VeriLogger Pro 操作チュートリアル

本チュートリアルでは,基本的なプロジェクトの作成,プロジェクト内にある Verilog-HDL ファイルのシミュレーションについて説明します.

1) プロジェクトにファイルを追加する

VeriLogger Pro シミュレータは、「プロジェクト」を使ってシミュレーション・オプションの制御、シミュレーションされる Verilog-HDL ファイルの設定、シミュレーション時の波形の選択を行います.

プロジェクトを作成するには,まずはじめに Verilog-HDL ファイルをプロジェクト・ツリー・ウィンドウに加えます. Verilog-HDL ファイルをプロジェクトに加えるには以下の2種類の方法があります:

- プロジェクト・ツリー・ウィンドウ内で右クリックし,コンテキスト・メニュを開き,[Add File(s)...]を選択(図1).
- メニュの[Project]-[Add File(s)....]を選択(図2).
- どちらの操作でも、ファイル選択ダイアログが開くので、プロジェクトに加えたいファイルを選択して、「開く」ボタンをクリックしてダイアログを閉じます、そして、選択したファイルがプロジェクト・ウィンドウに表示されることを確認します、なお、プロジェクト・ウィンドウ内のファイル名をダブル・クリックすると、内蔵のエディタが起動し、Verilog-HDL ソース・コードを見ることができます(図3)、





図1 プロジェクト・ツリー・ウィンドウ



図3 sec_b.v ファイルを内蔵のエディタで表示した例

2) プロジェクトのビルド

以上で,プロジェクトに Verilog-HDL ファイルを加えることができましたが,このままではモジュール内の階層 構造を表示できません.プロジェクトに含まれるモジュール内部の階層構造をツリー状に表示するには,まず, ビルド,またはシミュレーションを行わなければなりません.ビルド・コマンドは Verilog-HDL ファイルをコンパイ ルし, Verilog-HDL ファイル内部のモジュールのツリーを作ります.ビルドでは,モジュールをコンパイルするの みで,シミュレーションは行いません.大きなプロジェクトでも,ビルド・コマンドを使うことで,シミュレーション・プ ログラムを実行しなくとも,短時間でモジュールのツリー構造を得ることができます.プロジェクトをビルドするに は以下の3種類の方法があります:

- シミュレーション・ボタン・バーにある黄色のビルド・ボタンをクリック.
- メニュの[Simulate]-[Build]を選択.
- <F7>キーを押す.

ー度プロジェクトがビルドされると、Verilog-HDL ファイル内のモジュール,信号,ポート、コンポーネント、すべての情報を得ることができます.あるモジュール名は<<モジュール名>のように表示されますが、これはプロジェクトのトップ・レベル・モジュールであることを示しています.このモジュールがプロジェクトの一番上のインスタンスであり、これに続くサブ・モジュールは、トップ・レベル・モジュールのツリーをたどることで見ることができます(図4).

プロジェクトをビルドした後,トップ・レベル・モジュールの信号,ポートは自動的に波形表示ウィンドウに追加されます.ここで,トップレベル・モジュールがポート信号を持っていない場合,モジュール内部の信号が表示されます(図 4).また,トップレベル・モジュールがポート信号を持っている場合,出力ポート信号が紫色で,入力ポート信号が黒色で表示されます.黒の入力信号は,テスト・スティミュラスをトップレベル・モジュールに入力するのに使われます.波形描画機能については,WaveFormer Pro マニュアルの一章,またはオンライン・マニュアルをご覧ください(WaveFormer Pro マニュアルは,本CD-ROM またはホーム・ページよりダウンロード可能).

