Glitch Checks at Netlist Level using VC Spyglass CDC

Sayantan Maiti (smaiti@nvidia.com)
Alex Li (alexli@nvidia.com)
Pratik Suthar (psuthar@nvidia.com)

NVIDIA

Bhaumik Matholia (matholi@synopsys.com)
Samrat Das (samratd@synopsys.com)

Synopsys

Agenda

- Introduction
- Types of Glitch Check
- Methodologies & Performance
- Summary and Future Works

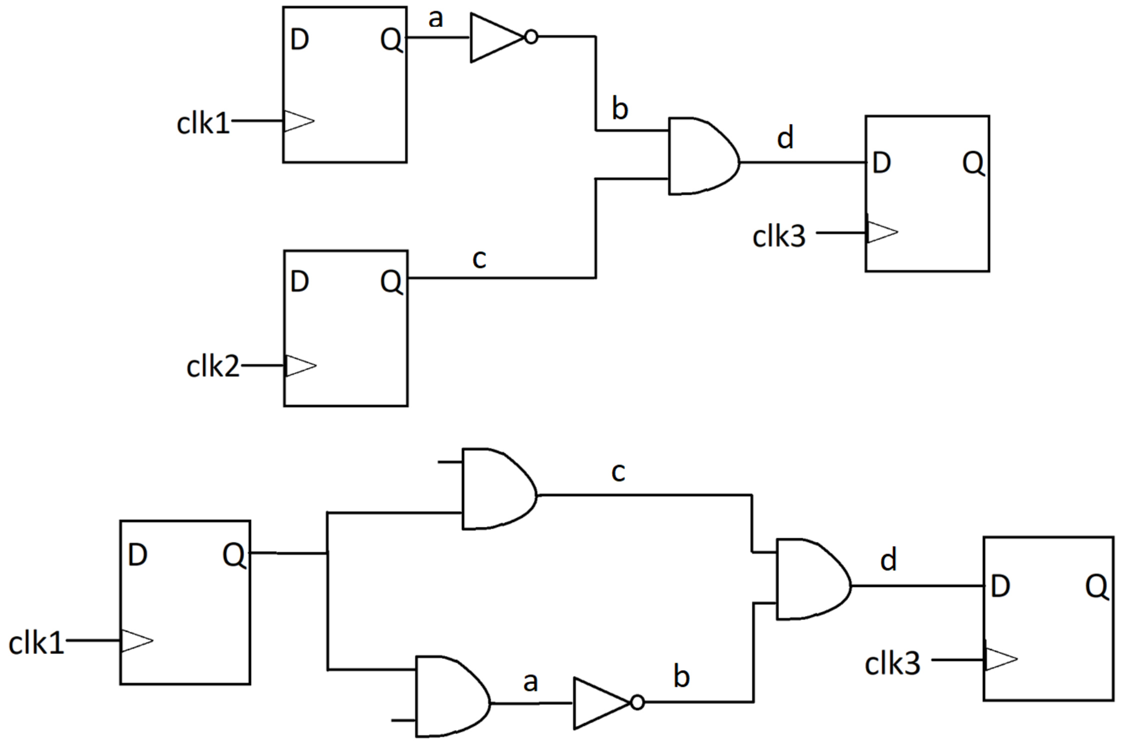
Introduction

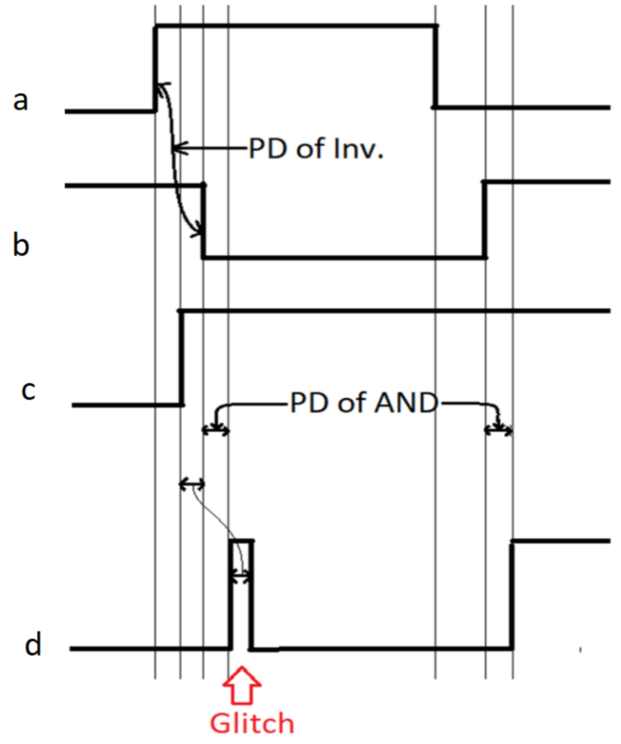
What is glitch

Importance of glitch check at netlist level

What is Glitch?

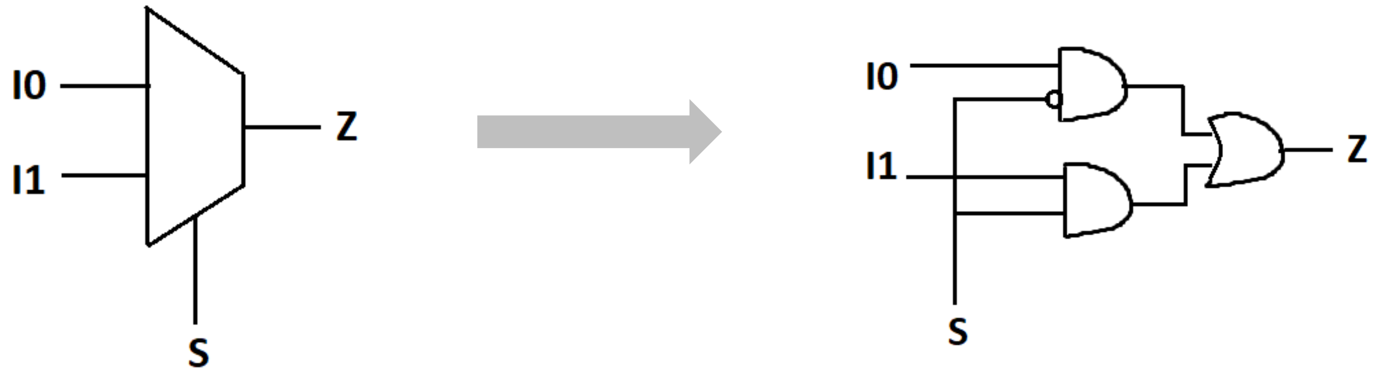
