

VC Formal Lab

Sequential Equivalence Checking (SEQ) App Setup and Standard Usage

Learning Objectives

In this lab, you will use the multiplier example to learn to do the following:

- Setup compile two designs spec and impl
- Mapping of signals
- Set up clocks and resets
- Establish initial state for formal
- Run checks
- Debug failure

Familiarity with the SVA assertion language and know about of the basic formal verification concepts are required for this lab.



Lab Duration:
30 minutes

Testcase Location

All files for this VC Formal lab are in directory:

`$VC_STATIC_HOME/doc/vcst/examples/SEQ/`

Directory Structure	
SEQ	Current working directory
README_SEQ_setup.pdf	Lab instructions
design/	Verilog RTL code of the multiplier design
run/	Run directory
solution/	Solutions for the lab

Resources

The following resources are available for in-depth guidance regarding VC Formal usage, commands, and variables.

VC Formal User Guide:

`$VC_STATIC_HOME/doc/vcst/VC_Formals_Docs/VC_Formals_UG.pdf`

VC Formal Apps Quick References Guides:

`$VC_STATIC_HOME/doc/vcst/VC_Formals_Docs/Quick_Reference_Guides/`

VC Formal Apps Tcl Templates:

`$VC_STATIC_HOME/doc/vcst/VC_Formals_Docs/Quick_Reference_Guides/vcf_tcl_templates/`

Prepare your Environment

1. Load VC Formal and Verdi to set up the tool and required licenses. For example:

```
% module load vcstatic
% module load verdi
```

2. Change your working directory to run

```
$> cd run
```

Now you are ready to begin the lab.

Create a run.tcl file for SEQ setup

VC Formal has a tcl based command interface. The most common way is to start with a tcl file to setup and compile the design. At this step, user will create a template file for the multiplier design used in this Lab.

The design files are under design/ directory.

1. Open a new file run.tcl (any arbitrary name is ok to use) using an editor. e.g.

```
% vi run.tcl
```

2. Use tcl variable to switch to SEQ app mode:

```
set_fml_appmode SEQ
```

3. Enter following commands to compile spec and impl design:

```
analyze -format sverilog -library spec ../design/rtl1.v
analyze -format sverilog -library impl ../design/rtl1cg_bad.v
elaborate_seq -spectop mul -impltop mul
```

4. Use following command to map signals. It will create assert properties for top module outputs, blackbox inputs, abstracted points and assume properties for top module inputs, blackbox outputs, undriven nets.

```
map_by_name
```

VC Formal can be run in 3 modes: Interactive GUI mode, Interactive using vcf shell without GUI and batch mode.

