Formality: Debugging Failing Verifications

CONFIDENTIAL INFORMATION

The following material is being disclosed to you pursuant to a non-disclosure agreement between you or your employer and Synopsys. Information disclosed in this presentation may be used only as permitted under such an agreement.

LEGAL NOTICE

Information contained in this presentation reflects Synopsys plans as of the date of this presentation. Such plans are subject to completion and are subject to change. Products may be offered and purchased only pursuant to an authorized quote and purchase order. Synopsys is not obligated to develop the software with the features and functionality discussed in the materials.

Scope of this Training

- This presentation covers debugging tips for failing verifications
- Other Formality presentations will address
  - Recognizing and addressing verification problems in the UPF flow
  - Debugging and resolving hard verifications
- Labs are included with the training

Agenda

- Debugging Flow
- Frequently Used Debugging Tools
- Common Problems
- Additional Debugging Tools
- Labs

Help!

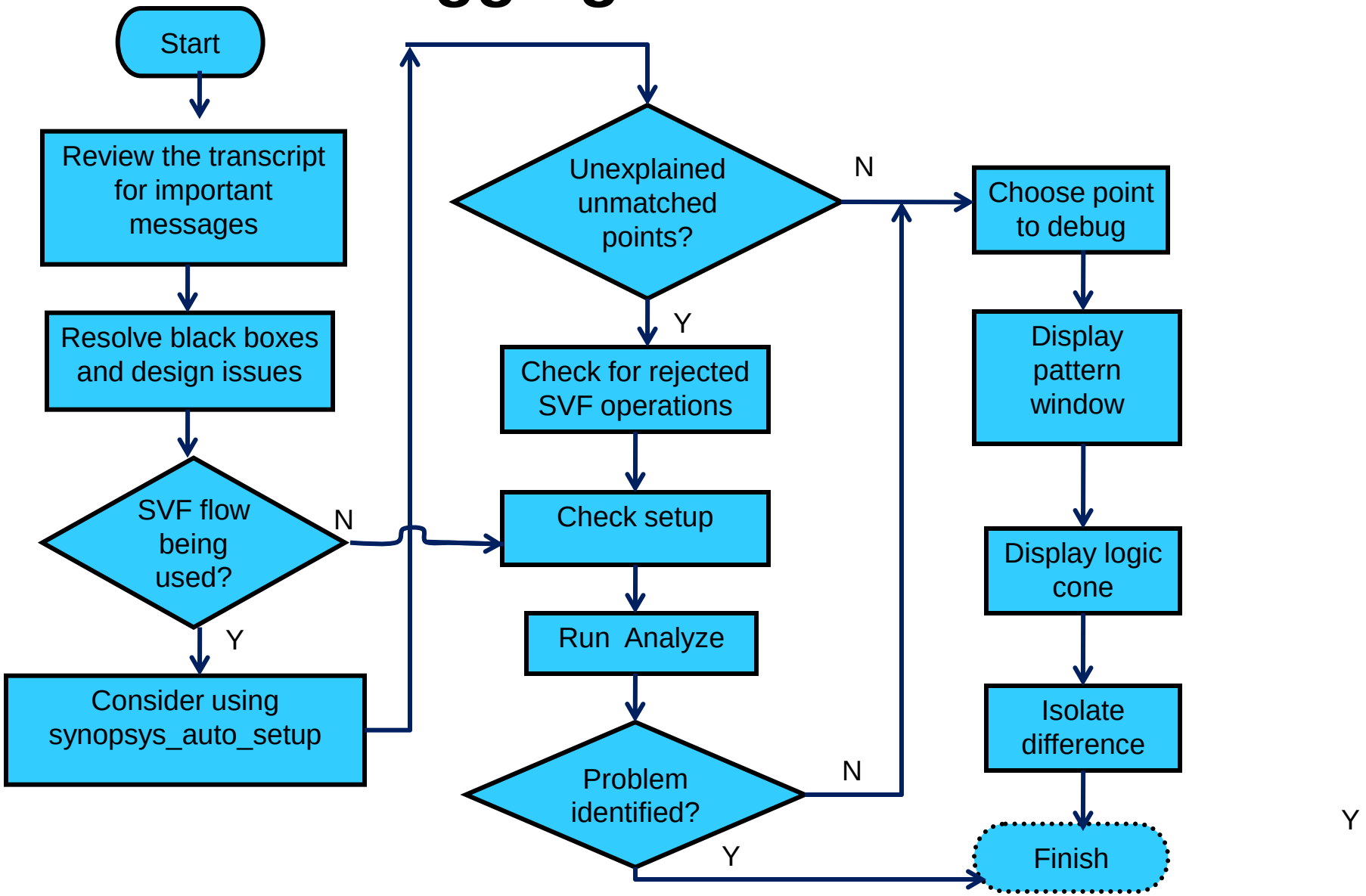
My verification failed!

Now what do I do?

Debugging Flow

- Step 1: Look at the transcript for clues
- Step 2: Use debugging tools and commands
- Step 3: Identify and resolve problem areas
- Step 4: Try the verification again
- Step 5: Ask for help

Debugging Flow Chart



Look at the Transcript

- Check for simulation/synthesis mismatch errors
- Check RTL interpretation messages in transcript
- Were full_case and parallel_cases pragmas interpreted?
- Check for black-box warnings in the transcript
- Check for rejected SVF guidance commands
- Check for unmatched compare points
- Unmatched compare points only in implementation? Clock-gating latches?
- Is there a setup problem? Did you disable scan?

Auto Setup Mode

- Reduces the need for debugging
  - A large percentage of failing verifications are “false failures” caused by incorrect or missing setup in Formality
- `set synopsys_auto_setup true`
  - Assumptions made in DC will also be made in FM
  - Increases out-of-the-box (OOTB) verification success rate
  - Use before command: `set_svf file.svf`
- Works with or without SVF, does more with SVF
  - Handles undriven signals like synthesis
  - RTL interpretation like synthesis
  - Auto-enable clock-gating and auto-disable scan (requires SVF)

What Auto Setup Mode Does

- Performs all of the following automatically (and more):

```
set hdlin_error_on_mismatch_message false
set hdlin_ignore_parallel_case false
set hdlin_ignore_full_case false
set verification_set_undriven_signals synthesis
set verification_verify_directly_undriven_output false
```

- DC places additional setup information in SVF used in this mode
  - Clock-gating notification
  - Disabling scan mode
- Variable and command changes can be overwritten by user
  - Transcript summary shows variable settings
  - Variables will take the last value that was set

Agenda

- Debugging Flow
- Frequently Used Debugging Tools
- Common Problems
- Additional Debugging Tools
- Labs

Frequently Used Debugging Tools

- `analyze_points` command
- Pattern viewer
- Logic cone viewer

Debugging Tools: analyze_points

- Provides possible problem cause and next steps for failing and hard verifications:
  - Unconstrained inputs (scan inputs)
  - Rejected SVF guidance (constants, reg merging, etc)
  - Undriven signals in the reference design
  - Pipelined DesignWare components that have not been retimed automatically
  - Incorrect retention register models in UPF designs
  - Complex datapath modules with related SVF rejections resulting in a hard verification
- New causes are added in future releases

Debugging Tools: analyze_points

- Command `analyze_points`
  - Options for failing verifications: `-failing, -all`
  - Takes a single or list of compare points as an argument
- Command `report_analysis_results`
  - Options: `-summary`
- Variable `verification_run_analyze_points`
  - Default value is false
  - When enabled runs `analyze_points -all`

Debugging Tools: analyze_points

```
fm_shell (verify)> analyze_points -failing

********************************** Analysis Results **********************************
Found 1 Unconstrained Implementation Input
--------------------------------
Unmatched input ports in the implementation typically result
from test logic insertion. Constraining the unmatched ports
to a constant value may correct the failures.
--------------------------------
i:/WORK/aes_cipher_top/test_se
    Unmatched in the implementation cones for 20 compare point(s):
        i:/WORK/aes_cipher_top/text_in_r_reg_112_
        i:/WORK/aes_cipher_top/us22/sbox2/dreg_reg_2_
        i:/WORK/aes_cipher_top/us22/sbox2/dreg_reg_4_
        i:/WORK/aes_cipher_top/us22/sbox2/dreg_reg_7_
        {…}

    Try adding this command before verify:
        set_constant i:/WORK/aes_cipher_top/test_se 0
-----------
--------------------------------
*******************************************************************************
Analysis Completed
```

Debugging Tools: analyze_points

Formality (R) Console - Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/top

Implementation: i:/WORK/top

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

	Type	Reference	Size	Implementation	Size	+/-
1	DFF	/Fp_reg[1]		Fp_reg_1_		
2	DFF	/Fp_reg[4]		Fp_reg_4_		

Number of Failing Points: 2

Display names: Original Mapped

Analyze

Analyze Selected Points

Filter:

Debugging Tools: analyze_points

Formality (R) Console - Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/top

Implementation: i:/WORK/top

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

Possible Failure Causes

- Unconstrained Implementation Input (1)
  - i:/WORK/top/test_se
- Rejected Guidance Type (0)
- Undriven Reference Signal (0)
- Unretimed DesignWare Component (0)
- Missing Retention Register (0)
- Modules With Rejected Datapath Guidance (0)
- Modules With Datapath Components (0)

Description - Unconstrained Implementation Input:
Unmatched input ports in the implementation typically result from test logic insertion. Constraining the unmatched ports to a constant value may correct the failures.

Recommendations:
i:/WORK/top/test_se
Unmatched in the implementation cones for 2 compare point(s):

- i:/WORK/top/Fp_reg_1_
- i:/WORK/top/Fp_reg_4_

Try adding this command before verify:
set_constant i:/WORK/top/test_se 0

Debugging Tools: Pattern Viewer

- Formality automatically creates sets of vectors to illustrate failure at the compare point
  - These counter-examples are failing patterns
  - Failing patterns are applied on the inputs of each logic cone
  - Proof of non-equivalence performed mathematically
  - No failing patterns exist for passing or hard to verify compare points
- Viewing the logic cone inputs and failing patterns are extremely helpful in debugging

Debugging Tools: Pattern Viewer

Formality (R) Console – Synopsys Inc.

File Edit View Designs Run Window Help

Reference: r:/WORK/mR4000

Implementation: i:/WORK/mR4000

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match

Failing Points | Passing Points | Aborted Points | Unverified Points | Errors

	Type	Reference	Size	Implementation
28	DFF	Instruction_reg_8_	3	Instruction_reg_8_
29	DFF	Instruction_reg_9_	3	Instruction_reg_9_
30	DFF	Instruction_reg_7_	3	Instruction_reg_7_
31	DFF	Instruction_reg_0_	3	Instruction_reg_0_
32	DFF	Instruction_reg_3_	3	Instruction_reg_3_
33	DFF	ALUOp_reg_1_	195	ALUOp_reg_1_
34	DFF	ALUSelB_reg_0_	195	ALUSelB_reg_0_
35	DFF	IRWrite_reg	195	IRWrite_reg
36	DFF	ALUSelB_reg_1_	195	ALUSelB_reg_1_

Number of Failing Points: 252 Display names: Original Mapped

Filter:

Diagnose | Diagnose Selected Points

Implementation design: i:/WORK/mR4000
98 Passing compare points
262 Failing compare points
0 Aborted compare points

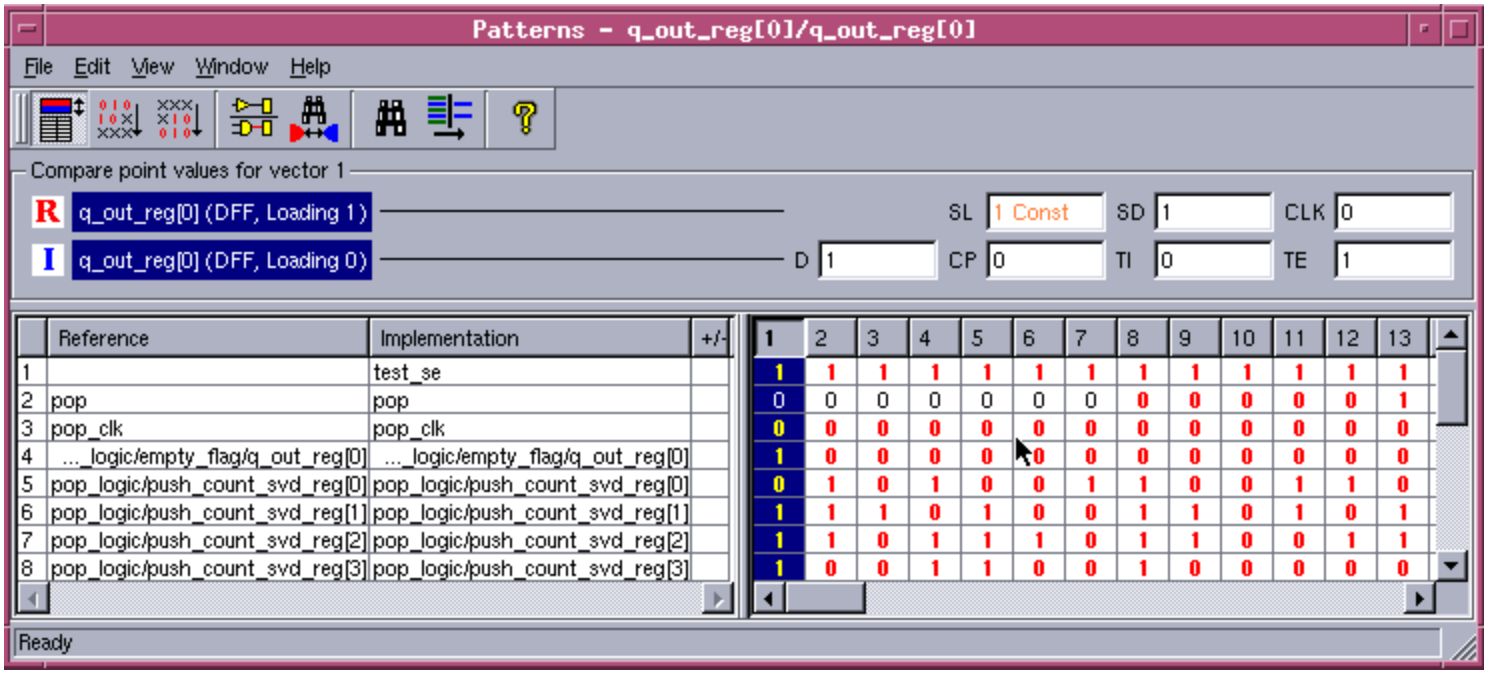
Log | Errors | Warnings | History | Last Command

Formality (verify)>

Ready Shell State: verify

- Show Logic Cones
- Show Selected Cone Sizes
- Show All Cone Sizes
- Show Patterns
- Show Matching Tool
- View Reference Object
- View Implementation Object
- View Reference Source
- View Implementation Source
- Set Don't Verify
- Analyze
- Analyze Selected
- Diagnose
- Diagnose Selected Points
- Copy

Debugging Tools: Pattern Viewer



- Allows quick identification of issues with setup and matching
  - For this example, note failure when scan enable “test_se” has “1” value
  - Try using “set_constant $impl/test_se 0” to get a successful verification

Debugging Tools: Pattern Viewer

Failing Compare Point values annotated

Vector annotated in schematic (logic cone view)

Patterns - q_out_reg[0]/q_out_reg[0]

File Edit View Window Help

Compare point values for vector 1

R q_out_reg[0] (DFF, Loading 1)

I q_out_reg[0] (DFF, Loading 0)

SL 1 Const SD 1 CLK 0

D 1 CP 0 TI 0 TE 1

	Reference	Implementation	+/-
1		test_se	
2	pop	pop	
3	pop_clk	pop_clk	
4	..._logic/empty_flag/q_out_reg[0]	..._logic/empty_flag/q_out_reg[0]	
5	pop_logic/push_count_svd_reg[0]	pop_logic/push_count_svd_reg[0]	
6	pop_logic/push_count_svd_reg[1]	pop_logic/push_count_svd_reg[1]	
7	pop_logic/push_count_svd_reg[2]	pop_logic/push_count_svd_reg[2]	
8	pop_logic/push_count_svd_reg[3]	pop_logic/push_count_svd_reg[3]	

1	2	3	4	5	6	7	8	9	10	11	12	13
1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0		0	1
0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0		0
0	1	0	1	0	0	1	1	0	0	1		0
1	1	1	0	1	0	0	1	1	0	1	0	
1	1	0	1	1	1	0	1	1	0	0	1	
1	0	0	1	1	0	0	1	0	0	0	0	0

Ready

Cone Schematics - q_out_reg[0]/q_out_reg[0]

Schematic Edit View Window Help

Color Mode Error Candidates

Reference>>> pop_logic/empty_flag/q_out_reg[0]

pop

0

pop_logic/pop

pop_logic/empty_flag/q_out_reg[0]

c1 SL Q 1 pop_logic/empty_flag/q_out[0]

