

VC Formal

Formal Coverage Analyzer (FCA) App

Lab 2: Unreachability Analysis Continued

Learning Objectives

In this VC Formal lab, you will use a traffic light controller example to learn to do the following:

- Set up design
- Run interactively with Verdi GUI
- Run FCA on specified coverage goals
- Generate coverage database
- Generate exclusion file for items found
- View coverage in Verdi Coverage GUI
- Save and restore session
- Run in interactive shell without Verdi GUI
- Run in batch mode



Lab Duration:
30 minutes

Familiarity with VCS coverage, knowledge of the basic formal verification concept and basic setup using VC Formal are required for this lab

Files Location

All files for this VC Formal lab are in directory:

`$VC_STATIC_HOME/doc/vcst/examples/FCA/FCA_without_SimCovdb/`

Directory Structure	
FCA_without_SimCovdb	Lab main directory
README_VCFormal_FCA_Lab2.pdf	Lab instructions
design/	Verilog RTL code of the Device Under Test (DUT)
run/	Run directory
solution/	Solution directory

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_Forma Docs/VC_Forma_UG.pdf`

VC Formal Apps Quick References Guides:

`$VC_STATIC_HOME/doc/vcst/VC_Forma Docs/Quick_Reference_Guides/`

VC Formal Apps Tcl Templates:

`$VC_STATIC_HOME/doc/vcst/VC_Forma Docs/Quick_Reference_Guides/vcf_tcl_templates/`

Prepare your Environment

1. Set environment variable pointing to your VC Formal installation directory:

```
%setenv VC_STATIC_HOME /tools/synopsys/vcstatic
```

2. Add path \$VC_STATIC_HOME/bin to the PATH environment variable.
3. Change your working directory to FCA/FCA_without_SimCovdb/run:

```
%cd FCA/FCA_without_SimCovdb/run
```

Now you are ready to begin the lab.

Create a run.tcl Setup File

VC Formal has a Tcl-based command interface. It is common to start with a Tcl file to set up and compile a design. In this step, you will create a VC Formal Tcl file for the DUT, a traffic light controller, used in this lab.

4. Open file run.tcl (any arbitrary name is ok to use) using any text editor:

```
%vi run.tcl
```

5. Add command to enable FCA App mode:

```
set_fml_appmode COV
```

6. Specify DUT top level module name as Tcl variable:

```
set design traffic
```

7. Add command to compile DUT and enable automatic code coverage extraction:

The DUT files and filelist are located under directory FCA_with_SimCovdb/design.

```
read_file -top $design -format sverilog \
  -vcs {-f ../design/filelist} -cov all
```

Note: Switch `-cov all` instruments the RTL for line, condition, fsm_state, fsm_trans and toggle coverage metrics. To enable a single coverage metric, specify `-cov <metric>`. To specify multiple coverage metrics, specify `-cov <metric1 + metric2>`, e.g., `-cov line` or `-cov line+cond`.

Note: To use unified usage model to compile design, use these commands instead of

read_file to compile design and SVA properties:

```
analyze -format sverilog \  
-vcs {-f ../design/filelist}  
elaborate $design -sva -cov all
```

In this lab, since we did not import a simulation coverage database, Formal Coverage Analyzer will work on all the extracted coverage goals.

8. Save run.tcl file and exit editor.

VC Formal can be run in three modes: interactive Verdi GUI mode, interactive without Verdi GUI using shell mode, and non-interactive batch mode.

