SYNOPSYS®
Silicon to Software™

VC SpyGlass Lint

Early Design Analysis for IP and SoC Signoff

May 2022

Agenda

- **Technical Overview**
- Lint Check flow
  - Running Lint Checks
  - Results Analysis
  - Creating Waivers

Lint –Quick Coding Style Check without Test Case

Wrong RTL Code

```
17 wire [1:0] hresp_flat ;
18 reg        lcyc_o ;
19 reg        sstb_o ;
20 reg        hwrite_flag;
21 wire [2:0] hburst;
22 wire [2:0] hsize;
23 wire [1:0] htrans;
24 wire [2:0] hlong;
25 wire [1:0] hshort;
26
27 always @ (posedge hclk or negedge hresetn ) begin
28   if (hresetn) begin
29         hresp_flat  <= 2'b00;
30         lcyc_o      <= 1'b0;
31         hwrite_flag <= 1'b0;
32   end
33   else if(hlong < hshort) begin
34         case (htrans)
35               1'b0 :  begin
36                     hresp_flat <= 2'b10;
37                     lcyc_o     <= 1'b0;
38                     sstb_o     <= 1'b1;
39               end
40               1'b1 :  begin
41                     hresp_flat <= hburst && hsize ;
42                     lcyc_o     = 1'b1;
43                     sstb_o     <= 1'b0;
44               end
45         endcase
46   end
47 end
```

IBLHS-NT

STARC05-2.3.1.6

InferLatch

W362

W263

ResetFlop-ML

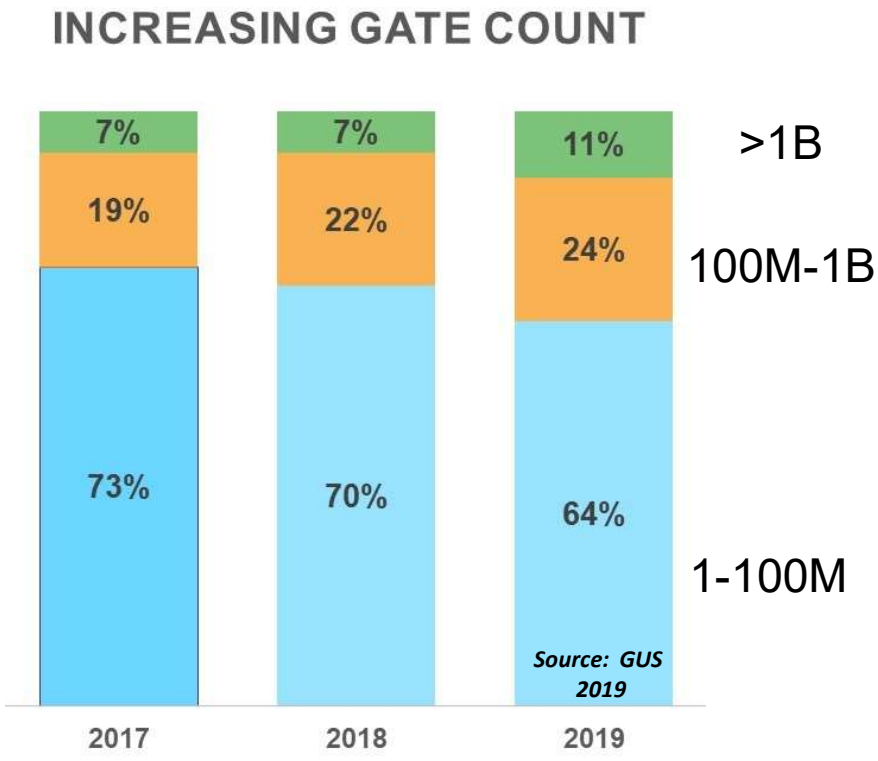
STARC05-2.1.4.5
SM_BNP, W336

Correct RTL Code

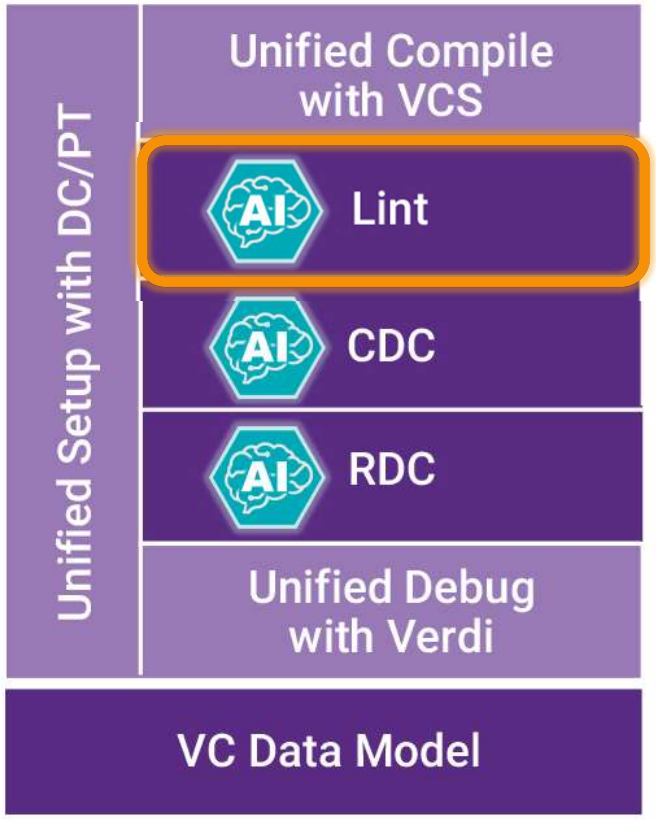
```
17 reg [1:0] hresp_flat ;
18 reg        lcyc_o ;
19 reg        sstb_o ;
20 reg        hwrite_flag;
21 wire [2:0] hburst;
22 wire [2:0] hsize;
23 wire [1:0] htrans;
24 wire [2:0] hlong;
25 wire [1:0] hshort;
26
27 always @ (posedge hclk or negedge hresetn ) begin
28   if (!hresetn) begin
29         hresp_flat  <= 2'b00;
30         lcyc_o      <= 1'b0;
31         sstb_o      <= 1'b0;
32   end
33   else if(hlong < {hshort,1'b0}) begin
34         case (htrans[1])
35               1'b0 :  begin
36                     hresp_flat <= 2'b10;
37                     lcyc_o     <= 1'b0;
38                     sstb_o     <= 1'b1;
39               end
40               1'b1 :  begin
41                     hresp_flat <= hburst & hsize ;
42                     lcyc_o     <= 1'b1;
43                     sstb_o     <= 1'b0;
44               end
45         endcase
46   end
47 end
```

VC SpyGlass Lint Addresses Leading Causes of Issues

- Chip Killer Bugs/Escapes
  - Delayed schedule by fixing problems late
  - Combinational loops, tri-state nets cause functional failure
- Long Verification Time
  - Incomplete coverage (limited simulation vectors)
  - Time-consuming & complex test vector/assertion creation
  - Synthesis/simulation mismatch, incomplete initialization
- Poor Area, Power, Testability and Timing
  - Unintended redundant logic results in poor QoR
- No Predictability Effecting Time-market
  - Manual, ad-hoc and subjective design reviews
  - 10+ structural bugs (find and fix takes 2-4 days each)



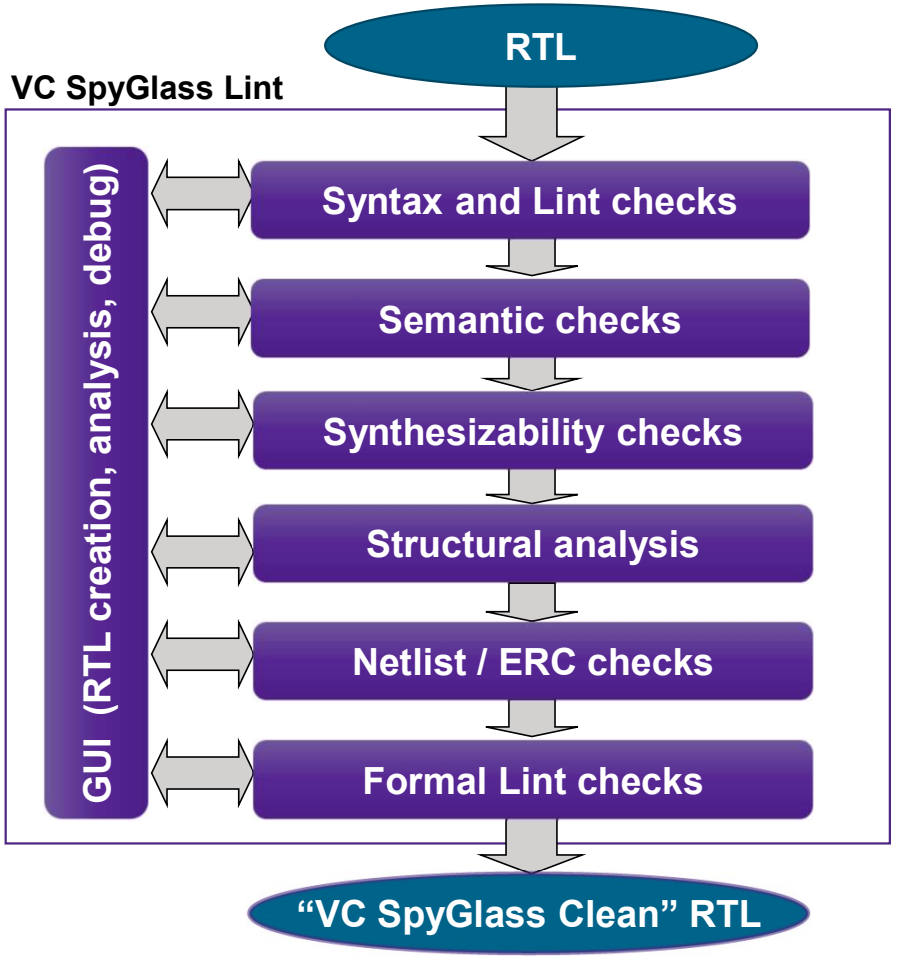
VC SpyGlass: Next Generation Static Platform