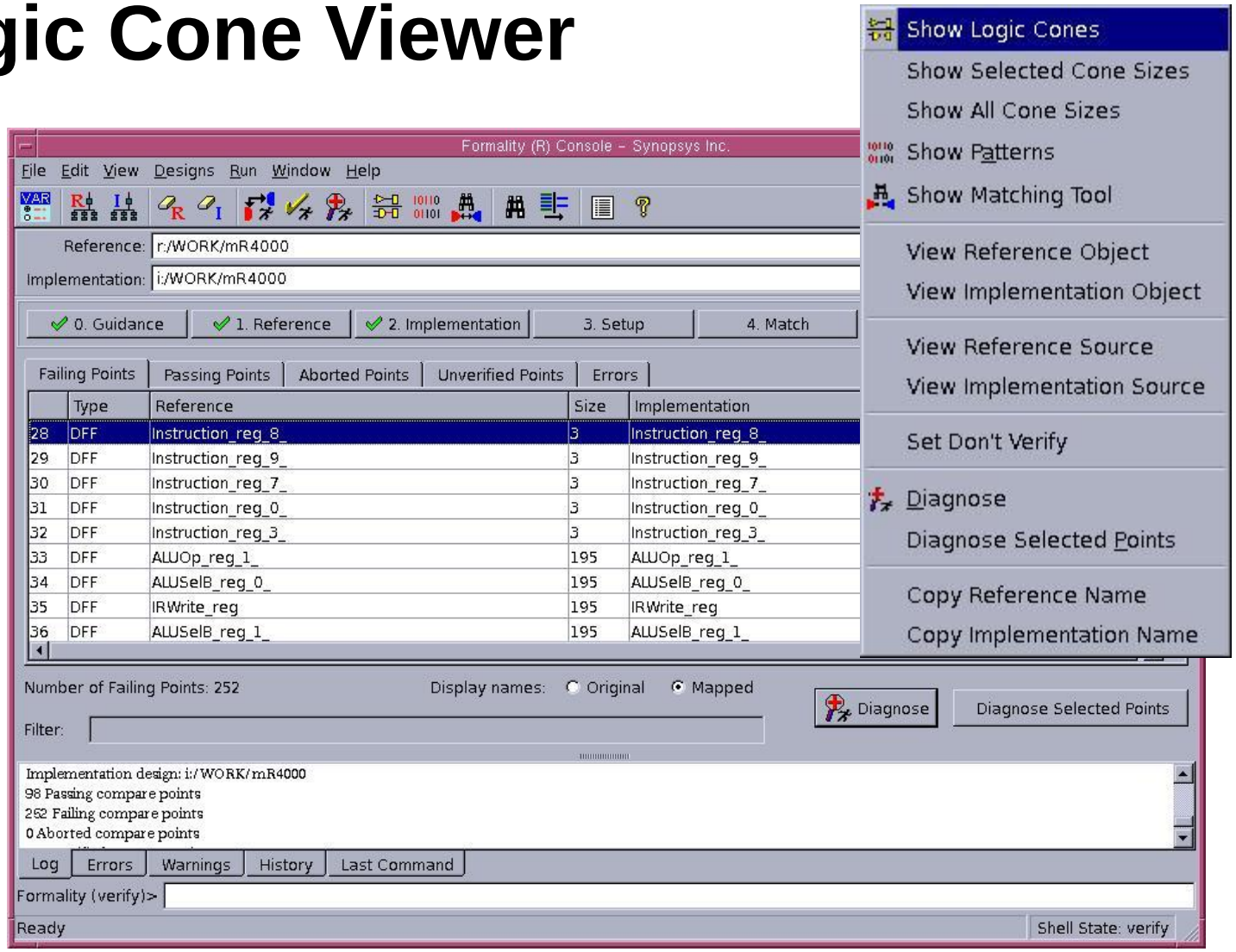
1 SD

0 CLK

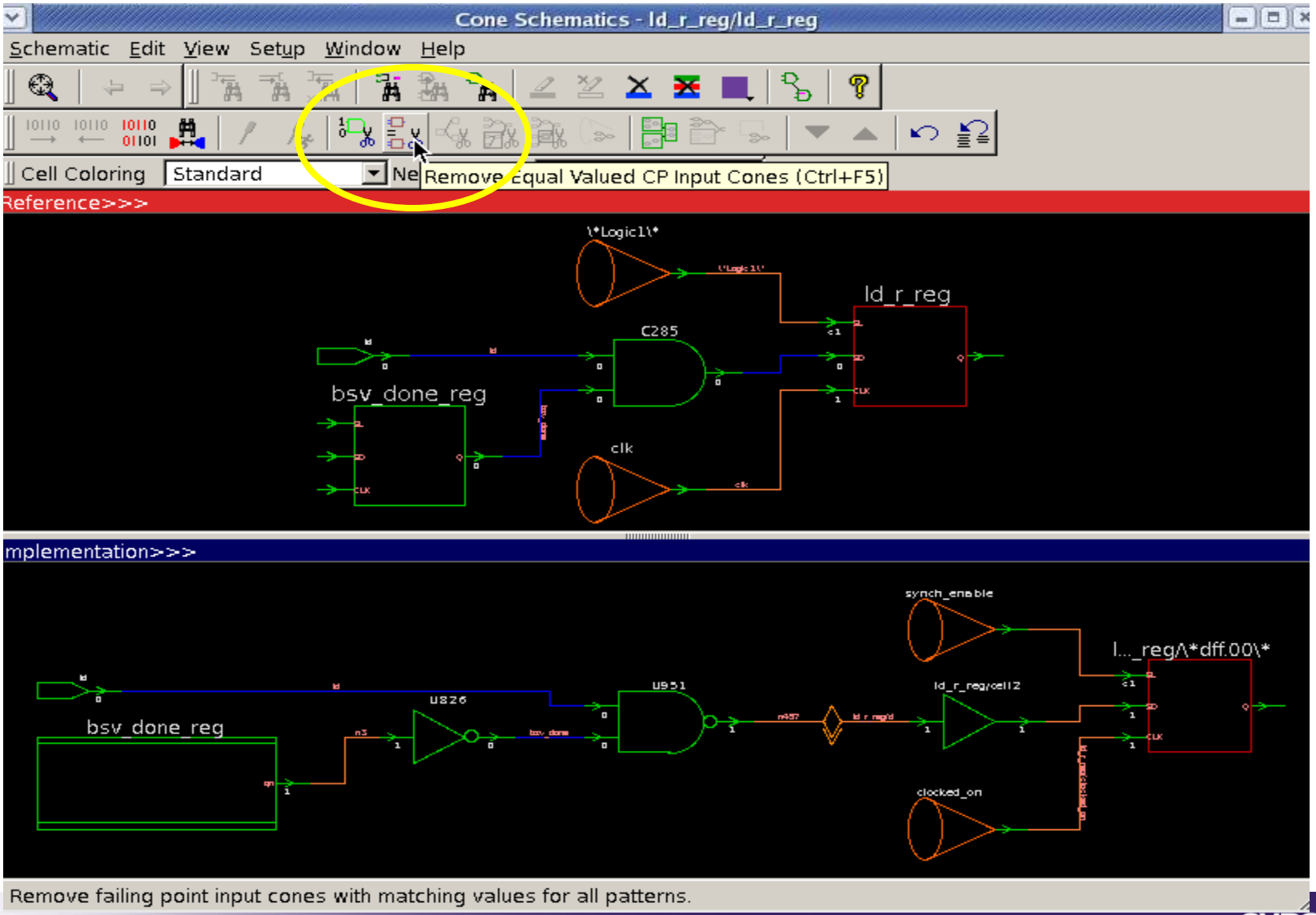
pop_logic/empty_flag/q_out_reg[0]
(SEQ)
Loading 1
SL = synchronous load (enable)
SD = synchronous data

Ready

Debugging Tools: Logic Cone Viewer

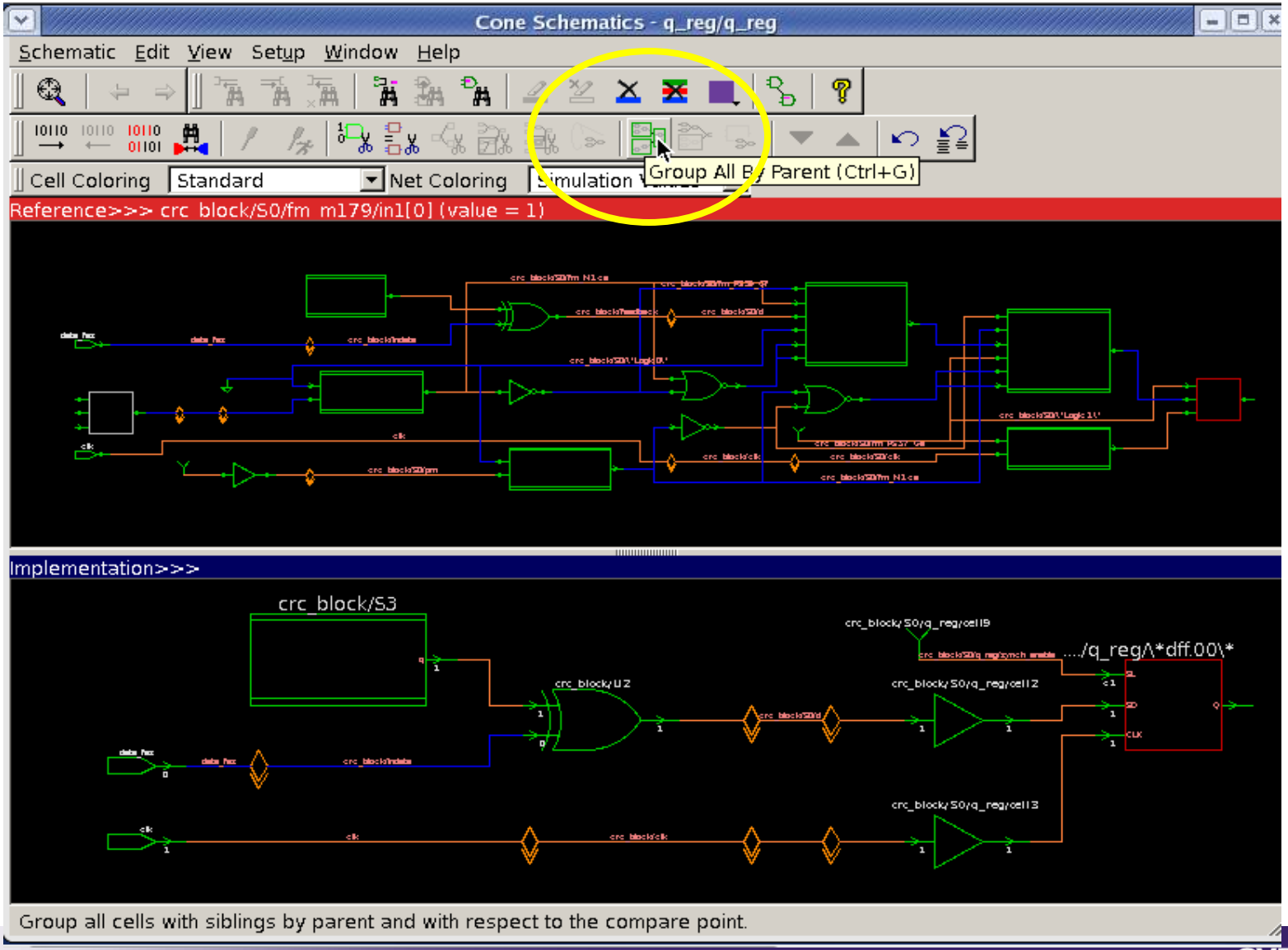


Debugging Tools: Logic Cone Viewer



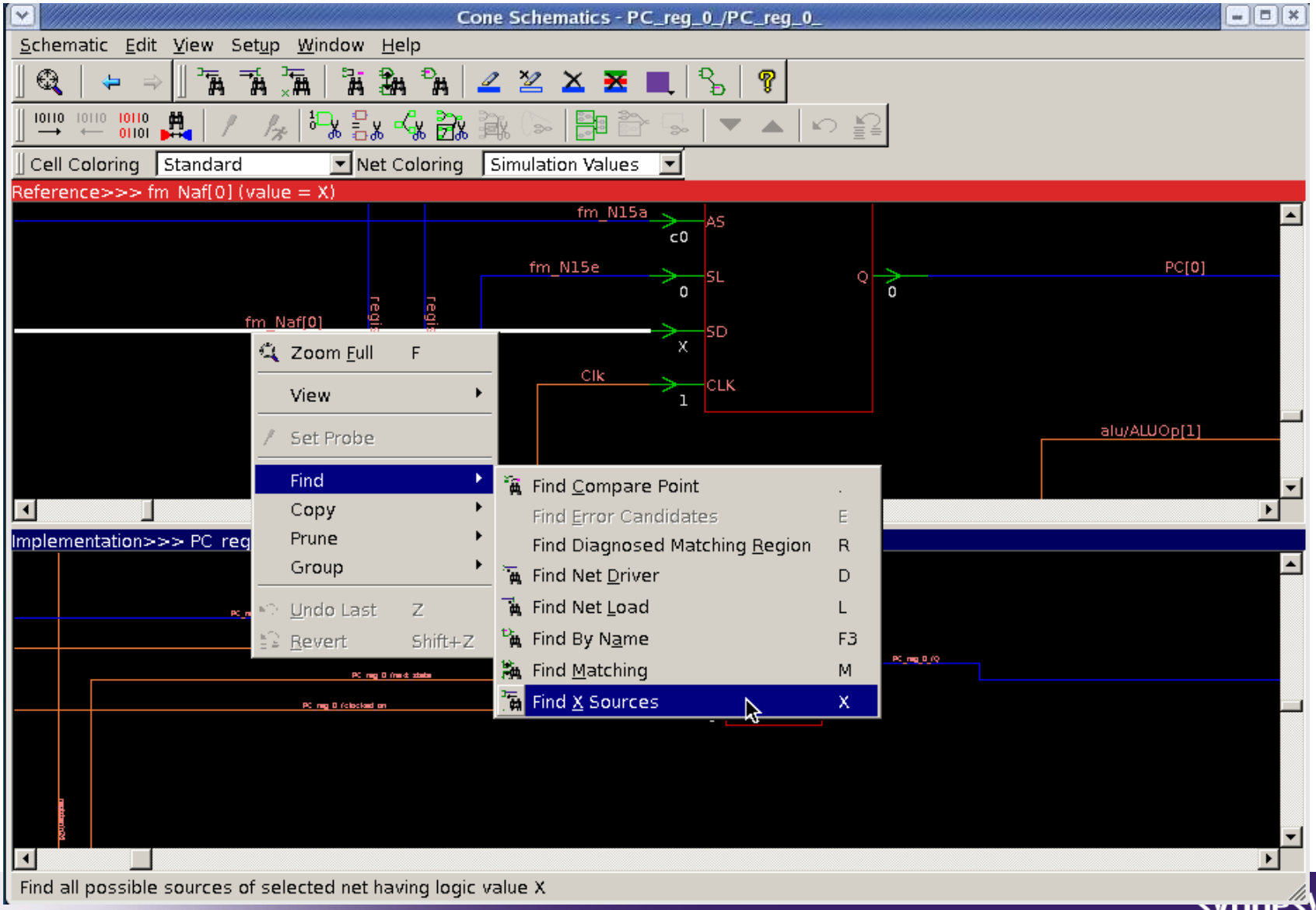
SYNOPSYS® Predictable Success

Debugging Tools: Logic Cone Viewer



Predictable Success

Debugging Tools: Logic Cone Viewer



Predictable Success

Debugging Tools: Logic Cone Viewer

Cone Schematics - q_reg/q_reg

Schematic Edit View Setup Window Help

Cell Coloring Standard Net Coloring Simulation Values

Reference>>> crc_block/S0/eq_17/in1[0] (value = 1)

Implementation>>> crc_block/S0/q_reg/CP (value = 1

Zoom Full F
View
Set Probe
Find
Copy
Prune
Group
Undo Last Z
Revert Shift+Z

Find Compare Point .
Find Error Candidates E
Find Diagnosed Matching Region R
Find Net Driver D
Find Net Load L
Find By Name F3
Find Matching M
Find X Sources X

Find driver of current net

Debugging Tools: Logic Cone Viewer

Cone Schematics - q_reg/q_reg

Schematic Edit View Setup Window Help

Cell Coloring Standard Net Coloring Simulation Values

Reference>>> crc_block/S0/*Logic1* (value = Const 1)

Zoom Full F
View
Set Probe
Find
Copy
Prune
Group
Undo Last Z
Revert Shift+Z

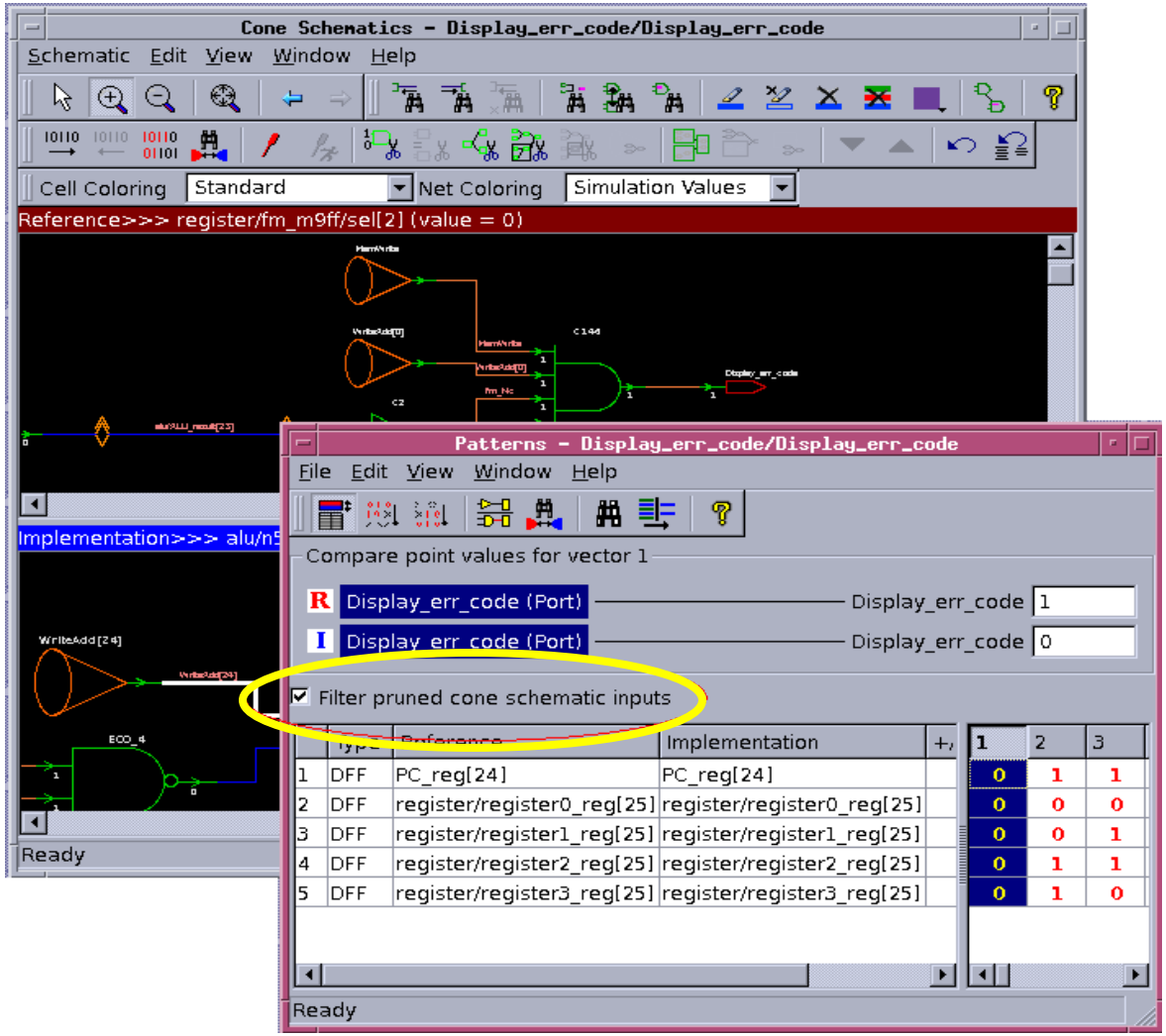
Remove Non-Controlling F5
Remove Equal Valued CP Inputs Ctrl+F5
Remove Subcone F6
Isolate Subcone F7
Isolate Error Candidates F8
Return Subcone Ctrl+F6

Implementation>>> crc_block/S0/q_reg/synch_enable (value = Const 1)

crc_block/S3/q_reg
crc_block/U2
crc_block/S0/q_reg/cell2
crc_block/S0/q_reg/cell3

Remove subcones of selected nets.

