（TestBuilder）

VeriLogger Pro :

- グラフィカルな入力によるテストベクタ生成機能付きの、Verilogシミュレータです。
- また、WaveFormer Proの機能を有します。
- 特徴として、タイミングダイアグラムからテストベクタを生成する機能があり、テストパターンを用意しなくとも、タイミングダイアグラムからの入力によりMUTをシミュレーションすることができます（インタラクティブ・シミュレーション機能）。



InterLink



Synapt iCAD社製品群の中での、 TestBench Proの位置づけ（続き）

DataSheet Pro :

- タイミングダイアグラムを含むドキュメントの作成支援ツールです。
- WaveFormer Proの機能を有します。
- 主な特徴：
 - [] OLE機能は標準装備。
 - [] プロジェクト・ウィンドウにより、複数のタイミングダイアグラムを一つのプロジェクトとして管理できます。

WaveFormer Pro :

- タイミングダイアグラムエディタです。
- TimingDiagrammer Proの機能を有します。
- 主な特徴：
 - [] タイミングダイアグラムから、HDLスティミュラスを生成します。
 - [] 多くのファイル形式に対して入出力を対応しています。
 - ロジアナで吸い上げた実機データをシミュレータに利用
 - シミュレータのデータ（vcdファイル）をパタンジェネレータで実機環境で利用
 - ・・・など



InterLink



Synapt iCAD社製品群の中での、 TestBench Proの位置づけ（続き）

TimingDiagrammer Pro :

- ・ タイミングダイヤグラムエディタです。

GigaWave Viewer :

- ・ 波形ビューワです。



InterLink

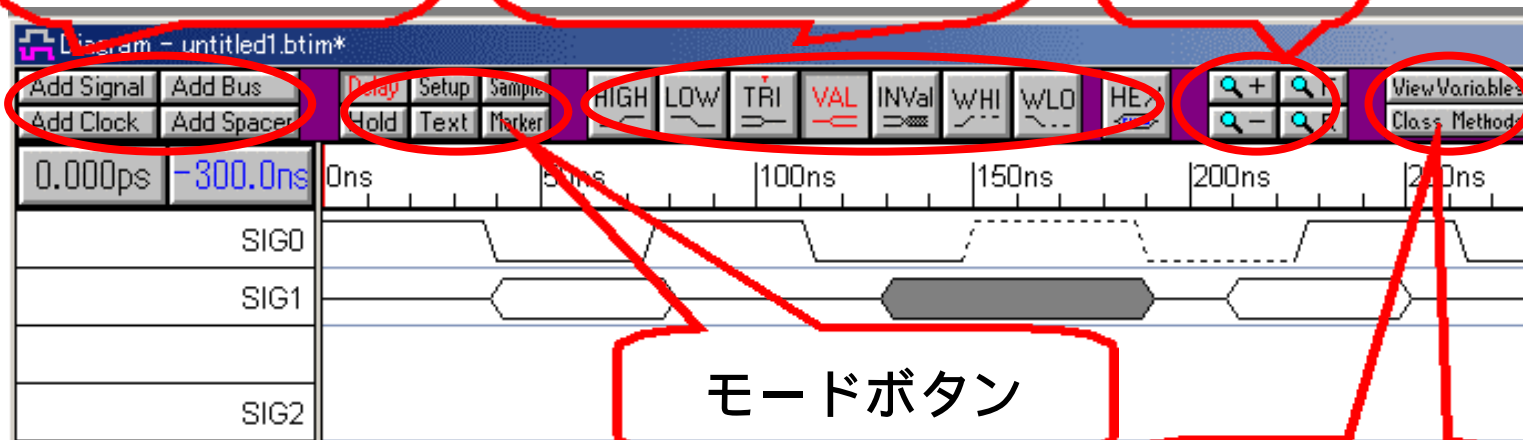


タイミングダイアグラム記述方法

新たな信号の追加ボタン

ステートの状況を表すボタン

ズームボタン



ご参照：スケーリング操作

http://www.syncad.com/ani_app.htm
(アニメーション・アプリケーション)

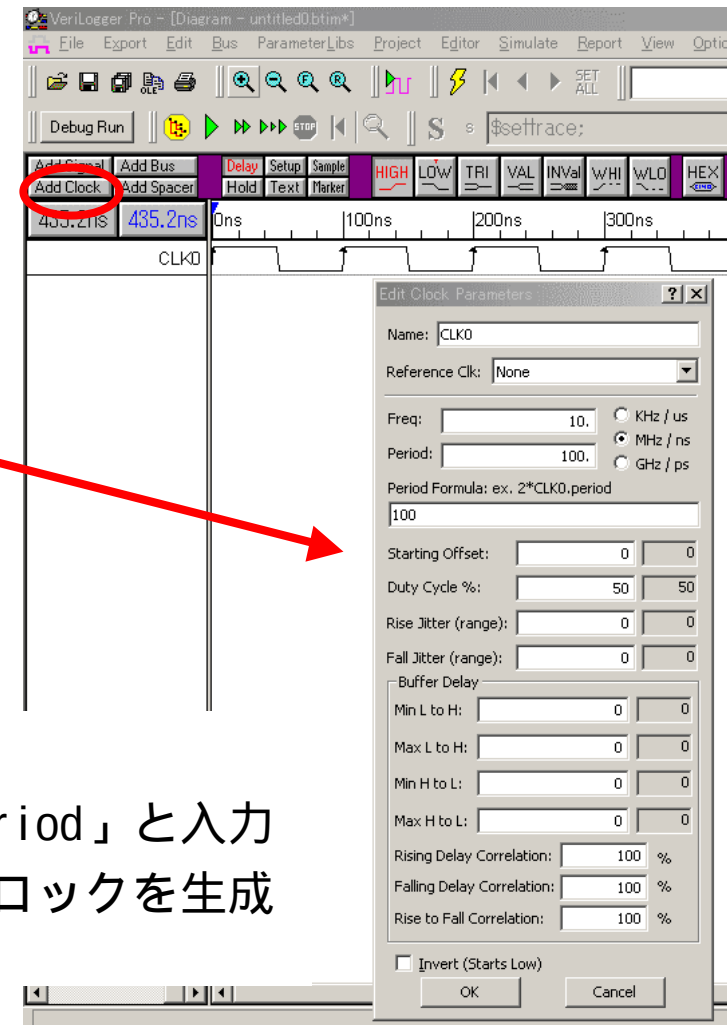
変数とクラス
、メソッドボ
タン

タイミングダイアグラム記述方法（続き）

クロック信号の定義：
Add Clockボタンを押し、
開くダイアログの中で、
クロックのプロパティ
（周期等）を定義する

式による定義も可能

例）CLK1のPeriod Formula欄に「 $2 * CLK0.period$ 」と入力
→ 信号CLK0の2倍のクロック信号のクロックを生成する



InterLink



タイミングダイアグラム記述方法（続き）

「時間式による波形生成」と「自動ラベル機能」

シグナルプロパティ・
ダイアログボックス

The screenshot displays the Verilogger Pro interface with a waveform diagram and the Signal Properties dialog box open. The waveform diagram shows a clock signal (SIG0) and a data signal (SIG3). The Signal Properties dialog box is configured for SIG3, showing the Wfm Eqn as $(20 = V) * 10$ and the Label Eqn as $\text{Hex}(\text{RandInt}(10,255))$. Red callout boxes highlight specific features:

- 信号名をダブルクリック (Double-click signal name)
- ダイアログ開く (Open dialog)
- 時間式と自動ラベル入力欄 (Time formula and automatic label input field)

タイミングダイヤグラム記述方法（続き）

- Setup、Holdパラメータによるタイミング解析例 -

[Setup]、[Hold]タイミング・パラメータを使用すると、タイミング違反が発生した場合、ダイヤグラムとパラメータウィンドウでレポート（右図例では、セットアップエラーをレポート）

使用回路

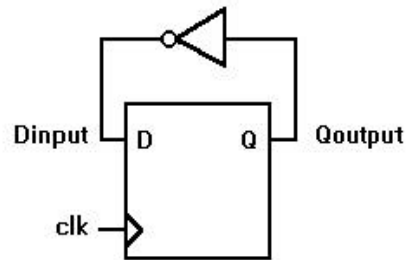
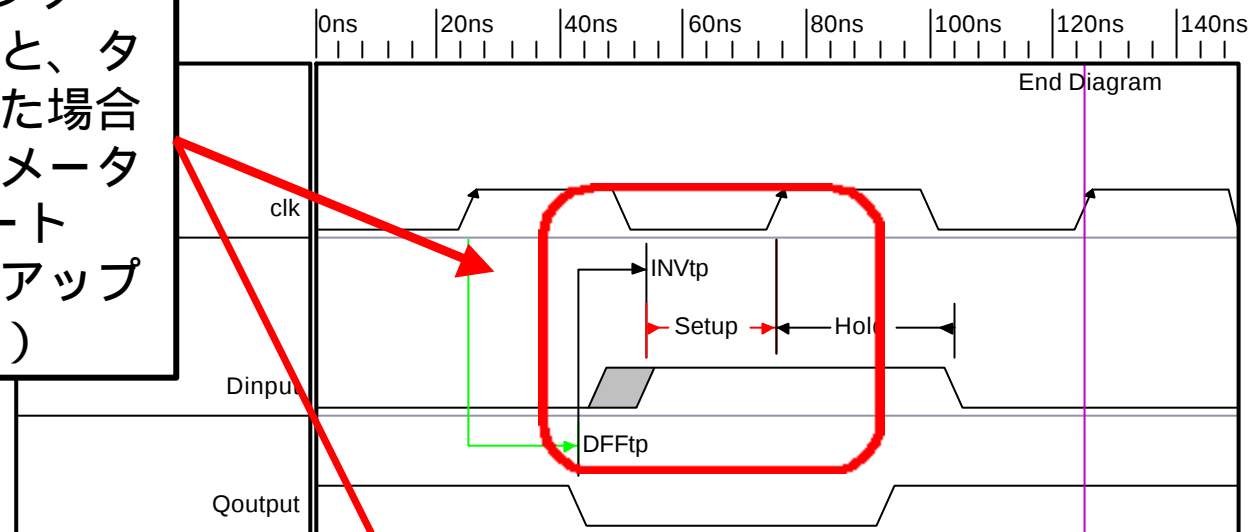


