Addressing Reset Domain Crossing Issues at Block-Level and Top-Level in Complex SoC Using VC-Spyglass

Harshavardhan Vajjaramatti & Isuru Athapattu
Nokia

Agenda

- Importance of resets in semiconductor development.
- Reset Domain Crossing (RDC)
- Handling Reset Domain Crossing (RDC) in complex SoCs.
- Using VC SpyGlass to identify and verify RDC issues
- Advanced RDC analysis using VC-Spyglass
- Conclusion

Resets in SoC

- Need of a reset
- Asynchronous and Synchronous resets
- Increased complexity in SoC requires multiple resets.

- Cold Power-on-Reset (PoR)

  Trigger : External signal or internal under-voltage condition.

  Scope : Entire chip.

  Impact : Deepest propagation; resets every circuit, processor, memory element, and interface.

  Use Case : Complete initialization from a powered-off state.

- Warm Power-on-Reset (PoR)

  Trigger : External signal or internal self-system recovery.

  Scope : Entire SoC or specific subsystems.

  Impact : Resets most operational states but may retain some settings; less propagation than Cold PoR.

  Use Case : Restart without complete power-off.

Resets in SoC

- Hardware Resets

  Trigger : Built-in hardware mechanisms.

  Scope : Designated partitions or IP blocks.

  Impact : Localized reset; more limited than PoR.

  Use Case : Automatic response to specific events/conditions.

- Software Resets

  Trigger : Software mechanisms (e.g., watchdog timer).

  Scope : Software-driven states or data paths.

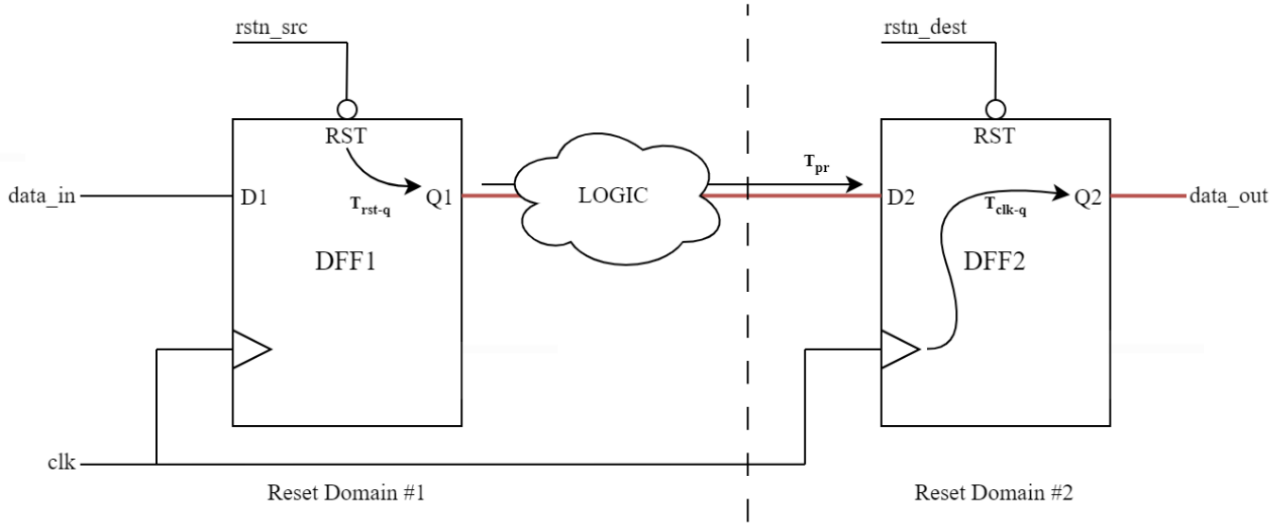
  Impact : Limited impact; does not affect hardware configuration.

  Use Case : Reinitialize software states without disrupting hardware.

Reset Domain Crossing

- A **reset domain** is a specific partition of the chip that operates under a unique reset signal or set of reset signals, asserted together during operation.

- How RDC occurs?



Block-Level Verification

Block-level verification aims to ensure the functionality, performance, and reliability of individual subsystems within a System-on-Chip (SoC) before delivering them to the subsystem or SoC top-level .

RDC Challenges

- Data Corruptions
- Clock corruptions
- Glitches
- Convergence.