Pruning Correlation From Logic Cone to Pattern Viewer



Queued Setup Commands

Cone Schematics - text_in_r_reg_112_/text_in_r_reg_112_

Schematic Edit View Setup Window Help

Set Constant 0
Set Constant 1
Set Black Box
Set Cutpoint
Set Clock
Remove Black Box
Remove Constant
Remove Cutpoint
Remove Clock

Cell Coloring Standa ng Simulation Values

Reference>>>

Command Queue

set constant -type port i:/WORK/aes_cipher_top/test_se 0

Clear Queue | Delete Selected | Execute Queue | Cancel

Implementation>>> test_se

test_se
1
test_se
CLK
1
U2681
CN
0
n2174
CLK
0
U2678
CN
1
n2171
bsv_done_reg

Ready

Agenda

- Debugging Flow
- Frequently Used Debugging Tools
- Common Problems
- Additional Debugging Tools
- Labs

Common Problems

- Design Read – Reading/linking libraries or designs
- Match – Incorrectly matched compare points, unmatched compare points, rejected SVF commands, scan mode not disabled
- Verification – Clock-gating circuitry not recognized, logic differences

Design Read

- Problems reading and linking libraries or designs
  - Simulation/synthesis mismatch errors
  - Synthesis pragmas full_case and parallel_case
  - RTL has instantiated DesignWare components but `hdlin_dwroot` not set
  - Undriven signals
  - Black boxes in one design but not in the other one

Design Read: Simulation/Synthesis Mismatch

- Performs conservative RTL interpretation by default
  - Formality stops processing when detects difference between simulation and synthesis
- Simulation/synthesis mismatch error messages:

  Warning: /users/training/lab2/rtl/DCT8_final.vhd line 1059
  Default initial value of signal will be ignored (FMR_VHDL-1002)
  Error: RTL interpretation messages were produced during read.
  Verification results may disagree with a logic simulator. (FM-089)
- Convert error messages into warning messages:

  -> `set hdlin_warn_on_mismatch_message "FMR_VHDL-1002"`
- Set variable before reading in the RTL into a container
- Investigate these differences before tape-out
- Auto Setup Mode uses this global setting
  - `set hdlin_error_on_mismatch_message false`

Design Read: Synthesis Pragmas

- Parallel case and full case synthesis pragmas

```
case (WriteRegSel)  // synopsys parallel_case full_case
```

- Formality ignores pragmas by default
  - Same as simulation
- Transcript information after `set_top` completes

```
************ RTL Interpretation Summary ************
************ Design: r:/WORK/mR4000
full_case ignored (7 total, 1 with unspecified cases)
parallel_case ignored (7 total, 1 with overlapping cases)

Please refer to the Formality log file for more details,
or execute report_hdlin_mismatches.
**************************************************
```

Design Read: Synthesis Pragmas

- See article "*full_case parallel_case, the Evil Twins of Verilog Synthesis*" by Clifford Cummings
  http://www.sunburst-design.com/papers/CummingsSNUG1999Boston_FullParallelCase.pdf
- Should examine RTL to ensure "case" statements are parallel or fully specified
- Use Formality variables:

```
set hdlin_ignore_full_case false
set hdlin_ignore_parallel_case false
```

- Or, use `set synopsys_auto_setup true`

Synthesis Pragma Issue: Analyze

Formality (R) Console - Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/mR4000

Implementation: i:/WORK/mR4000

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

Possible Failure Causes

- Unmatched Cone Input (17)
  - r:/WORK/mR4000/state_reg_0
  - r:/WORK/mR4000/state_reg_1
  - r:/WORK/mR4000/state_reg_2
  - r:/WORK/mR4000/state_reg_3
  - r:/WORK/mR4000/state_reg_4
  - r:/WORK/mR4000/state_reg_5
  - r:/WORK/mR4000/state_reg_6
  - r:/WORK/mR4000/state_reg_7
  - r:/WORK/mR4000/state_reg_8
  - r:/WORK/mR4000/state_reg_9
  - r:/WORK/mR4000/state_reg_10
  - r:/WORK/mR4000/Instruction_reg_29
  - r:/WORK/mR4000/Instruction_reg_26
  - r:/WORK/mR4000/Instruction_reg_27
  - r:/WORK/mR4000/Instruction_reg_28
  - r:/WORK/mR4000/Instruction_reg_30
  - r:/WORK/mR4000/Instruction_reg_31

Description - Unmatched Cone Input:
Unmatched cone inputs result either from mismatched compare points or from differences in the logic within the cones. Only unmatched inputs that are suspected of contributing to verification failures are included in the report.
The source of the matching or logical differences may be determined using the schematic. cone and source views.

Recommendations:
i:/WORK/mR4000/state_reg_0
Matched with cell i:/WORK/mR4000/state_reg_0_/*dff.00*
Exists in the ref cone but not in the impl cone for 23 compare point(s)

- i:/WORK/mR4000/ALUOp_reg_0
- i:/WORK/mR4000/ALUOp_reg_1
- i:/WORK/mR4000/ALUSelA_reg
- i:/WORK/mR4000/ALUSelB_reg_1
- i:/WORK/mR4000/IRWrite_reg
- i:/WORK/mR4000/IorD_reg
- i:/WORK/mR4000/MemWrite_reg
- i:/WORK/mR4000/MemtoReg_reg

Synthesis Pragma Issue: Pattern Viewer

Patterns - ALUOp_reg_0_/ALUOp_reg_0_

File Edit View Window Help

Compare point values for vector 1

R	ALUOp_reg_0_ (DFF. Loading 0)	SL	1 Const	SD	0	CLK 1
I	ALUOp_reg_0_/*dff.00* (DFF. Loading 1)	SL	1 Const	SD	1	CLK 1

Filter pruned cone schematic inputs

Appears to be a bad logic problem

	Type	Reference	Im				
1	DFF	r:/WORK/mR4000/state_reg_0_		0	0	0	0
2	DFF	r:/WORK/mR4000/state_reg_1_		0	0	0	0
3	DFF	r:/WORK/mR4000/state_reg_2_		1	0	1	0
4	DFF	r:/WORK/mR4000/state_reg_3_		0	0	0	0
5	DFF	r:/WORK/mR4000/state_reg_4_		0	0	0	0
6	DFF	r:/WORK/mR4000/state_reg_5_		0	0	0	0
7	DFF	r:/WORK/mR4000/state_reg_6_		0	1	0	0
8	DFF	r:/WORK/mR4000/state_reg_7_		0	0	0	1
9	DFF	r:/WORK/mR4000/state_reg_10_		0	0	0	0
10	DFF	r:/WORK/mR4000/ALUOp_reg_0_	...K/mR4000/ALUOp_reg_0_/*dff.00*	0	0	0	0
11	Port	r:/WORK/mR4000/Clk	i:/WORK/mR4000/Clk	1	1	1	1
12	Port	r:/WORK/mR4000/Reset	i:/WORK/mR4000/Reset	0	0	0	0
13	DFF	r:/WORK/mR4000/state_reg_8_	i:/WORK/mR4000/state_reg_8_	0	0	1	0
14	DFF	r:/WORK/mR4000/state_reg_9_	i:/WORK/mR4000/state_reg_9_	1	1	0	1

Ready

Design Read: Instantiated DesignWare

- Set variable **hdlin_dwroot** to top-level of DC installation
  - Example of transcript when variable is not set:

```
fm_shell> set_top chip
Setting top design to 'r:/WORK/chip'
Status:   Elaborating design chip   ...
Status:   Elaborating design pll   ...
Status:   Elaborating design ff   ...
Status:   Elaborating design dp   ...
Warning: Cannot link cell '/WORK/dp/u0' to its reference design 'DW01_add'. (FE-LINK-2)
Warning: Cannot link cell '/WORK/dp/u1' to its reference design 'DW01_add'. (FE-LINK-2)
Warning: Cannot link cell '/WORK/dp/u2' to its reference design 'DW01_add'. (FE-LINK-2)
Warning: Cannot link cell '/WORK/dp/u3' to its reference design 'DW01_add'. (FE-LINK-2)
Status:   Elaborating design cntrl   ...
Error: Unresolved references detected during link. (FM-234)
Error: Failed to set top design to 'r:/WORK/chip' (FM-156)
```

Design Read: Undriven Signals

- Undriven signals may control a downstream compare point which will cause failures
- Transcript information:

```
(Matching)
Status:  Checking designs...
    Warning: 603 (21) undriven nets found in reference (implementation) design;
    see formality.log for list (FM-399)
…
-----------------------------------------------------------------------
Unmatched Objects                                          REF        IMPL
-----------------------------------------------------------------------
 Cut-points (Cut)                                          603           0
…
(Verification Results)
ATTENTION: 84 failing compare points have unmatched undriven signals in their
                 reference fan-in.  To report such failing points, use
                 "report_failing_points -inputs unmatched -inputs undriven".
               To read about undriven signal handling, use
                 "man verification_set_undriven_signals".
```

Undriven Signals: Analyze

Formality (R) Console - Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/DCT8_final

Implementation: i:/WORK/DCT8_final

0. Guidance | 1. Reference | 2. Implementatio | 3. Setup | 4. Match | 5. Verify | 6. Debug

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

Possible Failure Causes

- Undriven Reference Signal (142)
  - r:/WORK/DCT8_final/I0/I3/I0/gm[1]
  - r:/WORK/DCT8_final/I0/I3/I0/fm[1]
  - r:/WORK/DCT8_final/I0/I3/I0/em[1]
  - r:/WORK/DCT8_final/I0/I3/I0/dm[1]
  - r:/WORK/DCT8_final/I0/I3/I0/qp[1]
  - r:/WORK/DCT8_final/I0/I3/I0/fp[1]
  - r:/WORK/DCT8_final/I0/I3/I0/ep[1]
  - r:/WORK/DCT8_final/I0/I3/I0/dp[1]
  - r:/WORK/DCT8_final/I0/I3/I0/qm[2]
  - r:/WORK/DCT8_final/I0/I3/I0/fm[2]
  - r:/WORK/DCT8_final/I0/I3/I0/em[2]
  - r:/WORK/DCT8_final/I0/I3/I0/dm[2]
  - r:/WORK/DCT8_final/I0/I3/I0/qp[2]
  - r:/WORK/DCT8_final/I0/I3/I0/fp[2]
  - r:/WORK/DCT8_final/I0/I3/I0/ep[2]
  - r:/WORK/DCT8_final/I0/I3/I0/dp[2]
  - r:/WORK/DCT8_final/I0/I3/I0/qm[3]

Description - Undriven Reference Signal:
An undriven signal in the reference design may be caused by either a legitimate 'dont care' condition or an error in the RTL.
If an examination of the RTL finds no unexpected undriven signals try '***set verification_set_undriven_signals 0:X***' to match synthesis.

Recommendations:
r:/WORK/DCT8_final/I0/I3/I0/gm[1]
Undriven in the reference cones for 1 compare point(s):

- r:/WORK/DCT8_final/I0/I3/I2/aout_reg_1

Undriven Signals: Pattern Viewer

- **Failing Pattern Window shows undriven signal names with the text "(originally undriven)"**

Patterns - aout_reg_1_/aout_reg_1_

File Edit View Window Help

Compare point values for vector 1

R aout_reg_1_ (DFF. Loading 1) — AC 0 AS 0 Const SL 1 Const SD 1 CLK 1

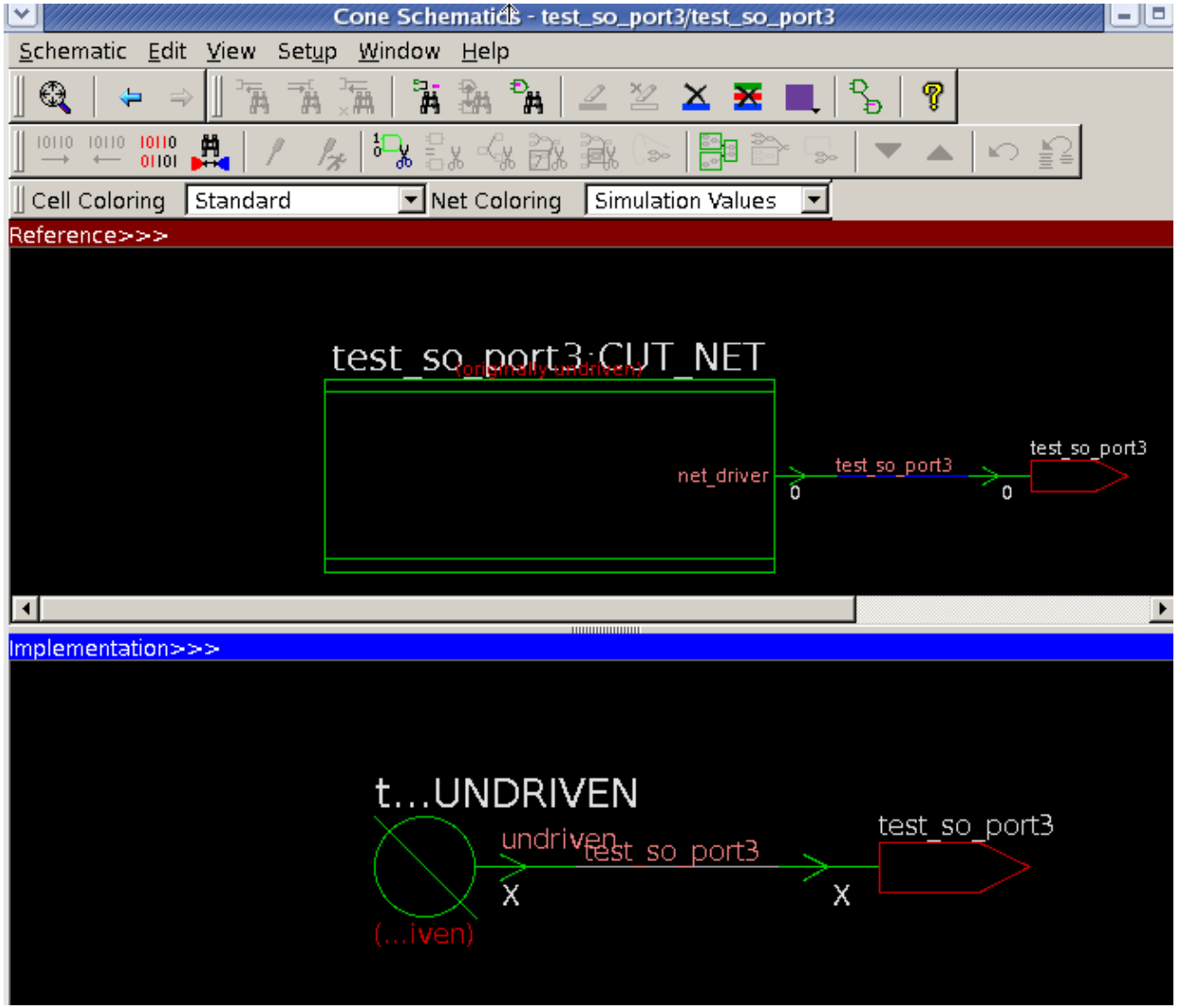
I I0/I3/I2/aout_reg_1_/*dff.00* (DFF0. Loading 0 (Constant)) — AC 0 SL 1 Const SD 0 Const CLK 1

Filter pruned cone schematic inputs

	Type	Reference	Implementation	+/	1	2	3	4	5	6	7	8
1	DFF	r:/WORK/DCT8_final/I0/I2/address_reg_0_			1	0	1	0	1	0	0	1
2	DFF	r:/WORK/DCT8_final/I0/I2/address_reg_1_			0	0	1	1	1	0	1	0
3	DFF	r:/WORK/DCT8_final/I0/I2/compl_reg			1	1	0	0	1	0	1	0
4	CutNet	r:/WORK/DCT8_final/I0/I3/I0/dm[1] (Originally undriven)			0	1	0	0	0	0	0	0
5	CutNet	r:/WORK/DCT8_final/I0/I3/I0/dp[1] (Originally undriven)			0	0	0	0	0	1	0	1
6	CutNet	r:/WORK/DCT8_final/I0/I3/I0/em[1] (Originally undriven)			1	0	0	0	0	0	0	0
7	CutNet	r:/WORK/DCT8_final/I0/I3/I0/ep[1] (Originally undriven)			0	0	0	0	0	0	0	1
8	CutNet	r:/WORK/DCT8_final/I0/I3/I0/fm[1] (Originally undriven)			0	0	0	0	0	0	1	0
9	CutNet	r:/WORK/DCT8_final/I0/I3/I0/fp[1] (Originally undriven)			0	0	0	1	0	0	0	0
10	CutNet	r:/WORK/DCT8_final/I0/I3/I0/gm[1] (Originally undriven)			0	0	0	0	1	0	0	0
11	CutNet	r:/WORK/DCT8_final/I0/I3/I0/gp[1] (Originally undriven)			0	0	1	0	0	0	0	0
12	Port	r:/WORK/DCT8_final/bit_in_odd			1	1	1	1	1	1	1	1
13	DFF	r:/WORK/DCT8_final/I0/I2/enable0_reg	...K/DCT8_final/I0/I2/enable0_reg		1	1	1	1	1	1	1	1
14	DFF/...	r:/WORK/DCT8_final/I0/I3/I2/aout_reg_1_	...al/I0/I3/I2/aout_reg_1_/*dff.00*		0	0	0	0	0	0	0	0
15	Port	r:/WORK/DCT8_final/clk	i:/WORK/DCT8_final/clk		1	1	1	1	1	1	1	1

Ready

Undriven Signals: Logic Cone



Undriven Signals: Variable Settings

- Variable `verification_set_undriven_signals`
  - Default value: `BINARY:X`
- New mode `synthesis` results in fewer failing verifications but will still detect undriven signals in implementation

```
set verification_set_undriven_signals synthesis
```

  - Treats undriven signals:
    - Reference the same as Design Compiler
    - Implementation as binary
- New mode `synthesis` used in Auto Setup Mode
- Command to ignore undriven failures on output ports (common for test output ports)

```
set_dont_verify -directly_undriven_output
```