Figure 1: Tutorial circuit

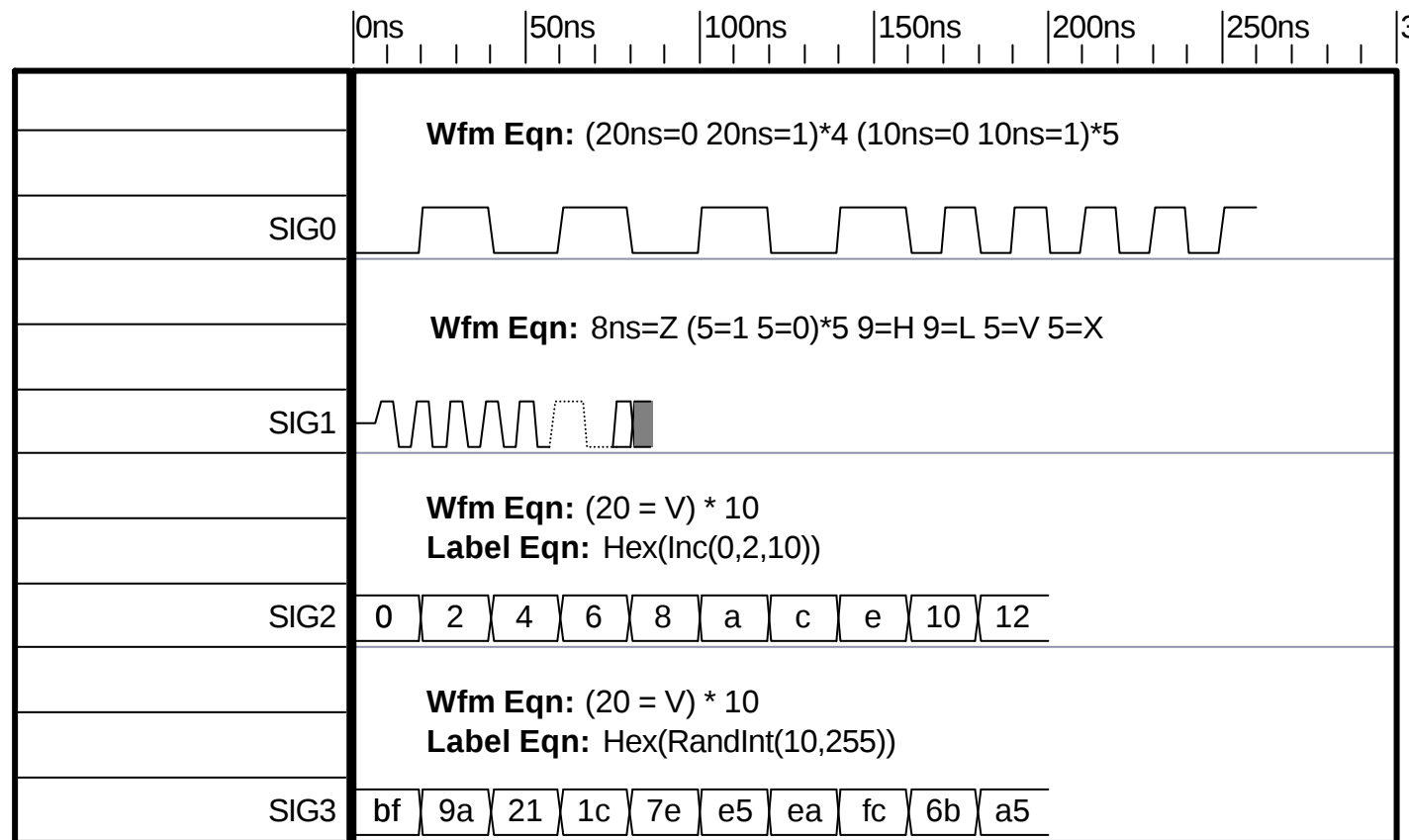


name	min	max	margin	comment
DFFtp		18	na (delay)	clk to Qoutput propagation time
INVtp	3	11	na (delay)	INV inverter delay
Setup	25		-4	Check for metastable condition
Hold	1		28	