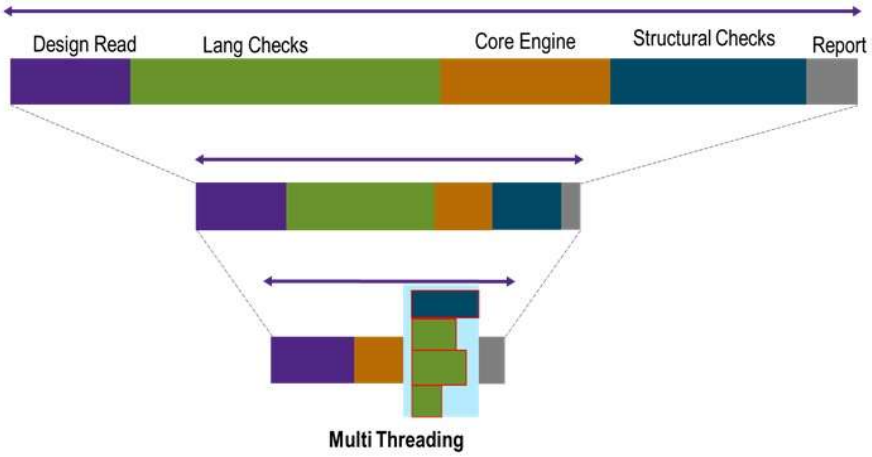
- 3X performance Over Native SpyGlass
  - Billion gate+ capacity with multi-core processing
- Lower noise
  - Built in Turbo Lint capabilities
  - Formal Analysis
  - Robust language support
- Accelerated Design Debug
  - Native Verdi debug
  - Extensive TCL Debug support

VC SpyGlass Lint

- Find & fix bugs at source (correct-by-construction RTL)
  - Compliance to coding guidelines, STARC, OpenMore, Morelint, etc.
  - Synthesizability & simulation issues
  - Structural, logical and connectivity issues
  - Electrical rule checks
- Verilog, VHDL, SystemVerilog and mixed RTL support
- Structured methodology and templates help tackle design issues systematically
- Comprehensive waiver support
- Easy debug with cross-probe to RTL in VC SpyGlass GUI



Accelerated Runtime with Multi-core Design Execution



Design Name	Gate Count (M)	SpyGlass Runtime (hh:mm)	VC SpyGlass Runtime	Runtime Gain
Design 1	140	07:19	01:31	5X
Design 2	28	00:59	00:25	2X
Design 3	1382	20:24	03:51	5X
Design 4	1600	04:50	01:28	3X
Design 5	750	07:46	03:00	3X

3x performance gain and half the memory

- Runtime acceleration for each process
- Parallelism enables more acceleration

Design Name	Gate Count (M)	SpyGlass Memory Usage	VC SpyGlass Memory Usage	Reduced Memory
Design 1	140	85	19.2	4X
Design 2	28	21.2	6.3	3X
Design 3	1382	223	62.5	4X
Design 4	1600	232	94	2X
Design 5	750	161	48	3X

VC SpyGlass Lint: Providing Faster Path to “Clean RTL”

Common Design Read

```
vc_static_shell> set search_path “<space separated list of directories >”
vc_static_shell> set link_library  “<space separated list of libs>“

vc_static_shell> analyze -format  vhdl -work lib_name file.vhd
vc_static_shell> analyze -format  verilog -work lib_name file.v
……
vc_static_shell> elaborate  -work libname DUT

vc_static_shell> read_file -format verilog -top DUT -netlist “design.v”
```

Setup

RTL Design

Netlist Design

Less Noisy Synthesis Engine

```
2 module TOP(a);
3 input [2:0] a;
4
5 wire [2:0] b;
6 wire [2:0] w1;
7 assign b = a;
8 assign a = b & w1;
9 endmodule
```

SpyGlass Bit level loop

```
1. Combloop at TOP.a[0]-TOP.b[0]
2. Combloop at TOP.a[1]-TOP.b[1]
3. Combloop at TOP.a[2]-TOP.b[2]
```

VC Lint Word level loop

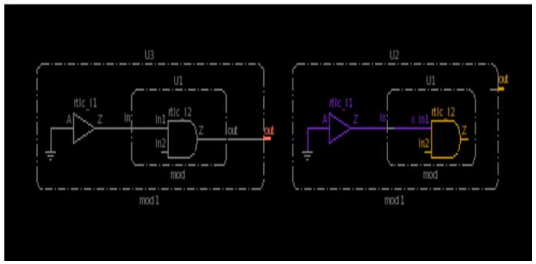
```
1. Combloop at TOP.a
```

Formal Aware Lint: Low Noise Architecture

```
 1 module top(in,enable,out);
 2
 3 input [3:0] in;
 4 input enable;
 5 output reg[3:0]out;
 6
 7 always @*
 8 begin
 9     if(!enable)
10         out<=4'b0000;
11     else
12         if(^in[3:0])
13             out<=in+1'b1;
14         else out<=4'b1111;
15
16 end
17
18 endmodule
```

- Violation reported at line 13.
- Condition @ line 12 will only be true for odd number of 1’s.
- No actual overflow possible.

Structural Rules Low Noise Mode (Turbo)



```
 2 module mod1(output out);
 3 supply0 s0;
 4 assign w1 = s0;
 5 mod U1(.in(w1), .out(out));
 6 endmodule
 7
 8 module mod(input in, in1, output out);
 9 assign out = in & in1;
10 endmodule
11
12
13 module TOP(output out);
14 mod1 U2(.out(out));
15 mod1 U3(.out(out));
16 endmodule
```

SpyGlass two viol

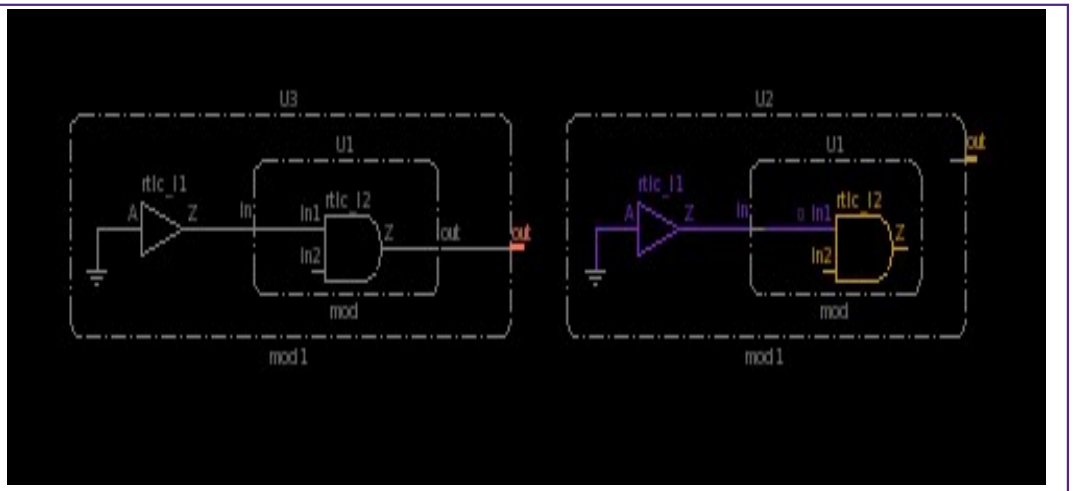
```
DisabledAnd Viol
1. TOP.out tiedlow thru TOP.U2.w1
2. TOP.out tiedlow thru TOP.U3.w1
```

VC Lint report smartly

```
DisabledAnd Viol
1. TOP.out tiedlow thru TOP.U2.w1
```

Low Noise Mode (Turbo) for Structural Rules

```
 2 module mod1(output out);
 3 supply0 s0;
 4 assign w1 = s0;
 5 mod U1(.in(w1), .out(out));
 6 endmodule
 7
 8 module mod(input in, in1, output out);
 9 assign out = in & in1;
10 endmodule
11
12
13 module TOP(output out);
14 mod1 U2(.out(out));
15 mod1 U3(.out(out));
16 endmodule
```



DisabledAnd Viol

1. TOP.out tiedlow thru TOP.U2.w1
2. TOP.out tiedlow thru TOP.U3.w1

SpyGlass Lint Multiple violations

DisabledAnd Viol

1. TOP.out tiedlow thru TOP.U2.w1

VC SpyGlass Lint reports module personality based

Functional Lint using Formal

Noise Reduction Examples

ImproperRangeIndex-ML

```
module encoder(clk, in1, sel, out);
input clk;
input [7:0]in1;
input [3:0]sel;
output out;

reg out;

always @(posedge clk) begin
   out <= in1[sel];
end

endmodule
```

Violation
'sel' signal value range (0-15) is greater than 'in1' signal index range (0-7)

```
module encoder(clk, in1, sel, out);
input clk;
input [7:0]in1;
input [3:0]sel;
output out;

reg out;

always @(posedge clk) begin
    if(sel <= 3'b111)
        out <= in1[sel];
    else
        out <= 1'b0;
end

endmodule
```

No Violation
Due to formal analysis, rule does not report false violation.

MissingCaseItems-ML

```
module case_multipleconcat(clk, in1, in2, out);
input clk;
input in1, in2;
output out;

reg out;

always @(posedge clk) begin
case({in1, in2})//synopsys full_case
2'b00 :
    out = 1'b0;
2'b01 :
    out = 1'b1;
2'b10:
    out = 1'b0;
endcase
end
endmodule
```

Violation
2'b11 case item is missing

```
module case_multipleconcat(clk, in1, in2, out);
input clk;
input in1, in2;
output out;

reg out;

always @(posedge clk) begin
case({in1, ~in1})//synopsys full_case
2'b01 :
    out = 1'b0;
2'b10:
    out = 1'b1;
endcase
end
endmodule
```

No Violation
Due to formal analysis, rule does not report false violation, although 2'b11 & 2'b00 case items are missing.