- For more information try `report_undriven_nets`

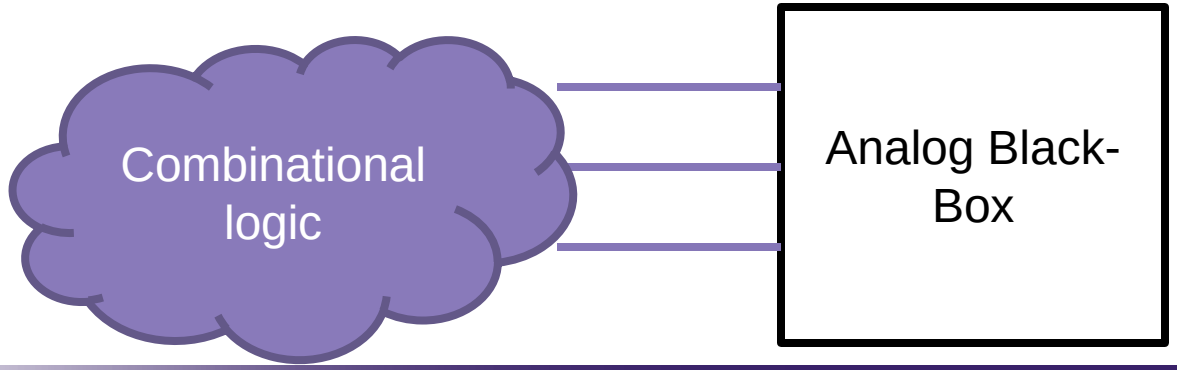
Design Read: Black-Box

- Black-boxes are a problem if one container has one and the other container has the design
- Formality can create black-boxes automatically if design is missing

  `set hdlin_unresolved_modules black_box`
- Some design modules have only I/O port declarations without behavior
  - Formality will perform best guess if I/O port direction are not specified
  - Use command `set_direction` to change if needed
- Memory library cells may only contain port and timing arcs

Design Read: Black-Box

- Can specify black-boxes using variable
  `set hdlin_interface_only "SRAM* dram16x8"`
  - Any module beginning with SRAM and the dram16x8 module will become a black-box
- Can use command `set_black_box designID`
- Command `report_black_boxes` shows list of black-boxes
- Note: Variable `verification_verify_unread_bbox_inputs` has default value of `true`



Design Read: Black-Box

- Transcript clues:

```
Status:   Elaborating design wb_dma_wb_slv  0 ...
Status:  Implementing inferred operators...
Status:  Creating black-box designs...
Created technology library 'FM_BBOX' in container 'r' for black-box designs
Created black-box design 'wb_dma_de' in library 'FM_BBOX'
Warning: 1 blackbox designs were created for missing references. (FM-064)
Status:  Attempting to resolve unlinked cells by using black-boxes...
Warning: 585 black-box pins of unknown direction found; see formality.log for list (FM-230)
Top design set to 'r:/WORK/wb_dma_top' with warnings
```

```
************************************ Matching Results ************************************
 550 Compare points matched by name
 0 Compare points matched by signature analysis
 0 Compare points matched by topology
 216 Matched primary inputs, black-box outputs
 430(199) Unmatched reference(implementation) compare points
 222(0) Unmatched reference(implementation) primary inputs, black-box outputs
 5200(0) Unmatched reference(implementation) unread points
-------------------------------------------------------------------------------------
Unmatched Objects                                                    REF        IMPL
-------------------------------------------------------------------------------------
 Black-boxes (BBox)                                                    1           0
 Black-box input pins (BBPin)                                        330           0
 Black-box output pins (BBPin)                                       222           0
 Registers                                                           100         199
   DFF                                                                 0         197
```

Black-Box: Match Tab

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Compare Rule Setup | User Match Setup | Matched Points | Unmatched Points | Summary

	Reference Object	Type
1	r:/WORK/wb_dma_top/u2/mast1_din[16]	BBPin
2	r:/WORK/wb_dma_top/u2/mast1_din[15]	BBPin
3	r:/WORK/wb_dma_top/u2/mast1_din[14]	BBPin
4	r:/WORK/wb_dma_top/u2/mast1_din[13]	BBPin
5	r:/WORK/wb_dma_top/u2/mast1_din[12]	BBPin
6	r:/WORK/wb_dma_top/u2/mast1_din[11]	BBPin
7	r:/WORK/wb_dma_top/u2/mast1_din[10]	BBPin
8	r:/WORK/wb_dma_top/u2/mast1_din[0]	BBPin
9	r:/WORK/wb_dma_top/u2/mast1_adr[9]	BBPin
10	r:/WORK/wb_dma_top/u2/mast1_adr[8]	BBPin
11	r:/WORK/wb_dma_top/u2/mast1_adr[7]	BBPin
12	r:/WORK/wb_dma_top/u2/mast1_adr[6]	BBPin
13	r:/WORK/wb_dma_top/u2/mast1_adr[5]	BBPin

Set User Match

	Implementation Object	Type
121	i:/WORK/wb_dma_top/u2/mast1_adr_reg_16_	DFF
122	i:/WORK/wb_dma_top/u2/mast1_adr_reg_17_	DFF
123	i:/WORK/wb_dma_top/u2/mast1_adr_reg_18_	DFF
124	i:/WORK/wb_dma_top/u2/mast1_adr_reg_19_	DFF
125	i:/WORK/wb_dma_top/u2/mast1_adr_reg_20_	DFF
126	i:/WORK/wb_dma_top/u2/mast1_adr_reg_21_	DFF
127	i:/WORK/wb_dma_top/u2/mast1_adr_reg_22_	DFF
128	i:/WORK/wb_dma_top/u2/mast1_adr_reg_23_	DFF
129	i:/WORK/wb_dma_top/u2/mast1_adr_reg_24_	DFF
130	i:/WORK/wb_dma_top/u2/mast1_adr_reg_25_	DFF
131	i:/WORK/wb_dma_top/u2/mast1_adr_reg_26_	DFF
132	i:/WORK/wb_dma_top/u2/mast1_adr_reg_27_	DFF
133	i:/WORK/wb_dma_top/u2/mast1_adr_reg_28_	DFF

Number of unmatched reference points: 651 Number of unmatched implementation points: 199 Display names: Original Mapped

Filter on reference list: Filter on implementation list:

Run Matching

Black-Box: Analyze

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

Possible Failure Causes

- Unmatched Cone Input (63)
  - r:/WORK/wb_dma_top/u2/de_adr1_we
  - i:/WORK/wb_dma_top/u2/adr1_cnt_req_14_/*dff.00*
  - i:/WORK/wb_dma_top/u0/u0/ch_adr1_r_req_14_/*d...
  - i:/WORK/wb_dma_top/u3/u0/mast_dout_req_16_/*...
  - i:/WORK/wb_dma_top/rst_i
  - i:/WORK/wb_dma_top/u2/adr1_cnt_req_15_/*dff.00*
  - i:/WORK/wb_dma_top/u0/u0/ch_adr1_r_req_15_/*d...
  - i:/WORK/wb_dma_top/u3/u0/mast_dout_req_17_/*...
  - i:/WORK/wb_dma_top/u2/adr1_cnt_req_16_/*dff.00*
  - i:/WORK/wb_dma_top/u0/u0/ch_adr1_r_req_16_/*d...
  - i:/WORK/wb_dma_top/u3/u0/mast_dout_req_18_/*...
  - i:/WORK/wb_dma_top/u2/adr1_cnt_req_17_/*dff.00*
  - i:/WORK/wb_dma_top/u0/u0/ch_adr1_r_req_17_/*d...
  - i:/WORK/wb_dma_top/u3/u0/mast_dout_req_19_/*...
  - r:/WORK/wb_dma_top/u2/dma_done_all
  - i:/WORK/wb_dma_top/u2/chunk_cnt_req_6_/*dff.00*
  - i:/WORK/wb_dma_top/u2/chunk_cnt_req_7_/*dff.00*

Description - Unmatched Cone Input:
Unmatched cone inputs result either from mismatched compare points or from differences in the logic within the cones. Only unmatched inputs that are suspected of contributing to verification failures are included in the report.
The source of the matching or logical differences may be determined using the schematic, cone and source views.

Recommendations:
r:/WORK/wb_dma_top/u2/de_adr1_we
Is globally unmatched affecting 4 compare point(s):

- i:/WORK/wb_dma_top/u0/u0/ch_adr1_r_req_14
- i:/WORK/wb_dma_top/u0/u0/ch_adr1_r_req_15
- i:/WORK/wb_dma_top/u0/u0/ch_adr1_r_req_16
- i:/WORK/wb_dma_top/u0/u0/ch_adr1_r_req_17

Black-Box: Pattern Viewer

Patterns - ch_adr1_r_reg_0_/ch_adr1_r_reg_0_

File Edit View Window Help

Compare point values for vector 1

R	ch_adr1_r_reg_0_ (DFF, Holding 0)	SL	0	SD	0	CLK 1
I	u0/u0/ch_adr1_r_reg_0_/*dff.00* (DFF, Loading 1)	SL	1 Const	SD	1	CLK 1

Filter pruned cone schematic inputs

Note the unmatched pin inputs

	Type	Reference	Imp	+/-	1	2	3	4	5
10	Pin	r:/WORK/wb_dma_top/u2/de_adr1[2]			0	0	0	0	
11	Pin	r:/WORK/wb_dma_top/u2/de_adr1_we			0	0	0	0	
12	Pin	r:/WORK/wb_dma_top/u2/dma_busy			0	0	0	0	
13	DFF		i:/WORK/wb_dma_top/u2/state_reg_9_		0	0	0	0	
14	DFF		i:/WORK/wb_dma_top/u2/state_reg_10_		0	0	0	0	
15	DFF		...ORK/wb_dma_top/u3/u0/mast_dout_reg_2_		1	1	1	1	
16	DFF		i:/WORK/wb_dma_top/u2/adr1_cnt_reg_0_		0	0	0	0	
17	DFF		i:/WORK/wb_dma_top/u2/state_reg_0_		0	0	0	0	
18	DFF		i:/WORK/wb_dma_top/u2/state_reg_8_		0	0	0	0	
19	DFF		i:/WORK/wb_dma_top/u2/state_reg_1_		0	0	0	0	
20	DFF		i:/WORK/wb_dma_top/u2/state_reg_6_		0	0	0	0	
21	DFF		i:/WORK/wb_dma_top/u2/state_reg_5_		0	0	1	0	
22	DFF		i:/WORK/wb_dma_top/u2/state_reg_7_		1	1	0	0	
23	DFF		i:/WORK/wb_dma_top/u2/state_reg_2_		0	0	0	0	

Match

- Unmatched or incorrectly matched compare points
- Rejected SVF commands

Unmatched Compare Points

- Incompletely matched logic cone inputs will lead to a failing verification of the compare point
- Successful use of guidance information from DC SVF should provide complete matching “out of the box”
  - change_names, group, ungroup, and uniquify
- Examine “Matching Results” summary table in the transcript for obvious matching issues
- Some unmatched points are expected
  - Constant registers may be optimized away
  - Test inputs (primary inputs or registers)

Unmatched Compare Points

```
********************************** Matching Results **********************************
 24774 Compare points matched by name
 66 Compare points matched by signature analysis
 0 Compare points matched by topology
 1837 Matched primary inputs, black-box outputs
 805(977) Unmatched reference(implementation) compare points
 0(0) Unmatched reference(implementation) primary inputs, black-box outputs
 26841(0) Unmatched reference(implementation) unread points
-----------------------------------------------------------------------------------------
Unmatched Objects                                    REF      IMPL
-----------------------------------------------------------------------------------------
 Cut-points (Cut)                                     68         0
 Registers                                           737       977
   DFF                                                72         0
   Clock-gate LAT                                      0       959
   Constant 0                                        659         0
   Constant 1                                          6        18
*****************************************************************************************
```

These undriven signals may cause problems

These unmatched registers <u>might</u> be a problem (or may be okay)

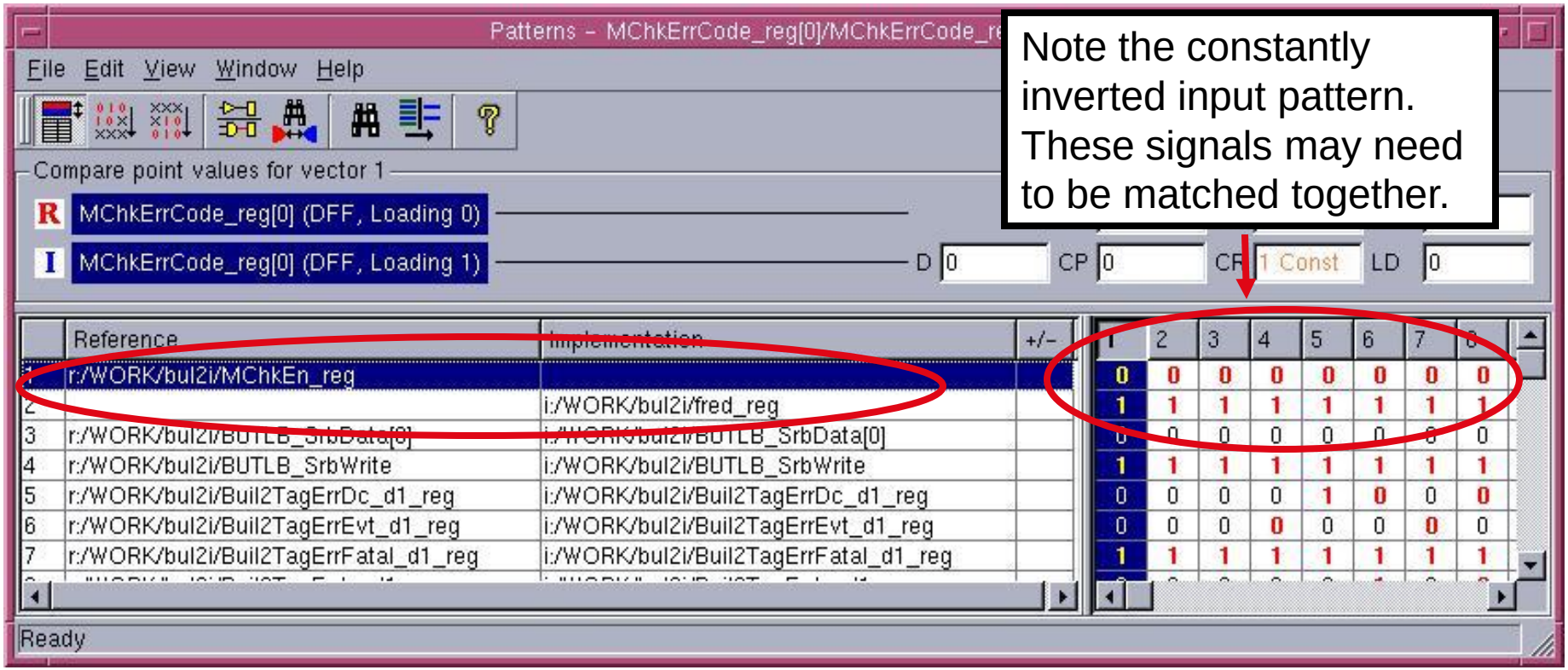
No problem with these unmatched registers

Unmatched Compare Points

- Points matched by signature analysis may indicate that SVF content was missing or rejected
  - Note: Signature analysis matching may compensate for missing SVF matching guidance. However, missing or rejected SVF content may lead to other verification issues
- Cut-points (Cut) listed as unmatched REF objects usually indicate the existence of undriven signals
- Large numbers of unmatched latches may indicate the presence of clock-gating

Unmatched Compare Points: Pattern Viewer

- Examine “Pattern Display” window in the GUI when debugging a failed point to check for matching issues



Note the constantly inverted input pattern. These signals may need to be matched together.

GUI Unmatched Point Report

Formality (R) Console – Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/tv80s

Implementation: i:/WORK/tv80s

0. Guidance | ✓ 1. Reference | ✓ 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Compare Rule Setup | User Match Setup | Matched Points | Unmatched Points | Summary

	Reference Object	Type
1	i_tv80_core/ACC_reg[0]	DFF
2	i_tv80_core/ACC_reg[1]	DFF
3	i_tv80_core/ACC_reg[2]	DFF
4	i_tv80_core/ACC_reg[3]	DFF
5		DFF
6		DFF
7	i_tv80_core/ACC_reg[6]	DFF
8	i_tv80_core/ACC_reg[7]	DFF
9	i_tv80_core/ALU_Op_r_reg[0]	DFF
10	i_tv80_core/ALU_Op_r_reg[1]	DFF

Set User Match + − ?