タイミングダイアグラム記述方法（続き）

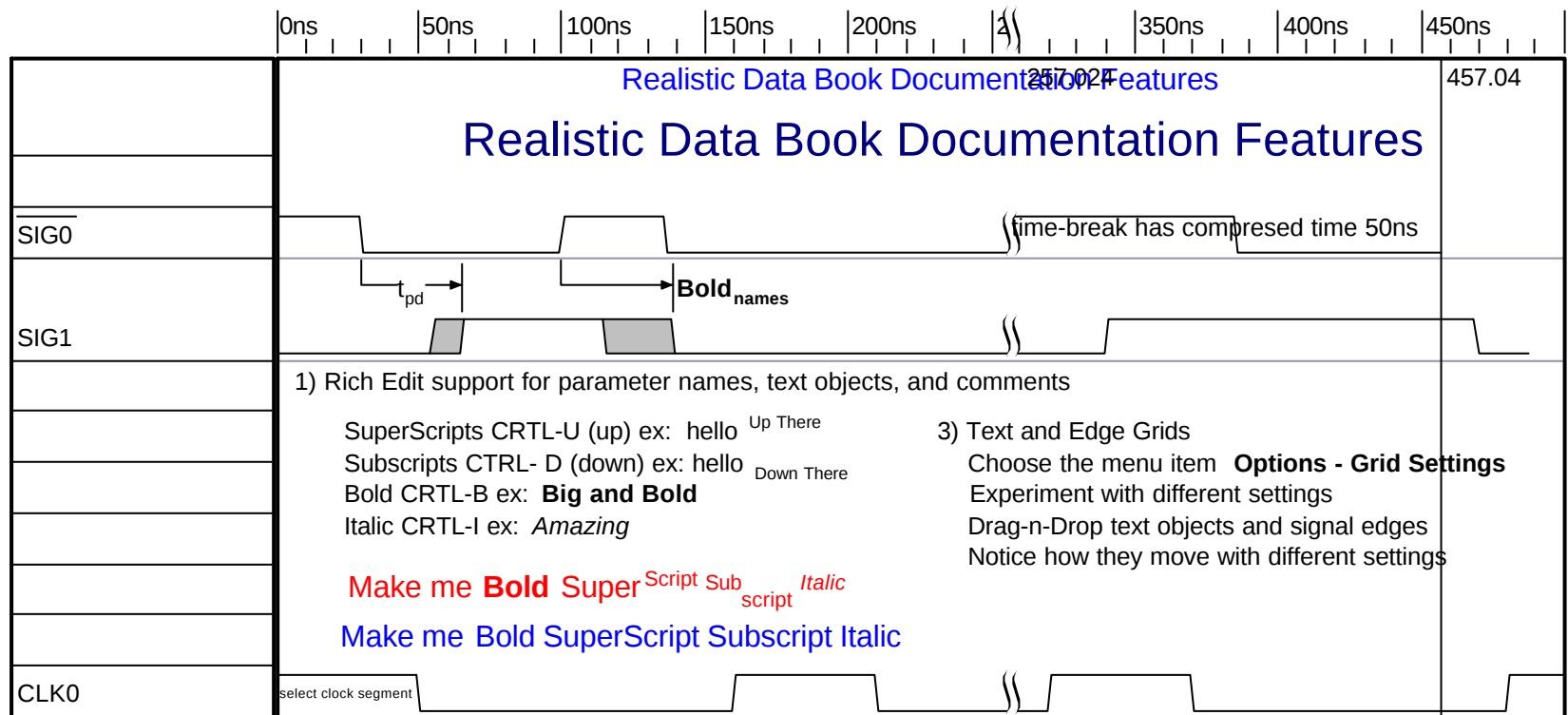
時間式による波形生成（[Wfm Eqn]ボタン）と

自動ラベル機能（[Label Eqn]ボタン）で生成した波形の例



波形の例

評価用プログラムの Examples ディレクトリの中には、この他にも色々な波形例がございます。ご確認ください。

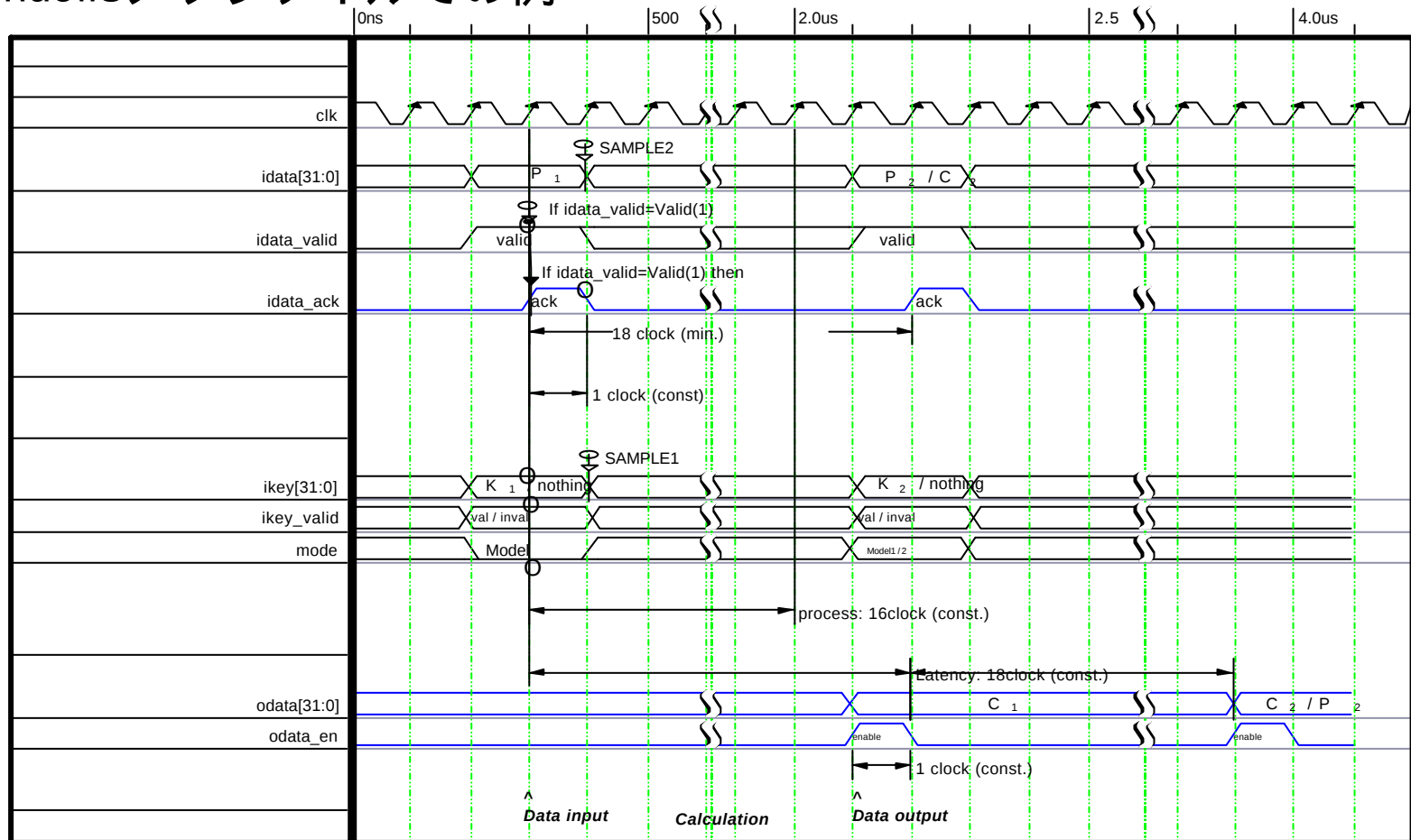


InterLink



MS-Wordへの波形貼り付け

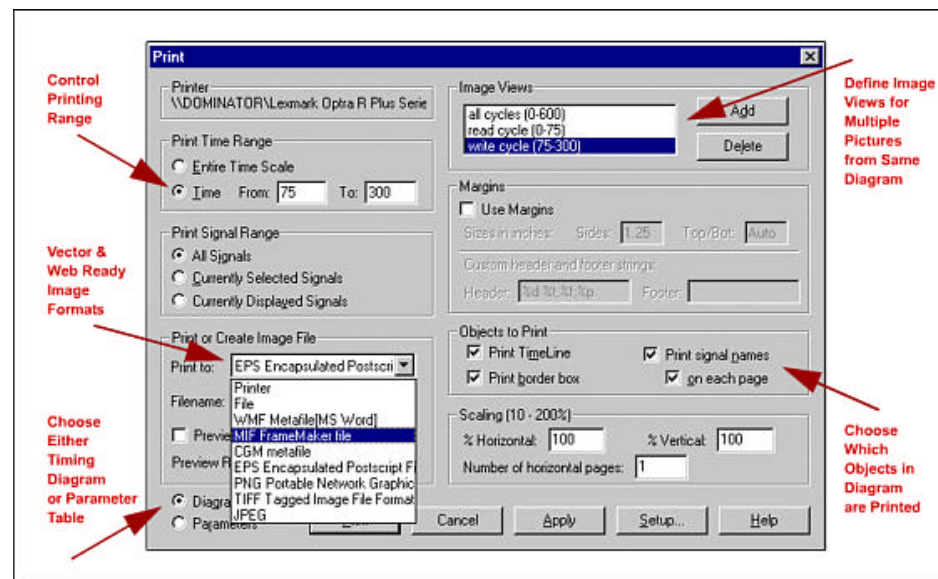
Windowsメタファイルでの例



MS-Wordへの波形貼り付け（続き）

対応フォーマット

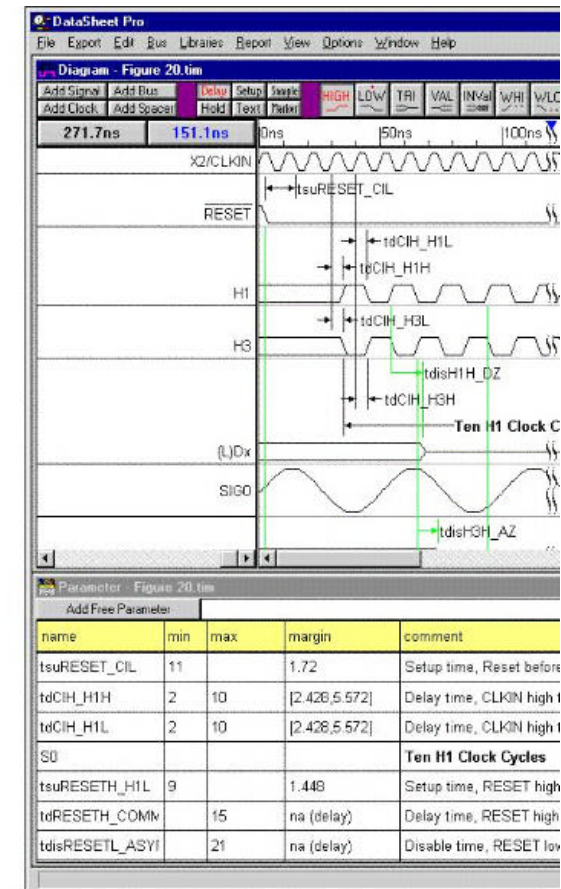
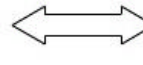
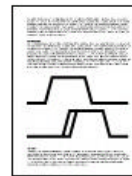
- Windowsメタファイル（.wmf）、フレームメーカーMIFファイル、EPSファイル、CGMファイル、EMFファイル
- TIFF、PNG、JPEGはDataSheet Proで対応



MS-Wordへの波形貼り付け（続き）

OLE機能

TestBench Proでは、
オプション機能として
追加が必要



InterLink



TestBencher Pro 製品概要

設計が大規模・複雑になるにしたがい、HDL設計全行程のうち、テストベンチ生成の工数の占める割合が多くなりました。

=> テストベンチ作成の為の工数を減らす手段が必要。。。>

- ✧ TestBencher は、テストベンチ生成とメンテナンスに要する時間を削減します。
- ✧ テストベンチ開発の局面で遭遇する自動化できる作業（ポート情報の管理など）をTestBencherに行わせる事で、設計者は、本来の設計作業（トランザクション、シーケンスなど）に注力できます。
- ✧ 言語に非依存のタイミングダイアグラムから、テストベンチを生成します。



InterLink



TestBencher Pro 製品概要（続き）

- ✧ グラフィカルなタイミングダイアグラムからバスファンクション・モデルを生成します。
- ✧ サイクルベース、イベントベース、両方のテストベンチをサポートします。
- ✧ MUT信号を抽出 - MUTからポート情報を自動抽出し、正しい信号属性の状態で、ダイアグラムにセットアップします。
=>方向、タイプ、サイズは自動でセットアップされます。
- ✧ Sample 機能 - MUTの出力結果や、他のダイアグラムトランザクション結果に応答するテストベンチを生成します。
- ✧ クラス生成 - ユーザ定義のクラス（データ構造）を生成キュー、アレイなど



InterLink



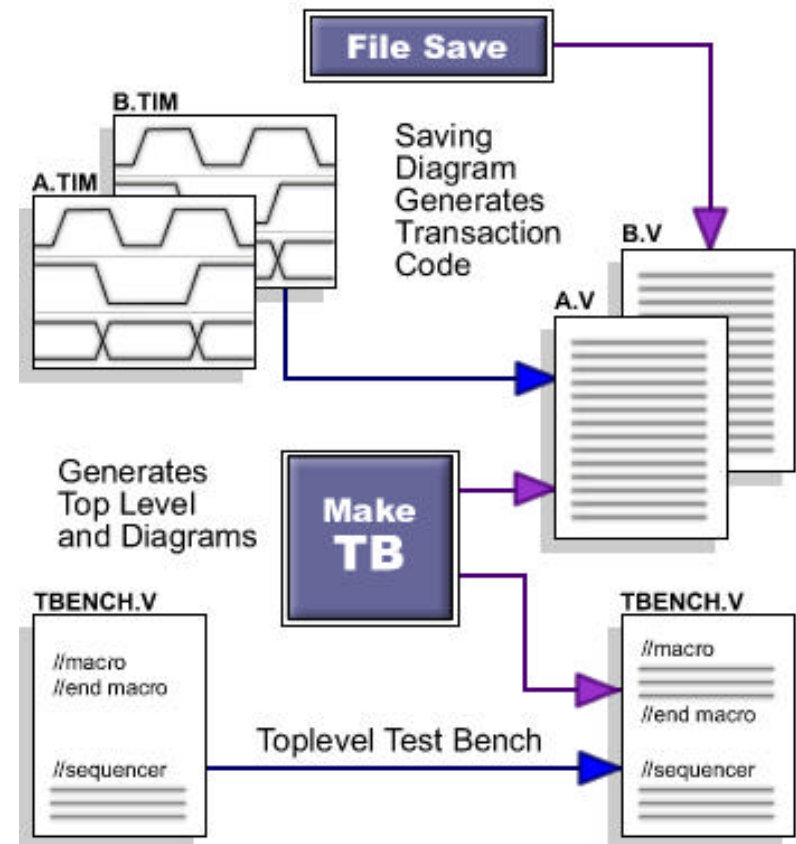
コード生成アーキテクチャ

✓「タイミング・ダイアグラム」と「トップレベル・テンプレートファイル」からテストベンチを生成します。

✓トップレベル・コードはトランザクションのシーケンスやシミュレーション中の信号モニタをコントロールします。

✓ダイアグラム保存時に、各トランザクションのコードは生成されます。
→ダイアグラムの編集結果をすぐに確認できます。

TestBencher Code Generation



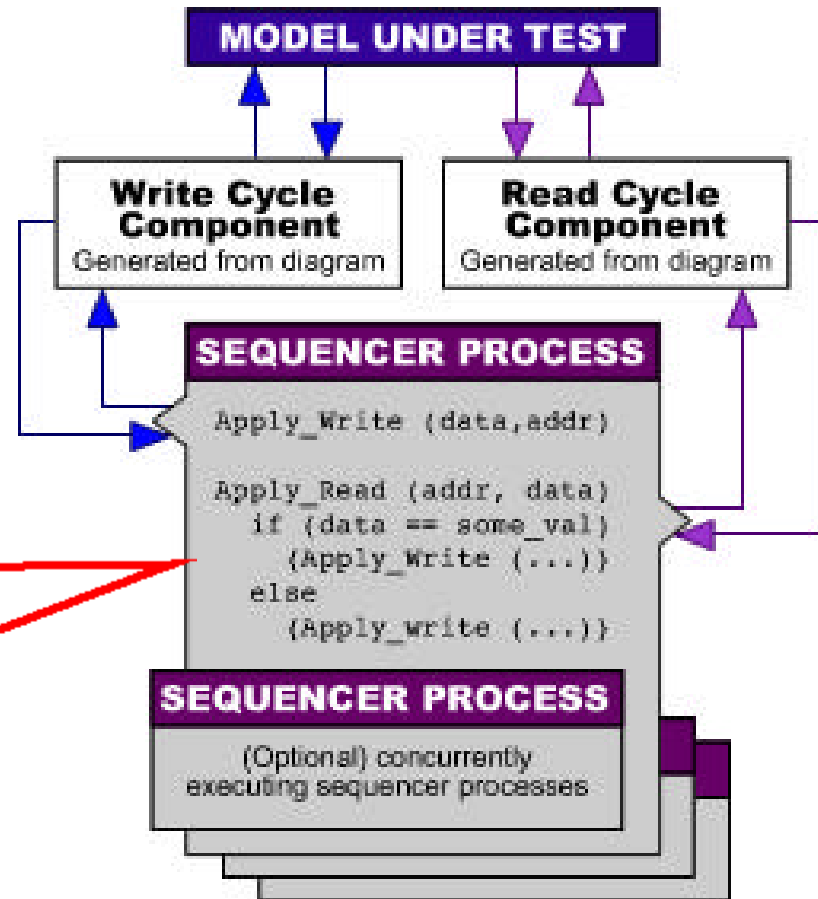
InterLink



コード生成アーキテクチャ（続き）

各タイミング・トランザクションごとのタスクが生成されます。

シーケンサ・プロセスで、MUTに対するタイミング・トランザクションの実行順序をコントロール



コード生成アーキテクチャ（続き）

トップレベルテンプレートファイルの展開（makeTBボタン実行）前後

makeTB実行前

makeTB実行後

The image shows two side-by-side code editors. The left editor, titled 'sramtest_v_before_makeTB.v', shows a template with comments like '// \$InitializeDiagramsTask' and '// \$DefineDiagramTasks'. The right editor, titled 'sramtest_v_after_makeTB.v', shows the same template but with actual tasks generated, such as 'task tb_initialize_sramtest_v;' and 'task Apply_tbwrite;'. Red boxes and arrows highlight the transformation. A central text box explains that diagram and transaction tasks are generated.

```
// $FileOutputComponentInstantiation
// End $FileOutputComponentInstantiation

// $InitializeDiagramsTask
// End $InitializeDiagramsTask

// $DefineDiagramTasks
// End $DefineDiagramTasks

// $ResolverProcess
// End $ResolverProcess

// $UserDefinedMethods
// End $UserDefinedMethods

// $MiscFunctions
// End $MiscFunctions

//*****
// Instatiation of the Model

// $FileOutputComponentInstantiation
// There are no output file associations being used (set i
// End $FileOutputComponentInstantiation

// $InitializeDiagramsTask
task tb_initialize_sramtest_v;
begin
    #0;
end
endtask
// End $InitializeDiagramsTask

// $DefineDiagramTasks
task Apply_tbwrite;
input [7:0] addr;
input [7:0] data;
begin
    tbwrite_addr = addr;
    tbwrite_data = data;
    tbwrite_tb_trigger = `TB_ONCE;
end
endtask
```