Ø.: \	/eriLogger	Pro										- 🗆 ×
<u>F</u> ile	e E <u>x</u> port	<u>E</u> dit	<u>B</u> us	<u>L</u> ibraries	<u>P</u> roject	E <u>d</u> itor	<u>S</u> imulate	<u>R</u> eport	⊻iew	<u>O</u> ptions	<u>W</u> indow	<u>H</u> elp
	Auto Run		• И	D D D D D D D D D D D D D D D D D D D	NR.	S	<u>B</u> uild Bun	F7				7
		Bu	Id The	Active Proj	iect	T is a	Step Ov	er F10	É			
	Project -	nthee	nib) .			1.51.11	Step Into	5 F11				
	bject Hieran Ω∎ ⊖¥Pro	ony Daram	Files¥	VI ogger¥ser	av.	Directio	Pause					
1	i⊡ (12) rip	ple car	ry cou	v Losson +300 Inter	d.v		 Re <u>s</u> tart	F7				
	ēla	Ports	-						-			
		= q[3	:0]			output	vector	of wire				
1		⊡]clk ⊡°r∽o	 +			input	wire					
	ا ب	⊡ res Como	ei nnents			input	wire					
		Instan	tiation	IS								
	电 🍫 <	< stimu	ilus >>	»>								
	⊡l <u>@</u> I] ⇔Ma	FF .										
		Ports Bio				output	mire					
		≘iч Ēìclk				input	wire					
		🖹 res	et			input	wire					
	₽�	Signal	s									
	E S	Comp	onents	_								-
╘		Instan	tlation	15								ات.
10-1	Diagram –	untitle	d.tim		1		- 1		_			
Ad	id Signal . id Clock .	Add Bu Add Sp	is acer	Hold Tex	ip Sample (t Marker	нідн			Val Wi X Val		HEX €	Simulatio Built
9.	4.72ns	94.72	Ins ()ns	, 50n	s , ,	100r	ns I	15	iOns	200)ns
st	mulus.q[3	3:0]										
st	mulus.re:	set										
st	mulus.cl											
												-
•				•								
	Report - v	eril		🗵 🞇 Para	ameter – u	리미	\mathbf{X}					
SIML	lator is in auto s	sim mode i	auto simu	dates when input w	vaveforms chan	ge). Press to	switch to debug H	Hode.			Simulation	Built

図4 sec_a.vと sec_b.v Verilog-HDL ファイルからなるプロジェクトのビルド

スティミュラス波形の作成/編集機能や,作成したスティミュラスで即座にシミュレーションができる機能は,他の Verilog-HDL シミュレータにはない VeriLogger Pro 特有のものです.トップ・レベル・モジュールがポートを持っている場合,プロジェクトはトップレベル・モジュールに対するテスト・ベンチを自動的に取り込みます.また,トップ・レベル・モジュールの動作を見たり,ドライブするための信号を,そのテスト・ベンチ内に作成します.これにより,設計を完成させる前に設計の一部を即座にテストすることが容易になります.

3) 信号,コンポーネントを見る

VeriLogger Pro では,シミュレーション結果を表示するのに,2つのレポート出力があります.一つは, Verilog.log.ログ・ファイルで,シミュレータのメッセージ,Verilog-HDL ソースに記述された\$display ステートメント による出力などが書き出されます.もう一つは,波形表示ウィンドウ上で,デフォルトの状態では,トップ・レベル・ モジュールの信号またはポートのみが表示されます.トップ・レベル・モジュール内で実装されている特定のコ ンポーネントの(特定の)信号を表示するには以下のようにします:

- <<モジュール名>>表示のトップ・レベル・モジュールのプラス「+」印を左クリックし、ツリーを展開.
- 目的の信号,またはコンポーネントが見つかるまで,トップ・レベルのツリーを展開.
- 目的の信号,またはコンポーネント上で右クリックし,コンテキスト・メニュを開く.
- Watch Connection または Watch Component メニュ・オプションを選択 .これで(モジュールのフル・パスを 指定した)特定の信号を波形表示ウィンドウに追加できる(図 5).コンポーネントを選択した場合は,そのコ ンポーネントに属するすべての信号が,波形表示ウィンドウに追加される(図 6).