	Implementation Object	Type
1	ACC_reg_0_	DFF
2	ACC_reg_1_	DFF
3		
4		
5	ACC_reg_4_	DFF
6	ACC_reg_5_	DFF
7	ACC_reg_6_	DFF
8	ACC_reg_7_	DFF
9	ALU_Op_r_reg_0_	DFF
10	ALU_Op_r_reg_1_	DFF

Create a compare rule

Select two points and set match

Number of unmatched reference points: 314 Number of unmatched implementation points: 351 Display names: Original Mapped

Filter on reference list: Filter on implementation list: Run Matching

Unmatched Compare Points

- Compare rules
  - If SVF is missing or is rejected
  - Easier to globally remove part of the path name of compare points:
    - Ref: r:/WORK/tv80s/**i_tv80_core**/ALU_Op_r_reg[0]
    - Impl: i:/WORK/tv80s/ALU_Op_r_reg_0_

```
set_compare_rule $ref -from {i_tv80_core} -to {}
```

  - Formality tries different bus-bit delimiters automatically
    - No need to specify reg[0] versus reg_0_
- Command **`set_user_match`**

SVF Command Rejections Causing Failures

- `guide_fsm_reencoding` (if exists and is ignored – default mode)
- `guide_inv_push`
- `guide_multiplier` (when applied to DW_multp)
- `guide_reg_constant`
- `guide_reg_duplication`
- `guide_reg_merging`
- `guide_retiming`*
- `guide_scan`* (if you are using Auto Setup Mode)

Command: report_guidance –summary

```
**************************** Guidance Summary ****************************
                                        Status
Command                  Accepted   Rejected  Unsupported  Unprocessed  Total
---------------------------------------------------------------------------
architecture_netlist:         786          0            0            0    786
boundary             :       3062          0            0            0   3062
boundary_netlist     :        782          0            0            0    782
change_names         :      29410      15021            0            0  44431
constraints          :       1314        107            0            0   1421
datapath             :       2945        117            0            0   3062
environment          :          7          0            0            0      7
implementation       :         50          0            0            0     50
instance_map         :       2218        143            0            0   2361
inv_push             :       1098          1            0            0   1099
merge                :       2756        101            0            0   2857
multiplier           :        901        130            0            0   1031
reg_constant         :      10756          0            0            0  10756
reg_merging          :       2215          0            0            0   2215
rename_design        :         33          1            0            0     34
replace              :      11402        466            0            0  11868
scan_input           :          0          6            0            0      6
ungroup              :        582         17            0            2    601
uniquify             :       2838        220            0            0   3058
ununiquify           :          1          0            0            0      1

Note: If verification succeeds you can safely ignore unaccepted guidance commands.
```

Rejected SVF Commands

- Use command **`report_svf_operation`** to isolate the problem
- If naming concordance problem, modify SVF manually

```
_
fm_shell (verify)> report_svf_operation -status rejected r:/WORK/aes_cipher_top/us10/sbox1/dreg_reg_*_

## SVF Operation 16 (Line: 128) - inv_push.  Status: rejected
## Operation Id: 16
guide_inv_push \
  -design { aes_cipher_top } \
  -register { badname_ld_r_reg }

Info:  guide_inv_push 16 (Line: 128)  Cannot find register 'badname_ld_r_reg'..
```

Modifying SVF

- Design Compiler default name: `default.svf`
  - Compressed binary file
- DC and Formality use command `set_svf`
  - Formality automatically converts to ASCII
  - Located under: `./formality_svf/svf.txt`
- Modify ASCII file to address naming concordance issues:

```
unix> cp -r formality_svf debug_svf
unix> vi debug_svf/svf.txt
Change name of object (see next slide)
fm_shell> set_svf debug_svf/svf.txt
```

Modifying SVF

- Change SVF to match Formality
  - Use favorite text editor to change SVF name to match Formality name
- Or, change Formality to match SVF
  - Use guidance commands

    `guide_change_names` (instance name change)

    `guide_rename_design` (design name change)

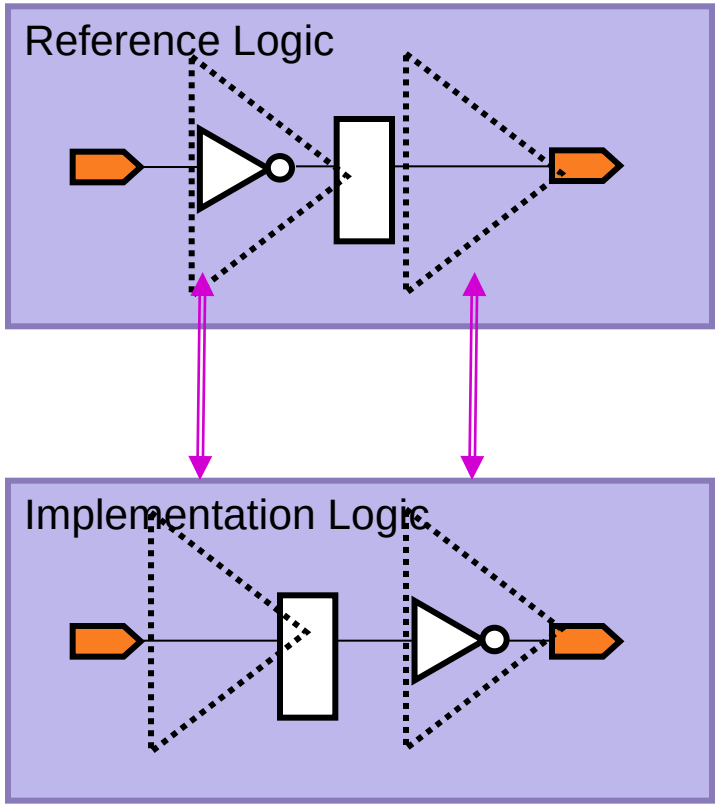
    - Add to ASCII SVF

SVF Command: guide_fsm_reencoding

- Finite State Machine re-encoding
  - Typically a FPGA only optimization
  - State registers and state encodings are captured in SVF
  - SVF content for FSMs is ignored by default because Formality cannot validate it
    - User should confirm that the original state information is complete and accurate
    - To enable:

```
set svf_ignore_unqualified_fsm_information false
```

Register Phase Inversion



▪▪▪ Aligned Cone

- Phase inversion
  - Moves inversions across register boundaries to improve performance and area
- Phase inversion impact
  - Logic cone pairs are no longer functionally equivalent
  - Verification will fail, a false difference
- Fully supported in SVF Flow

SVF Command: guide_inv_push

- DC Ultra automatically pushes inversions across register boundaries to improve QoR
- SVF flow is necessary for verification
  - DC lists the impacted registers in the SVF
  - Formality applies the `set_inv_push` command to them
  - No additional user action is required

Inversion Push: Analyze

Formality (R) Console - Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/aes_cipher_top

Implementation: i:/WORK/aes_cipher_top

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

Possible Failure Causes

- Rejected Guidance Command (1)
  - inv_push
- Unconstrained Implementation Input (0)
- Unmatched Cone Input (0)
- Undriven Reference Signal (0)
- Directly Undriven Reference Port (0)
- Unmatched Blackbox Net (0)
- Failing Blackbox Net (0)
- Unretimed DesignWare Component (0)
- Missing Retention Register (0)
- X Propagation to Implementation Compare Point (0)
- Rejected Datapath Guidance Module (0)
- Hard Datapath Component Module (0)

Description - Rejected Guidance Command:
The rejection of some SVF guidance commands will almost invariably cause verification failures. For more information use:
'report_svf_operation -status rejected -command command_name

Recommendations:
inv_push

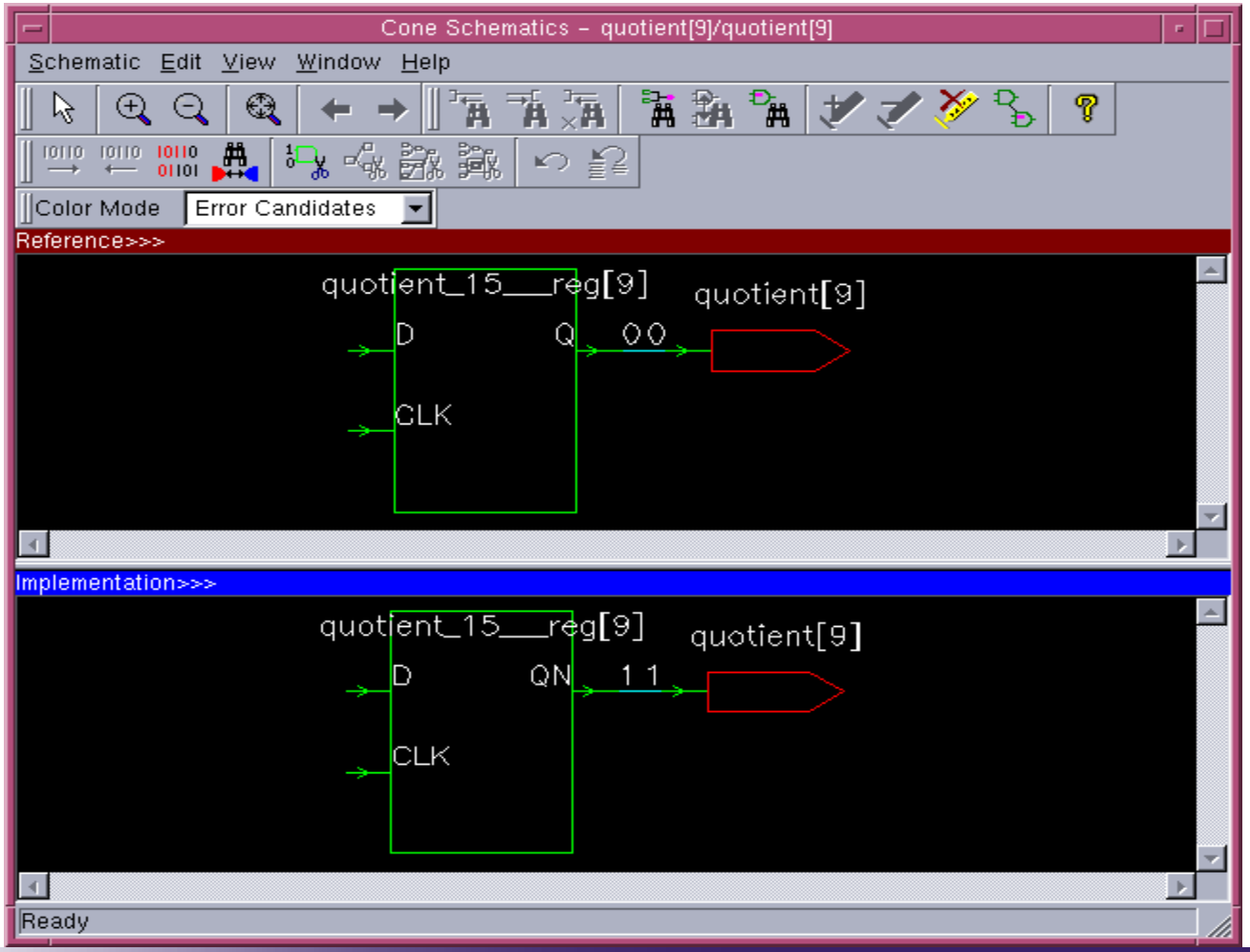
```
## SVF Operation 16 (Line: 128) - inv_push.  Status: rejected
## Operation Id: 16
guide_inv_push \
  -design { aes_cipher_top } \
  -register { badname_ld_r_reg }

Info:  guide_inv_push 16 (Line: 128)  Cannot find register 'badname_ld_r_reg'..
```

Log | Errors | Warnings | History | Last Command

Formality (verify)>

Inversion Push: Logic Cone



SVF Command: guide_reg_constant

- Constant registers
  - How to detect them:
    - Unmatched DFF in REF matching summary table
    - Unmatched register input in reference logic cone displayed in pattern window…failing patterns showing constant value
  - What to do:
    - Always use SVF guidance file
    - Create your own "`guide_reg_constant`" command
      - Formality will validate before using constant
    - Use `set_constant` command if you know register is constant
    - Validate with verification against a constant0 or constant1

```
verify r:/WORK/top/potentially_constant_reg -constant0
```

Formality (R) Console – Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/usb20h_top

Implementation: i:/WORK/usb20h_top

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Compare Rule Setup | User Match Setup | Matched Points | Unmatched Points | Summary

	Reference Object	Type
9	...nonlgcy_inst/rh_cfg_inst/port_0/prt_rst_rsm/Resume_TSe0_reg	DFF
10	...u_ehci_ahb/u_uhc2_ahbiu/u_uhc2_ahm/ahm_hsize_r_o_reg[2]	DFF
11	...st/uhostc_inst/rh_hregs_inst/rh_top_inst/sie_0/tx_inst/Txd_reg	DFF
12	...t/uhostc_inst/rh_hregs_inst/rh_top_inst/sie_0/tx_inst/TSe0_reg	DFF
13	...c_inst/rh_hregs_inst/rh_top_inst/rst_rsm_inst/KAlive_TSe0_reg	DFF
14	...c_nonlgcy_inst/rh_cfg_inst/port_1/prt_sm/KAlive_TxenL_d_reg	DFF
15	...onlgcy_inst/rh_cfg_inst/port_1/prt_rst_rsm/Resume_TxdPls_reg	DFF
16	...c_nonlgcy_inst/rh_cfg_inst/port_0/prt_sm/KAlive_TxenL_d_reg	DFF
17	...onlgcy_inst/rh_cfg_inst/port_0/prt_rst_rsm/Resume_TxdPls_reg	DFF
18	...u_uhc2_rh/u_uhc2_rhutm/u_uhc2_rhprt_0/prt_trm_sel_r_reg[1]	DFF
19	...ORK/usb20h_top/u_sh_top/u_sh_slave/s1_hsize_reg/q_reg[2]	DFF0
20	...sh_master_tx/data_ram/B13RF16X32RR1_bistctl/dual_test_reg	DFF0
21	...sh_master_tx/addr_ram/B13RF16X32RR1_bistctl/dual_test_reg	DFF0
22	...h_top/u_sh_master/u_sh_master_rx/s2_ocp_addr_reg/q_reg[1]	DFF0

Set User Match + − ?

	Implementation Object	Type
1	i:/WORK/usb20h_top/imdio/clk_gate_MdiotoReg_reg_latch	LATC
5	...i_ahb/u_uhc2_ahbiu/u_uhc2_ahm/clk_gate_state_r_reg_latch	LATC
6	.../u_uhc2_ahbiu/u_uhc2_ahs/clk_gate_aci_ben_r_o_reg_latch	LATC
7	...ci_ahb/u_uhc2_ahbiu/u_uhc2_ahs/clk_gate_hrdata_reg_latch	LATC
8	..._ehci/u_uhc2_aux_eor/clk_gate_portsc0_22_0_r_reg_0_latch	LATC
9	...c2_ehci/u_uhc2_aux_eor/clk_gate_portsc0_22_0_r_reg_latch	LATC
10	..._ehci/u_uhc2_aux_eor/clk_gate_portsc1_22_0_r_reg_0_latch	LATC
11	...c2_ehci/u_uhc2_aux_eor/clk_gate_portsc1_22_0_r_reg_latch	LATC
12	..._uhc2_ehci/u_uhc2_eor/clk_gate_eor_00_23_16_r_reg_latch	LATC
13	...b/u_uhc2_ehci/u_uhc2_eor/clk_gate_eor_08_5_0_r_reg_latch	LATC
14	.../u_uhc2_ehci/u_uhc2_eor/clk_gate_eor_0c_13_0_r_reg_latch	LATC
15	...i_ahb/u_uhc2_ehci/u_uhc2_eor/clk_gate_eor_10_r_reg_latch	LATC

These might be undetected constant registers.

Number of unmatched reference points: 79

Filter on reference list:

Number of unmatched implementation points: 306

Filter on implementation list:

Display names: Original Mapped

Run Matching

```
********************************** Matching Results **********************************
 8725 Compare points matched by name
 0 Compare points matched by signature analysis
 0 Compare points matched by topology
 407 Matched primary inputs, black-box outputs
 79(305) Unmatched reference(implementation) compare points
 0(1) Unmatched reference(implementation) primary inputs, black-box outputs
 571(0) Unmatched reference(implementation) unread points
```

Log | Errors | Warnings | History | Last Command

Formality (verify)>

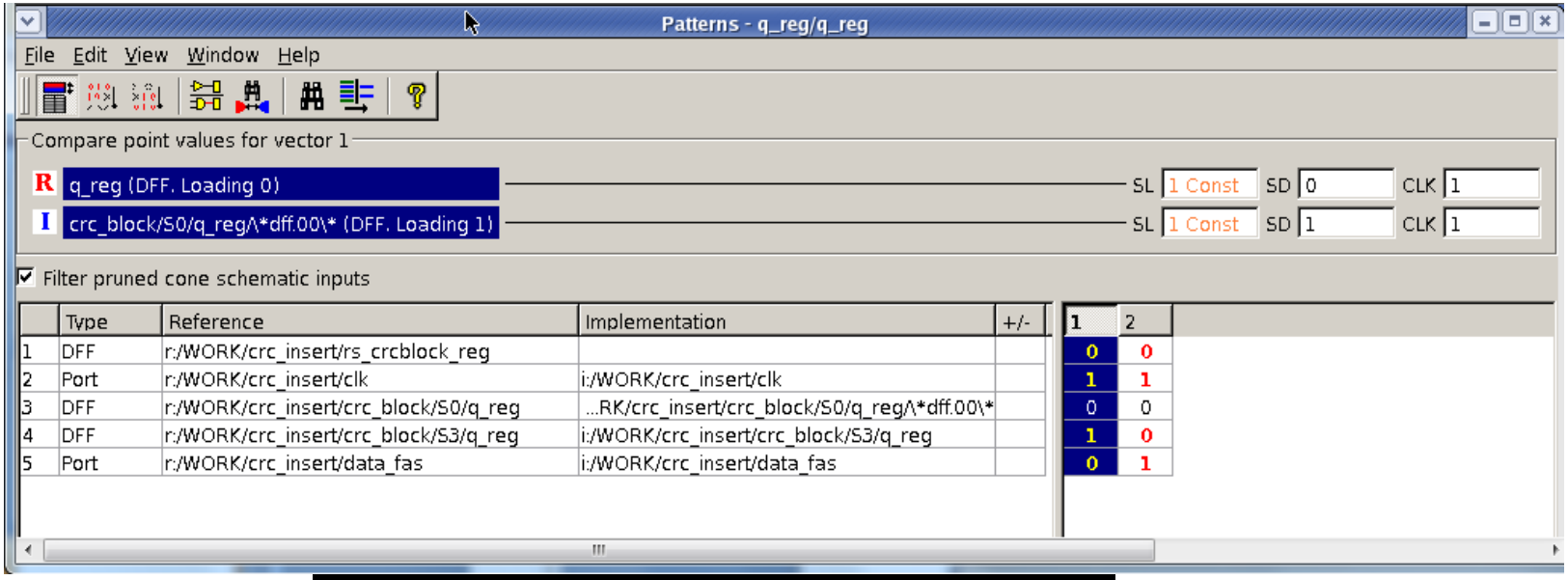
Ready

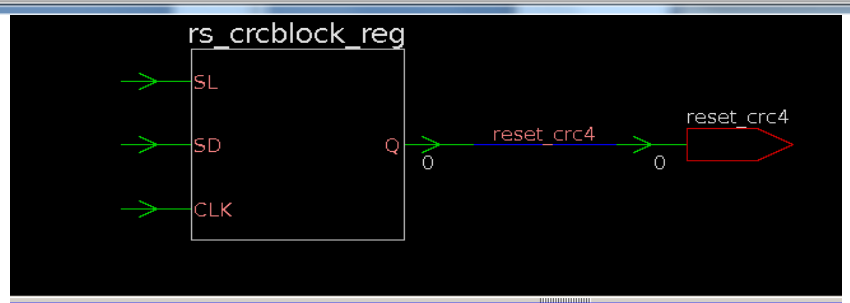
Shell State: verify

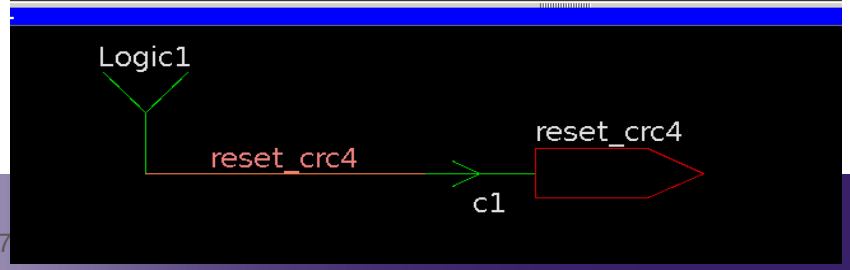
SVF Command: guide_reg_constant

- Note about SVF constant register processing
  - SVF command `guide_reg_constant` is only an assertion that the register is constant
  - Formality verifies the constant before using the guidance
  - If the constant value cannot be proven the guidance is rejected
    - Register will remain in the reference design
- Infrequently concordance naming issues between Design Compiler and Formality prevent processing of `guide_reg_constant` guidance
  - SVF text file may be edited to change module names to match Formality