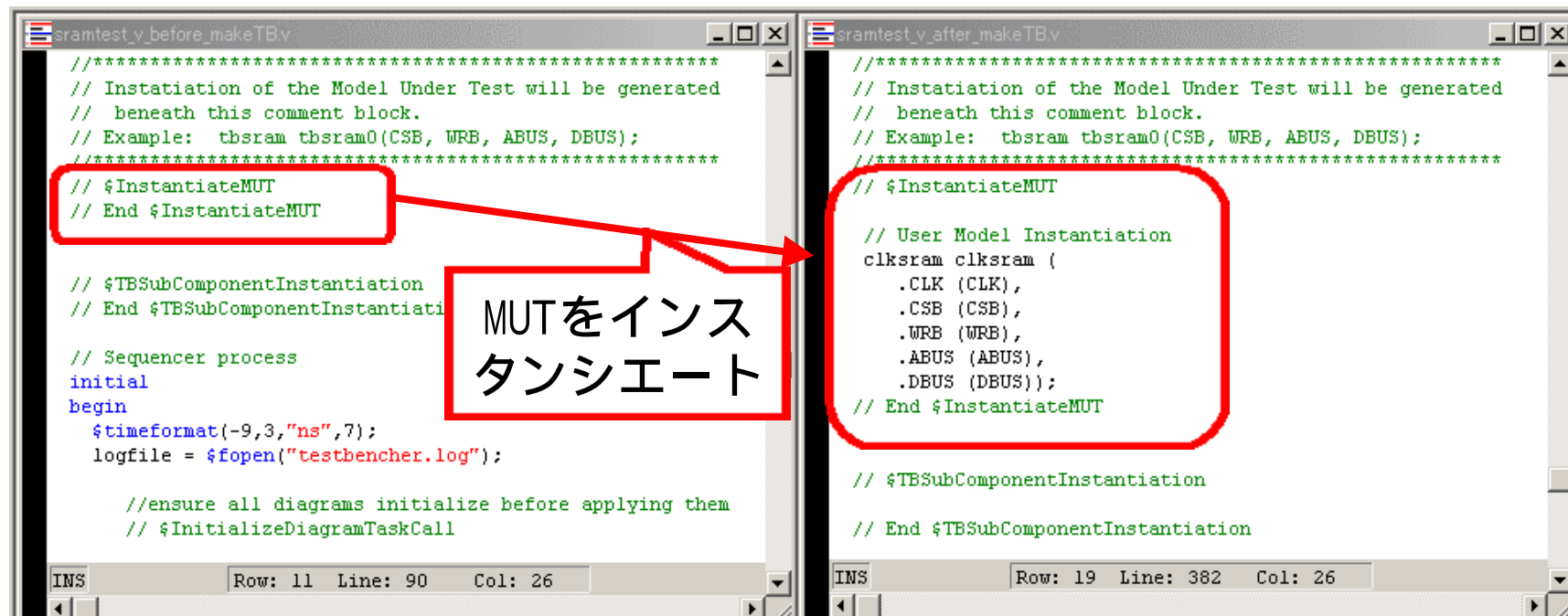
ダイアグラム・トランザクションをタスクに生成

コード生成アーキテクチャ（続き）

トップレベルテンプレートファイルの展開（makeTBボタン実行）前後

makeTB実行前

makeTB実行後



```
//*****
// Instatiation of the Model Under Test will be generated
// beneath this comment block.
// Example: tbsram tbsram0(CSB, WRB, ABUS, DBUS);
//*****
// $InstantiateMUT
// End $InstantiateMUT

// $TBSUBComponentInstantiation
// End $TBSUBComponentInstantiation

// Sequencer process
initial
begin
$timeformat(-9,3,"ns",7);
logfile = $fopen("testbencher.log");

//ensure all diagrams initialize before applying them
// $InitializeDiagramTaskCall

INS Row: 11 Line: 90 Col: 26
```

MUTをインスタンス化

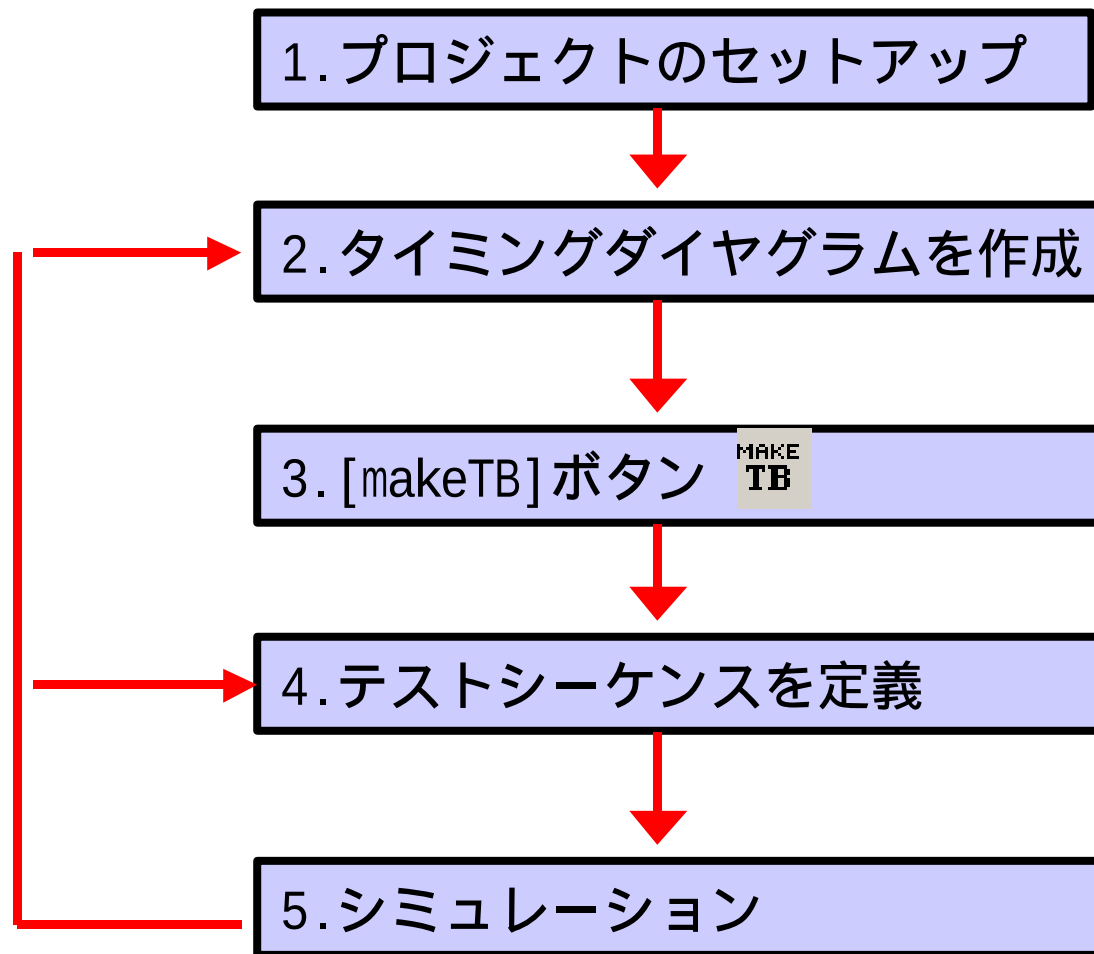
```
//*****
// Instatiation of the Model Under Test will be generated
// beneath this comment block.
// Example: tbsram tbsram0(CSB, WRB, ABUS, DBUS);
//*****
// $InstantiateMUT

// User Model Instantiation
clkgram clkgram (
.CLK (CLK),
.CSB (CSB),
.WRB (WRB),
.ABUS (ABUS),
.DBUS (DBUS));
// End $InstantiateMUT

// $TBSUBComponentInstantiation
// End $TBSUBComponentInstantiation

INS Row: 19 Line: 382 Col: 26
```

TestBencher デザインフロー



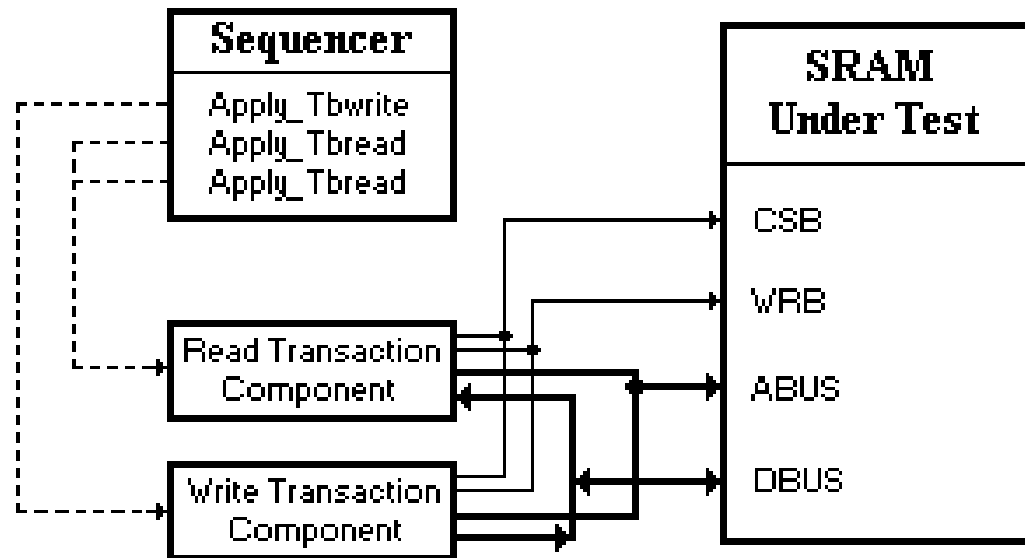
TestBencher デザインフロー

- 例：SRAMテスト（Tutorial5） -

操作デモ用回路

テストベンチ

Module Under Test



InterLink



TestBencher デザインフロー

- 例：SRAMテスト（Tutorial5） -

デモシナリオ

- ✓ SRAMモデルを例にとり、実際のフローのご説明と操作デモを行います。
- ✓ このデモではまず、使用言語として Verilog を選択し、TestBencher 内蔵の Verilog シミュレータを使って、シミュレーションまでをご覧ください。
- ✓ Sample 機能をご確認いただく為に、ワーニングが発生するよう、故意にエラーを記述しています。
- ✓ 次に、e /Verilog 言語を選択し、e 言語が生成される事をご覧ください。
- ✓ また、Sample の設定内容が、e のキーワード"expect"を用いて生成される例をご覧ください。