Top-Level Verification

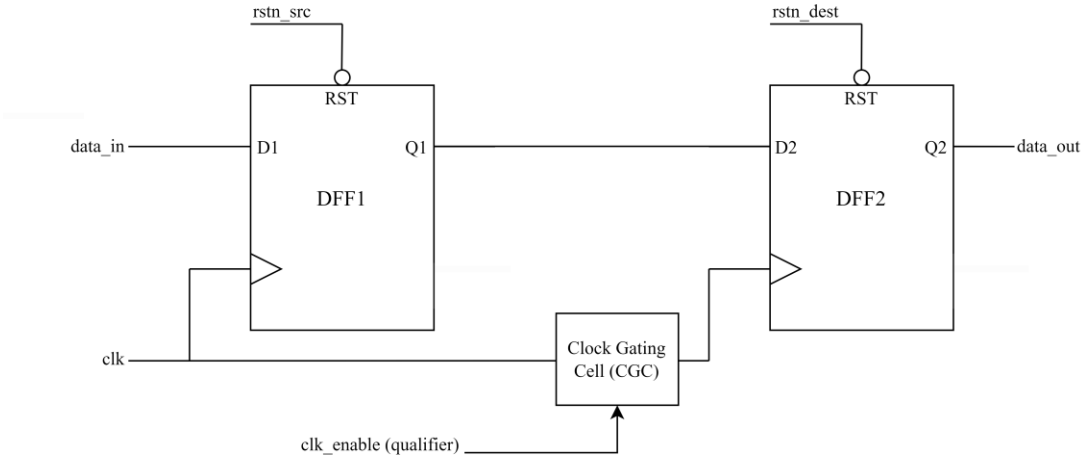
The top-level of a System-on-Chip (SoC) is where all subsystems and some blocks are integrated, making it the most complex layer.

RDC Challenges

- Data corruptions between sub-blocks
- Constraint complexity and migration
- Extreme flat-level violation count
- Resource utilization (Processing Power/Memory)

Addressing RDC issues

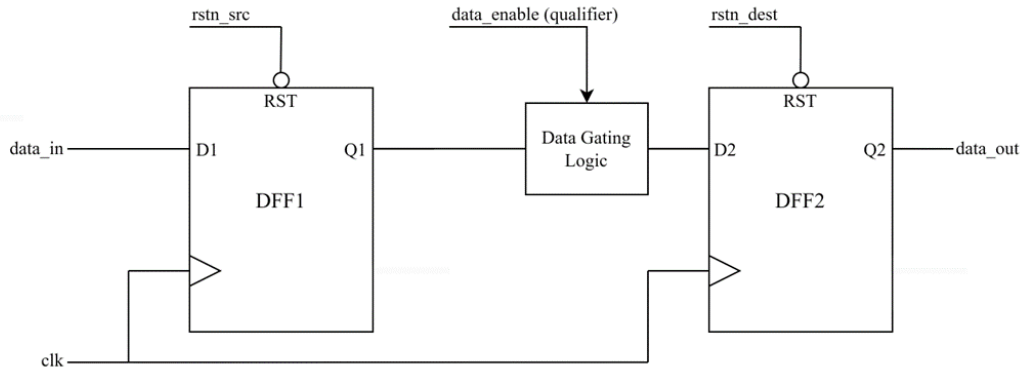
Clock Gating



- **Mechanism:**
  - **Clock Gating Cell (CGC):** Controls clock input of flip-flops.
  - **Control Signal (clk_enable):** Qualifier signal to initiate gating.
- **Implementation:**
  - Lower clk_enable before asserting reset.
  - Block active clock edges to prevent data capture.
  - Synchronize clk_enable to destination clock in case of CDC.
- **Clock Revival:**
  - **Immediate Capture:** Reactivate clock one cycle after reset assertion.
  - **Hold Value:** Reactivate clock after de-asserting source reset.