Undetected Constant Register: Pattern Viewer and Logic Cone







SVF Command: guide_reg_merging

- Design Compiler may merge registers together and place information in SVF
- Potential workarounds other than fixing naming concordance in SVF are complex
  - Submit STAR

Register Merging: Analyze

Formality (R) Console - Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/mR4000

Implementation: i:/WORK/mR4000

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

Possible Failure Causes

- Unmatched Cone Input (1)
  - r:/WORK/mR4000/cntrl/PCSource_reg[0]
- Rejected Guidance Command (1)
  - reg_merging
- Unconstrained Implementation Input (0)
- Undriven Reference Signal (0)
- Directly Undriven Reference Port (0)
- Unmatched Blackbox Net (0)
- Failing Blackbox Net (0)
- Unretimed DesignWare Component (0)
- Missing Retention Register (0)
- X Propagation to Implementation Compare Point (0)
- Rejected Datapath Guidance Module (0)
- Hard Datapath Component Module (0)

Description - Rejected Guidance Command:
The rejection of some SVF guidance commands will almost invariably cause verification failures. For more information use:
report_svf_operation -status rejected -command command_name

Recommendations:
reg_merging

```
1
Formality (verify)> report_svf_operation -status rejected -command reg_merging

## SVF Operation 16 (Line: 121) - reg_merging.  Status: rejected
## Operation Id: 16
guide_reg_merging \
  -design { mCntrl } \
  -from { PCSource_reg_0_ } \
  -to { ALUOp_reg[0] }

Info:  guide_reg_merging 16 (Line: 121) Cannot find reference cell 'PCSource_reg_0_'.
```

Log | Errors | Warnings | History | Last Command

Formality (verify)>

Register Merging: Pattern Viewer

Patterns - PC_reg_0_/PC_reg_0_

File Edit View Window Help

Compare point values for vector 1

R PC_reg_0_ (DFF. Holding 0) — AC 0 AS 0 Const SL 0 SD X CLK 1

I PC_reg_0_/*dff.00* (DFF. Loading 1) — AC 0 SL 1 Const SD 1 CLK 1

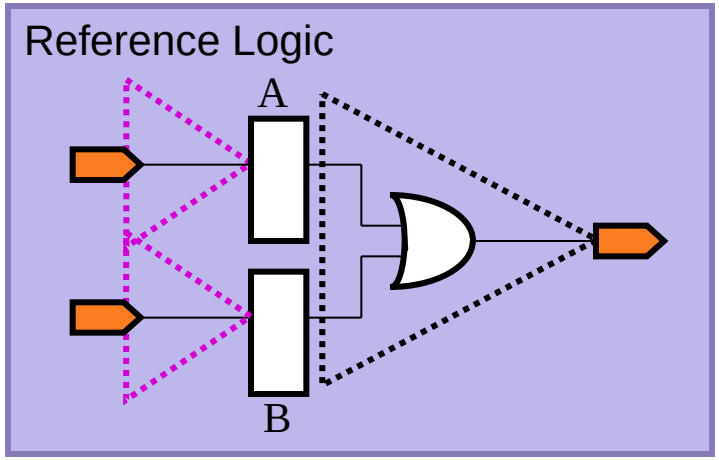
Filter pruned cone schematic inputs

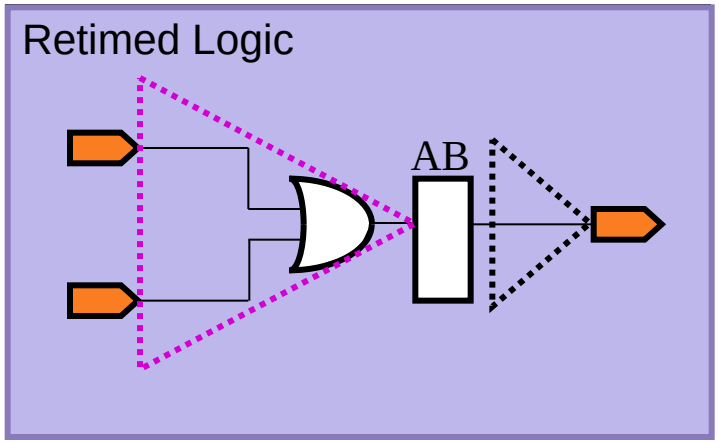
	Type	Reference	Implementation	+/-	1	2	3	4	5	6
1	DFF	r:/WORK/mR4000/cntrl/PCSource_reg[0]			1	1	1	0	1	1
2	Port	r:/WORK/mR4000/Clk	i:/WORK/mR4000/Clk		1	1	1	1	1	1
3	DFF	r:/WORK/mR4000/Instruction_reg_0_	i:/WORK/mR4000/Instruction_reg_0_		1	1	1	1	1	1
4	DFF	r:/WORK/mR4000/Instruction_reg_1_	i:/WORK/mR4000/Instruction_reg_1_		1	1	0	1	1	0
5	DFF	r:/WORK/mR4000/Instruction_reg_2_	i:/WORK/mR4000/Instruction_reg_2_		0	1	0	1	1	0
6	DFF	r:/WORK/mR4000/Instruction_reg_3_	i:/WORK/mR4000/Instruction_reg_3_		0	0	0	1	1	1
7	DFF	r:/WORK/mR4000/Instruction_reg_4_	i:/WORK/mR4000/Instruction_reg_4_		0	1	1	0	0	1
8	DFF	r:/WORK/mR4000/Instruction_reg_5_	i:/WORK/mR4000/Instruction_reg_5_		0	1	0	0	1	0
9	DFF	r:/WORK/mR4000/Instruction_reg_6_	i:/WORK/mR4000/Instruction_reg_6_		1	1	1	1	0	1
10	DFF	r:/WORK/mR4000/Instruction_reg_7_	i:/WORK/mR4000/Instruction_reg_7_		0	0	1	0	0	1
11	DFF	r:/WORK/mR4000/Instruction_reg_8_	i:/WORK/mR4000/Instruction_reg_8_		1	0	1	0	0	1
12	DFF	r:/WORK/mR4000/Instruction_reg_9_	i:/WORK/mR4000/Instruction_reg_9_		0	0	0	0	0	1
13	DFF	r:/WORK/mR4000/Instruction_reg_10_	i:/WORK/mR4000/Instruction_reg_10_		0	1	1	0	1	0
14	DFF	r:/WORK/mR4000/Instruction_reg_11_	i:/WORK/mR4000/Instruction_reg_11_		1	1	0	0	1	1
15	DFF	r:/WORK/mR4000/Instruction_reg_12_	i:/WORK/mR4000/Instruction_reg_12_		0	1	1	0	0	1
16	DFF	r:/WORK/mR4000/Instruction_reg_13_	i:/WORK/mR4000/Instruction_reg_13_		0	0	0	0	0	0

SVF Command: guide_retiming

- Retimed designs result from DC commands: `optimize_registers` or `compile_ultra -retime`
  - Improves timing of design
- Use SVF verification flow to verify retimed designs
  - Use of old flow (`set_parameter -retime designID`) will disable SVF retiming flow
- Successful verification with retiming depends on SVF guidance acceptance

The Retiming Challenge

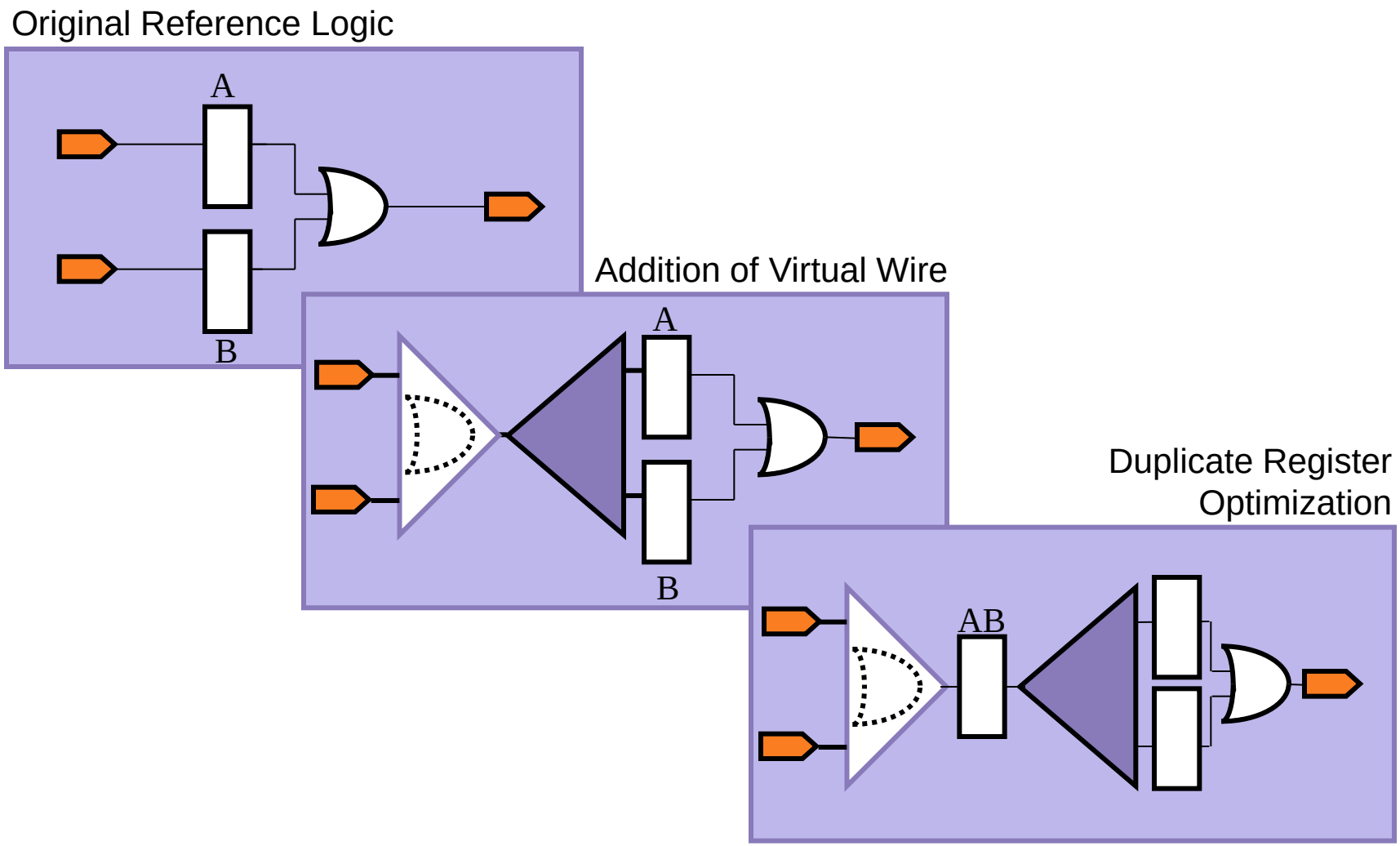




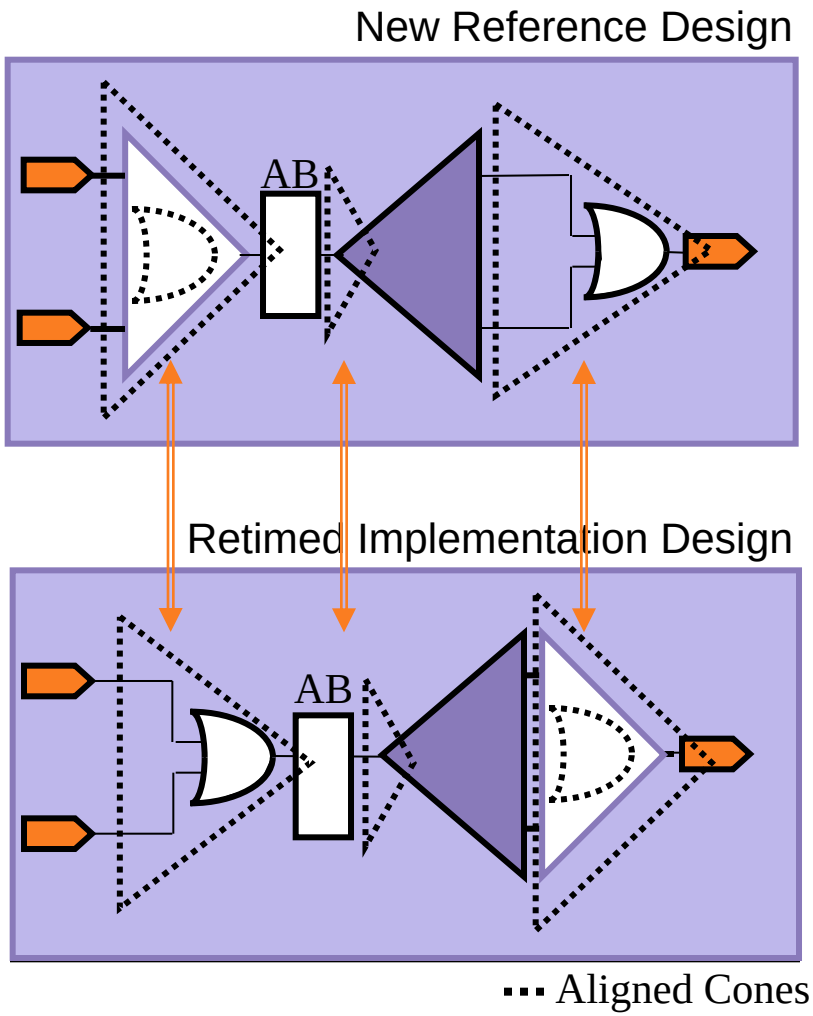
Aligned Cone Cannot be Aligned

- Equivalency checking bounds logic between registers
  - Functionally tractable verification
  - Bounded logic is called “logic cone”
  - Cones between designs are aligned then verified
- The Retiming Impact
  - Loss of logic cone correlation
  - Cones that do align are no longer functionally equivalent
  - Verification cannot succeed

SVF Retiming Guidance



Retiming Verification



- Logic cones now align
- All logic is verified
- Retiming verification is a reality only in Formality

Whitepaper: http://www.synopsys.com/cgi-bin/verification/pdfr1.cgi?retimingwp.pdf

SVF Command: guide_scan_input

- In Auto Setup Mode, Formality uses `guide_scan_input` to disable scan automatically
- Use command `set_constant` to disable scan mode manually

Scan Disable: Analyze

Formality(R) Console - Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/aes_cipher_top

Implementation: i:/WORK/aes_cipher_top

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

Possible Failure Causes

- Unconstrained Implementation Input (1)
  - i:/WORK/aes_cipher_top/test_se
- Unmatched Cone Input (0)
- Rejected Guidance Command (0)
- Undriven Reference Signal (0)
- Directly Undriven Reference Port (0)
- Unmatched Blackbox Net (0)
- Failing Blackbox Net (0)
- Unretimed DesignWare Component (0)
- Missing Retention Register (0)
- X Propagation to Implementation Compare Point (0)
- Rejected Datapath Guidance Module (0)
- Hard Datapath Component Module (0)

Description - Unconstrained Implementation Input:
Unmatched input ports in the implementation typically result from test logic insertion. Constraining the unmatched ports to a constant value may correct the failures.

Recommendations:
i:/WORK/aes_cipher_top/test_se
Unmatched in the implementation cones for 20 compare point(s):

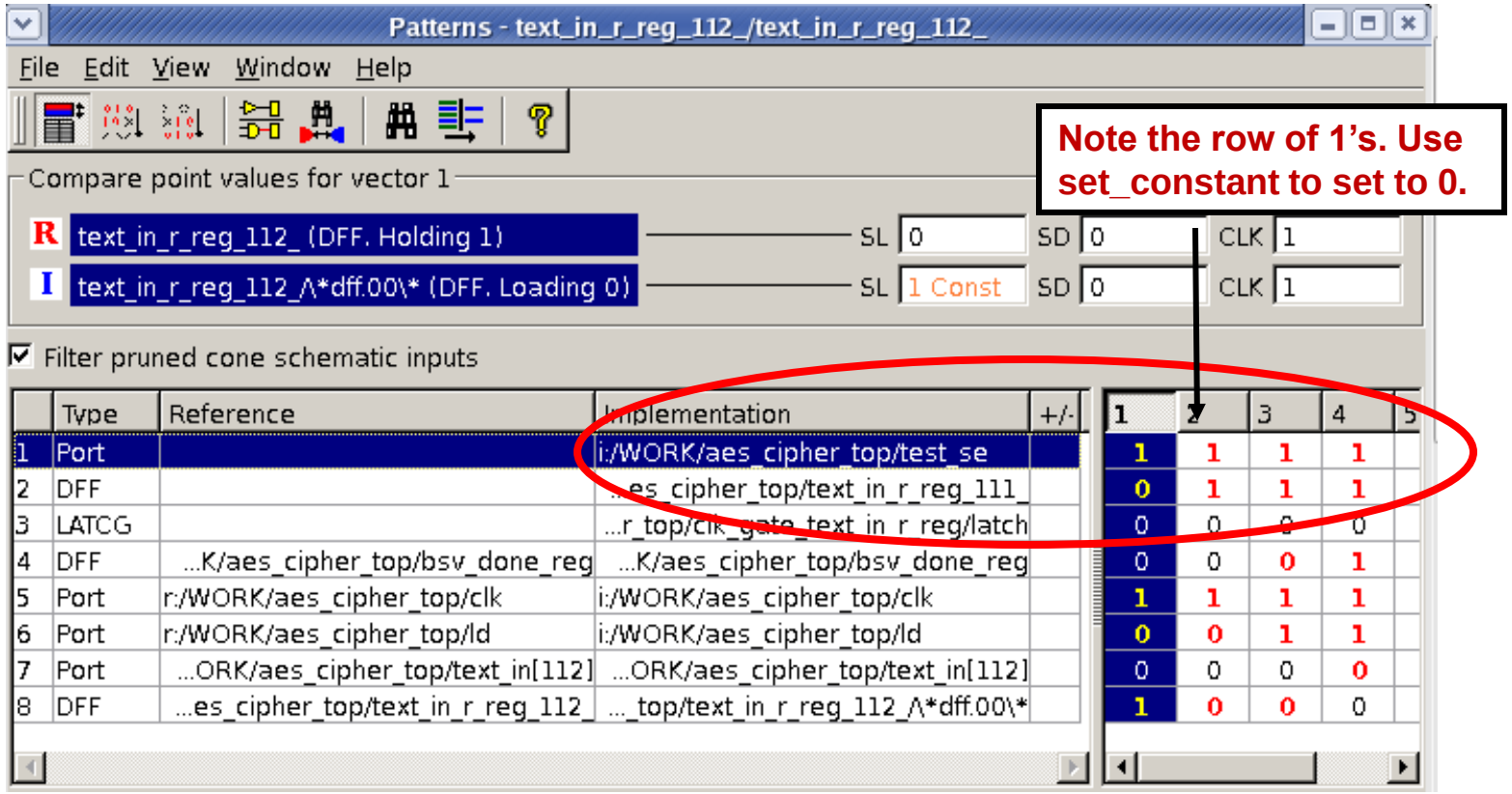
- i:/WORK/aes_cipher_top/text_in_r_reg_112
- i:/WORK/aes_cipher_top/text_in_r_reg_113
- i:/WORK/aes_cipher_top/text_in_r_reg_114
- i:/WORK/aes_cipher_top/text_in_r_reg_115
- i:/WORK/aes_cipher_top/text_in_r_reg_116
- i:/WORK/aes_cipher_top/text_in_r_reg_117
- i:/WORK/aes_cipher_top/text_in_r_reg_118
- i:/WORK/aes_cipher_top/text_in_r_reg_119
- i:/WORK/aes_cipher_top/text_in_r_reg_56
- i:/WORK/aes_cipher_top/text_in_r_reg_57
- i:/WORK/aes_cipher_top/text_in_r_reg_58
- i:/WORK/aes_cipher_top/text_in_r_reg_59

```
    Try adding this command before verify:
        set_constant i:/WORK/aes_cipher_top/test_se 0
-----------
```