Descene variabilitie Add Start Add St	Zoom Zoom Ons
Add Spyse Add Spyse Mail	Zoom Zoom
Add Clock Add Space Hold Text	Zoom Ons
Tot.o.4/ns Dns ZUns HUns BUns TUDns TUDns <thtudns< th=""> <tht< th=""><th>Ons</th></tht<></thtudns<>	Ons
stimulus q[3:0] stimulus q[3:0] stimulus reset stimulus reset stimulus reset stimulus reset	
stmulus reset stimulus reset stimulus reset	
l stimulus cik	
stimulus n. tru. g	
summus r. u.s. q Simulas r. U.	
ammushinushinushinushinushinushinushinushin	-
stimulas / 182 d	
	-
Reprised = unititadity i Reprised = unititadity i Development Development Development	
Index interactive procession in the state of the state o	
Et@ ripple_carry_counter B ⊕ Ports Et@ ripple_carry_counter	
the Components	
Build from Life the test of te	
Dept wife U2 T_FF	
reset input wite Add File(s). Watch all batterese of Component and Veronment and Vero	nt a
B P Simple Specific Stating Port B Set Stating Port B Compared Stating Port B Compared Stating Port B Set S	
So titl TFF Event Tem - Bound to the second tem - Bound to the second tem - Bound to the second tem - Bound tem -	
the first transform the set of t	
B ♥ Instantiations B ♥ (instantiations)	
Only to per warder distance in residual tation Mandator MAT	

図5 インスタンス r1 内の tff0 の信号 q を選択

図6 インスタンス r1 内の tff3 の全信号を選択

4) プロジェクトのシミュレーション

プロジェクトのシミュレーションを実行するには以下の3種類の方法があります:

- シミュレーション・ボタン・バーにある,大きな緑色の Run ボタンをクリック(図7).
- メニュの[Simulate]-[Run]を選択.
- <F5>キーを押す.

プロジェクトに対してシミュレーションが行われると,波形表示ウィンドウにシミュレーション結果波形が表示されます.特定の信号だけを波形表示ウィンドウから見えないようにするには,その信号名の上を左クリックし, <Delete>キーを押します.

VeriLogger Pro は, AutoRun, DebugRun の2つのシミュレーション・モードを持っています.シミュレーション・ モードはシミュレーション・ボタン・バーの一番左側に表示されます. DebugRun モードでは, Run ボタン, または シングル・ステップ・ボタンがクリックされたときだけシミュレーションが実行されます(通常の Verilog-HDL シミュレータと同様の動作). AutoRun モードでは, 波形表示ウィンドウ内で波形が描かれたときや, 波形が変更されたときなど変更を加えたときは常にシミュレーションが自動的に実行されます. このモードを活用することで,小さなモジュールのテストを素早く行うことができ, ボトムアップ・テストを行うことができます. モード・ボタンをクリックすると, 2つのシミュレーション・モードを交互に変更することができます(図7).

🙅 VeriLogger Pro - [Diagra	am – untitled.tim]					
📇 Eile Export Edit B	<u>B</u> us <u>L</u> ibraries	<u>P</u> roject E <u>d</u>	itor <u>S</u> imulate	<u>R</u> eport <u>)</u>	⊻iew <u>O</u> ptions	<u>₩</u> indow <u>H</u> elp	
Auto Run 📴 💽	> >>> 🥶 🕅		s]∄s			Ŧ	
Add Signal Add Bus Run/P Add Clock Add Spacer	Resume Simulat Hold Text M	ion HIGH		L INVal w ⊐œ∎ ∽		X Simulation ▶ Good	Zoom In Zoom Full Zoom Out Zoom Range
25.09ns 25.09ns	Ons	20ns	40ns	60ns	80ns	100ns	120ns 140ns
stimulus.q[3:0]	0000	0001 0010	0011 0100	0101 0	0110 0111 1	000 1001 10	010 1011 1100 1101 1
stimulus.reset							
stimulus.clk							
stimulus.r1.tff0.q							
stimulus.r1.tff3.q							
stimulus.r1.tff3.clk							
stimulus.r1.tff3.reset							
stimulus.r1.tff3.d							
							•
▲ →							Þ
Simulate the active project							Simulation Good