Functional Lint Analysis Results

Nominal runtime impact with high-noise reduction using formal engines

Customer Design	VC SpyGlass Lint Runtime (min)	VC SpyGlass Lint - Functional mode Runtime (min)	VC SpyGlass Lint Violations	VC SpyGlass Lint - Functional mode Violations	Noise Reduction (%)
Design 1	12	22	2035	1344	**34%**
Design 2	25	35	2068	1561	**24%**
Design 3	3	3.5	850	676	**20%**
Design 4	47	62	8880	6517	**27%**
Design 5	31	59	8621	6138	**29%**
Design 6	9	14	439	243	**45%**

Agenda

- Technical Overview
- **Lint Check Flow**
  - Running Lint Checks
  - Results Analysis
  - Creating Waivers

Tool Setup – General

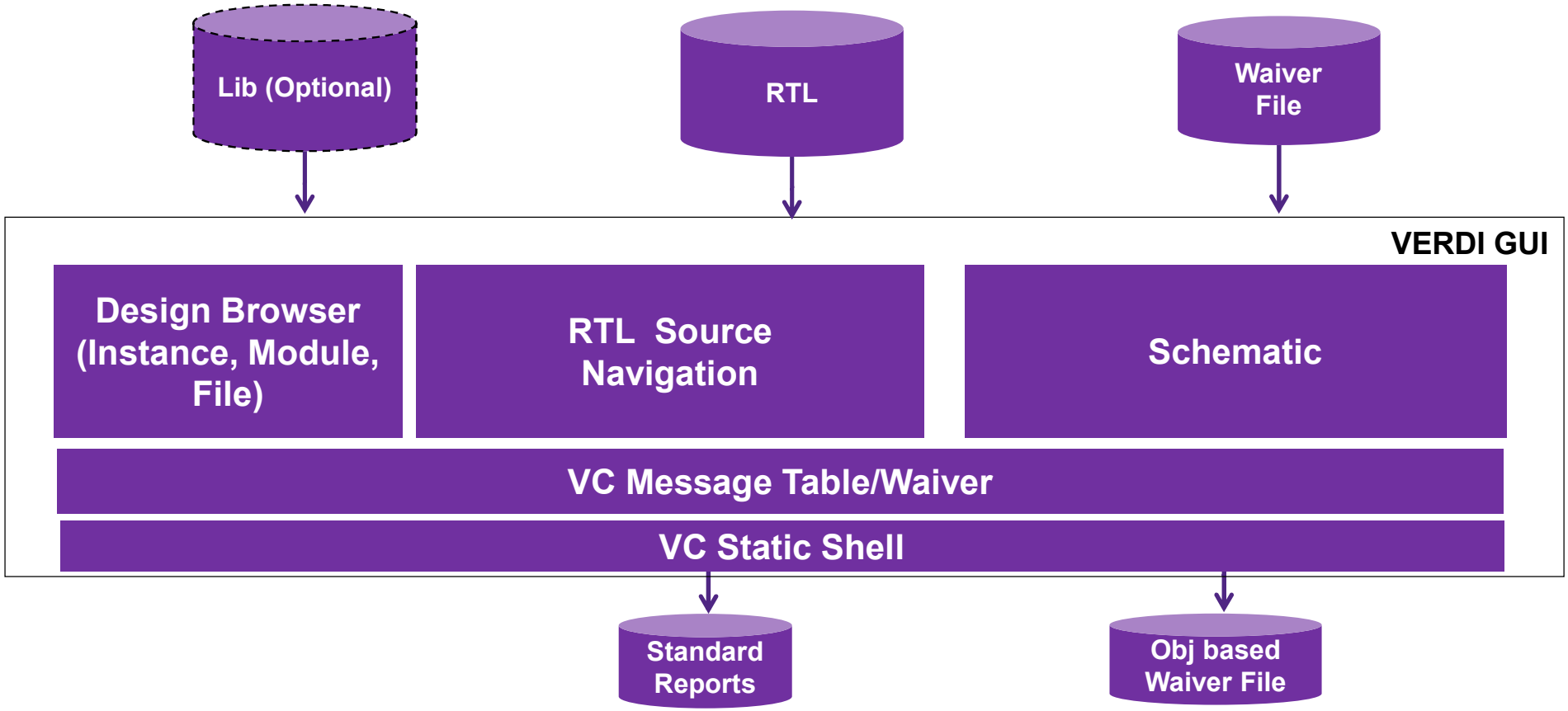
Two approaches:

- Native VC SpyGlass Lint Environment
  - VC Design Read and “synthesis” : analyze, elaborate
  - Lint configuration : app_vars, config_lint* commands
  - Lint analysis : check_lint
  - Reporting commands

- Migration from SG project file – Through `sg_read_project` command.
  - Same project file used in SpyGlass can be read (converted) on the fly to VC TCL equivalent
  - “Most” SpyGlass options and constraints will be translated to equivalent VC TCL script.

  .

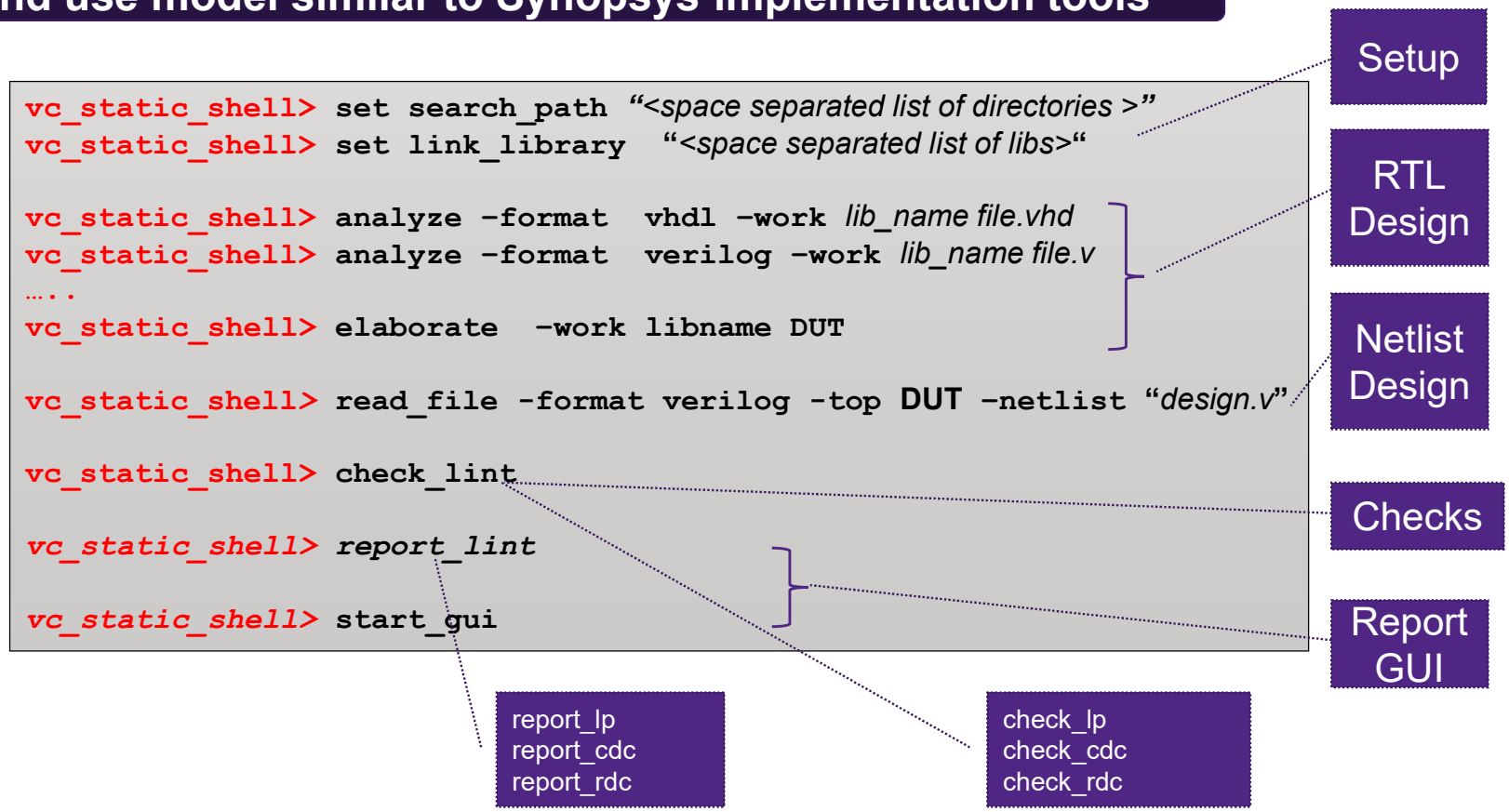
VC SpyGlass Lint Setup flow



Sample VC Static Command File

Loading a Design and Running Checks

Setup and use model similar to Synopsys implementation tools



VC SpyGlass- Sample Script

Lint TCL Script

```
set search_path <>
set link_library <>
set_app_var enable_lint true


 #### Reading design RTL files
<analyze_commands>
elaborate <design-top>

#### Reading SDC file (optional)
#read_sdc <>

#### Perform Lint checks
check_lint

#### Waive Lint violations
## waive_lint <>

#### Report Generation
report_lint -verbose -limit 0 –file lint_verbose.rpt

#### Save design database
## checkpoint_session -session <session-name>
```

%vc_static_shell -f lint.tcl

CDC TCL Script

```
set search_path <>
set link_library <>
set_app_var enable_cdc true


 #### Reading design RTL files
<analyze_commands>
elaborate <design-top>

#### Reading SDC file
read_sdc <>

#### Perform CDC checks
check_cdc

#### Waive CDC violations
## waive_cdc <>

#### Report Generation
report_cdc -verbose -limit 0 -file cdc_verbose.rpt

#### Save design database
## checkpoint_session -session <session-name>
```

%vc_static_shell -f cdc.tcl

RDC TCL Script

```
set search_path <>
set link_library <>
set_app_var enable_rdc true


 #### Reading design RTL files
<analyze_commands>
elaborate <design-top>

#### Reading SDC file
read_sdc <>

#### Perform CDC checks
check_rdc

#### Waive RDC violations
## waive_rdc <>

#### Report Generation
report_rdc -verbose -limit 0 –file rdc_verbose.rpt

#### Save design database
## checkpoint_session -session <session-name>
```