InterLink



TestBencher デザインフロー

- 例：SRAMテスト (Tutorial5) -

1. プロジェクトのセットアップ

- ✓ウィザードによりプロジェクト設定
- ✓MUTをプロジェクトへ追加
- ✓MUTの信号情報を抽出 

2. タイミングダイアグラムを作成

リセット (tbreset)、
ライト (tbwrite)、
リード (tbread) の
各トランザクションを作成

3. [makeTB] ボタン

4. テストシーケンスを定義

ダイアログで各トランザクションを
トップレベルテストベンチ上に呼び
出す

5. シミュレーション



InterLink

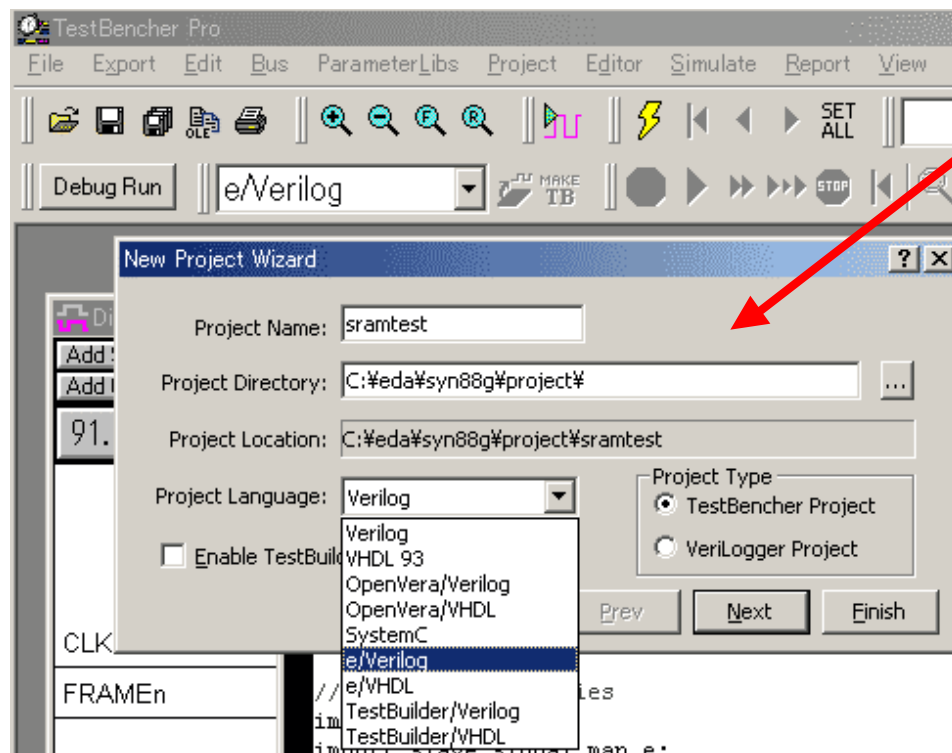


TestBencher デザインフロー

- 例：SRAMテスト（Tutorial5） -

1. プロジェクトのセットアップ

✓ ウィザードによりプロジェクト設定



✓プロジェクト名、作業ディレクトリ、使用する言語（e、OpenVera等）を選択

✓サイクルベースに必要なデフォルトクロック名なども設定



InterLink



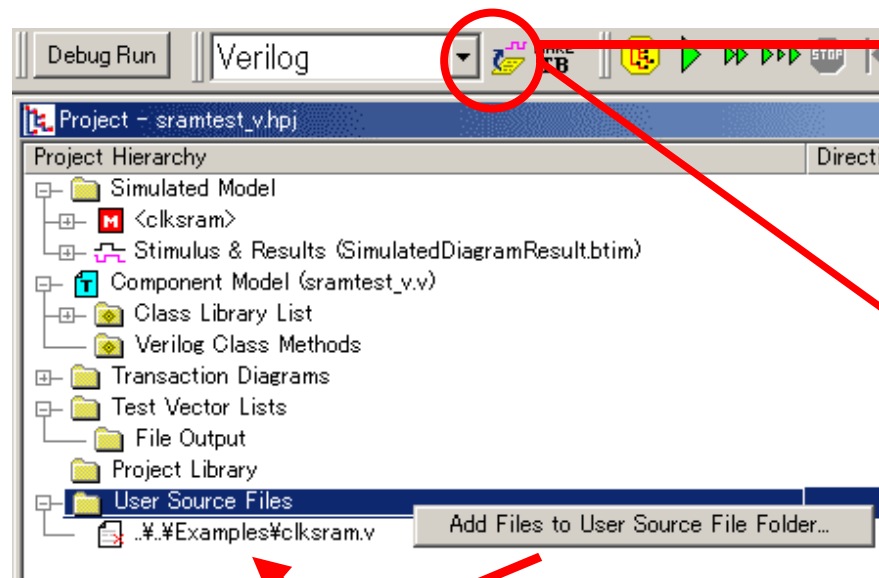
TestBencher デザインフロー

- 例：SRAMテスト (Tutorial5) -

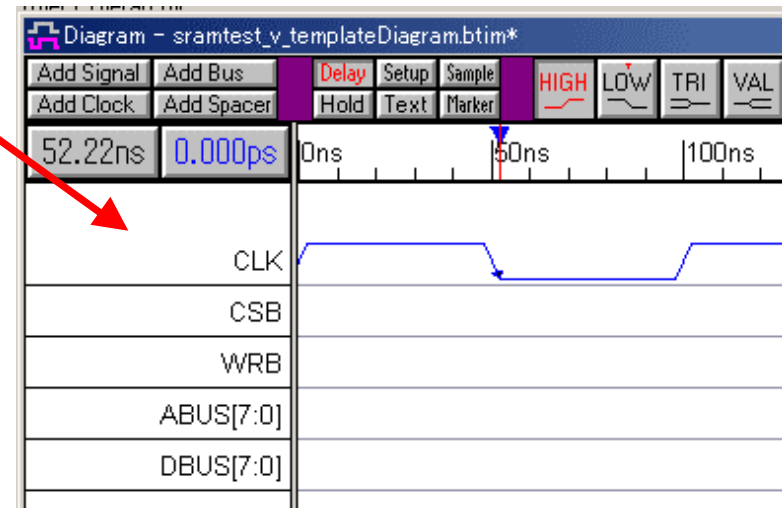
1. プロジェクトのセットアップ

✓ MUTをプロジェクトへ追加

✓MUTの信号情報を抽出



信号情報がダイアグラムエディタにセットアップされる



コンテキストメニューを使って
MUTをプロジェクトへ追加



InterLink



TestBencher デザインフロー

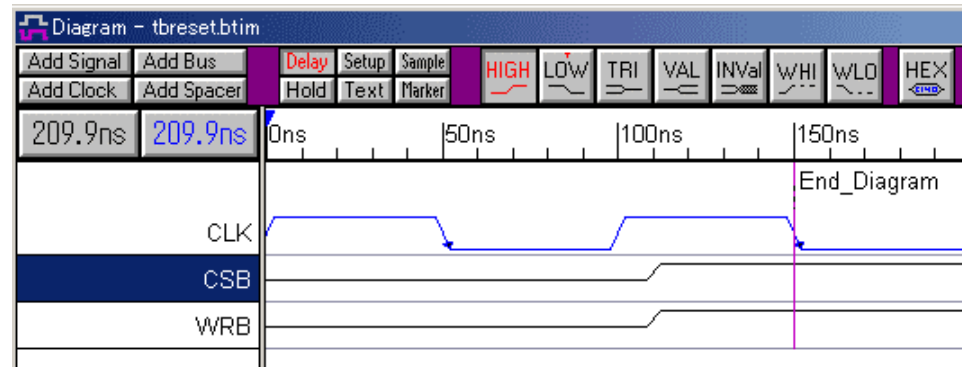
- 例：SRAMテスト (Tutorial5) -

2. タイミングダイアグラムを作成

✓ リセット (tbreset)

状態ボタン、モードボタン、マーカー等を使用してタイミングダイアグラムを描画する

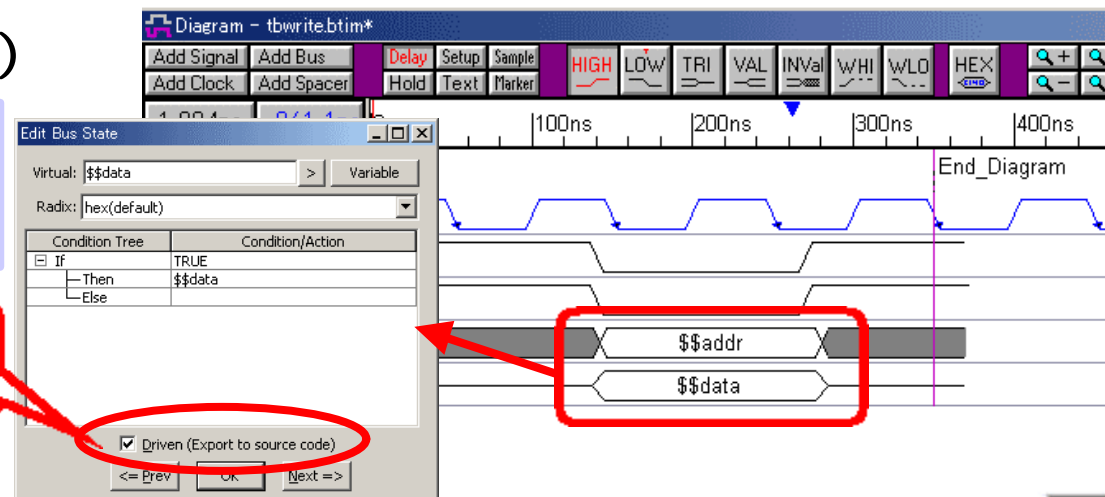
描画したダイアグラムはプロジェクトへ追加する



✓ ライト (tbwrite)

ABUS (アドレス・バス)、DBUS (データ・バス) の値は変数を用いる

信号方向のコントロール



TestBencher Pro

InterLink



TestBencher デザインフロー

- 例：SRAMテスト（Tutorial5） -

2. タイミングダイアグラムを作成

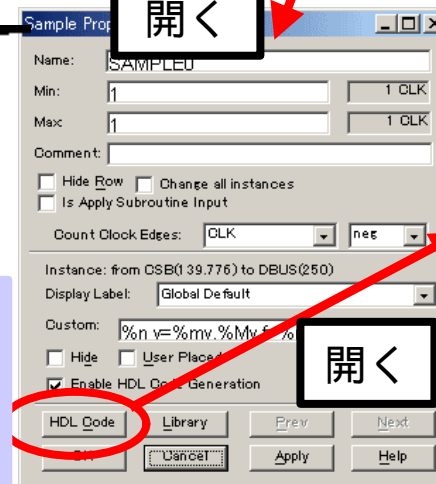
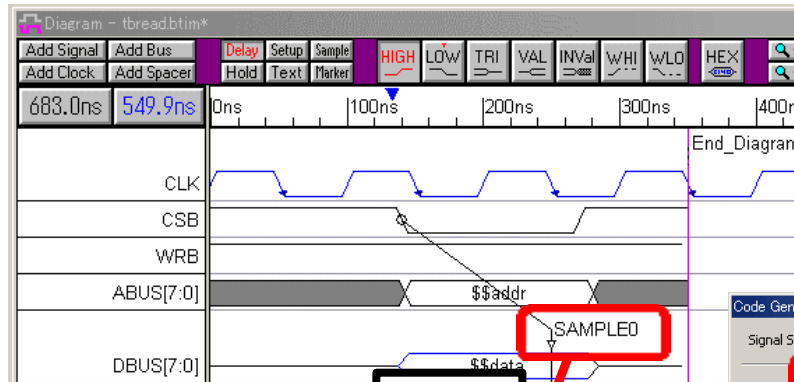
✓ リード（tbread）