図7 シミュレーション・モード・ボタンと Run ボタン

5) プロジェクトのデバッグ

ボタン・バー上と, 画面の右下にあるステータス・バーにはインジケータがあり(図 8), シミュレーションの正常 終了, 異常終了を示します. シミュレーションのエラー, またはコンパイルのエラーがボタン・バー上に表示され たときは, Verilog-HDL のソース・コードに何らかの誤りがあります.

シミュレーションに失敗したとき,その原因を見つけるためには,レポート・ウィンドウ内に表示される,次の二 つの情報を活用します.一つは Verilog.log で,実行したシミュレーションについてのすべての情報が出力され ています.もう一つは Errors タブに出力されている情報です.ここにはエラーと,その簡単な説明が出力されて います.Verilog-HDLファイルのエラー箇所にジャンプするには以下のようにします(図8):

- レポート・ウィンドウ内の Errors タブをクリックして, エラー・ウィンドウを開く.
- エラーの表示されている部分をダブル・クリック.すると,エラー箇所にカーソルが置かれた状態で,内蔵の エディタが開く.
- インタラクティブ・シミュレーションを使って, 論理代数式を Signal Properties ダイアログに入力したときなどは, 第3のファイル waveperl.log の参照もデバッグに役立つ.詳細はオンライン HELP を参照のこと.

eVeriLogger Pro	
jile Export <u>E</u> dit <u>B</u> us Libraries <u>P</u> roject E <u>d</u> itor <u>S</u> imulate <u>R</u> eport ⊻iew <u>O</u> ptions <u>W</u> indow <u>H</u> elp	
Debug Run 🐚 🕨 🕪 🕬 🔤 🔀 🕄 S 🛸 🎼	
🗜 Project – untitledhpj *	
Project Hierarchy Direction Type Information	
E w rope carry counter	
🕫 🔖 <<< stimulus >>>	
B TIFF	
m 🞸 CAPPOgram Files+VLogger+sec_LAV	
-Diagram - untitled tim	
Add Signal Add Bus Delay Setup Savok HIGH LINV/ TRL VAL INVALVEL WILD HEX Simulation Zoom In Zoom	Ful
Add Clock Add Spacer Hold Text Nexer 🔑 🖂 🦳 🖂 🛥 🖉 🕬 🕬 Error Zoom Out Zoom	Range
153.1ns 153.1ns Ons 20ns 40ns 160ns 180ns 1100ns 1120ns 1140	ns
timescale ins / ips	-
<pre>module ripple_carry_counter(q,clk,reset);</pre>	
output[3:0] q;	
T FF tff0(q[0],clk, reset);	
<pre>T_FF tff1(q[1],q[0], reset);</pre>	
Feport - Messages T_FF tff2(q[2],q[1], reset);	
Error in sec_av(b): parse error, expecting error or , or ; i I_r Clis(q[s], q[2], fese();	
Error in sec_a.v(6): parse error, expecting `error' or ``," or ``,"	
Error in sec_av(6) parse error, expecting 'error' or '" module T_FF(q, clk, reset);	
Error in sec av(0): (reset is not in the port list	
Error in sec.a.v(6): parse error, expecting 'error' or '." or '." vire d;	
<pre>D_FF dff0(q,d,clk,reset);</pre>	
not n1(d,q);	
verilge log (wevener log (Presknoint) From (untitled Tip	
INS Row: 6 Line: 6 Col: 2	-
Parameter - u. BOX	<u>в</u> Г./
aft olds and degrozzen in. Right olds to zoon out.	