- **Advantages:**
  - No additional data path delay during normal operation.
  - Single CGC can handle high fan-out, simplifying RDC management.
  - Reduced dynamic power consumption during reset assertion.
- **Disadvantages:**
  - Requires additional qualifier signal, increasing design complexity and verification workload.
  - Qualifier signal synchronization needed if generated from a different clock domain.
  - Adds complexity to clock distribution tree and potential gate delays.(Remove)

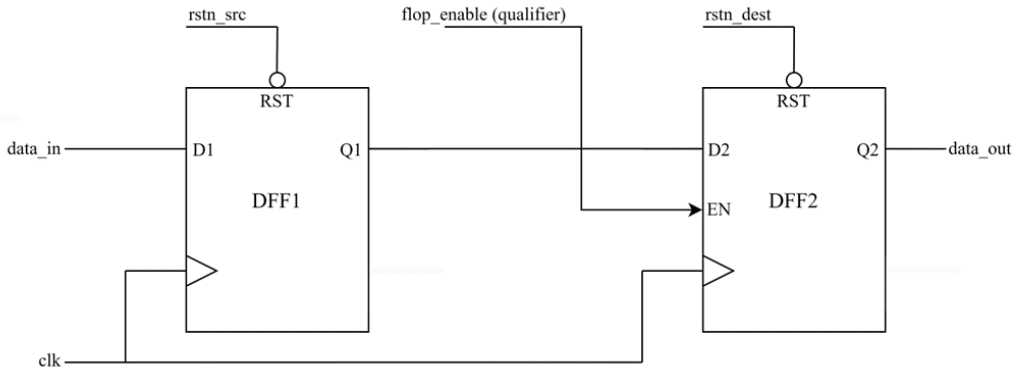
Data Gating



- **Mechanism:**
  - **Data Blocking:** Uses AND/OR gates with a qualifier signal (data_enable) to block data transitions.
  - **Implementation:**
    - AND gate for capturing logic level 0.
    - OR gate for capturing logic level 1.
    - Synchronize data_enable with the destination clock.
    - Alter data_enable to block data propagation before source reset assertion.

- **Advantages:**
  - Simplifies clock network by focusing on data paths.
  - Easier to implement than complex clock gating logic.
- **Disadvantages:**
  - Requires additional qualifier signal and synchronization scheme.
  - Each RDC path needs separate implementation, introducing combinational delays.
  - Potentially higher power utilization due to added logic and continued clock switching. -> Static power