%vc_static_shell -f rdc.tcl

Overview
Sample script for running VC SpyGlass Lint

```
%> vc_static_shell -f vc_sg_lint.tcl out_dir run_lint -output_log_file vc_sg_lint.log          ;# start vc_static_shell and execute tcl script

# 0: Set parameter/options that can impact the design read.
vc_static_shell> set_app_var enable_lint true                                                 ;# Configures vc_static_shell to run lint

# 1: Configure rule tags and goal
vc_static_shell> configure_lint_tag -enable -tag "STARC05-2.10.2.3" -goal custom_lint -severity Warning
vc_static_shell> configure_lint_tag_parameter -tag "STARC05-2.10.2.3" -parameter HANDLE_NEGATION_SEMANTICS -value {no} -goal custom_lint
vc_static_shell> configure_lint_tag -enable -tag "W287b" -goal custom_lint
# 2: Specify where to find Source code, Analyze and Elaborate:
vc_static_shell> analyze -verbose -format sverilog -vcs { -f filelist }                       ;# Source code to Analyze + options
vc_static_shell> elaborate TOP                                                                ;# Elaboration + top

# 3: set goal name to run check_lint
vc_static_shell> configure_lint_setup -goal custom_lint                ;# Configuration which goal to run

# 4: Run lint check
vc_static_shell> check_lint                                            ;# Run lint check
# 5: report violations
vc_static_shell> report_lint -limit 0 -verbose -include_waived -gen_empty -file report_lint_verbose_limit_0.rpt -report {all} ;# report violations

# 6: save a session
vc_static_shell> checkpoint_session -session lint_database             ;# Save a session

#Open GUI for analysis, schematics
vc_static_shell> view_activity                                         ;# Open the GUI
```

VC SpyGlass Setup

Inputs Required To Run VC Apps

- Design (RTL)
  - RTL can be synthesizable subset of SV, Verilog, VHDL or any mix of these
  - Non-synthesizable constructs are NOT supported and will lead to errors
- DBs (SpyGlass requires .libs)

Setup

```
setenv VC_STATIC_HOME <VC STATIC Install Path>
setenv PATH       ${VC_STATIC_HOME}/bin:${PATH}
```

Lint app_vars

- "***set_app_var***" is similar to set_parameter/set_option of SpyGlass terminology
- You can override the tool default behavior using this command.
- "***get_app_var <variable name***" used to get the current value of app var.

Few examples :

- Required app_var for Lint.
  - `set_app_var enable_lint true ;#` Enable Lint flow inside of vc_static

```
#One Line TCL script to return all app_var's and values
foreach x [ get_app_var * -list ] { puts $x=[get_app_var $x]  }
```

Library Setup

```
vc_static_shell> set search_path ". path1 path2 … pathN"
vc_static_shell> set link_library "lib1 lib2 … libN"
```

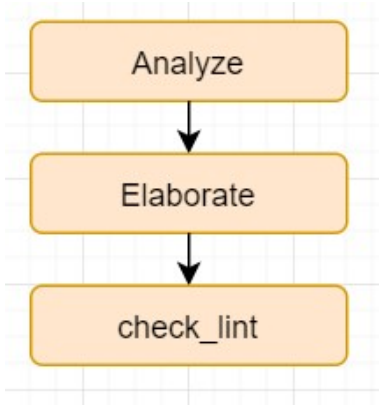
- search_path
  - Space separated list of relative or absolute directory paths
- link_library
  - Space separated list of liberty DB file names (.db's)
  - It does not support text form of liberty files (.libs)

- If user has a DC environment, the search_path and link_library settings can be copied from that environment
- Use .synopsys_vcst.setup to load initial setup

Read Design

- Using Analyze and Elaborate

```
analyze -format verilog top.v
analyze -format vhdl { -f ./vhdl.f }
analyze -format sverilog block1.sv
elaborate top
```



- VCS mode to read files

```
analyze -format verilog -vcs { +define+g90d -f sources_rtl.f }
analyze -format verilog -vcs { +incdir+src/./sim_1 -f sources_rtl.f –sv=2005 }
elaborate top
```

- Using the read_file

```
read_file -format verilog -top top top.v
read_file -format verilog -top top -vcs { -f file_list }
```

Configure Rules and Parameters

configure_lint_tag -enable -tag <rule-name> [-goal <goal-name>]

Language Rules:

configure_lint_tag -**enable** -tag "ArrayIndex" -goal test_goal

configure_lint_tag -**disable** -tag "BitOrder-ML" -goal test_goal

Language Rules： Checks performed on the HDL language construct

Structural Rules :

configure_lint_tag -enable -tag "FloatingInputs" -goal test_goal

configure_lint_tag -disable -tag "STARC05-1.3.2.2" -goal test_goal

Structural Rules： Checks performed on the hardware inferred by the tool

configure_lint_tag_parameter -tag <rule-name> -parameter <param-name> -value <value> [-goal <goal name>]

configure_lint_tag_parameter -tag "W287b" -parameter FAST -value {no} -goal test_goal

Lint Setup – General Commands

Command	Description
set_app_var/report_app_var	Sets/Reports the value of an application variable.
configure_lint_methodology -path <> -goal <>	Configure the methodology directory and goal name
configure_lint_tag –enable| -disable	Enable or Disable a rule tag
configure_lint_tag_parameter	Configure parameter for a rule tag
configure_lint_setup -goal lint_rtl	Configure which goal to run
check_lint	Run lint check for a goal
report_lint -verbose -file lint.verbose.rpt	Report the violations to a file
checkpoint_session -session complete_lint_database	Save a session
restart_session -session complete_lint_database	Load a session
view_activity or start_gui	Start GUI
waive_violation/waive_lint	Waive Lint violations
source template.tcl	Source all commands in template.tcl

* In vc_static_shell we can use “man” to get more detail explanation for each command

Agenda

- Technical Overview
- Lint Check Flow
  - Running Lint Checks
  - Results Analysis
  - Creating Waivers

Running Lint Checks

`check_lint` command

- Run Lint checks after importing the design

`check_lint`

It will check both language and structure rules

`-goal`

Specify which goal to run.

If we don't configure any lint tags, the default goal is lint_rtl goal which is in the following directory

```
vc_static_shell> configure_lint_methodology   \
                 -path $VC_STATIC_HOME/auxx/monet/tcl/GuideWare/block/rtl_handoff/lint \
                 -goal lint_rtl
```

```
1 set_app_var enable_lint true
2 check_hdl_lib -all
3
4 analyze -verbose -f sverilog -vcs test.v
5 elaborate test
6
7 check_lint
8 report_lint -gen_empty -file report_vc_sg.txt -verbose -limit 0
9
10 exit
```

Add incremental rule tag based on default lint_rtl goal

```
set_app_var enable_lint true
configure_lint_tag -enable -tag "W164a" -goal lint_rtl
analyz -verbose -f sverilog -vcs test.v
elaborate test

check_lint
report_lint -verbose -limit 0 -file report_lint.txt
```

Running Lint Checks

`check_lint` command

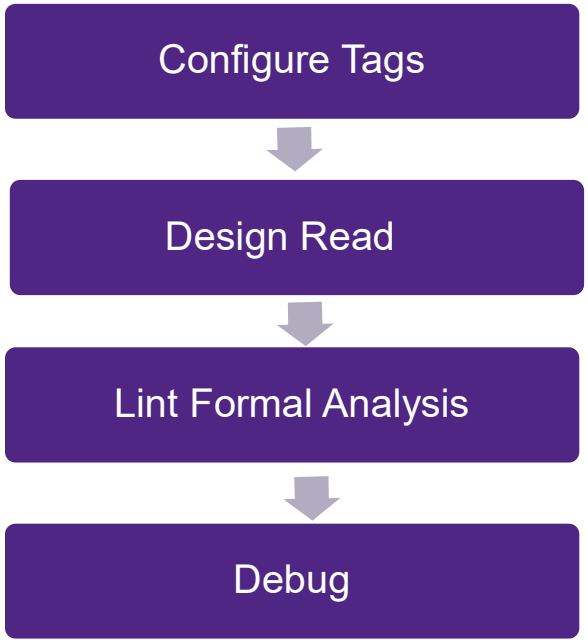
- Run Lint with customized goal

```
configure_lint_tag -enable -tag "badimplicitSM1" -goal custom_lint
configure_lint_tag -enable -tag "badimplicitSM2" -goal custom_lint
configure_lint_tag -enable -tag "badimplicitSM4" -goal custom_lint
configure_lint_tag -enable -tag "BlockHeader" -goal custom_lint
configure_lint_tag -enable -tag "bothedges" -goal custom_lint
configure_lint_tag -enable -tag "STARC05-2.1.6.5" -goal custom_lint -severity Warning
configure_lint_tag -enable -tag "STARC05-2.3.1.2c" -goal custom_lint -severity Error
configure_lint_tag -enable -tag "W421" -goal custom_lint -severity Error
configure_lint_tag -enable -tag "W442a" -goal custom_lint -severity Error
configure_lint_tag -enable -tag "W442b" -goal custom_lint
configure_lint_tag -enable -tag "W442c" -goal custom_lint
configure_lint_tag -enable -tag "W442f" -goal custom_lint
… …
configure_lint_setup -goal custom_lint

analyze …
elaborate …

check_lint
report_lint
```

Formal Aware Lint Delivering Low Noise

- Integration with VC Formal for advance debugging and interactivity
- Combine Formal abilities with traditionally dominant lint technology
- Easy setup: Automatic invocation of formal in existing lint setup
- Leverage existing comprehensive Lint checks in Formal flow

Flow for Formal Aware Lint



Noise Reduction Using Formal Engines: Supported 21 Rules

Plan to support additional Lint rules in upcoming releases

Rule Name	Description
DuplicateCaseLabel-ML	A case choice is covered more than once in a case statement when "parallel_case" pragma is specified
W71	A case statement(or selected signal assignment) does not have a default or OTHERS clause
W116	Unequal length operands in bitwise logical/arithmetic/ternary operator
W263	A case expression width does not match case select expression width
STARC05-2.10.3.2a	Bit-width of operands of a logical operator must match.
W415a	Signal may be multiply assigned (beside initialization) in the same scope.
sim_race02	Multiple assignments to a signal
STARC05-2.2.3.3	Do not assign over the same signal in an always construct for sequential circuits
STARC05-2.1.3.1	Bit-width of function arguments must match bit-width of the corresponding function inputs.
W110	An instance port connection has incompatible width compared to the port definition
W164a	LHS width is less than RHS width of assignment (Truncation)
W362	Unequal length in arithmetic comparison operator
W467	Use of don't-care except in case labels may lead to simulation/synthesis
SignedUnsignedExpr-ML	Signed and Unsigned are mixed together
ImproperRangeIndex-ML	Possible discrepancy in the range index or slice of an array
UniqueIfMissingCond-ML	Check missing conditions corresponding to unique if statements
W551	Ensure that a case statement marked full_case or a priority/unique case statement does not have a default clause.
MissingFsmStateTransition	FSM may have a missing transition pair from STATE - to STATE
NotReachableFsmState	Reports unreachable states of an FSM
NoExitFsmState	Do not have states without exit in FSMs
RegisterStuckInResetState-ML	Catches the registers which are stuck in a constant initial state due to bad coding style.