ライト・トランザクションと
リード・トランザクションとは、
「WRBの極性」と「DBUSの方向」
が違う

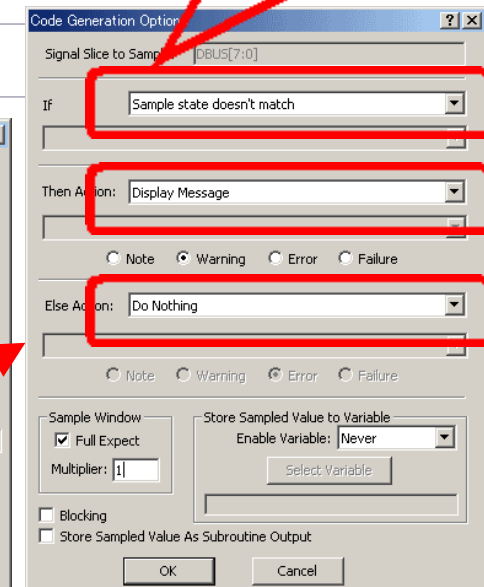
↓
tbwrite の名前を変えて保存 &
編集して tbread を作成する

Sample（サンプル）の振る舞い
は、ダイアログで設定する

SRAMテストでの Sample0 設定内容：
If「DBUSの値が期待値と合っていない場合には」
Then「ワーニングメッセージを表示させる」
Else「（期待値通りなので）何もしない」



If ~
Then ~
Else ~
のコンディシ
ョン設定



InterLink

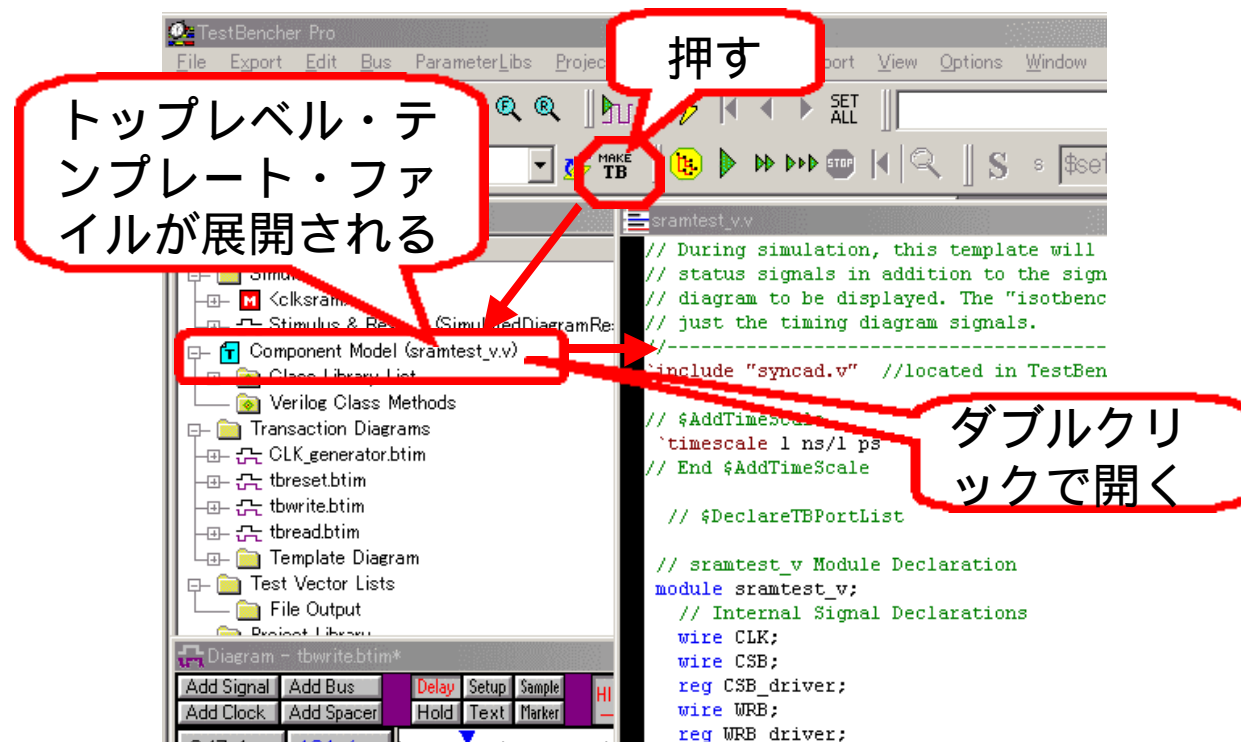


TestBencher デザインフロー

- 例：SRAMテスト（Tutorial5） -

3. [makeTB] ボタン

タイミングトランザクションを描画し終えたら、[makeTB]ボタンを押し、トップレベル・テンプレート・ファイルを展開します。



TestBencher デザインフロー

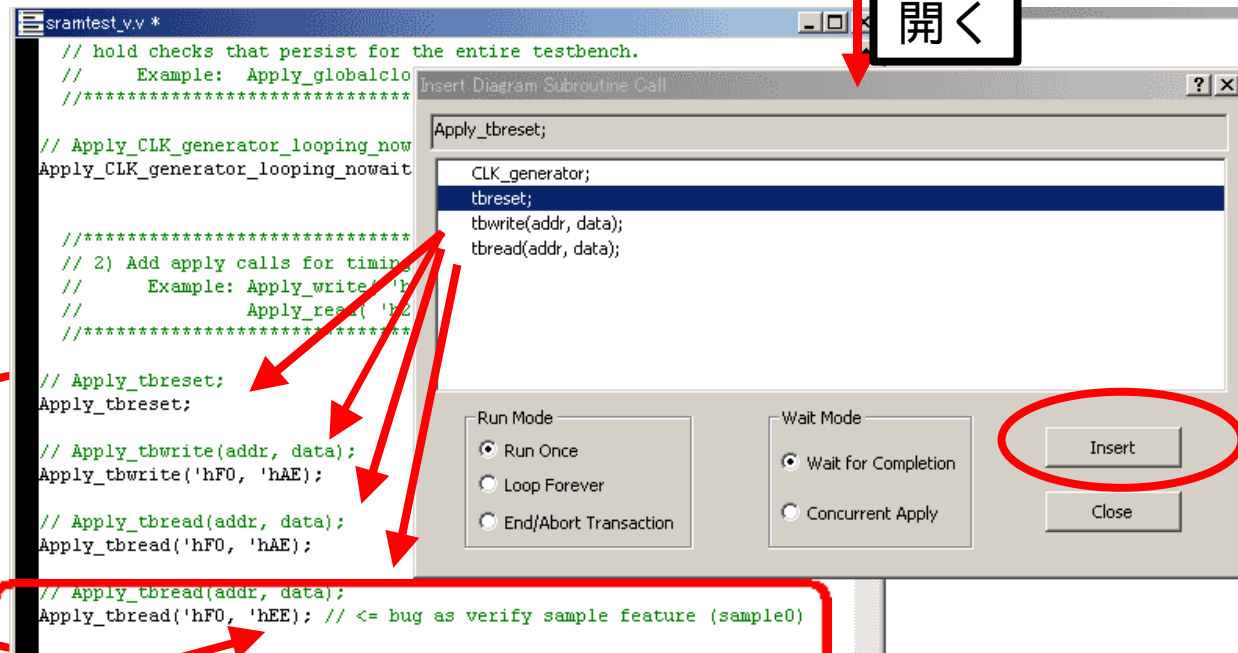
- 例：SRAMテスト（Tutorial5） -

4. テストシーケンスを定義

トップレベル・テストベンチ・エディタ上でマウス右クリック
→メニューから[Insert Diagram Calls...]を選択する

ここでの定義
リセットx1
ライトx1
リードx2

Sample 機能を確認
する為、2回目のリ
ードでは誤った値を
リードしてみる



TestBencher デザインフロー

- 例：SRAMテスト（Tutorial5） -

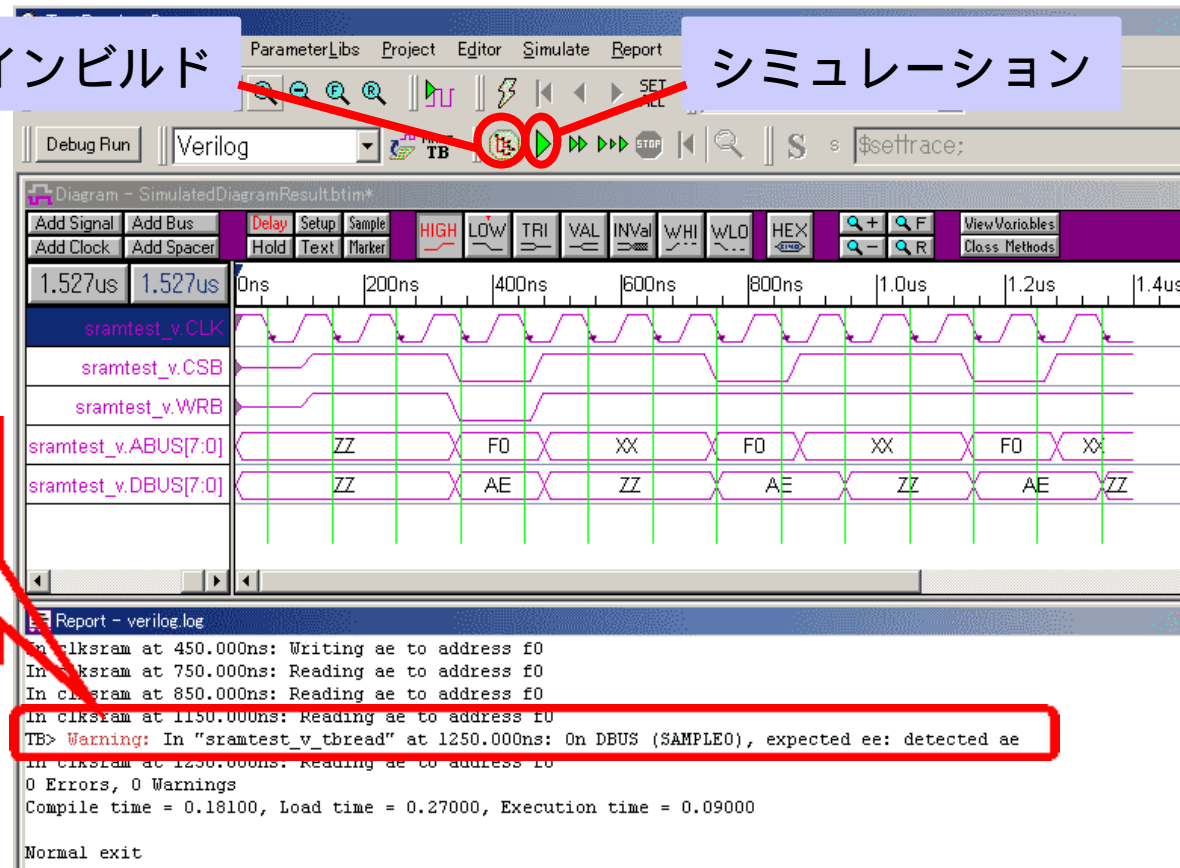
5. シミュレーション

TestBencher Pro内蔵のVerilogシミュレータでシミュレーション

デザインビルド

シミュレーション

Sample ダイアログの設定内容により、レポートが表示される



TestBencher Pro

InterLink



TestBencher デザインフロー

- 例 : SRAMテスト (Tutorial5) -

SRAM テストで e/Verilog 言語を選択の場合 :