図8 インジケータ, Errors タブと自動エラー・ロケーション機能

6) ブレーク・ポイント,シングル・ステップ,終了

VeriLogger Pro は, グラフィカルなブレーク・ポイントおよびシングル・ステップ実行, そして\$display などのテキスト表示のシステム・タスクをサポートしています. プレーク・ポイントを設定するには以下のようにします:

- プロジェクト・ウィンドウ上でファイル名をダブル・クリックするか,またはメニュの[Editor]-[Open HDL File]を 選択し, Verilog-HDL ファイルを内蔵エディタ上に開く.
- エディタ・ウィンドウ内の左側に黒色の帯(この部分がブレーク・ポイント・ウィンドウ)がある.
- ソース・コード上のシミュレーションを停止させたい部分(行)で、ブレーク・ポイント・ウィンドウを左クリックする、すると、赤い丸印がブレーク・ポイント・ウィンドウに加えられ、レポート・ウィンドウ内の Breakpoints タブ をクリックして開くブレーク・ポイント・ウィンドウ内には、指定したポイントが表示される(図9)、

Note:ブレーク・ポイントはエディタ・ウィンドウ内でそのブレーク・ポイントをもう一度クリックすることでキャンセル することができます.すなわちトグルします.

<mark>⊈ev</mark> VeriLogger Pro Eile Export <u>E</u> dit <u>B</u> us Libraries <u>P</u> roject E <u>d</u> itor <u>S</u> imulate <u>R</u> eport <u>V</u> iew Options <u>M</u>	X (indow <u>H</u> elp
Debug Run to → → → → → → → → → → → → → → → → → →	7
Report - Break Points Type File Name Source Line C#Program Files¥VLogger¥sec_av Source Line C#Program Files¥VLogger¥sec_av Source Line C#Program Files¥VLogger¥sec_av	
<pre>verilog.log / waveperLlog Breakpoints Errors / untitledTim.v /</pre>	X
<pre>bodule T_FF(q, clk, reset); output q; input clk, reset; wire d; D_FF dff0(q,d,clk,reset); not n1(d,q); endmodule</pre>	
INS Row: 5 Line: 12 Col: 28	Simulation Good

図9 ブレーク・ポイントの設定

シングル・ステップ・ボタンには, Step Over Calls と Step Into Calls の2種類があります(図10).

Step Over Calls ボタン中には,中くらいの大きさの三角形が二つあり,現在の階層レベル内で次の行を実行することを示しています.

Step Into Calls ボタンには,小さな三角形が二つの中くらいの三角形に挟まれて描かれています.これは,例 えばインスタンスされたモジュールなどが次の行に関わる場合は階層を潜り,そのモジュールの先頭1行を実 行することを示しています.ブレーク・ポイント近辺では,シングル・ステップ・ボタンが便利ですが,ループに入った場合は,Run ボタンを使うことで即座に次のブレーク・ポイントに移ることができます.

💁 VeriLogger Pro - [sec_a.v]					- 🗆 ×		
<u>Eile Export E</u> dit <u>B</u> us <u>L</u> ibraries	<u>P</u> roject E <u>d</u> itor	<u>S</u> imulate <u>R</u> eport	<u>∨</u> iew <u>O</u> ptions	<u>W</u> indow <u>H</u> elp _	_ 8 ×		
🛛 Debug Run 🔃 🕨 🕨 🕨	♀ S ₅	🔥 myclock =	1′ЬО;	•			
<pre>module ripple_carry_counter(q,clk,reset); output[3:0] q; input clk, reset; T_FF tff0(q[0],clk, reset);</pre>							
INS	Row: 4 Lin	ne: 5 Col: 2	:		-		
				Simulation Sto	opped		

図10 シングル・ステップ・ボタンと現在実行中の行の表示

このほかに,デバッグを支援する三つのボタンがあります(図11).