- Glitch is a temporary unintended state of logic that might create functional issues in designs
- Glitch might propagate to downstream logics and create hazards
- Happens mostly due to interactions of several combinational logic paths (structural)
- Needs to be captured early in designs





Why We Need to Perform Glitch Check at Netlist Level

- Synthesis can introduce discrete logic structures that might be glitch prone



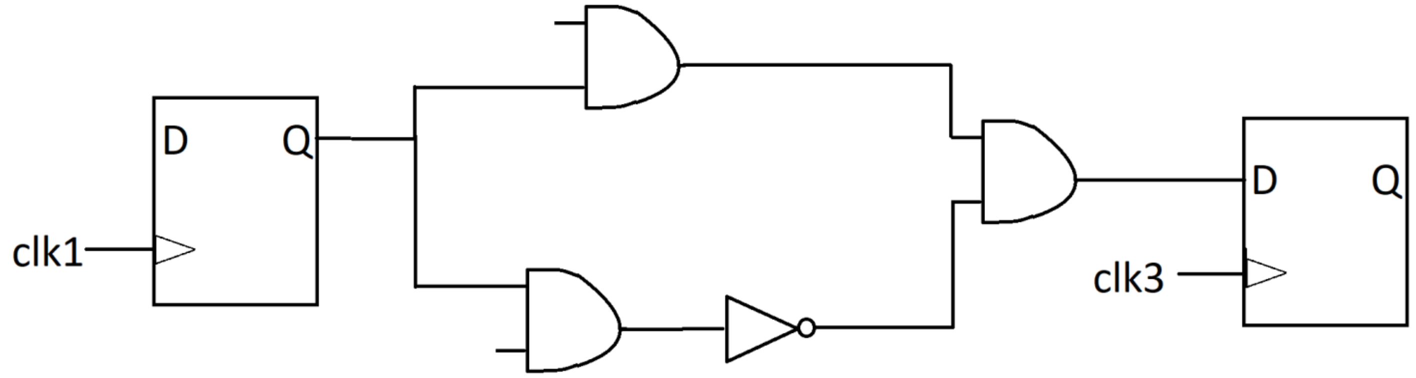
- DFT logics & Power logics are added post synthesis which needs to be checked for glitch