Log | Errors | Warnings | History | Last Command

Scan Disable: Pattern Viewer

- Unmatched inputs to logic cone require user setup



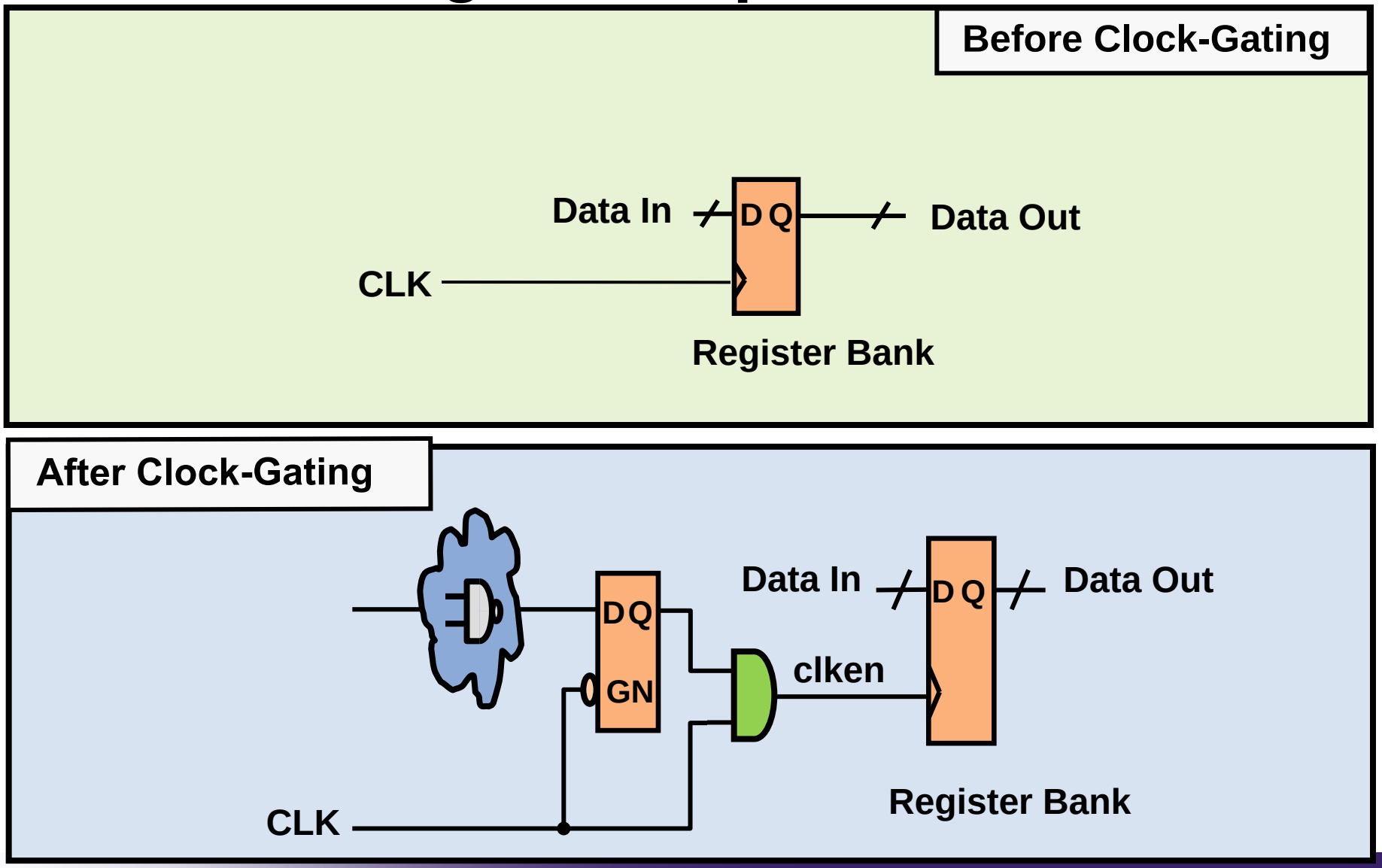
Verification

- Clock-gating circuitry not recognized
- Logic differences

Clock-Gating: Why Is It an Issue?

- Without intervention failing compare points will result
  - Compare points created for clock-gating latches
    - These compare point do not have matching point in the other design and will fail
  - The logic feeding the clock input of the register bank has changed
    - The register bank compare points will fail

Clock-Gating: Description



Clock-Gating: Verification Solution

- Use variable

```
set verification_clock_gate_hold_mode low
```

  - Use option `low` or `any` if clock-gating net drives the clock pin of positive edge-triggered DFF
  - If clock-gating net also drives primary outputs or black-box inputs use option `collapse_all_cg_cells`
  - Use `set_clock` command to identify the primary input clock net if clock-gating cells do not drive any clock pin of a DFF
- Auto Setup Mode will enable clock-gating automatically
- Use new variable if clock-gating verification issues continue

```
set verification_clock_gate_edge_analysis true
```

Clock-gating Not Recognized: Match

Formality (R) Console – Synopsys Inc.

File Edit View Designs Run Window Help

Verification Inconclusive

Reference: r:/WORK/tv80s

Implementation: i:/WORK/tv80s

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Compare Rule Setup | User Match Setup | Matched Points | Unmatched Points | Summary

	Reference Object	Type

Set User Match

	Implementation Object	Type
1	i:/WORK/tv80s/clk_gate_ACC_reg/latch	LAT
2	i:/WORK/tv80s/clk_gate_A_reg/latch	LAT
3	i:/WORK/tv80s/clk_gate_BusA_reg/latch	LAT
4	i:/WORK/tv80s/clk_gate_Fp_reg/latch	LAT
5	i:/WORK/tv80s/clk_gate_IR_reg/latch	LAT
6	i:/WORK/tv80s/clk_gate_I_reg/latch	LAT
7	i:/WORK/tv80s/clk_gate_PC_reg/latch	LAT
8	i:/WORK/tv80s/clk_gate_Pre_XY_F_M_reg/latch	LAT
9	i:/WORK/tv80s/clk_gate_R_reg/latch	LAT
10	i:/WORK/tv80s/clk_gate_Read_To_Reg_r_reg/latch	LAT
11	i:/WORK/tv80s/clk_gate_RegAddrA_r_reg/latch	LAT
12	i:/WORK/tv80s/clk_gate_RegsH_reg_0_/latch	LAT
13	i:/WORK/tv80s/clk_gate_RegsH_reg_1_/latch	LAT
14	i:/WORK/tv80s/clk_gate_RegsH_reg_2_/latch	LAT
15	i:/WORK/tv80s/clk_gate_RegsH_reg_3_/latch	LAT

Clock-gating Not Recognized: Analyze

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

Possible Failure Causes

- Unmatched Cone Input (16)
  - i:/WORK/tv80s/clk_gate_Fp_reg/latch/*lat.00*
  - i:/WORK/tv80s/Fp_reg_1_/*dff.00*
  - i:/WORK/tv80s/Fp_reg_4_/*dff.00*
  - i:/WORK/tv80s/clk_gate_Read_To_Reg_r_reg/latch/*...
  - i:/WORK/tv80s/clk_gate_I_reg/latch/*lat.00*
  - i:/WORK/tv80s/I_reg_0_/*dff.00*
  - i:/WORK/tv80s/clk_gate_R_reg/latch/*lat.00*
  - i:/WORK/tv80s/clk_gate_di_reg_reg/latch/*lat.00*
  - i:/WORK/tv80s/di_reg_reg_0_/*dff.00*
  - i:/WORK/tv80s/di_reg_reg_1_/*dff.00*
  - i:/WORK/tv80s/di_reg_reg_2_/*dff.00*
  - i:/WORK/tv80s/di_reg_reg_3_/*dff.00*
  - i:/WORK/tv80s/di_reg_reg_4_/*dff.00*
  - i:/WORK/tv80s/di_reg_reg_5_/*dff.00*
  - i:/WORK/tv80s/di_reg_reg_6_/*dff.00*
  - i:/WORK/tv80s/di_reg_reg_7_/*dff.00*
- Unconstrained Implementation Input (0)

Description - Unmatched Cone Input:
Unmatched cone inputs result either from mismatched compare points or from differences in the logic within the cones. Only unmatched inputs that are suspected of contributing to verification failures are included in the report.
The source of the matching or logical differences may be determined using the schematic, cone and source views.

Recommendations:
i:/WORK/tv80s/clk_gate_Fp_reg/latch/*lat.00*
Is globally unmatched affecting 2 compare point(s):

- r:/WORK/tv80s/Fp_reg_1_
- r:/WORK/tv80s/Fp_reg_4_

Clock-gating Not Recognized: Pattern Viewer

Patterns - Fp_reg_1_/Fp_reg_1_

File Edit View Window Help

Compare point values for vector 1

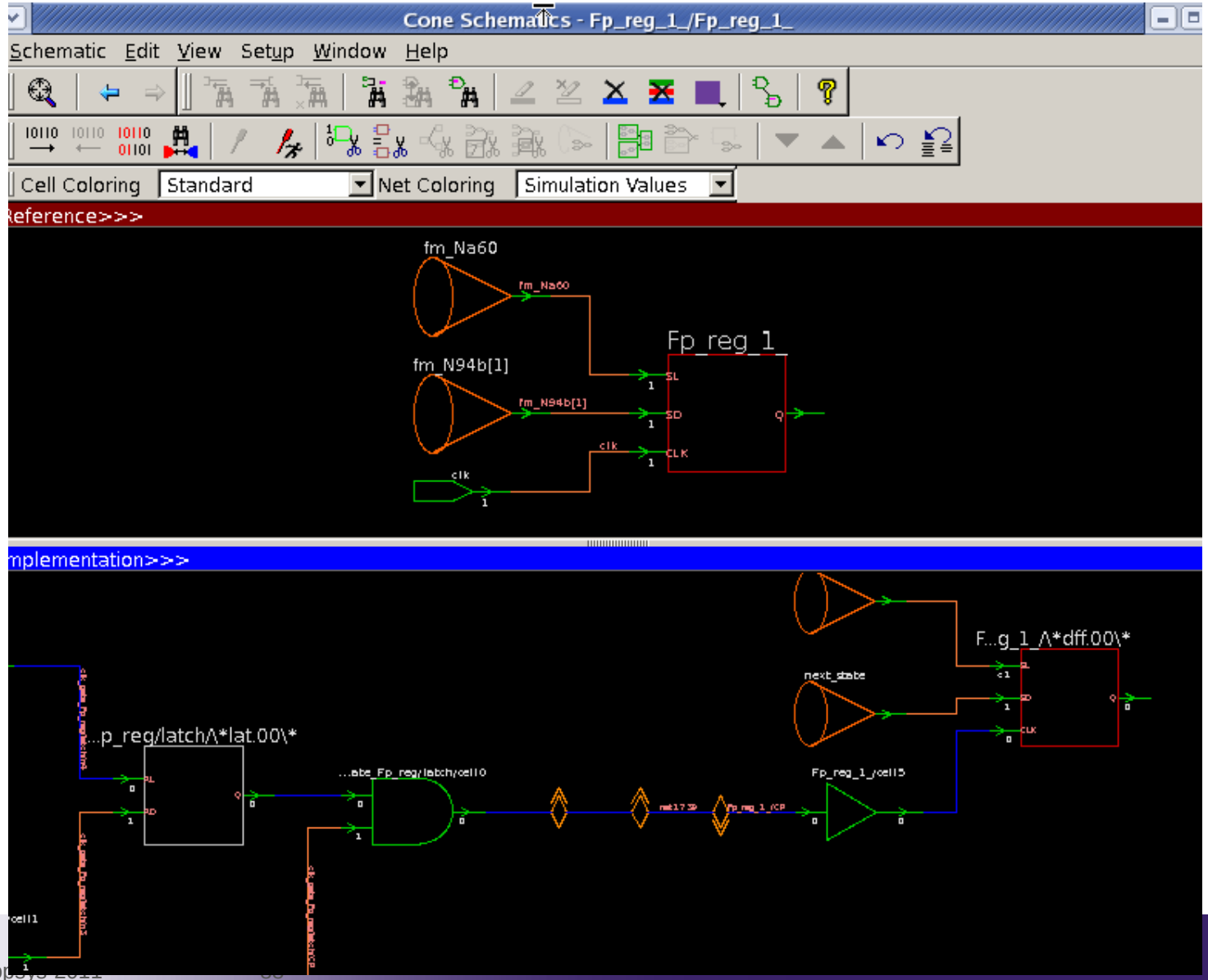
R Fp_reg_1_ (DFF. Loading 1) SL 1 SD 1 CLK 1

I Fp_reg_1_/*dff.00* (DFF. Holding 0) SL 1 Const SD 1 CLK 0

☑ Filter pruned cone schematic inputs

	Type	Reference	Implementation	+/-	1	2	3	4	5
1	Port		i:/WORK/tv80s/test_se (Const 0)		0	0	0	0	0
2	LAT		i:/WORK/tv80s/clk_gate_Fp_reg/latch		0	1	1	1	1
3	DFF	r:/WORK/tv80s/BusAck_reg	i:/WORK/tv80s/BusAck_reg		0	1	0	0	0
4	DFF	r:/WORK/tv80s/F_reg_1_	i:/WORK/tv80s/F_reg_1_		0	0	0	1	0
5	DFF	r:/WORK/tv80s/Fp_reg_1_	i:/WORK/tv80s/Fp_reg_1_/*dff.00*		0	1	1	0	1
6	DFF	r:/WORK/tv80s/IR_reg_0_	i:/WORK/tv80s/IR_reg_0_		0	0	0	0	0
7	DFF	r:/WORK/tv80s/IR_reg_1_	i:/WORK/tv80s/IR_reg_1_		0	0	0	0	0
8	DFF	r:/WORK/tv80s/IR_reg_2_	i:/WORK/tv80s/IR_reg_2_		0	0	0	0	0
9	DFF	r:/WORK/tv80s/IR_reg_3_	i:/WORK/tv80s/IR_reg_3_		0	0	0	0	0
10	DFF	r:/WORK/tv80s/IR_reg_4_	i:/WORK/tv80s/IR_reg_4_		0	0	0	0	0
11	DFF	r:/WORK/tv80s/IR_reg_5_	i:/WORK/tv80s/IR_reg_5_		0	0	0	0	0
12	DFF	r:/WORK/tv80s/IR_reg_6_	i:/WORK/tv80s/IR_reg_6_		0	0	0	0	0
13	DFF	r:/WORK/tv80s/IR_reg_7	i:/WORK/tv80s/IR_reg_7		0	0	0	0	0

Clock-gating Not Recognized: Logic Cone



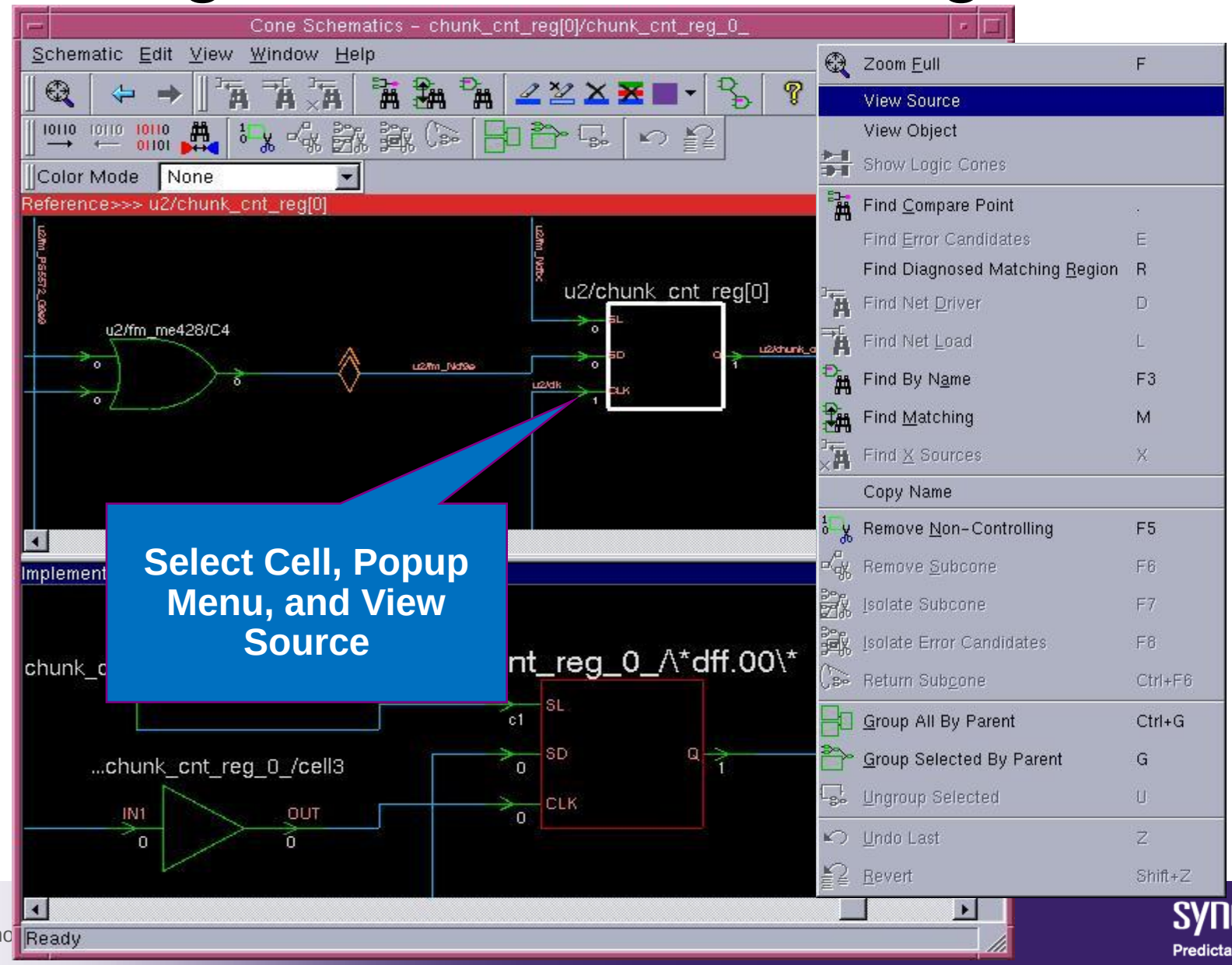
Agenda

- Debugging Flow
- Frequently Used Debugging Tools
- Common Problems
- Additional Debugging Tools
- Labs