- Stop ポタンは,シミュレーションを一時中断する.
- (黒色のダイオードのように見える) Restart ポタンは現在のシミュレーションを中止し,波形表示ウィンドウをクリアする.そして, Run またはシングル・ステップ・ボタンがクリックされたときにプロジェクトのはじめから 実行されるよう,シミュレーションをリセットする.
- (虫眼鏡のような)Goto ポタンは,エディタ・ウィンドウを開き,現在シミュレーションを行っている Verilog-HDL ソース・コードを表示する.

VeriLogger Pro は Verilog-HDL の\$display などテキスト出力のシステム・タスクもサポートしています.これらの命令で出力された結果は,直接ログ・ファイルに書き込まれ,レポート・ウィンドウで確認することができます(図11).

💁 VeriLogger Pro							
<u>File Export Edit Bus</u>	Libraries <u>P</u> roject Editor <u>S</u> imulate <u>R</u> eport <u>V</u> iew <u>O</u> ptions <u>W</u> indow <u>H</u> elp						
Debug Run 🔃 🕨 🕨	♦ ▶▶ <mark>ﷺ ا ♀</mark> S ≈ ₩						
🕂 Diagram – untitled.	tim 💶 🖂						
Add Signal Add Bus Add Clock Add Spacer	Delay Setup Sample HIGH LOW TRI VAL INVal WHI HEX Simulation Zoom In Zoom Full Hold Text Marker Image Image Image Image Image Stopped Zoom Out Zoom Range						
44.29ns 37.38ns	0ns 20ns 40ns 60ns 80ns 100ns 120ns 140ns						
stimulus.q[3:0]	0 <u>1 1 2 3 1 4 5 1 6 7 1 8 9 1 A 8 7 C 0 6</u>						
stimulus.reset							
stimulus.clk							
× •	× •						
Report - verilog.log							
L14 "C:\VLogger\sec_a	.v" (syncad_top.stimulus.r1): ASSIGN clk >>> clk = 1'h1, 1;						
** Triggering Continuous	s Assignment:						
L6 "C:\VLogger\sec_a.v	/" (syncad_top.stimulus.r1.tffD): ASSIGN clk >>> clk = 1'h1, 1;						
L6 "C:\VLogger\sec_b.v" (syncad_top.stimulus.r1.tftD.dftD): WIRE clk >>> NET = 1'h1, 1							
INS	Row: 6 Line: 826 Col: 1						
verilog.log / waveperl.log / Breakpoints / Errors / untitledTim.v /							
🎇 Parameter 🗗 🗖	🔀 隆 Project – u 🗗 🗆 🔀						
	Simulation Stopped						

図11 デバッグ支援ボタン群とレポート・ウィンドウ

7) プロジェクト, ソース・コード, 波形ファイルの保存

VeriLogger Pro には, Verilog-HDL プロジェクトと関係する三つのファイルがあります.

- プロジェクト・ファイルは, HPJ の拡張子を持ち, メニュの[Project]-[Save HDL Project]で保存できる. これにより, ファイル・リスト, シミュレーション・オプションが保存される.しかし, プロジェクト・ファイルを保存 しても, 波形表示信号の設定などは保存されない.
- Verilog-HDL **のソース・コード・ファイル**は, V の拡張子を持ち, エディタ・ウィンドウがアクティブの状態で, メニュの[Editor]-[Save HDL File]を選択して保存できる.
- <u>波形ダイヤグラム・ファイルは</u>, TIM の拡張子を持ち,メニュの[File]-[Save]を選択して保存できる.
 このファイルには,表示された信号の波形情報が記録されている.プロジェクト・ファイルとは別のファイル
 に信号の波形情報を保存することで,さまざまなテスト・ケースを作成することが可能で,これにより,ほか
 のシミュレーション結果と比較,検討することができる.

プロジェクト・ウィンドウ内での操作一覧

- Verilog-HDL ファイルを開く: ファイル名をダブル・クリックするとエディタ・ウィンドウが開く.
- 信号, コンポーネントの宣言を見る: 階層構造ツリーで信号名, またはコンポーネント名をダブル・クリックすると, Verilog-HDL ソース内の対応するオブジェクトにジャンプする.
- **階層構造ツリーをすべて展開する**: ツリー部分を右クリックするとコンテキスト・メニュが開くので, Expand Item メニュを選択(図 12).
- プロジェクトの保存: メニュの[Project]-[Save HDL Project]を選択.
- プロジェクトを開く: メニュの[Project]-[Open HDL Project...]を選択.