VC SpyGlass: Easy Setup of Functional Lint

- Minimal Changes in the setup.
- User needs to set an app-var to enable Functional Lint.
- Customized & extremely fast Formal engine for Functional Lint analysis
- No Formal expertise required

JUST an app_var : “lint_functional_mode”

```
 1 puts "Running VC_SpyGlass ..."
 2
 3 set_app_var enable_lint true
   ------------------------------------------------------------
 4
 5 check_hdl_lib -all
 6 get_resource_cost
 7
 8 set_blackbox -designs ##bbox_list
 9 set_black_block_list ##bbox_string
10 +-- 71 lines: set lint_ignore_rule_pragmas false-----------
~
```

```
 1 puts "Running VC_SpyGlass ..."
 2
 3 set_app_var enable_lint true
 4 set_app_var lint_functional_mode true
 5
 6 check_hdl_lib -all
 7 get_resource_cost
 8
 9 set_blackbox -designs ##bbox_list
10 set_black_block_list ##bbox_string
11 +-- 71 lines: set lint_ignore_rule_pragmas false---------
~
```

Agenda

- Technical Overview
- Lint Check Flow
  - Running Lint Checks
  - Results Analysis
  - Creating Waivers

Generating Lint Reports

- Generates a report from the violation database

report_lint [–verbose] [-file <*report.file*>]
[-tag <*violation_tag*>] [-filter <*expression*>]
[-severity <list>]
[-limit]

  - Run and analyze this report after every `check_lint` step
  - Default limit 100 (max 100 messages per error tag)

    Use -limit 0 switch to report all message per error tag

- Default (no options): Returns a violation summary
  - Verbose report of setup violations and save to file:

    report_lint -verbose -file report.viol
  - Example: Report all W110 violations in the module usbHost

    report_lint -tag W110 -verbose –limit 0 -filter {(Module =~ "usbHost")}

Tip: use verbose report to get complete list of filter selections

Generating Lint Reports

Use report_lint TCL proc as below:

```
report_lint -report { all } -file report_hdl.txt -verbose
```

- report { **all** } will generate :
  1. SpyGlass format **moresimple.rpt** **(***./vcst_rtdb/reports/moresimple.rpt)*
  2. *SpyGlass format waiver.rpt* **(***./vcst_rtdb/reports/waiver.rpt)*
  3. VC Static default **report_hdl.txt** **(**./report_hdl.txt)

- report { **sg_moresimple** } will only generate SpyGlass moresimple report

  Note : Generated filename is “moresimple.rpt”

- report { sg_waiver } will only generate SpyGlass waiver report

  Note : Generated filename is “waiver.rpt”

Analyzing Reports

Lint stages, families and tags

Examples for stages

- DESIGN_READ
- LANGUAGE_CHECK
- STRUCTURE_CEHCK

Management Summary

Stage	Family	Fatals	Errors	Warnings	Infos
BUILTIN_CHECK	CODING	0	0	0	13
DESIGN_READ	WarnAnalyzeBBox	0	0	1	0
LANGUAGE_CHECK	CODING	0	1	18	1
STRUCTURAL_CHECK	CODING	0	1	3	17
Total		0	2	22	31

Tree Summary

Severity	Stage	Tag	Count
error	LANGUAGE_CHECK	W122	1
error	STRUCTURAL_CHECK	UndrivenInTerm-ML	1
warning	DESIGN_READ	WarnAnalyzeBBox	1
warning	LANGUAGE_CHECK	STARC05-2.2.3.3	2
warning	LANGUAGE_CHECK	W240	6
warning	LANGUAGE_CHECK	W287a	1
warning	LANGUAGE_CHECK	W287b	2
warning	LANGUAGE_CHECK	W415a	1
warning	LANGUAGE_CHECK	W528	6
warning	STRUCTURAL_CHECK	STARC05-1.3.1.3	3
info	BUILTIN_CHECK	PragmaComments-ML	13
info	LANGUAGE_CHECK	ReportPortInfo-ML	1
info	STRUCTURAL_CHECK	RegInputOutput-ML	17
Total			55

Generating Lint Report – Use Tag

```
report_lint -verbose -tag  W287a
```

```
--------------------------------------------------------------------------------
Management Summary
--------------------------------------------------------------------------------
Stage           Family      Fatals    Errors  Warnings     Infos
-----           --------  --------  --------  --------  --------
LANGUAGE_CHECK  CODING           0         0         1         0
-----           --------  --------  --------  --------  --------
Total                            0         0         1         0


--------------------------------------------------------------------------------
Tree Summary
--------------------------------------------------------------------------------
Severity  Stage           Tag      Count
--------  -----           -------  -----
warning   LANGUAGE_CHECK  W287a        1
--------  -----           -------  -----
Total                               1


--------------------------------------------------------------------------------
W287a  (1 warning/0 waived)
--------------------------------------------------------------------------------
Tag           : W287a
Description   : Input '[Signal]' of instance '[InstName]' is undriven.[Hierarchy: '[HIERARCHY]']
Violation     : Lint:26
Goal          : rtl_lint
Module        : ethmac
FileName      : ./rtl/verilog/ethmac.v
LineNumber    : 635
Statement     :   .dbg_dat(wb_dbg_dat0),
Signal        : wb_dbg_dat0
InstName      : ethreg1
HIERARCHY     : ethmac
```

Save and Load Lint Session

- Save Lint session

  vc_static_shell > checkpoint_session -session complete_lint

- Load Lint session

  % vc_static_shell -restart complete_lint_cpdb -verdi

  Or

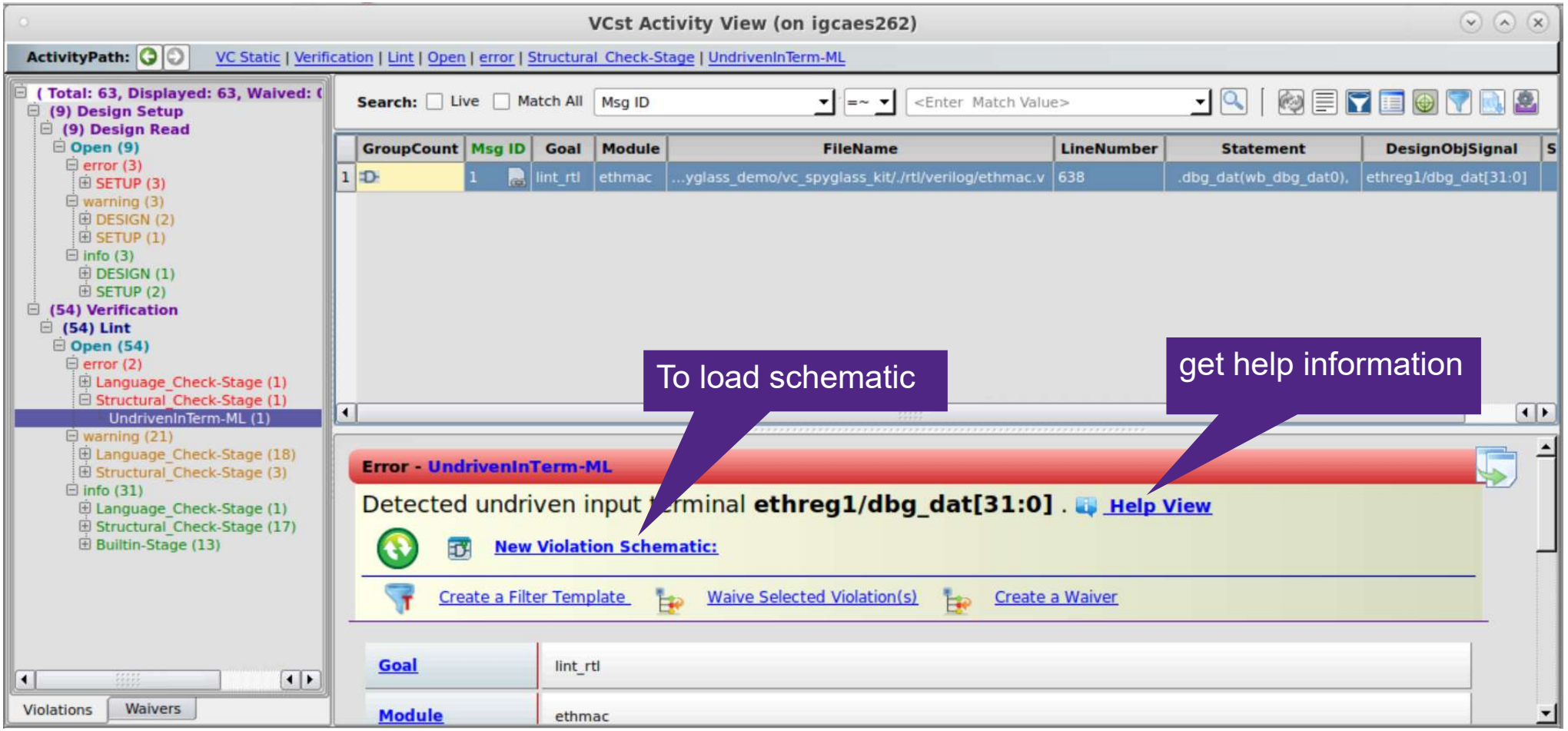
  vc_static_shell > restart_session -session complete_lint_cpdb