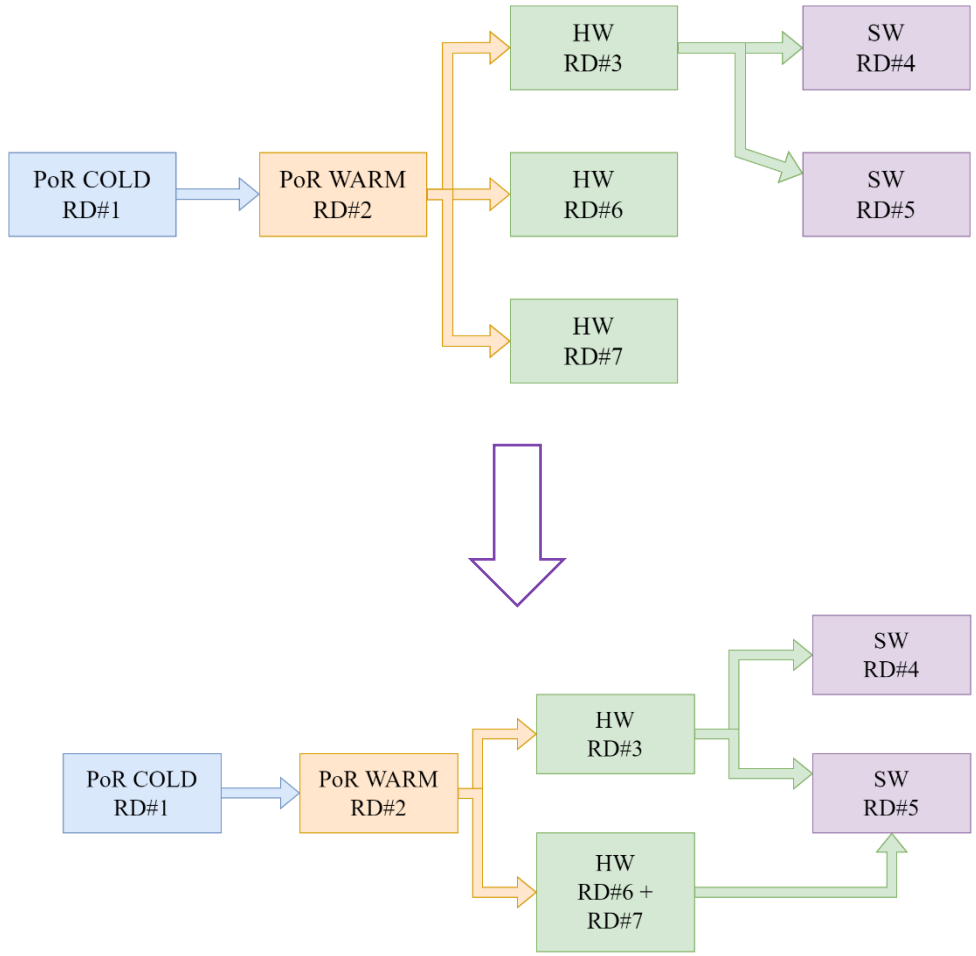
Flop Enable/Disable



- **Mechanism:**
  - **MUX and D Flip-Flop Combination:** Most cell libraries include D flip-flops with an enable pin, avoiding separate gating mechanisms.
- **Implementation:**
  - **flop_enable:** Acts as the select signal for the MUX inside DFF2.
  - **Operation:** Before source reset, flip flop_enable to select constant tied input.
  - **Synchronization:** Ensure flop_enable is synchronous with the destination clock to avoid asynchronous output.

- **Advantages:**
  - Simplifies integration using existing library cells with enable pins.
  - Effectively blocks data propagation across entire reset domain with a single high fan-out qualifier signal.
  - Easier implementation compared to adding separate logic gates.
- **Disadvantages:**
  - Propagation Delay: Potential increase depending on MUX structure.
  - Shares common disadvantages with data gating (e.g., need for additional qualifier signal, synchronization, and verification).