Additional Debugging Tools

- Source Code Browser
- Probe points
- Ref/Impl container browser and viewer
- Helpful commands and variables

Viewing RTL Source From Logic Cone



Select Cell, Popup Menu, and View Source

Source Code Browser

Reference Source Browser

File Edit View Window

/remote/fmcae4/users/ebhatch/dc_fest/FM_debugging_labs/lab5/rtl/...

```
197                 LD_DESC4        = 11'b000_1000_0000,
198                 LD_DESC5        = 11'b001_0000_0000,
199                 WB              = 11'b010_0000_0000,
200                 PAUSE           = 11'b100_0000_0000;
201
202 reg     [10:0] /* synopsys enum state */ state, next_state;
203 // synopsys state_vector state
204
205 reg     [31:0] mast0_adr, mast1_adr;
206
207 reg     [29:0] adr0_cnt, adr1_cnt;
208 wire    [29:0] adr0_cnt_next, adr1_cnt_next;
209 wire    [29:0] adr0_cnt_next1, adr1_cnt_next1;
210 reg            adr0_inc, adr1_inc;
211
212 reg     [8:0]  chunk_cnt;
213 reg            chunk_dec;
214
215 reg     [11:0] tsz_cnt;
216 reg            tsz_dec;
217
218 reg            de_txsz_we;
219 reg            de_csr_we;
220 reg            de_adr0_we;
221 reg            de_adr1_we;
```

Ready

Gate and line number highlighted

Debugging Tools: Probe Points

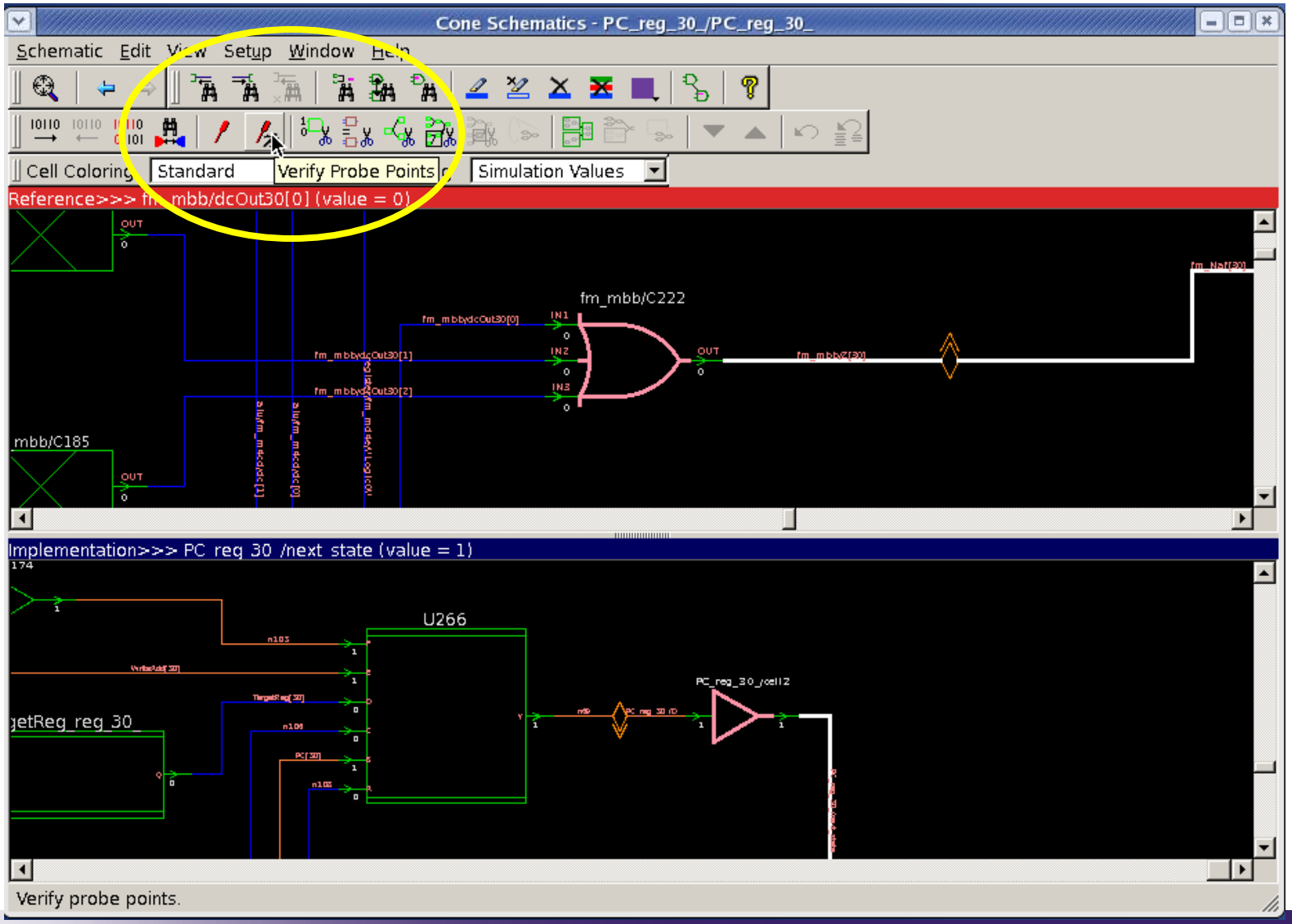
- `set_probe_points <ref_net> <impl_net>`
- Allows users to more easily debug failing or hard verifications
- Select a net pair between reference and implementation for probe verification
- Verification will determine if logic is equivalent up to those probe points
- Probe points can be set at any time after both designs are read-in and linked
- Can specify one or more probe point pairs

Debugging Tools: Probe Points

- Verification of probe points can only be done in "verify" mode
  - Command: `verify -probe`
- Verification of these probe points will neither destroy nor alter the current compare point matching and overall verification results
- Supports 1-N and inversion between probe points
- Commands for removing probes and reporting status:

```
remove_probe_points <net> |-all
report_probe_status
```

Debugging Tools: Probe Points



Debugging Tools: Probe Points

Formality (R) Console - Synopsys Inc.

File Edit View Designs Run Window Help

Verification Failed

Reference: r:/WORK/mR4000

Implementation: i:/WORK/mR4000

0. Guidance | 1. Reference | 2. Implementation | 3. Setup | 4. Match | 5. Verify | 6. Debug

Failing Points | Passing Points | Aborted Points | Unverified Points | Probe Points | Analyses

	Reference	Size	Implementation	Size	Status
1	r:/WORK/mR4000/fm_Naf[30]	2764	i:/WORK/mR4000/PC_reg_30_/next_state	5200	FAIL
2	r:/WORK/mR4000/MemData[12]	1	i:/WORK/mR4000/Instruction_reg_12_/next_state	10	FAIL
3	r:/WORK/mR4000/Clk	1	i:/WORK/mR4000/Instruction_reg_12_/clocked_on	3	PASS
4	r:/WORK/mR4000/register/fm_Na51[0]	459	i:/WORK/mR4000/register/mregister2_reg_0_/D	469	FAIL
5	r:/WORK/mR4000/register/Reset	2	i:/WORK/mR4000/register/register2_reg_0_/clocked_on	4	NOT RUN

Number of Probe Points: 5

Display names: Original Mapped

Filter:

Verify Probes | Remove Selected Probes

```
 3 Failing probe points
 0 Aborted probe points
 0 Unverified probe points
*****************************************************************************
0
Formality (verify)> set_probe_points r:/WORK/mR4000/register/Reset i:/WORK/mR4000/register/register2_reg_0_/clocked_on
Set user probe between 'r:/WORK/mR4000/register/Reset' and 'i:/WORK/mR4000/register/register2_reg_0_/clocked_on'
1
```

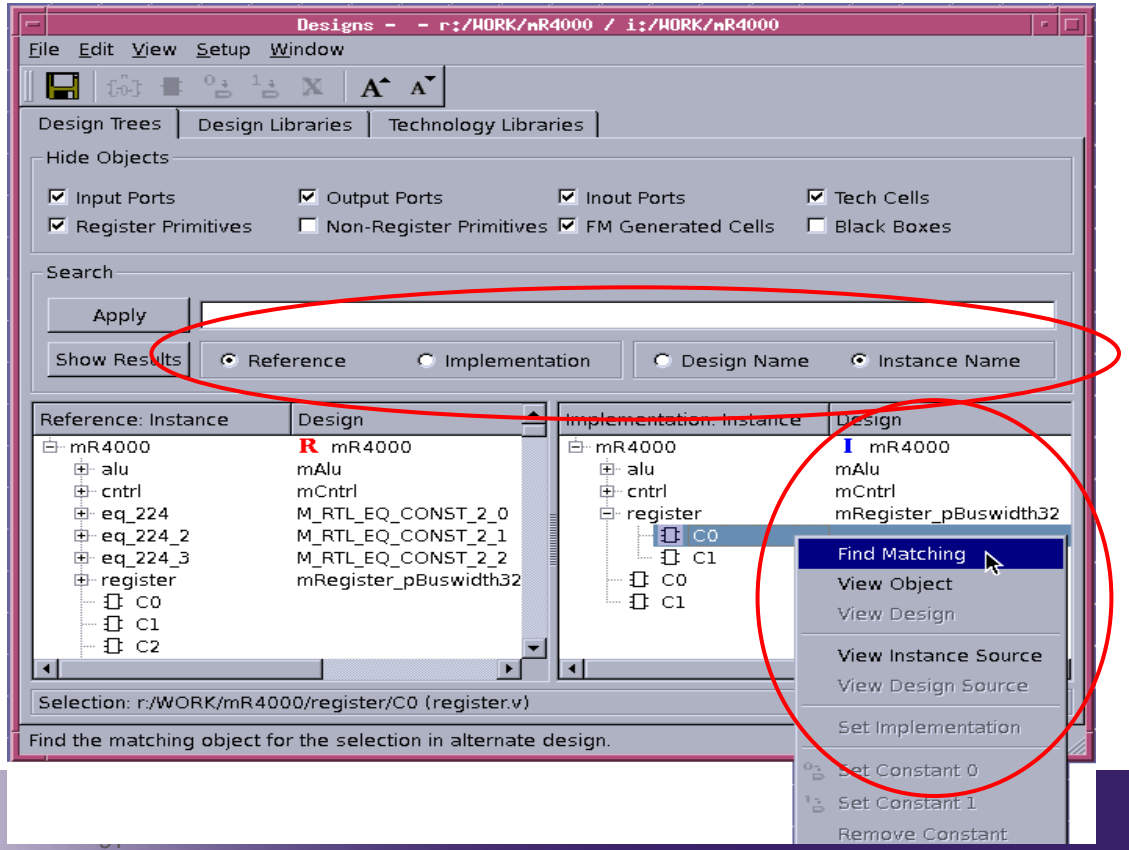
Log | Errors | Warnings | History | Last Command

Formality (verify)>

Ready | Shell State: verify

Debugging Tools: Double Design Browser

- Reference and implementation browser now integrated together
- Search feature
- “Find Matching” feature
  - Select an object and find corresponding object in other container



Helpful Formality Commands

- `report_guidance -summary`
- Reports the summary table of SVF commands and their disposition
- This table is automatically displayed in the transcript log after SVF processing is complete

```
***************************** Guidance Summary *****************************
                                        Status
Command                  Accepted   Rejected  Unsupported  Unprocessed  Total
-------------------------------------------------------------------------------
architecture_netlist:        786          0            0            0    786
boundary            :       3062          0            0            0   3062
boundary_netlist    :        782          0            0            0    782
change_names        :      29410      15021            0            0  44431
constraints         :       1314        107            0            0   1421
datapath            :       2945        117            0            0   3062
environment         :          7          0            0            0      7
implementation      :         50          0            0            0     50
instance_map        :       2218        143            0            0   2361
inv_push            :       1098          1            0            0   1099
merge               :       2756        101            0            0   2857
multiplier          :        901        130            0            0   1031
reg_constant        :      10756          0            0            0  10756
reg_merging         :       2215          0            0            0   2215
rename_design       :         33          1            0            0     34
replace             :      11402        466            0            0  11868
scan_input          :          0          6            0            0      6
ungroup             :        582         17            0            2    601
uniquify            :       2838        220            0            0   3058
ununiquify          :          1          0            0            0      1

Note: If verification succeeds you can safely ignore unaccepted guidance commands.
```

Helpful Formality Commands

- **`report_svf_operation`**
- Reports specific information about selected SVF commands and why rejected
- Options: **`-status rejected -summary [compare_points]`**
- Can specify certain compare points or groups of compare points using wildcard *

```
fm_shell (verify)> report_svf_operation -status rejected r:/WORK/aes_cipher_top/us10/sbox1/dreg_reg_*

## SVF Operation 16 (Line: 128) - inv_push.  Status: rejected
## Operation Id: 16
guide_inv_push \
  -design { aes_cipher_top } \
  -register { badname_ld_r_reg }

Info:  guide_inv_push 16 (Line: 128)  Cannot find register 'badname_ld_r_reg'..
```

Helpful Formality Commands

- `report_setup_status`
- Reports the following in summary format
  - Design statistics
  - Design read warning messages
  - User Specified setup
- Check critical design setup before committing time to run match or verify commands

Helpful Formality Commands

```
fm_shell (setup)> report_setup_status

#### Design Information ####

 # Design Settings #

 set_top reference design:      ref:/WORK/top
 set_top implementation design: impl:/WORK/top
 set_reference_design:      ref:/WORK/top
 set_implementation_design: impl:/WORK/top

 # Design Statistics Ref(Imp)  #

 Ports:                          359(362)
 Registers:                      65410(65014)
 Black boxes:                    173(173)
   - Unresolved modules :        173(173)
   - User specified     :        0(0)
 Undriven nets :                 334(320)
 Multiply-driven nets:           2120(1736)

 #### HDL Read Message Summary ####

Message ID:          Occurrences:  Ref(Imp)
 ---------------------------------------------
-
 FMR_VHDL-1002             :       7(   0)
 FMR_VHDL-1027             :    8374(   0)
 FMR_ELAB-149              :    2365(   0)
 FMR_ELAB-147              :     772(   0)
 FMR_ELAB-146              :     305(   0)
 FMR_VHDL-1140             :       1(   0)
 FMR_ELAB-115              :      29(   0)
 FMR_VHDL-1014             :       3(   0)
 FMR_ELAB-154              :       2(   0)

 #### User Specified Setup ####

 Command Name:          Result:  Ref(Imp)
 ---------------------------------------------
-
 set_black_box             :       0(   0)
 set_clock                 :       0(   0)
 set_compare_rule          :       5(   0)
 set_constant              :       0(   3)
 set_constraint            :       0(   0)
 set_cutpoint              :       2(   2)
 set_dont_verify_point     :       0(   0)
 set_equivalence           :       0(   0)
 set_factor_point          :       0(   0)
 set_inv_push              :       0(   0)
 set_user_match            :     288( 288)
 set_input_value_range     :       0(   0)
```

Helpful Formality Commands

`report_undriven_nets`

- Reports undriven nets after the most recent match
  - Total number of undriven nets
  - Instance path name of each undriven net

`report_multidriven_nets`

- Reports multiply driven nets after the most recent match
  - Total number of multiply driven nets
  - Instance path name of each multiply driven net and its list of drivers
  - Resolution or wire type displayed if not type consensus
  - Cell and library names displayed if they exist

- Benefits
  - No longer need to view formality.log file to get more information about multiply and undriven signals
  - Much more information and improved formatting over formality.log file

Helpful Formality Commands

- **`write_hierarchical_verification_script`**
  - Formality generates TCL script that performs hierarchical verification on current reference and implementation designs
  - Helpful for debugging large designs to isolate problem blocks
  - Usage:

```
…
set_top i:/WORK/top
set_constant $impl/test_se  0
write_hier -replace -level 3 myhierscript
source myhierscript.tcl
quit
```

    - View results in file fm_myhierscript.log
  - Formality will create one session file, by default, if verification fails on a sub-design

Formality TCL Variables

- `set verification_clock_gate_edge_analysis true`
- Specifies Formality to use clock edges in the next state formulation of registers
- Use when current clock gating solution variable `verification_clock_gate_hold_mode` does not identify all clock-gating circuitry
  - If LATCGs are not identified properly they will cause failing compare points
- New variable will disable old variable so no need to remove it from FM TCL script
- May see special rising (r) and falling (f) notations as well as other transition notations on failing patterns and cone schematics representing edge values on signals

Formality TCL Variables

- `set verification_effort_level super_low`
- Variable setting specifies how hard verification works before aborting unsolved compare points
- Recommend setting `super_low` only for debugging failing verifications
  - Very useful in getting to failing compare points quickly
  - Will abort other somewhat complex compare points

DC TCL Variable and Command Settings as Workarounds

- **The following may be changed in DC as a potential workaround for some verification failures**

```
set compile_enable_register_merging false
```

  - Turns off the identification and merging of registers that are equal or opposite

```
set compile_seqmap_propagate_constants false
```

  - Turns off the identification and removal of constant sequential elements

```
compile_ultra -no_seq_output_inversion
```

  - Disables sequential output inversion

Agenda

- Debugging Flow
- Frequently Used Debugging Tools
- Common Problems
- Additional Debugging Tools
- Labs

SYNOPSYS®

Predictable Success