  vc_static_shell > start_gui

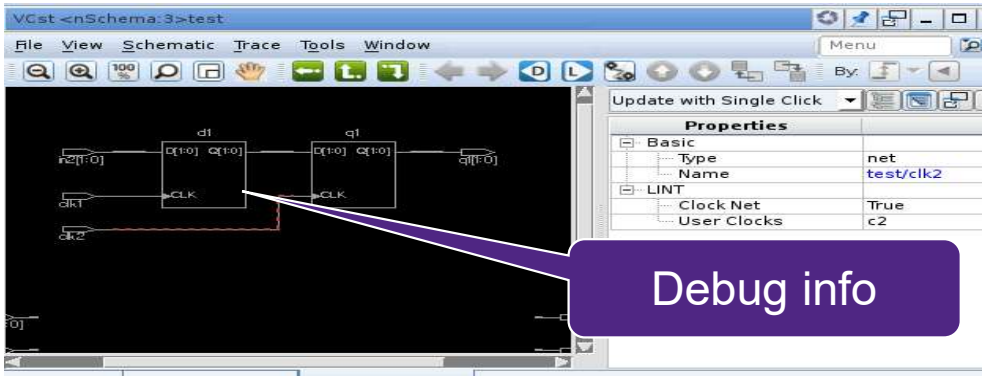
- Run lint in GUI mode

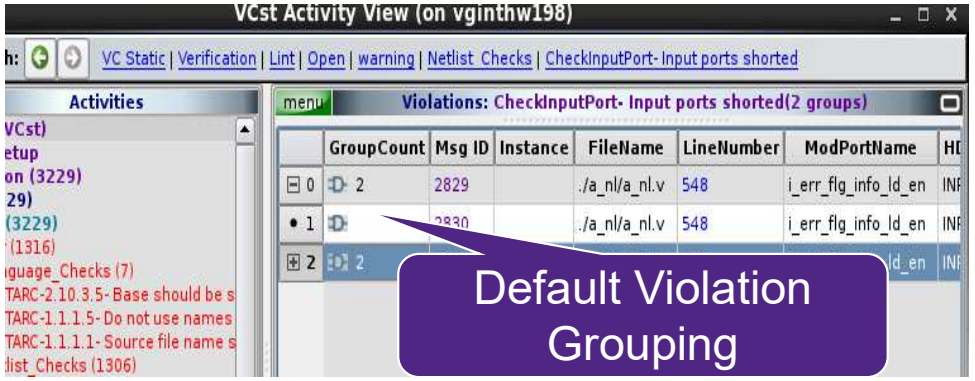
  % vc_static_shell -f vc_sg_lint.tcl -verdi

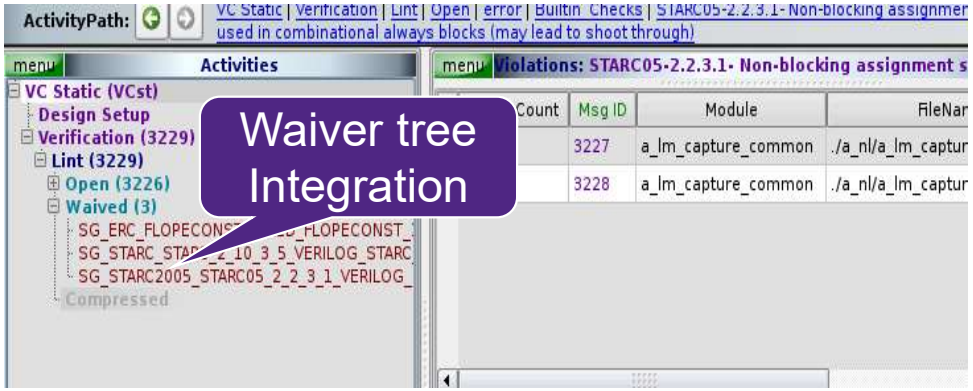
Review Lint Violations in GUI



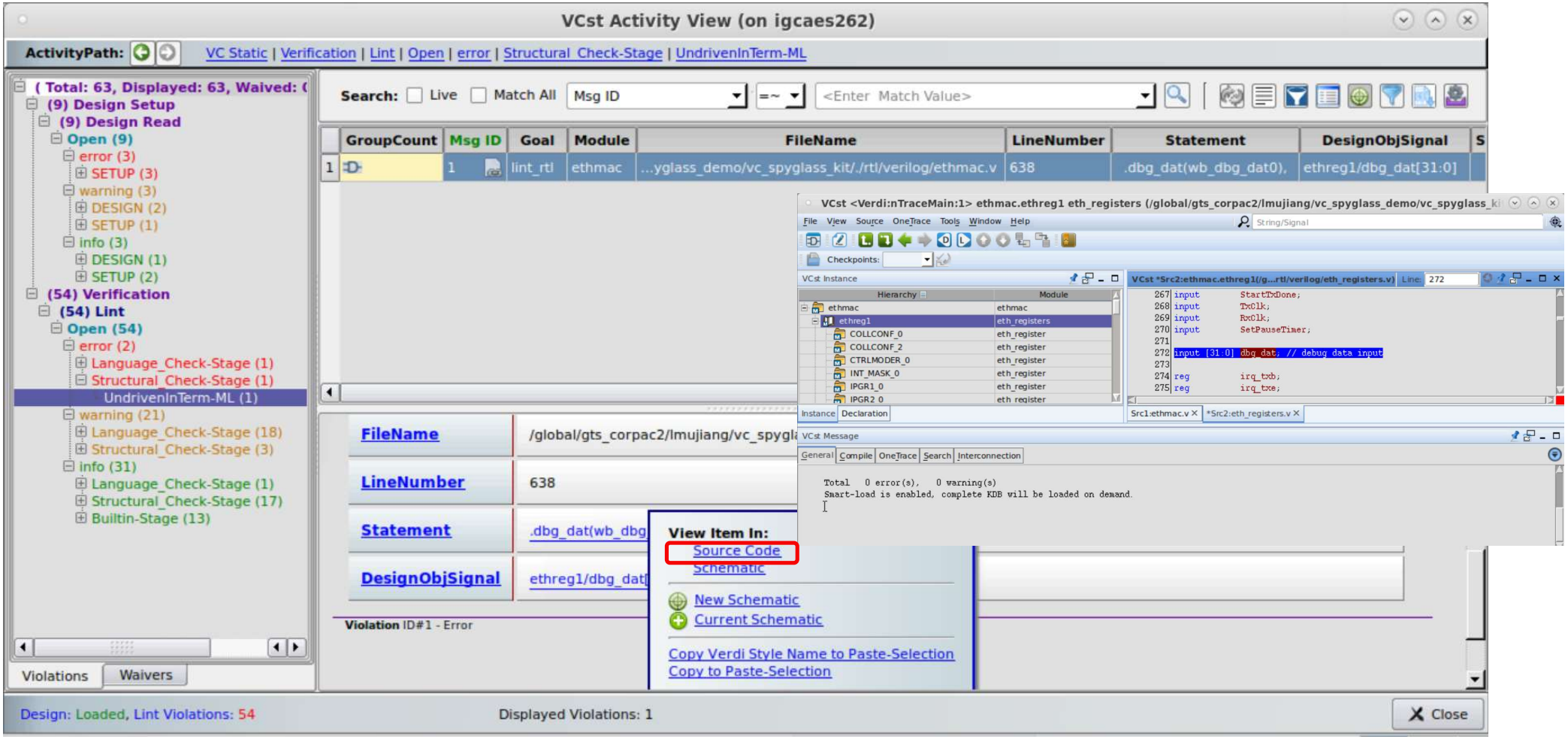
Review Lint Violations with Native Verdi Debug







Review Lint Violations with Native Verdi Debug



Agenda

- Technical Overview
- Lint Check Flow
  - Running Lint Checks
  - Results Analysis
  - Creating Waivers

Creating Waivers

waive_violation/waive_lint commands

- To ignore individual or groups of violations:

  [-add <name>] (Add waiver)

  [-comment <comment>] (Waiver comment)

  [-tag <tag>] (Waive violations based on tag)

  [-severity <list>] (Waive violations based on severity: Values: all, error, info, warning)

  [-filter <expression>] (Waive violations based on expression)

- To avoid false-positives, use filters to focus the scope of waivers
- In 2021.09 onwards release, new *waive_violation* command added which will support all options supported by existing app specific waiver commands.