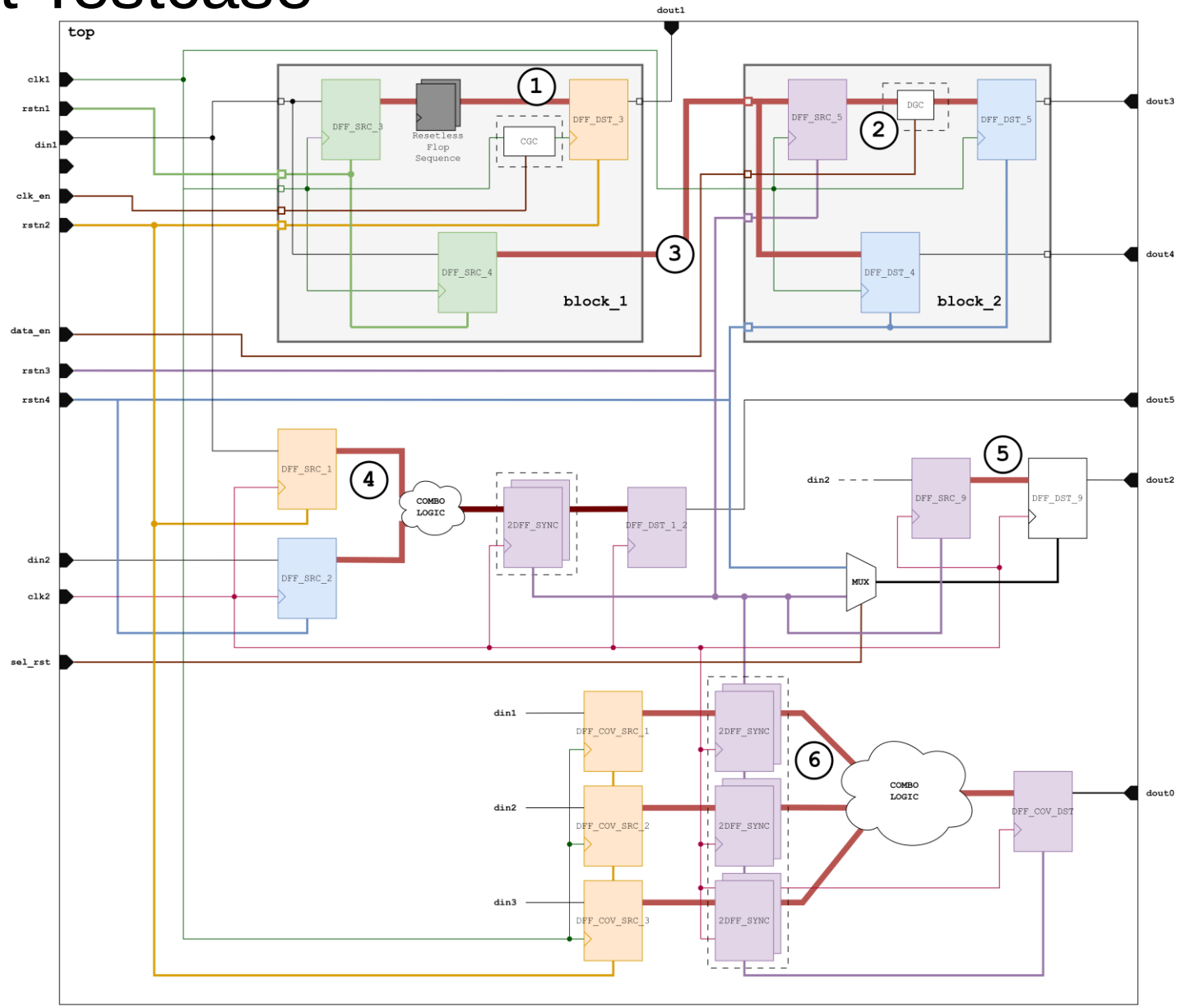
Reset Re-Sequencing



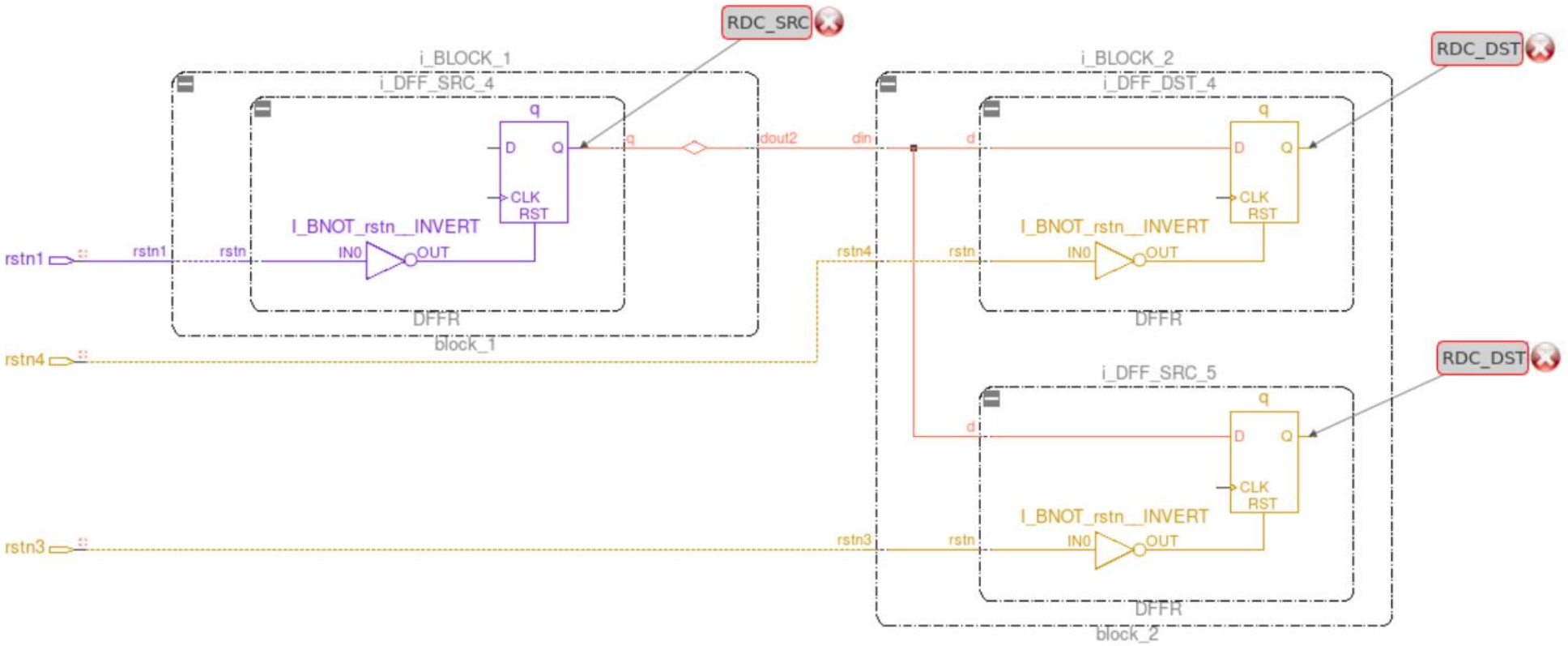
- **Hierarchy and Impact:**
  - Cold PoR: Most extensive impact.
  - Warm PoR: Forces assertion of all HW reset domains.
  - HW Resets: Localized to specific partitions or IP blocks.
  - SW Resets: Least impact, focused on software states.
- **Hierarchy of Assertion:**
  - **Directional RDC Paths:**
    - Safe by inherent reset assertion order (arrow direction).
    - Vulnerable in opposite direction, potentially causing metastability.
  - **Other Potential RDC Paths:**
    - No inherent assertion order.
    - Requires design optimization for proper reset sequence.
- **Optimization Strategies:**
  - **Reset Domain Merging:**
    - Minimize number of asynchronous reset domains.
    - Reduce RDC paths needing management.