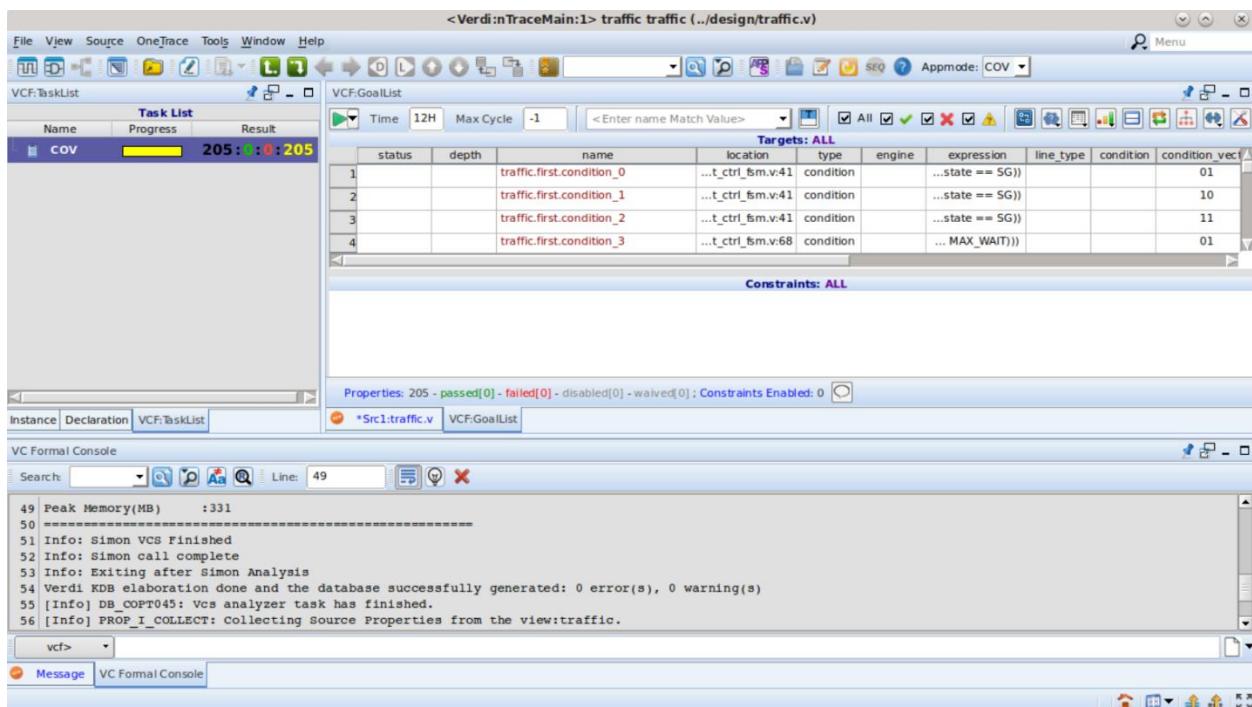
Mode 1: Start VC Formal in Verdi GUI Mode

- Start the tool in Verdi GUI mode:

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


VC Formal starts in the Verdi GUI mode, with icons, tables, tabs, and windows especially designed for verification with the FCA App.

Initial configuration is shown with the “VCF:TaskList” tab on the top left, the “VCF:GoalList” tab on the top right, and the “VC Formal Console” shell at the bottom.




- Get familiar with the Verdi GUI instance:

Check the source file tabs, properties to be checked under the “Targets: ALL” table as well as properties specified as constraints in the “Constraints: ALL” table.

If desired, customize the columns shown by clicking on the Customize View Settings icon  at the upper right of the property table.

Review Design Information and Setup

11. Review design information:

Click on the Show Complexity icon  on the upper right above the property table. Examine items under “Missing Clock” and “Missing Reset” and trace from the source file to see that the active phase of rst is high.

Revise Setup and Review Initial State

12. Add missing clock and reset information to the Tcl file:

Click on the Edit Tcl Project File icon  on the upper left and add the following commands to the Tcl file.

```
create_clock clk -period 100
create_reset rst -sense high
```

13. Add design initialization commands:

```
sim_run -stable
sim_save_reset
```

These commands initialize the DUT by holding reset active until sequential elements (latches and flip flops) values are stable.


14. Save edited run.tcl Tcl file:

Click on the Save icon .

15. Restart VC Formal:

Click on the Restart VCST icon .