Old command	New command	Usage
waive_<app>	waive_violation -app <>	Provides ability to add waive commands

  -app option - Provide list of apps on which waiver will be applied: lint, cdc, lp, rdc, sdc

- Existing waive_cdc are kept as tcl wrapper for providing backward compatibility. App specific commands will internally call waive_violations -app <APP>.
- Old wavie_lint will be still supported, but any dump writing will be using waive_violation

Creating Waivers

waive_violation commands

vc_static_shell > report_lint -verbose -tag W122

```
--------------------------------------------------------------------------------
W122  (1 error/0 waived)
--------------------------------------------------------------------------------
Tag          : W122
Description  : The signal/variable '[Signal]' (or some of its bits) read in the block is not in the sensitivity list[Hierarchy:
'[HIERARCHY]']
Violation    : Lint:27
Goal         : lint_rtl
Module       : eth_registers
FileName     : /global/gts_corpac2/lmujiang/vc_spyglass_demo/vc_spyglass_kit/rtl/verilog/eth_registers.v
LineNumber   : 879
Statement    :       `ETH_DBG_ADR         :  DataOut=dbg_dat;
Signal       : dbg_dat
HIERARCHY    : :ethmac:ethreg1@eth_registers
```

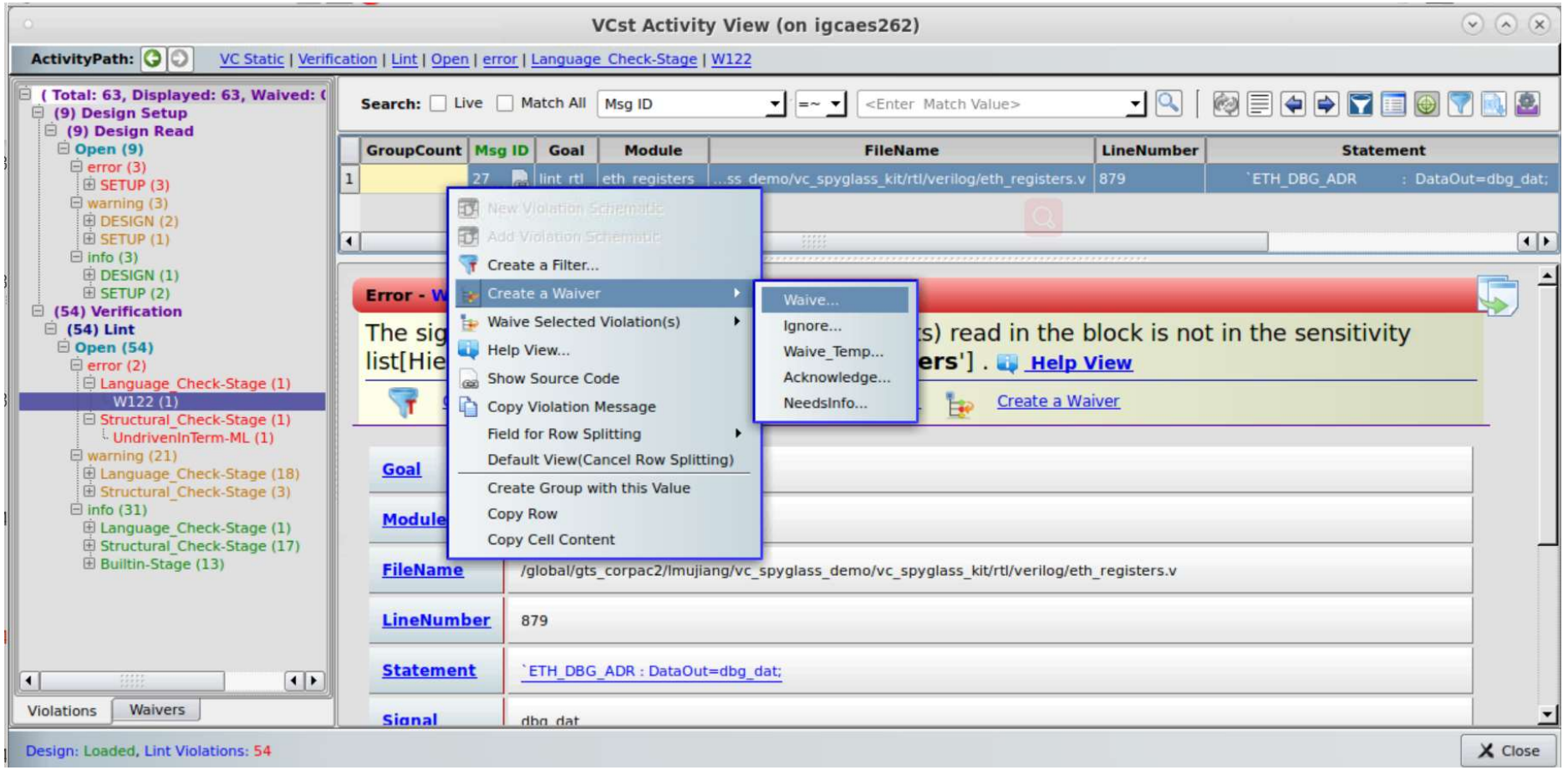
All fields can be used with filter option

- Create a lint waiver:

```
waive_violation -app lint -add W122_w1 -tag "W122" -filter {(Module == "eth_registers") AND (Signal == "dbg_dat") AND
(HIERARCHY == ":ethmac:ethreg1@eth_registers")}  -comment "Tom reviewed on 2020/4/1"
```

Creating Waivers

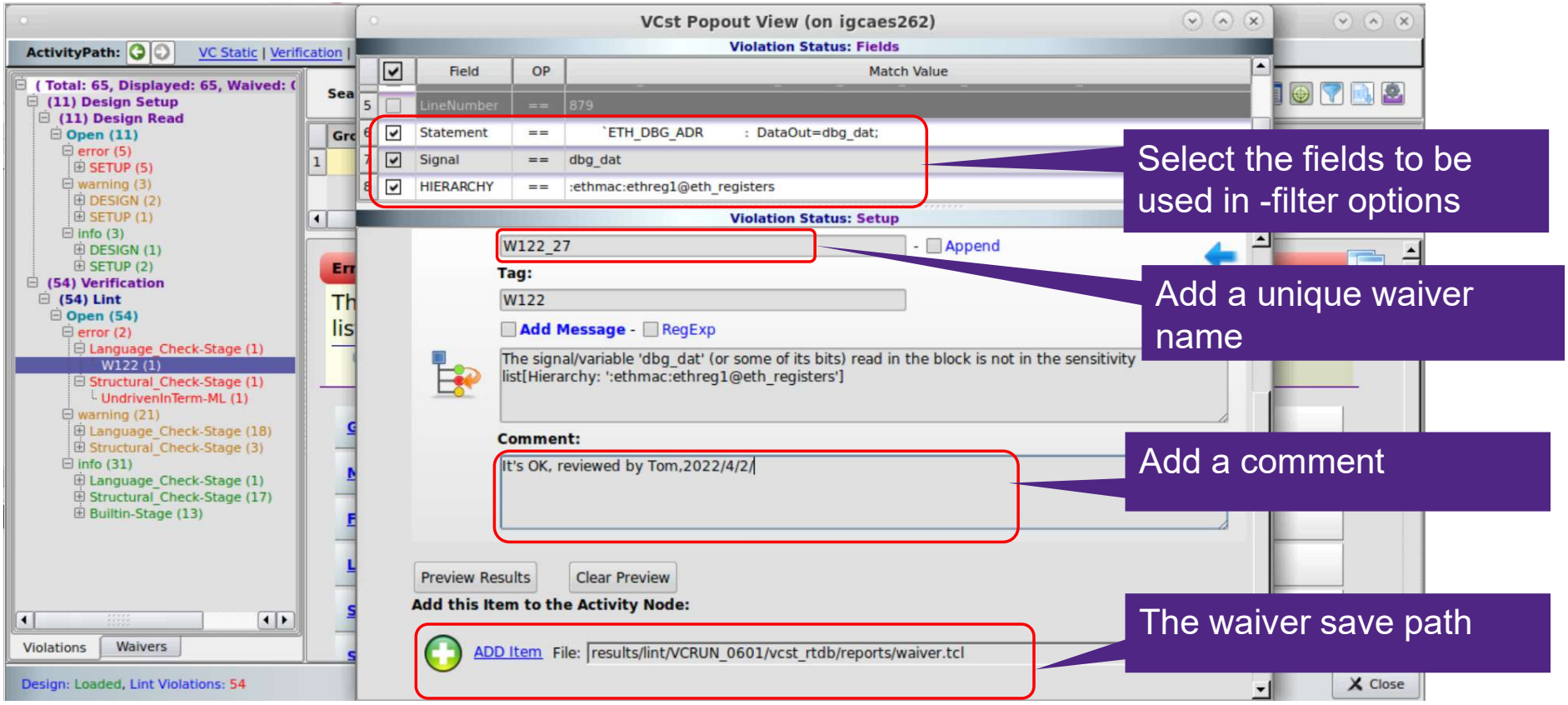
Add waiver in GUI



Creating Waivers

Add waiver in GUI

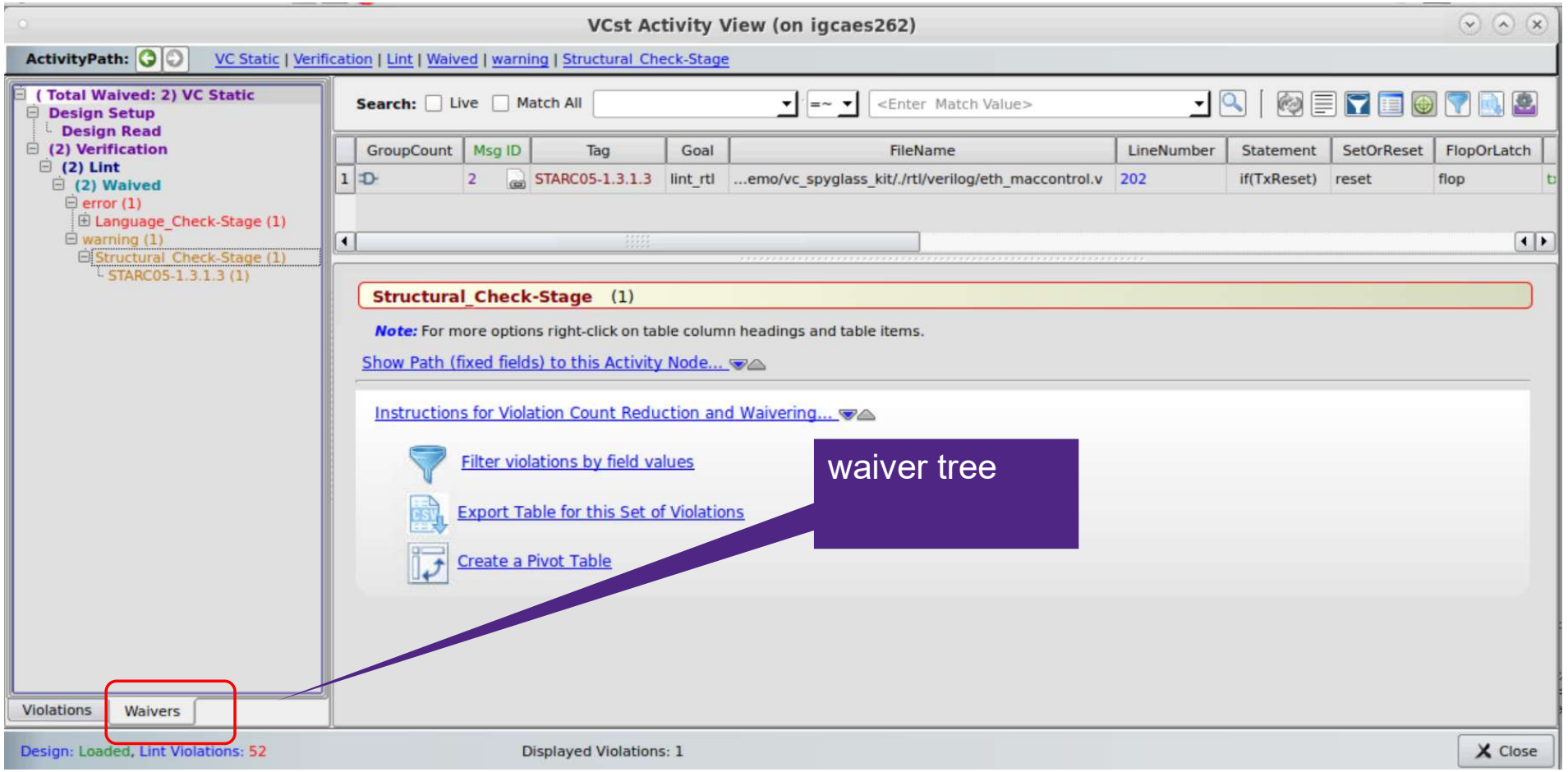
Configure which fields are selected by default to generate GUI waiver
set_app_var enable_signature_to_create_waiver false (disable signature)
configure_waiver_filter_field -tag <tag> -field <{<leaf path>}> <-enable/-disable>