Sampleダイアログの
内容が e 言語を使用
して記述される
(この例では テンポ
ラル記述のキーワー
ド "expect" が使用
されている)

Code Generation Options

Signal Slice to Sample: DBUS[7:0]

If: Sample state doesn't match

Then Action: Display Message

Min:

Max:

Com: ☐ Note ☒ Warning ☐ Error ☐ Failure

Else Action: Do Nothing

In:

Di:

Out:

Sample Window

☒ Full Expect

Multiplier: 1

Blocking ☐

Store Sampled Value As Subroutine Output ☐

Store Sampled Value to Variable

Enable Variable: Never

Select Variable

OK Cancel

```
// Expects generated from Samples
expect SAMPLED is
{ @start_of_transaction; [..]: @CSB_neg_event } => { [SAMPLED_min-1]*cycle; true('DBUS' == data) } @CLK_neg_event
else dut_error("Sample SAMPLED failed on signal DBUS");
```

操作デモ

- SRAMテスト - 一連の操作をご覧ください。
- Examplesディレクトリの中にある、PCIやATMのタイミングダイアグラムをご確認ください。
 - タイミングダイアグラム・ウィンドウ内の [Source Code] ボタンを押すことで、表示されている波形に対応する e や VERA などのコードが確認できます。



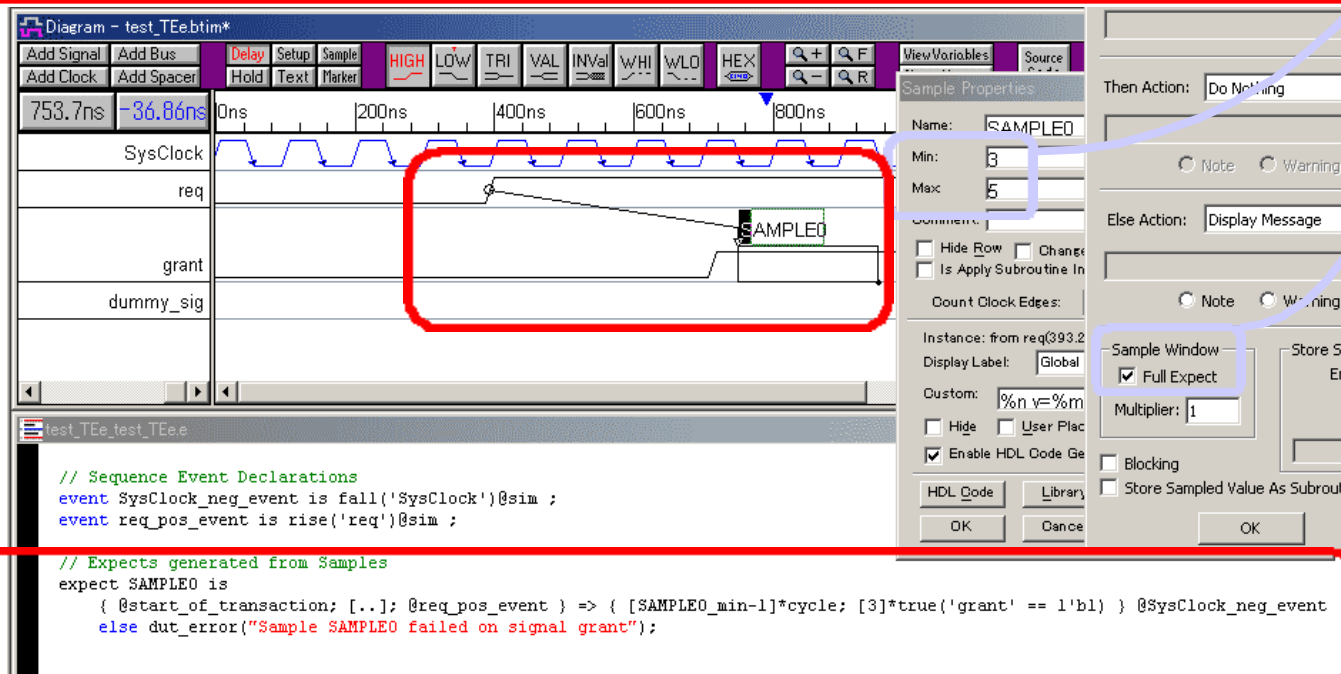
InterLink



e テンポラル記述 - 出力例

Window サンプル; Full Expect チェック : "req" の立ち上がり3クロック~5クロックで "grant" が立ち上がる → "grant" のスタビリティ・チェック

```
// Expects generated from Samples
expect SAMPLE0 is
    { @start_of_transaction; [...]; @req_pos_event } => { [SAMPLE0_min-1]*cycle;
[3]*true('grant' == 1'b1) } @SysClock_neg_event
    else dut_error("Sample SAMPLE0 failed on signal grant");
```



TestBench Pro

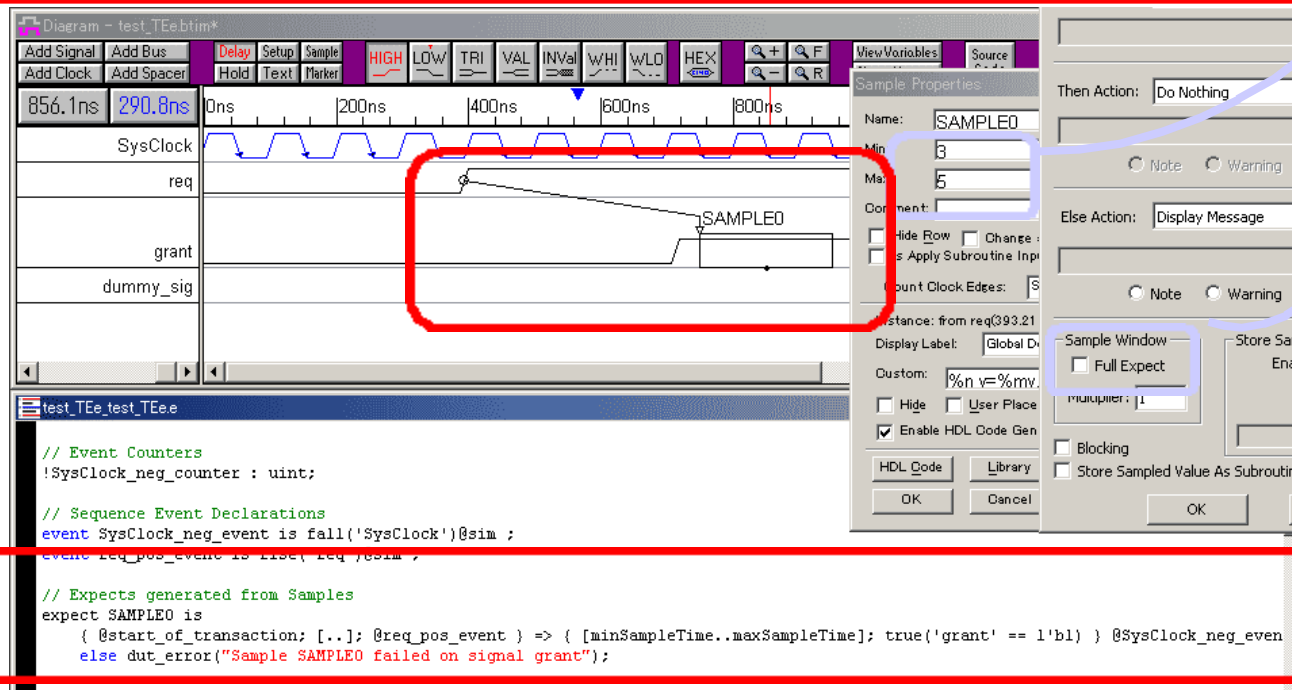
InterLink



e テンポラル記述 - 出力例

Window サンプル; Full Expect チェック外す: "req" の立ち上がり3クロック~5クロックで "grant" が立ち上がる → 指定クロック内で "grant" がアサートするかチェック

```
// Expects generated from Samples
expect SAMPLE0 is
{ @start_of_transaction; [...]; @req_pos_event } => {
[minSampleTime..maxSampleTime]; true('grant' == 1'b1) } @SysClock_neg_event
else dut_error("Sample SAMPLE0 failed on signal grant");
```