🖉 VeriLogger Pro - [Project - untitled.hpj *]								
<mark>]≵.</mark> Eile Export <u>E</u> dit <u>B</u> us Lib	raries <u>P</u> roject E <u>d</u> itor	<u>S</u> imulate	<u>R</u> eport	<u>V</u> iew	<u>O</u> ptions	<u>W</u> indow	Help _ B×	
Auto Run 🔃 🕨 ▶▶ 🚥 🖌 🔍 🛛 S 💈 ½S								
Project Hierarchy Direction Type Information								
C#VLogger¥sec_a.v								
	<u>W</u> atch All Instances of <u>W</u> atch All Instances of	f Componen f Componen	t t and Sub	-compc	onent			
🗄 🎨 C.¥VLogger¥sec_b.v	<u>A</u> dd File(s) <u>S</u> et Starting Point <u>R</u> eset Starting Point	コンテ	キスト	• X=	- <u>-</u>			
	Expand Item							
	•					Smula	ion Buik	

図12 コンテキスト・メニュ

シミュレーション時の操作一覧図13)

- ビルド: シミュレーション・バーにある, 黄色のボタンをクリックすると, Verilog-HDL ファイルをコンパイル し, Verilog-HDL ツリーを作成する. ビルド実行時シミュレーションは行なわれない.
- 実行/再開 : シミュレーション・バー上,緑色の大きな三角形のボタンをクリックすると、Veriog-HDL ファ イルをコンパイルし、シミュレーションを実行する.設定されているブレーク・ポイントでシミュレーションが停 止している時にこのボタンをクリックすると、シミュレーションを続けることができる.
- シングル・ステップ Step Over Calls : ボタンに,緑色で三角形が2つ描かれている. シングル・ステップ Step Into Calls : ボタンに,緑色で三角形が3つ描かれている.

🙅 VeriLogger Pro - [Pro	ject – i	untitled.hpj >	*]					
<u>Eile Export E</u> dit	<u>B</u> us	<u>L</u> ibraries	<u>P</u> roject	E <u>d</u> itor				
Auto Run	DD D0	Þ 💷 🚺	R	S s				
ビルド	\Box							
実行/再開	-	ノノクリ Stop b	レ・スフ いちゃ へ	ロッノ				
j	1	Step I		ans				
シングル・ステップ								
Step Over	Calls	S						
図13 シミュレーション時に使う	ラボタン群	ŧ						

エディタの機能

- **ブレーク・ポイントを** Verilog-HDL **コードに設定する** : エディタ・ウィンドウの左側にある黒い帯の上をクリックすると, ブレーク・ポイントを示す赤い丸印が加えられる.また, 設定したブレーク・ポイントが, レポート・ウィンドウ内の Breakpoints タブをクリックして開くブレーク・ポイント・ウィンドウ内に追加される(図9).

<u>コマンド入力の機能</u>

VeriLogger Pro は,シミュレーションを監視,制御,デバッグするために,Verilog-HDL コマンドを直接入力で きる,対話型コマンド・コンソールを装備しています.コマンド・コンソールはドロップ・ダウン・エディット・ボックス の形をし,シミュレーション・ボタン・バーにあります(図 14).シングル・ステップ・シミュレーションを実行中に, \$finish のようなシミュレーションの制御コマンドや,\$display のような変数を表示させるコマンドを入力すること ができます.

なお,入力できるコマンドは Verilog-HDL 標準のコマンドに加え,シミュレーションを制御するためのさまざまな VeriLogger Pro 特有のコマンドがあります.