static_shell> #waive_violation -app Lint -add W122_27 -tag "W122" -filter {(Goal == "lint_rtl") AND (Module == "eth_registers") AND (Statement == " `ETH_DBG_ADR: DataOut=dbg_dat;") AND (Signal == "dbg_dat") AND (HIERARCHY == ":ethmac:ethreg1@eth_registers")} -comment "It's OK, reviewed by Tom,2022/4/2"
#set_file_for_waiver W122_27 results/lint/VCRUN_0601/vcst_rtdb/reports/waiver.tcl

Creating Waivers

Add waiver in GUI

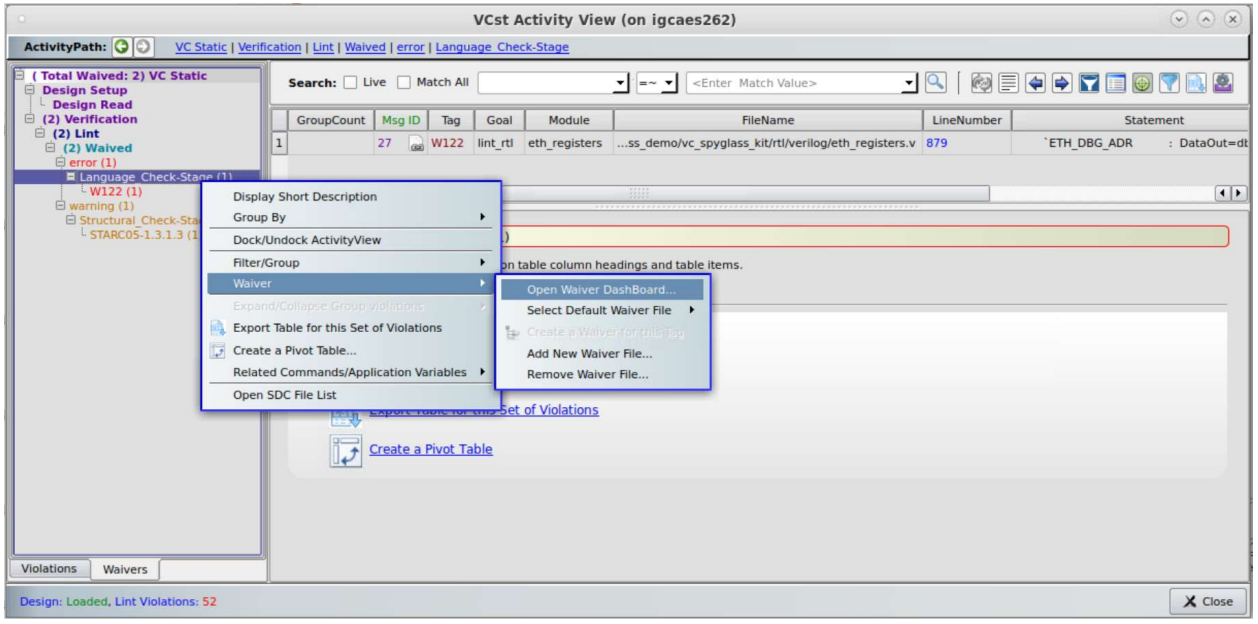


Manage Waiver with DashBoard

“Waiver DashBoard” is to manage and view details of waivers and waiver files.

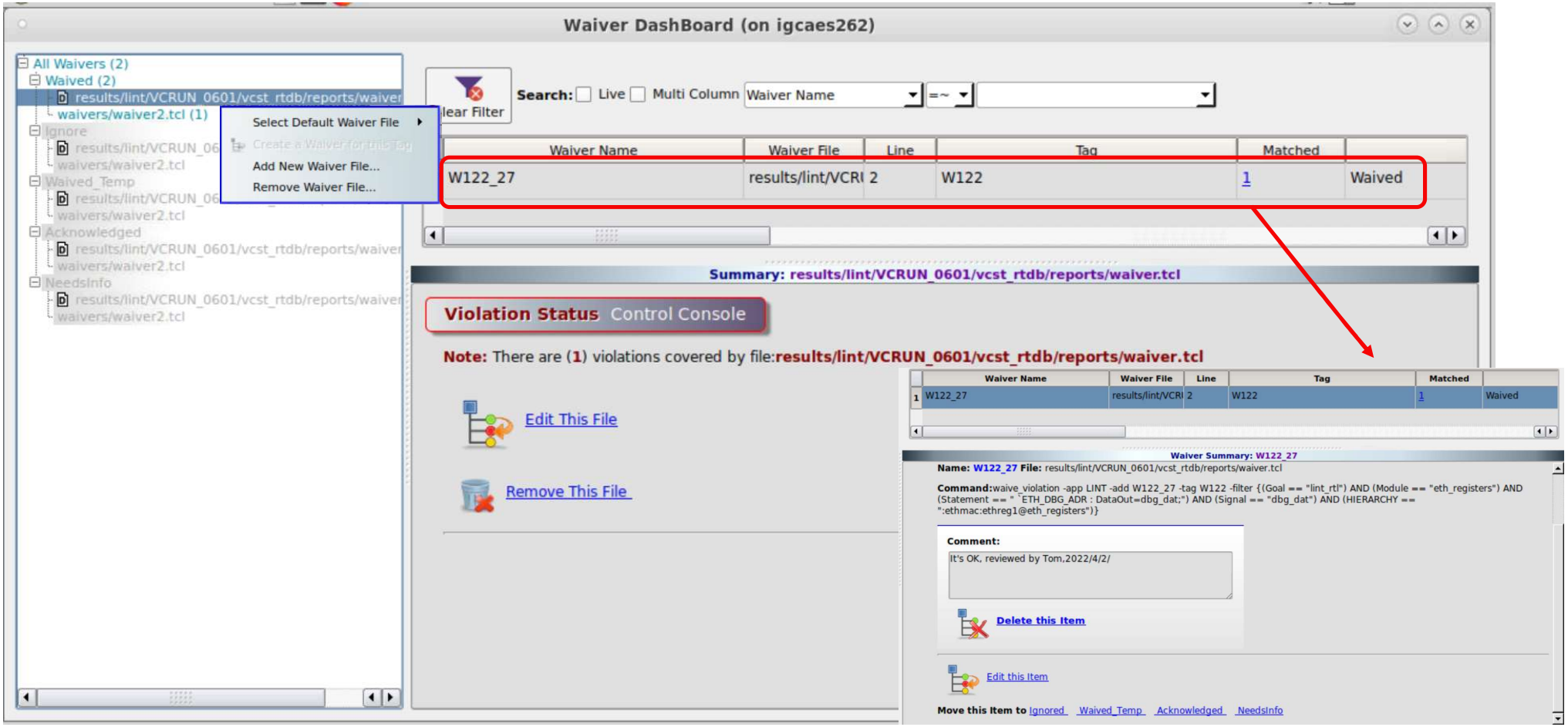
How to invoke

- Activity tree right-click menu (From both tabs) → Waiver → Open Waiver DashBoard
- Activity view toolbar icon “Show Waiver DashBoard”



Manage Waivers with DashBoard

Remove and add wavier file



FAQs in Lint Flow

How to Migrate SpyGlass Customized Lint Goal

Convert customized lint methodology to VC SpyGlass TCL script

- Greate convert tcl script

```
convert.tcl
sg_read_project –project lint.prj -goal lint/lint_rtl -app lint -tclfile vc_gen_lint.tcl \
                -debug_rule_conversion 1
```

  - Where lint.prj is spyglass project file
  - lint/lint_rtl is the goal name, the goal name decides by your own goal name in project file
- Run convert script

  %> vc_static_shell -f convert.tcl
- Review translated TCL script
  - Rule list file: vc_gen_lint_rules.tcl
  - Rule mapping list
    - ruleConversion.rpt
    - BuiltInRuleMapping
  - If there are STARC/STARC02 rules, need replace with STARC05 rules in VC SpyGlass

Documentation :

VC Static Platform Documentations
https://solvnet.synopsys.com/dow_retrieve/latest/ni/vc_static.html

VC Static Platform User Guide
VC Static Platform Command Reference Guide

VC Spyglass Lint Documentation

VC SpyGlass Lint User Guide
VC SpyGlass Lint Rules Reference Guide

Tcl command-line help

vc_static_shell> help report*
vc_static_shell> man <command>

SYNOPSYS®
Silicon to Software™

Thank You