16. Check setup and debug initial state:

Click on the Show Complexity icon  on the upper right to see design information and confirm there are more missing clock and reset shown.

Examine the initial state of latches and flip flops to check if anything is unexpected.

17. Exit VC Formal:

Click on File → Exit.

Run Formal Proofs and Review Results

18. Start VC Formal with existing Tcl file:


Now that you have a correct setup, start VC Formal pre-reading the existing run.tcl file.

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

19. Enable trace generation for covered goals:

```
vcf> set_fml_var fml_cov_gen_trace on
```


20. Start property verification:

Click on the Start Check icon .


21. Hide the constraints table:

Click on the Targets+Constraints icon  at the top right.

22. Filter “Targets” table to keep uncoverable targets:


Select to view only uncoverable targets .

23. Filter “Targets” table to keep covered targets:

Select to view only covered targets .

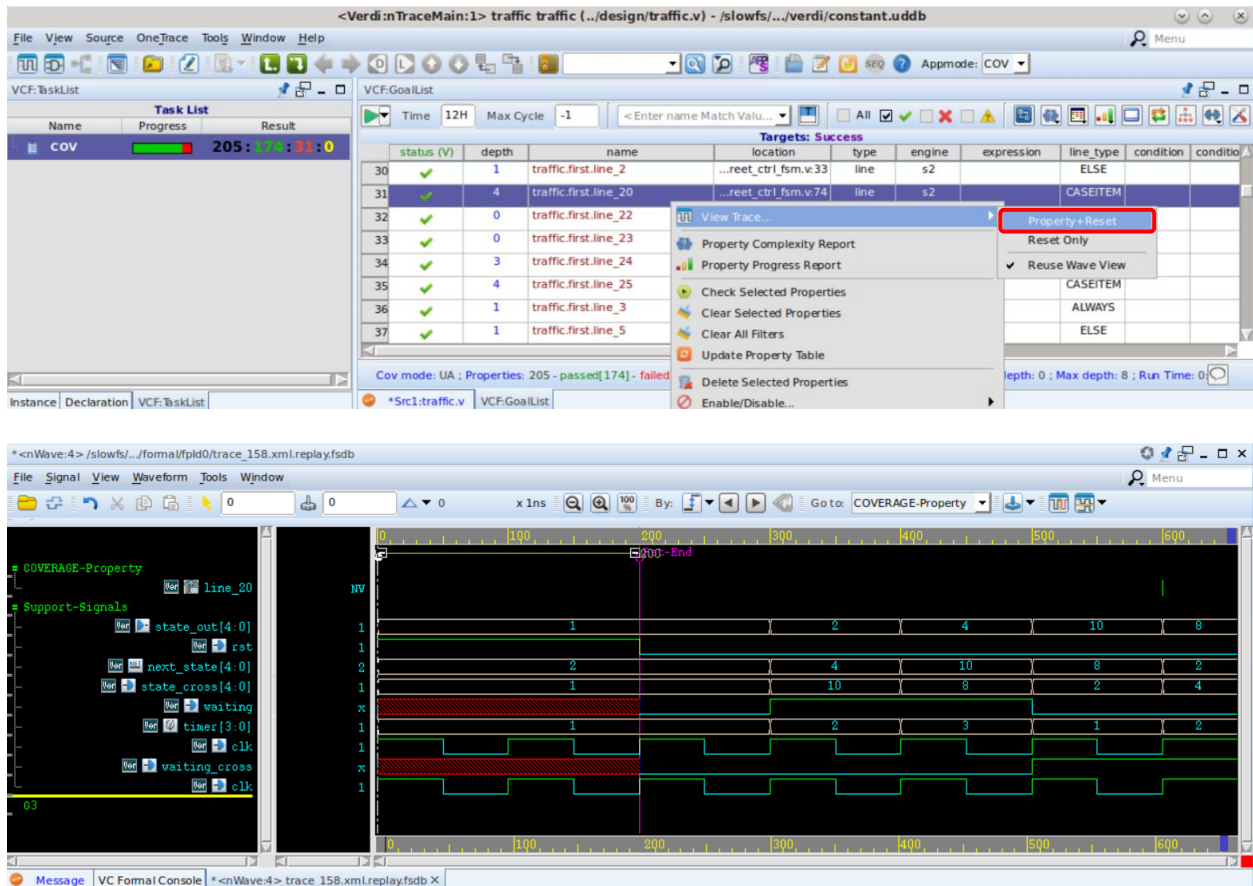
View Trace for Covered Targets

24. View trace for covered target:

Double click on  under the status of property “traffic.first.line_20” to open a waveform showing how the condition was covered.


(Note that a double click on the “name” of a property will open the source code window instead.)

Alternatively, open a trace using the right-click menu: View Trace → Property+Reset



View Coverage in Verdi Coverage GUI

25. Open coverage database in Verdi Coverage GUI:

Click on  to dump a coverage database with FCA results and an elfile (exclusion file) for the uncoverable targets. The Verdi Coverage GUI automatically comes up with the coverage and exclusion file information.

Save Session

26. Save session using default name from the VC Formal Shell:

```
vcf> save_session
```

Alternatively, click on File → Save Session... to open the “Save Session” dialog window.

27. Exit VC Formal:

Click on File → Exit

Restore Session

28. Start VC Formal using previous saved session:

```
%vcf -restore -verdi
```

29. Review setup and results then exit:

Click on File → Exit

Mode 2: Start VC Formal in Interactive Non-Verdi GUI Mode

30. Invoke VC Formal without Verdi GUI:

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