Using VC SpyGlass RDC

DUT – Unit Testcase

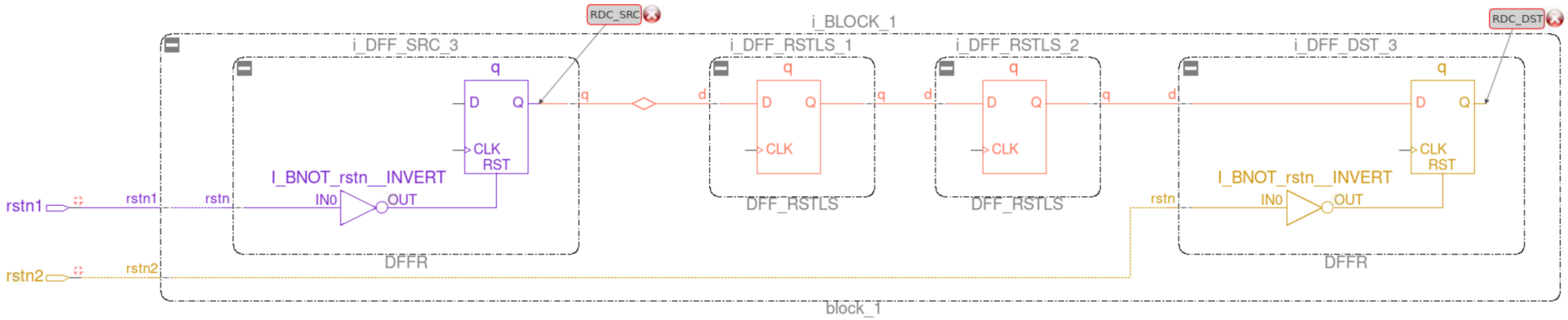


RDC resolved by reset assertion re-ordering

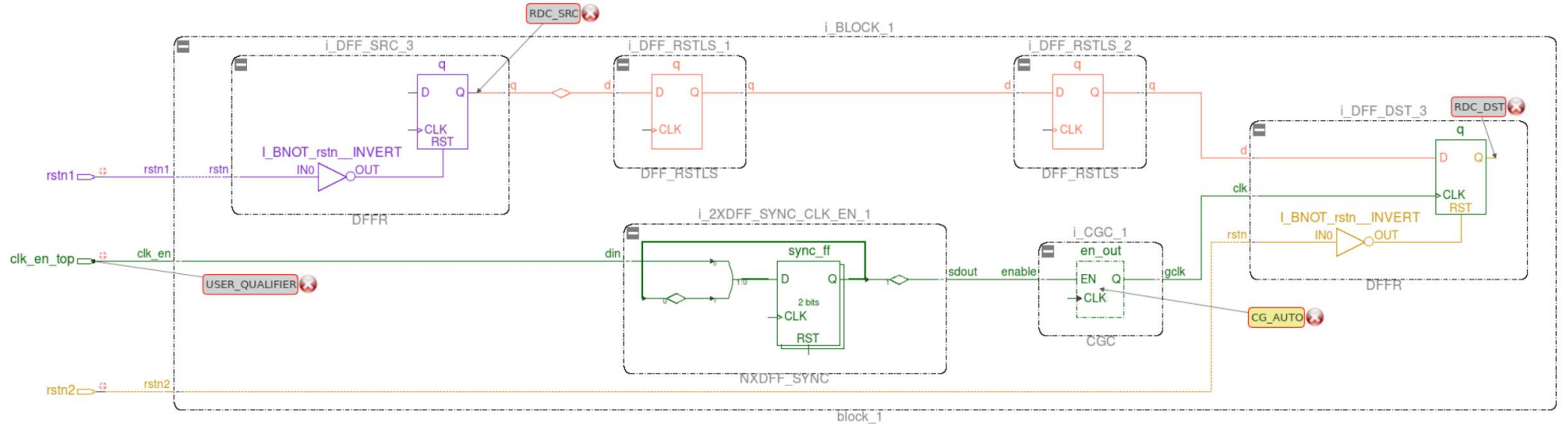



```
set_rdc_define_assertion_sequence -from_reset RSTN_1 -to_reset { RSTN_3 RSTN_4 }
```