TestBench Pro

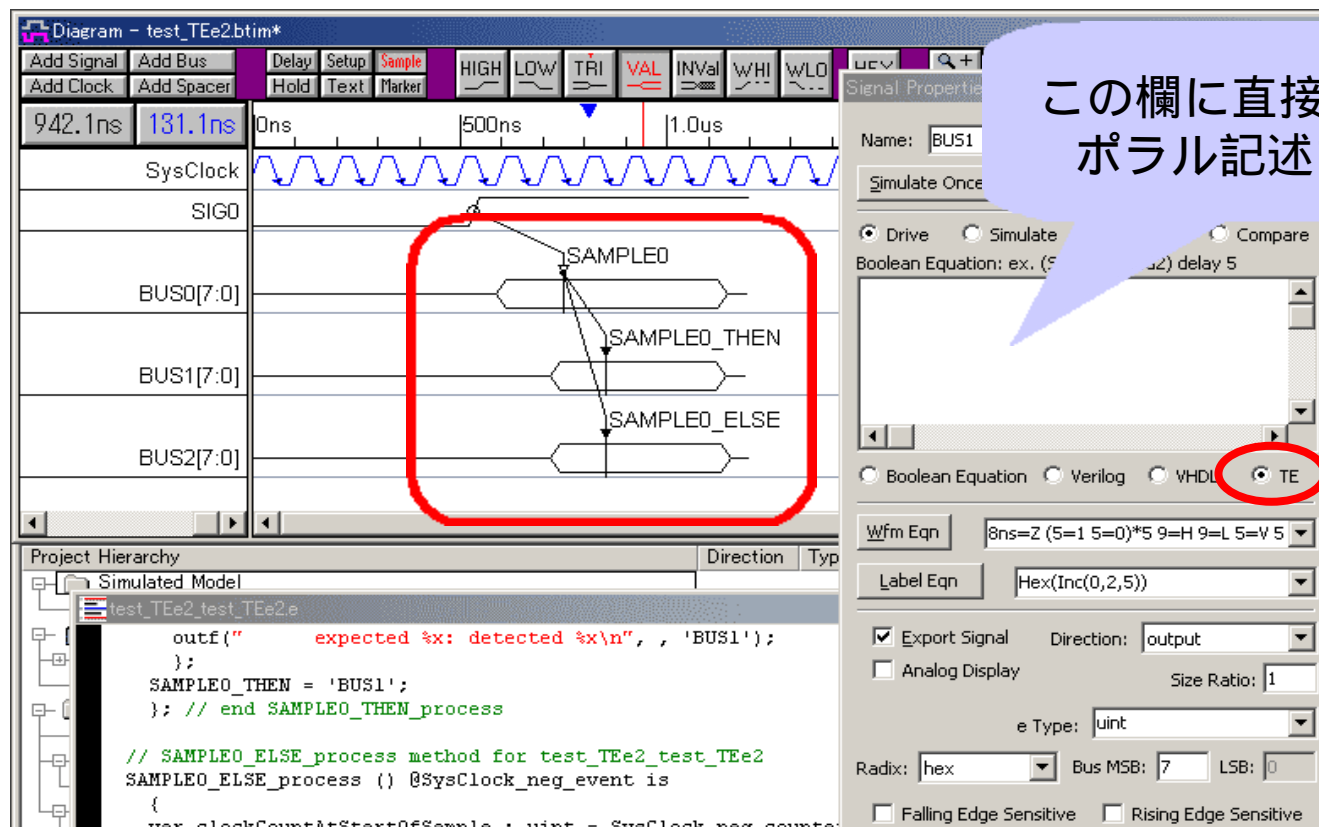
InterLink



e テンポラル記述 - 出力例

✓ Sample 間の If ~ Then ~ Else 定義も可能

✓ 「Signal Properties」ダイアログを使って、直接e言語を記述する事も可能



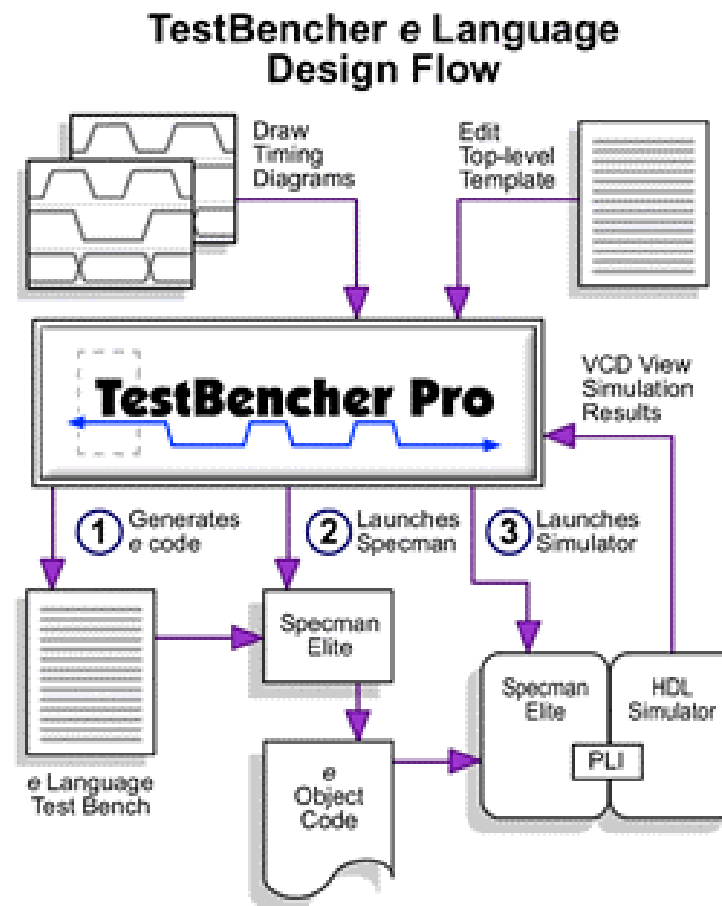
TestBench Pro

InterLink



TestBencher デザインフロー : e

eでのデザインフロー

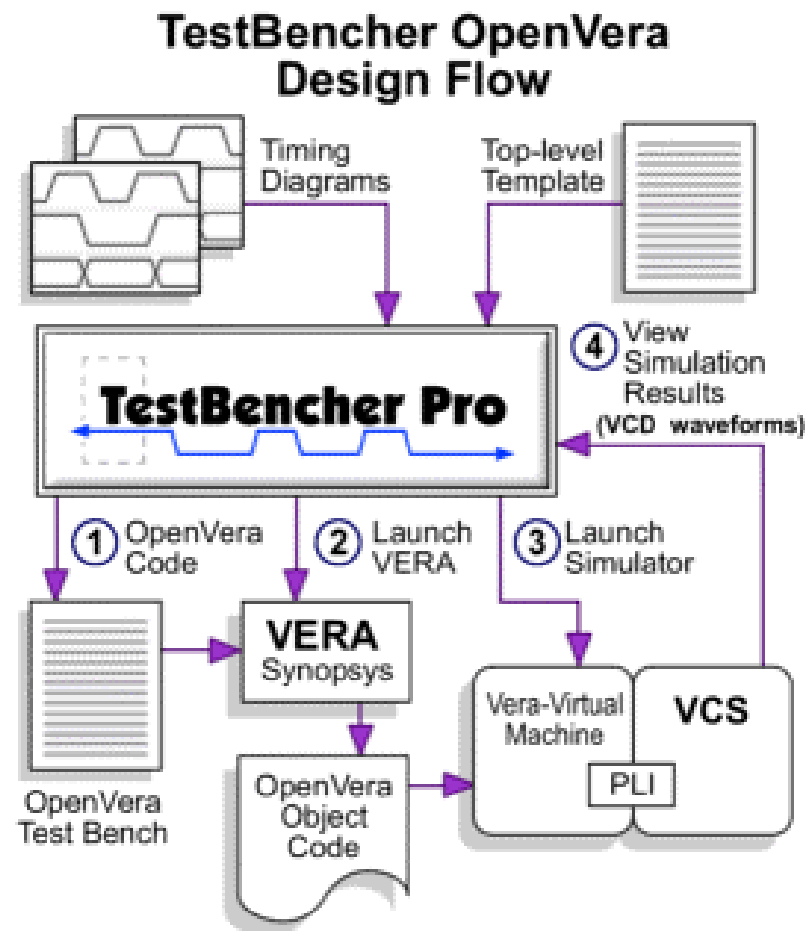


InterLink



TestBencher デザインフロー： OpenVera

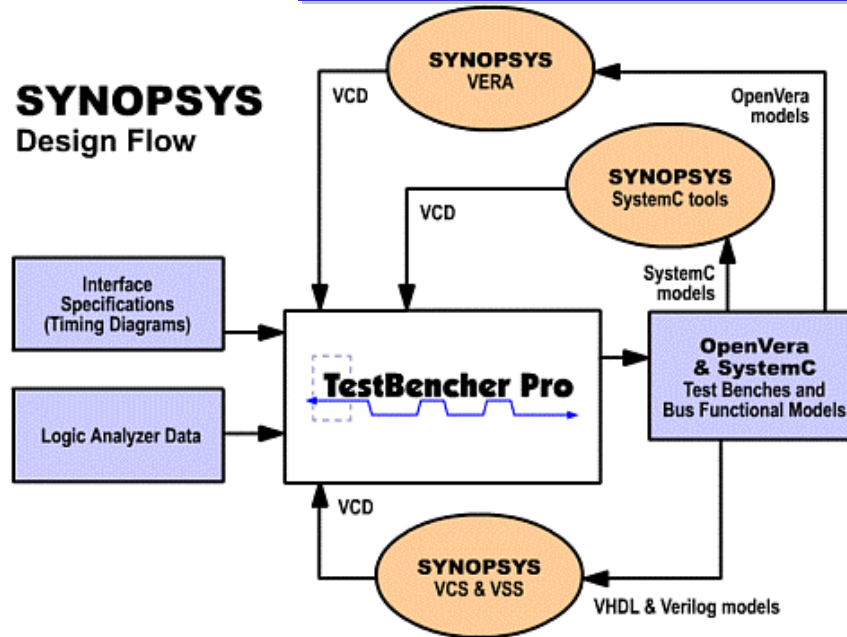
OpenVera での
デザインフロー



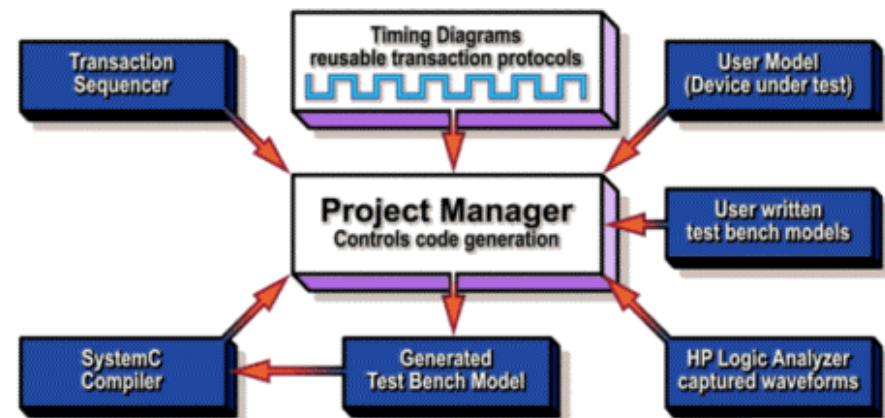
InterLink



TestBencher デザインフロー： OpenVera、SystemC



TestBencher Pro Design Flow

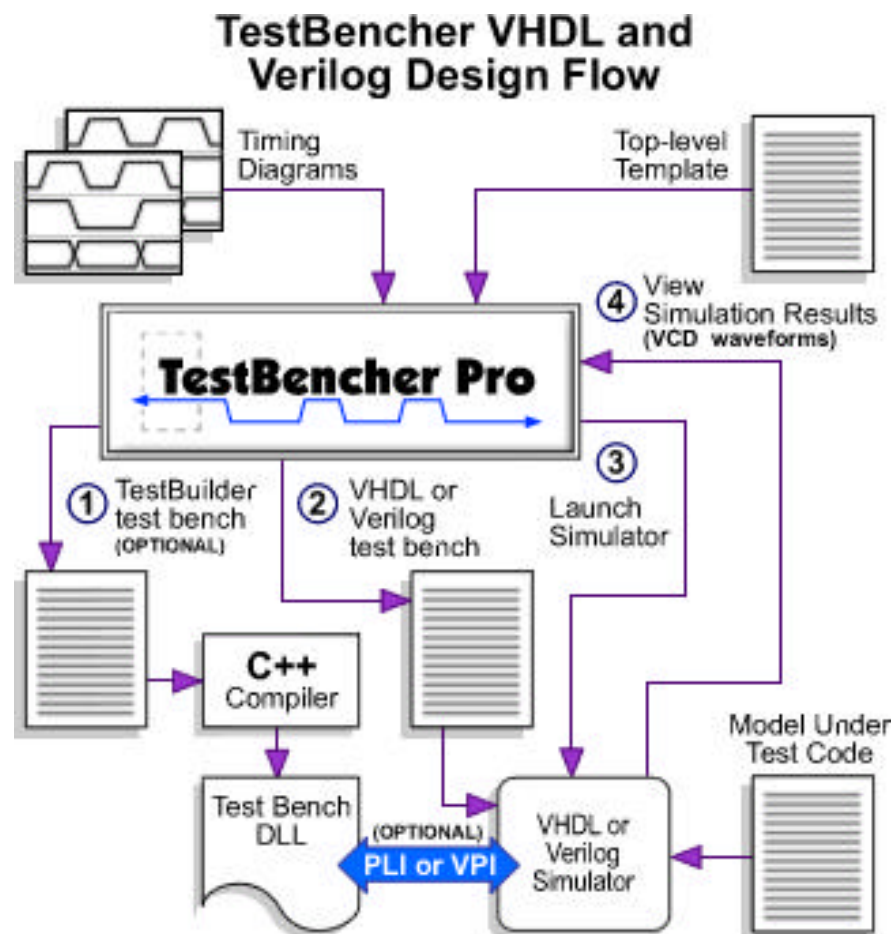


InterLink



TestBencherデザインフロー： VHDL、Verilog、TestBuilder

TestBuilder での
デザインフロー



InterLink