31. Enable trace generation for covered goals:

```
vcf> set_fml_var fml_cov_gen_trace on
```

32. Enter run check command in VC Formal Shell:

```
vcf> check_fv
```

33. When check completes, report results, save coverage database and exclusion file:

```
vcf> report_fv -list > results.txt
vcf> save_covdb -name cov -status covered -overwrite
vcf> save_cov_exclusion -file unr.el
```

Alternatively, enter run check command with callback task:


```
vcf> check_fv -run_finish {
report_fv -list > results.txt
save_covdb -name cov -status covered -overwrite
save_cov_exclusion -file unr.el
}
```

34. Start the Verdi GUI from within the VC Formal Shell:

```
vcf> start_verdi
```

To analyze traces of covered goals, it is recommended to use the Verdi GUI.

Note that the VC Formal Shell panel will not be available in the Verdi GUI. Any Tcl command will need to be entered from the `vcf>` prompt where you used the `start_verdi` command.

Also, the debug waveform may not be embedded in the same window as before. You can always click on  to dock (or undock) a window.

35. View coverage report from within the VC Formal Shell:

```
vcf> view_coverage -cov_input cov.vdb -elfile unr.el
```

36. View trace of covered goal “traffic.first.line_20” from within the VC Formal Shell:

```
vcf> view_trace -property traffic.first.line_20
```

Optionally, add switch `-composite` to show the reset phase in the trace.

37. Exit VC Formal:

```
vcf> quit
```

Mode 3: Set Up and Run VC Formal in Batch (Regression) Mode

38. Copy Tcl file run.tcl to run_batch.tcl:

```
%cp run.tcl run_batch.tcl
```

39. Edit run_batch.tcl and enable trace generation for covered goals:

```
set_fml_var fml_cov_gen_trace on
```

40. Add commands to run and save results:

```
check_fv -block  
report_fv -list > results.txt  
save_covdb -name cov -status covered -overwrite  
save_cov_exclusion -file unr.el
```

41. Add command to save session:

```
save_session -session batch_results
```

42. Save run_batch.tcl file and exit editor.

43. Start VC Formal in batch mode with switch -batch:

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

Note that in batch mode VC Formal exits automatically after the execution of the Tcl command file.