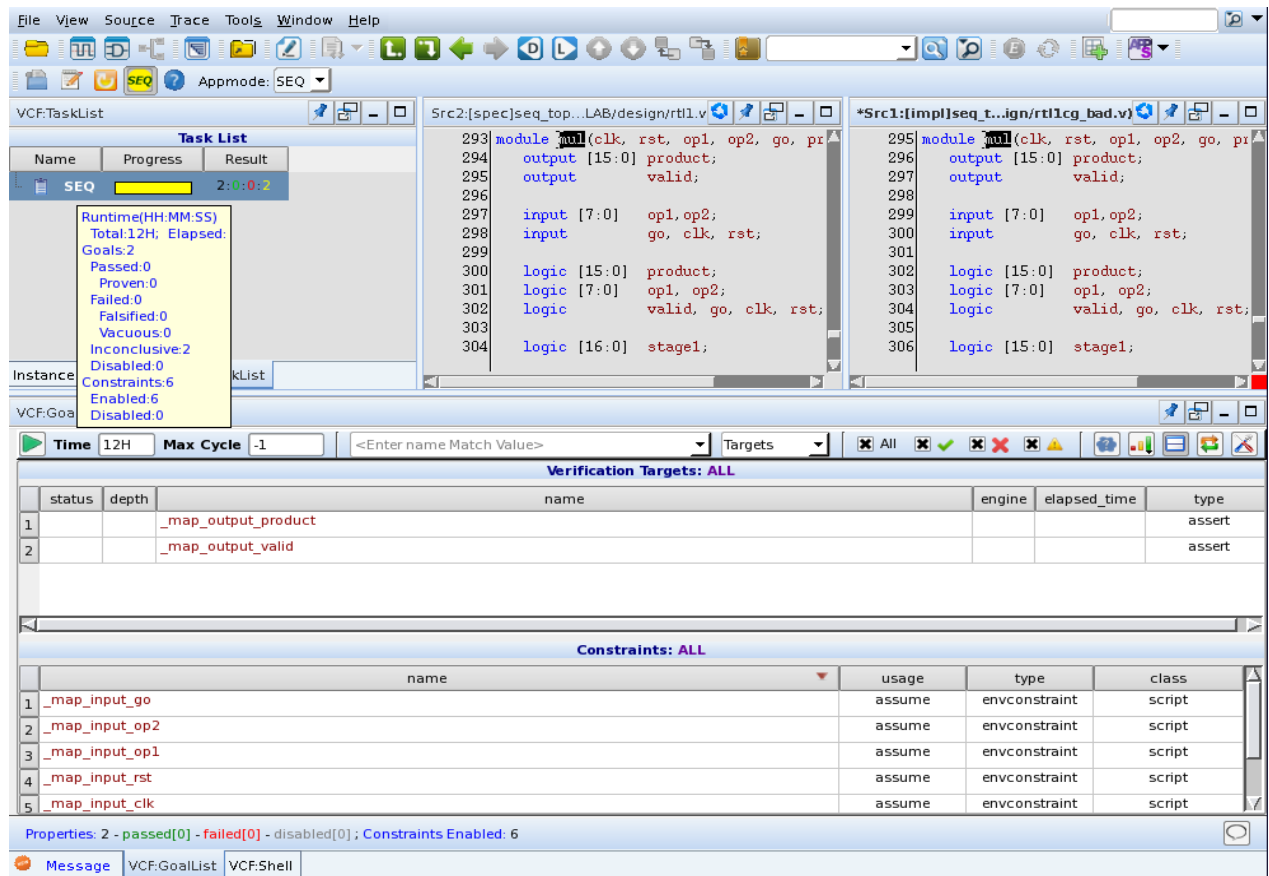
Mode 1 : Start VC Formal in GUI mode


5. Start the tool in GUI mode:

```
%vcf -verdi -f run.tcl
```

The GUI starts in the VC Formal mode, with icons, tables, tabs and windows especially designed for property verification.

The Appmode is by default set to FPV. Using SEQ app mode variable it will start in SEQ mode. Initial configuration is shown with VCF Task List tab on the top left, spec and impl source windows on the top right, VCF Goal List table on the bottom, and the VCF Shell cascaded on the bottom.



- Get familiar with the Verdi instance and source file tabs, properties to be checked under “Verification Targets: ALL” table as well as properties specified as constraints in the “Constraints: ALL table”. Customize the column if desired by clicking on  at the upper right of the property table.

Revise Setup and Review Initial State

- Add the clock and reset setup information to the tcl file:

Click on the Edit Tcl Project File icon  on the upper left.

Click on Edit to activate editing.

Add the following commands to the tcl file

```
create_clock spec.clk -period 100
create_reset spec.rst -sense low
```


Add commands to set up reset state by hold reset active until design is stable


```
sim_run -stable
sim_save_reset
```

8. Add commands to map uninitialized registers and x-assignments.


```
seq_config -map_uninit -map_x zero
```


9. Save edited run.tcl file: Click on .


10. Restart VCST: click on  to restart

11. Start property check: Click on  run to finish. This takes about 20-30 seconds.

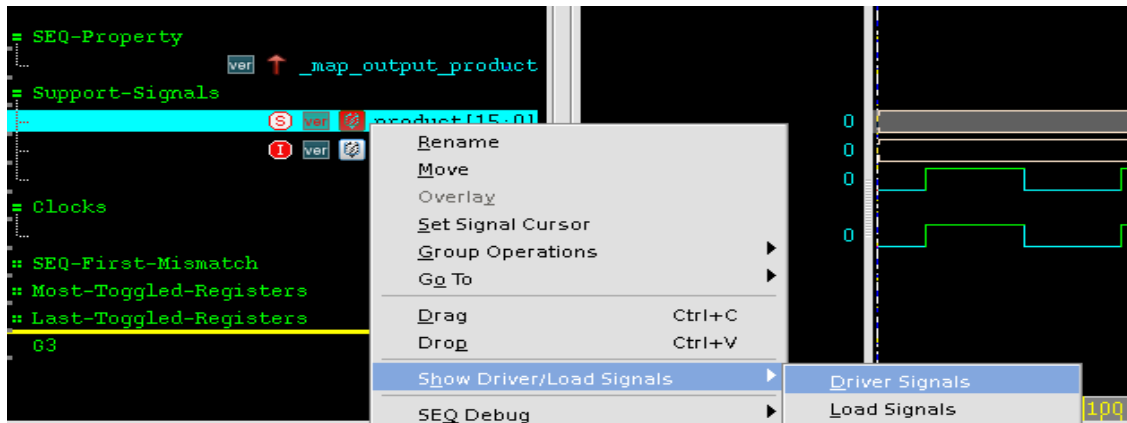
Debug Failures

12. Double click on  under the status of the property “_map_output_product” to start waveform. The failure occurred at 5th cycle after reset. Signal from spec design is shown