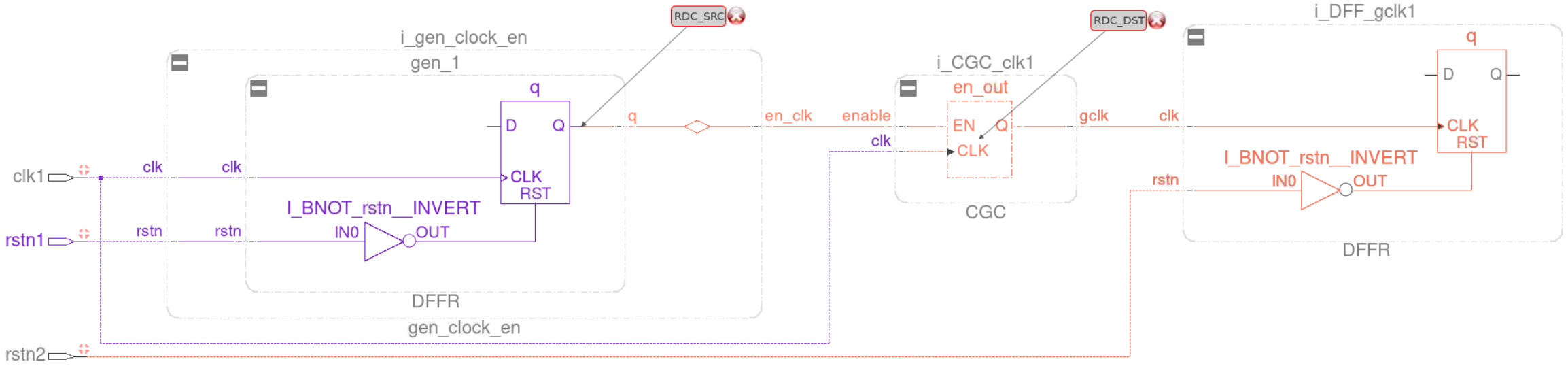
RDC with resetless flop chain handled with Clock-Gating




```
set_rdc_qualifier -object clk_en_top -from_reset RSTN_1 -to_reset RSTN_2 -depth 2
```