Types of Glitch Check

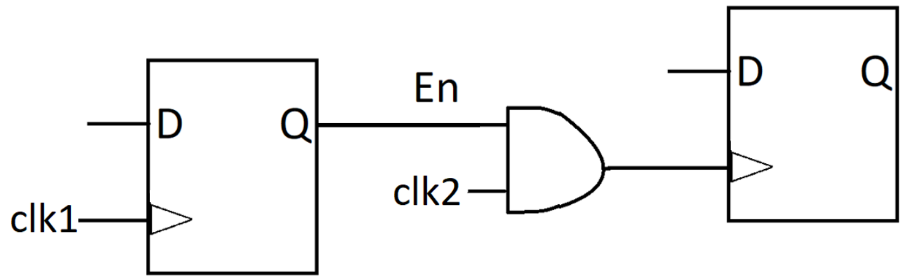
Glitch check on Asynchronous Path
Glitch check on Synchronous Path

Glitch check on Asynchronous Path

- Glitch on Synchronized Data path
- Glitch on Synchronized Control path
- Glitch on Unsynchronized path



- Glitch check on Clock paths
  - Asynchronous source(s) converging with different domain clock
  - Improper clock gating logic



Glitch Check on Synchronous Path

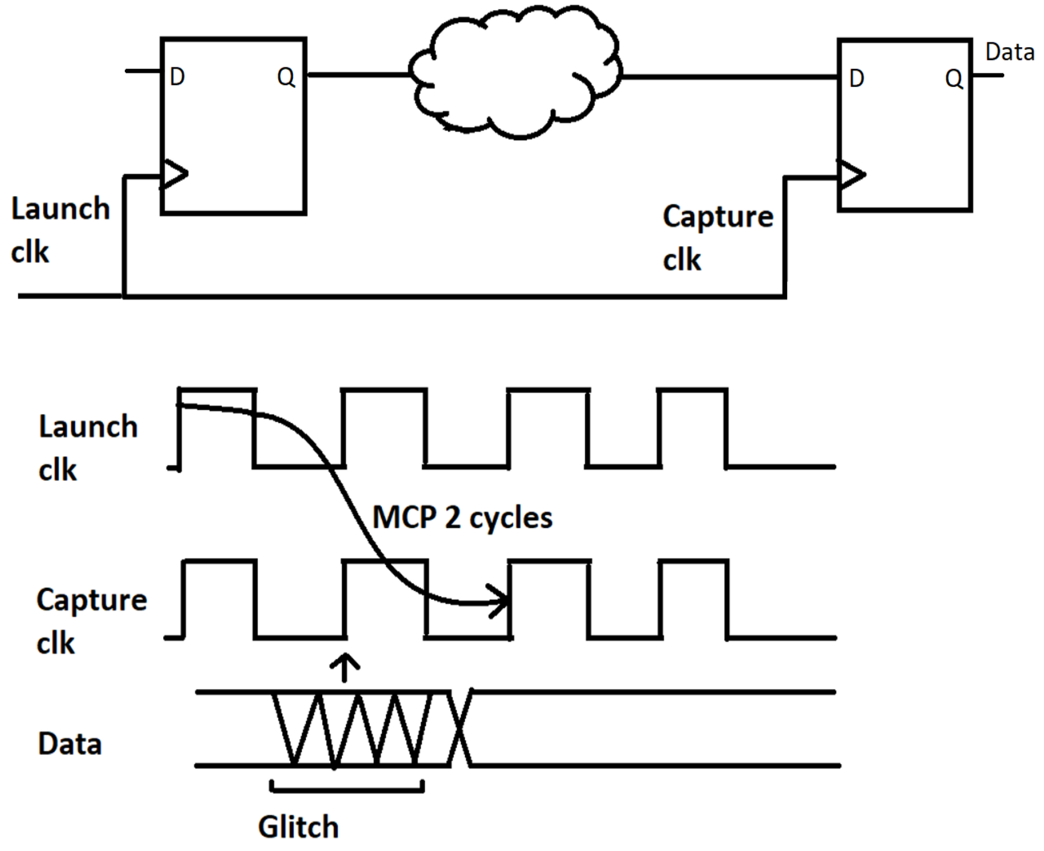
Why Glitches on sync paths are concern?