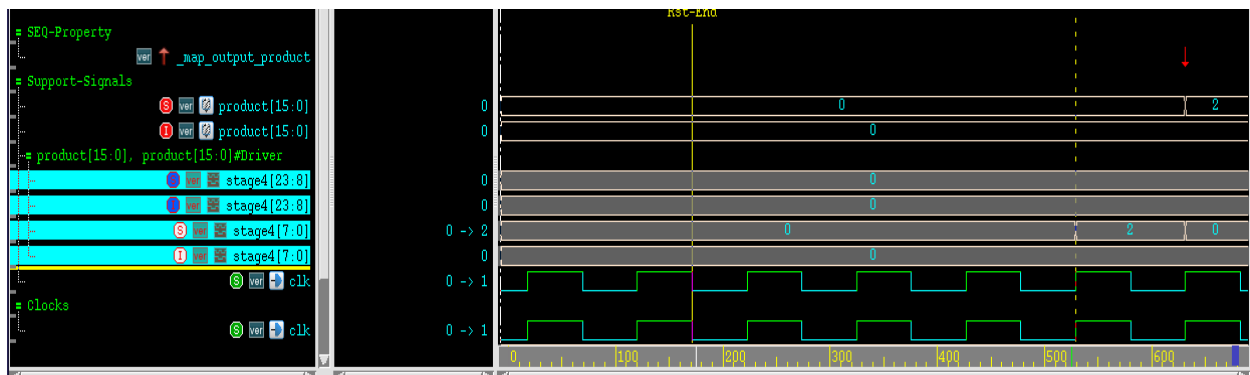
with “S” letter and impl with “I” letter. Red color  means mismatch signals and green

color  means no-mismatch.

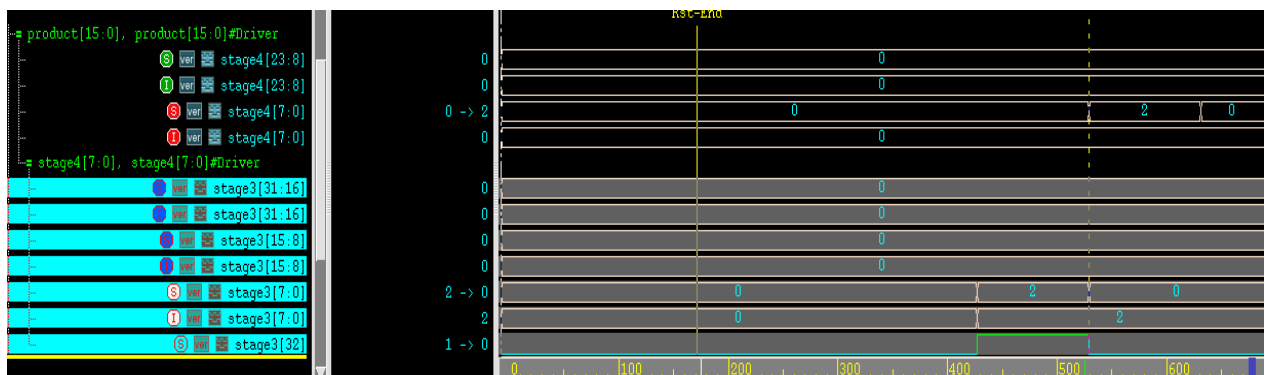
13. Right click on product[15:0] and select “Show Driver/Load signals -> Driver Signals”.