(例)

- **ピリオド**(..) : シミュレーションを続ける.
- セミコロン(;): Verilog-HDL コード内の次のステートメントに進む.
- コンマ(,) : 次のステートメントに進み, さらにトレース・メッセージを Verilog..log ファイルに書き出す.
- コロン(:) : 現在実行されているソース・コードの行を表示する.

💁 VeriLogger Pro							
<u>File Export Edit Bus Libraries Project Editor Simulate</u>	Report View Options Window Help						
Debug Run 🔃 🕨 🕪 🚥 🖊 🔍 🛛 S 🛭 🕵	stimulus.reset = 1'b1;						
🖶 Diagram – untitled.tim	timulus.reset = 1'b0;						
Add Signal Add Bus Delay Setup Sample HIGH LOW T Add Clock Add Spacer Hold Text Marker HIGH LOW T	Zoom In Zoom Full Zoom Out Zoom Range						
58.81ns -164.7ns Ons 100ns \$	settrace; 00ns 1500ns						
stimulus.q[3:0] 0 1 2 3 4 5 0 4	inno seton:						
stimulus.reset	Schowvars;						
stimulus.clk	Schowscopes;						
арана (1997) Тарана (1997) Тарана (1997)	scope(syncad_top); nyclock = 1'bil:						
\$	odisplay("myclock=b",myclock);						
Report - verilog.log	@(posedge myclock) \$stop; _□⊠						
。 L8 "C:\VLogger\sec b.v" (syncad top.stimulus.r1.tffD.dffD); ALWAYS							
stimulus.reset = 1'b1;	g(POSEDGE feset OR NEGEDGE Cik)						
コマンド入力の機能を使って,reset信号を強制的に変更してシミュレーションを行った							
INS Row: 4 Lir	ne: 939 Col: 1						
	`						
verilog.log / waveperl.log / Breakpoints / Errors / untitledTim.v /							
🔁 Project – u 🗗 🗆 🔀 Parameter 🗗 🗆 🗙							
	Simulation Stopped						

図14 コマンド入力によるシミュレーションの制御

かんたんな操作例

簡単な既存の Verilog-HDL コード sec_a.v, sec_b.v で, シミュレーションを実行してみましょう.

- 0. VeriLogger Pro を c:¥Vlogger ディレクトリにインストールします.
- 1.メニュの[Project]-[New HDL Project]をクリックします.
- 2 . メニュの[Project]-[Save HDL Project As ...]をクリックして, プロジェクト保存のダイアログを開き, sec_ab と入力して, c:¥vlogger にプロジェクト・テンプレート sec_ab.hpj を保存します.
- メニュの[Project]-[Add File(s)...]をクリックして、シミュレーションを行う Verilog-HDL ファイルをプロジェクトに登録するダイアログを開き、c:¥vlogger ディレクトリ内の sec_a.v、sec_b.vの両ファイルを登録します(順番は任意).
- 4. モードが Debug Run になっていることを確認して, Build ボタンをクリック(またはメニュの[Simulate]-[Build] をクリック)して, 階層のビルドとコンパイルを行います.
- 5 . 緑色の大きい三角の Run ボタンをクリックすると,シミュレーションが実行されシミュレーション結果波形 が表示されます.
- Zoom In/Zoom Out/Zoom Full/Zoom Range ボタンなどで, 波形表示を拡大/縮小して見やすくします.
 時間軸上で, 見たい観測時間の左端をクリックして, 右端までドラッグして離すと, 指定期間がウインドウいっぱいに広がるので, これでも見やすさを調整できます.
- 7. VeriLogger Pro を使っての詳しいデバッグ方法は,この資料の他の項やオンライン・ヘルプを参考にして ください.

VeriLogger Pro のお問い合わせ先

CQ 出版株式会社 デザインウェーブ企画室

TEL:03-5395-5673 FAX:03-5395-2127

E-mail: <u>edasupport@cqpub.co.jp</u> URL:http://www.cqpub.co.jp/