- Glitches on synchronous path with no exceptions defined should be settled in single cycle. Should meet timing

- If exceptions like MCP are defined, then glitches on such synchronous path can be concerning

- Not covered in STA

- If MCP, then glitch might persist for more than 1 cycle, might get captured by capture flop in next cycle



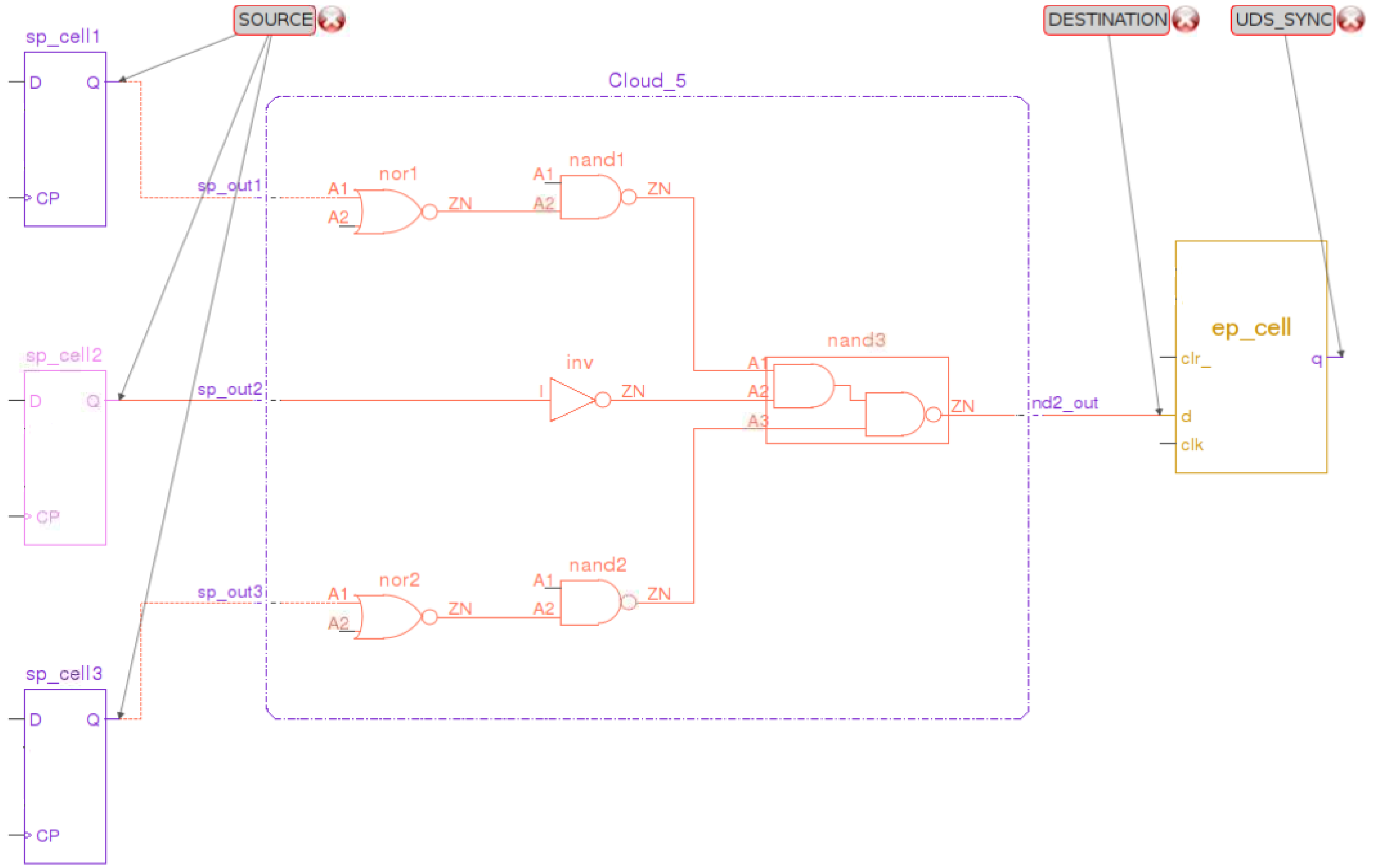
Methodologies and Performance

Glitch Check Methodologies

- We perform glitch analysis on controlpaths
  - Checking for multi source convergence and single source reconvergence
- We perform glitch check on user specified datapaths (point2point path)
  - Checking for single source divergence reconvergence
- Obtain startpoints, endpoints, clocks, converging gate, reasoncode from verbose reporting
- Reports to be analyzed by netlist CDC owners and sent to front end owners for review. Netlist owners can also add ECOs when required

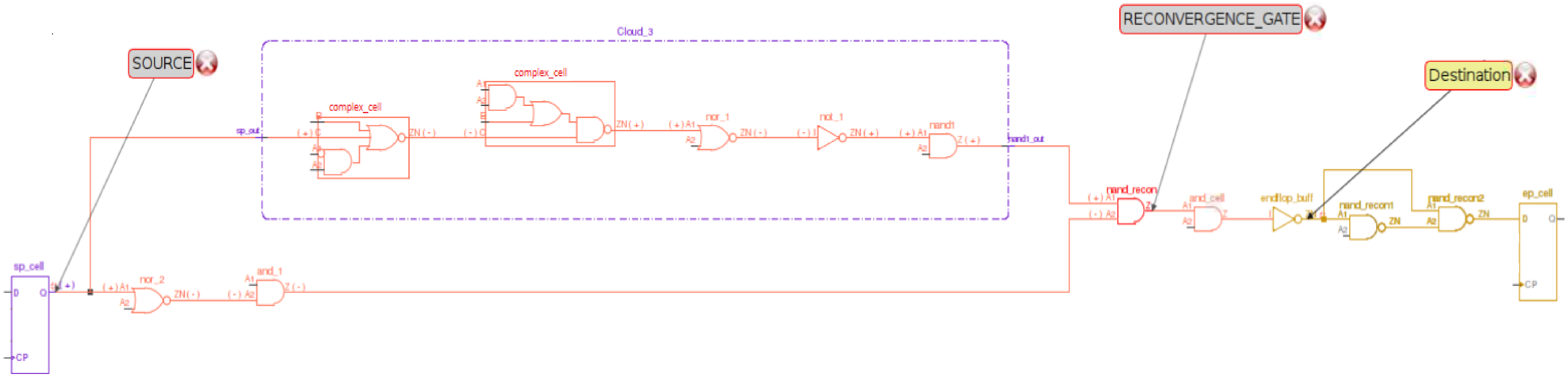
Example Structures for Glitch Check

- VC Spyglass CDC identifies glitches on controlpath



Example Structures for Glitch Check

- VC Spyglass CDC identifies glitch on user specified point2point paths



Glitch Check Performance

- Performing glitch checks on designs containing 250M instances, 700 clocks, 35M crossings

- Glitch check results:

Type of Glitch Check	Violations	Runtime
Controlpath Glitch Check	650K	0.75 hr*
Datapath Glitch Check	80K	1 hr*

  – Checks run using 8 threads
  – * This is additional runtime. Total runtime for regular CDC runs : ~20 hrs

Summary and Future Works

Summary

- Glitch check is a signoff check at netlist level
  - Ensuring designs are free from structural glitches due to combo logic
- Glitches needs to be checked on
  - Asynchronous crossings
  - Synchronous paths with exceptions
- VC Spyglass CDC tool provides various glitch check capabilities
  - Glitch checks on asynchronous, synchronous and special p2p paths
  - We utilize VC Spyglass CDC for netlist level CDC runs

Future Works

- Exhaustive glitch blocking analysis
  - Glitches blocked by potential qualifier(s) synced to destination
- Exhaustive multi source glitch analysis
  - Multi source convergence to report all type of converging fanins
- Incremental glitch check
  - Performing glitch check on multiple set of paths incrementally

References

VC Spyglass CDC User Guide
Solvnet

Questions

THANK YOU

Our
Technology,
**Your
Innovation**™