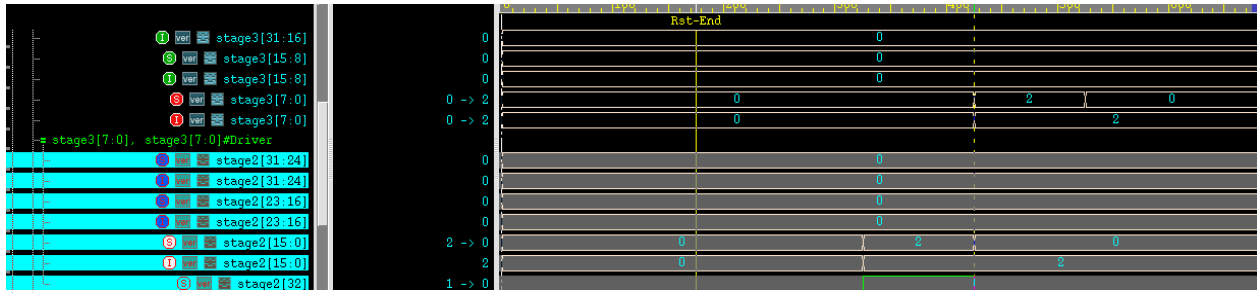
14. There is a mismatch between stage4[7:0] of spec and stage4[7:0] of impl design.



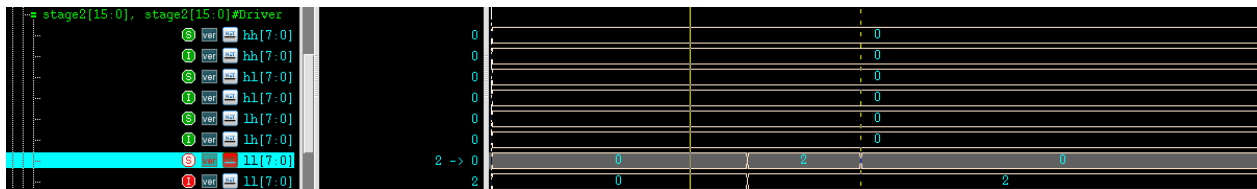
15. Right click on stage4[7:0] signal and select “Show Driver/Load signals -> Driver Signals”.
There is a mismatch between stage3[7:0] of spec and stage3[7:0] of impl design.




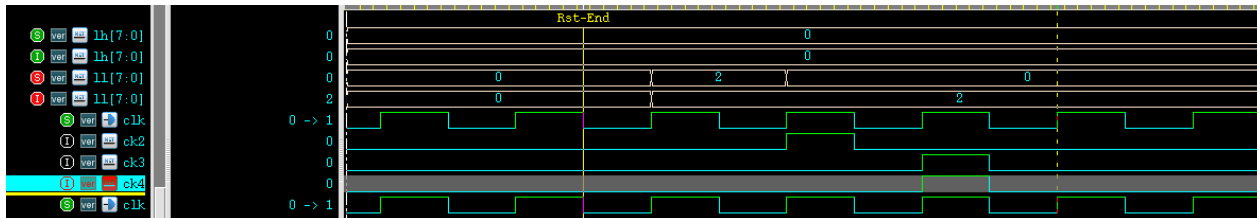
16. Right click on stage3[7:0] signal and select “Show Driver/Load signals -> Driver Signals”.
There is a mismatch between stage2[15:0] of spec and stage2[15:0] of impl design.



17. Right click on stage2[15:0] signal and select “Show Driver/Load signals -> Driver Signals”. There is a mismatch between ll[7:0] of spec and ll[7:0] of impl design.





18. Now search “ck” in verdi search pane  and select “ck2, ck3 and ck4”. Press ctrl+w to add these signals to the waveforms.
19. Issue is that clock ck4 should toggle next cycle after clock ck3 has toggled, but it is toggling at the same cycle as ck3.



20. Modify following line in the rtl file rtl1cg_bad.v. Replace go2 with go3.
Line: 318 cg cg4(clk, go2, ck4); // Error inserted here; go2 should be go3

Restart the Run and Verify Fix

21. Restart VCST: click on  to restart
 22. Click on  to rerun
- Observe that the same property is now proven.

Mode 2: Run interactively in vcf shell without GUI

23. Invoke without GUI

```
%vcf -f run.tcl
```

Enter run command in VCF and query


```
vcf> check_fv
vcf> report_fv
```

Or enter run command with callback task

```
vcf> check_fv -run_finish {report_fv -list > results.txt}
```

To debug failures, it is recommended to start the GUI. You can do that from within the vcf shell

```
vcf> start_gui
```

Please note that the VCF Shell panel will not be available in the GUI. Any tcl command will need to be entered from the vcf> prompt where you launched “start_gui”. Also, the debug waveform may not be embedded in the same window as the you have seen before. You can always click on  to dock or unlock a window.

Mode 3: Setup and Run Batch or Regression

24. Make a copy of run.tcl to batch.tcl
25. Edit batch.tcl and add commands to run and save results:

```
check_fv -block
report_fv -list > results.txt
```

26. Invoke to run batch

```
%vcf -f batch.tcl -batch
```