Clock Corruption

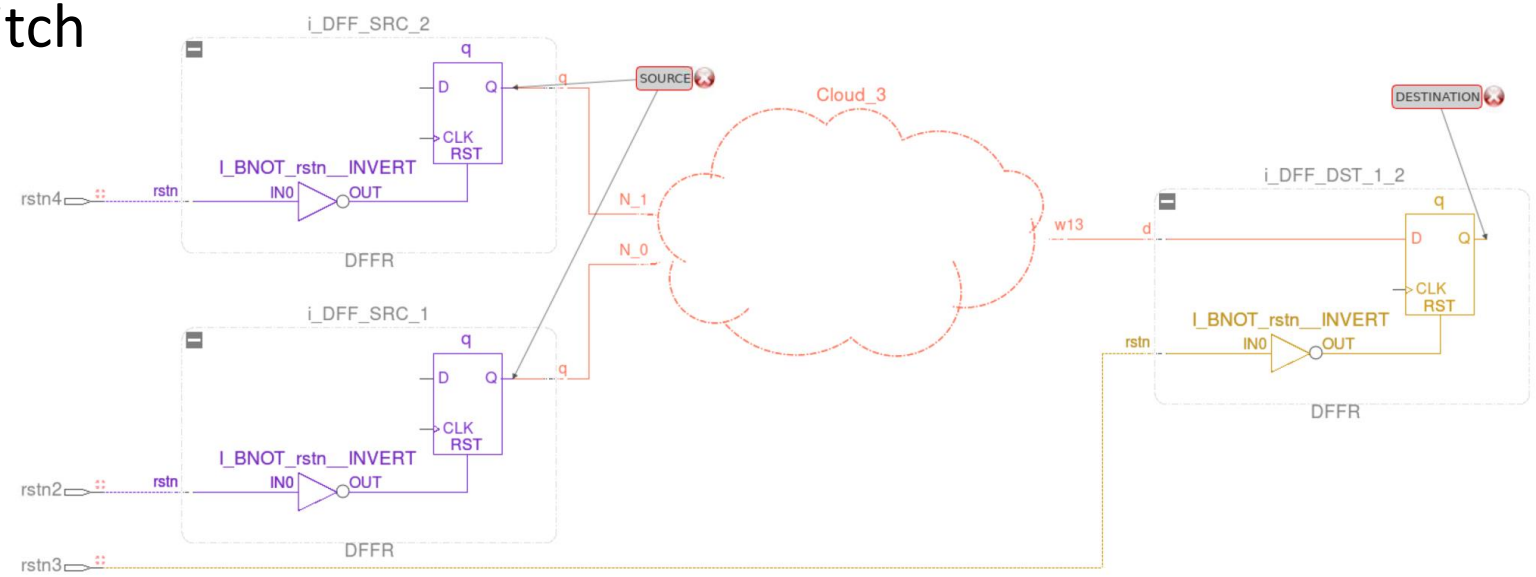


Why?

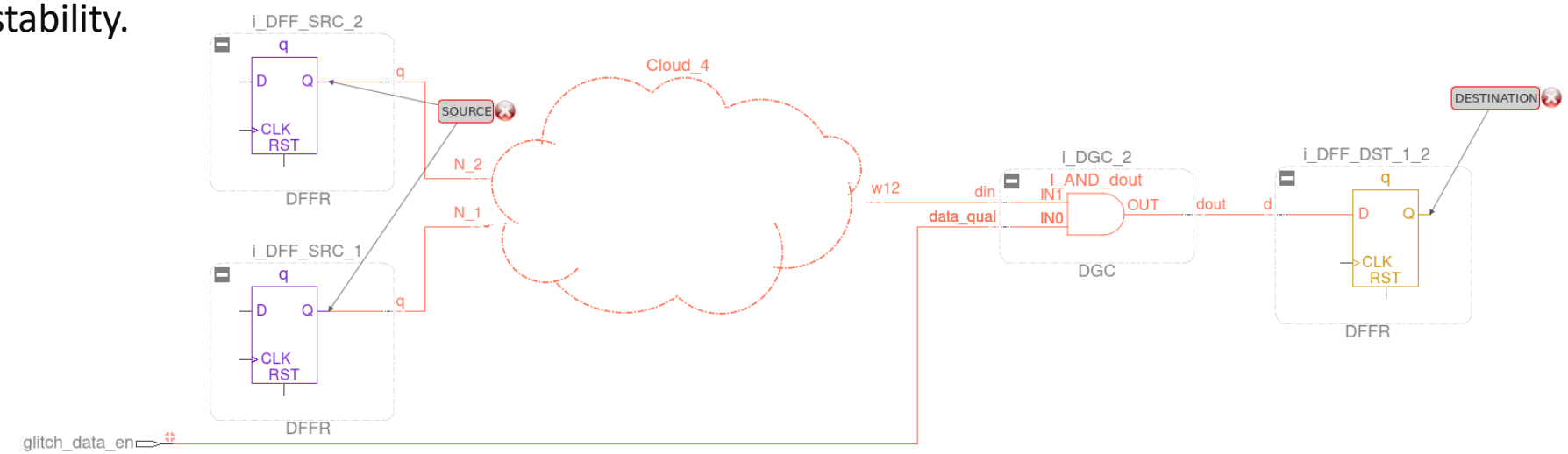
Clock enable signal at the clock gate input, transitions asynchronously because of the asynchronous reset assertion at its source.

The impact of clock corruption can be mitigated by using the same destination reset to assert the clock enable synchronizer reset.

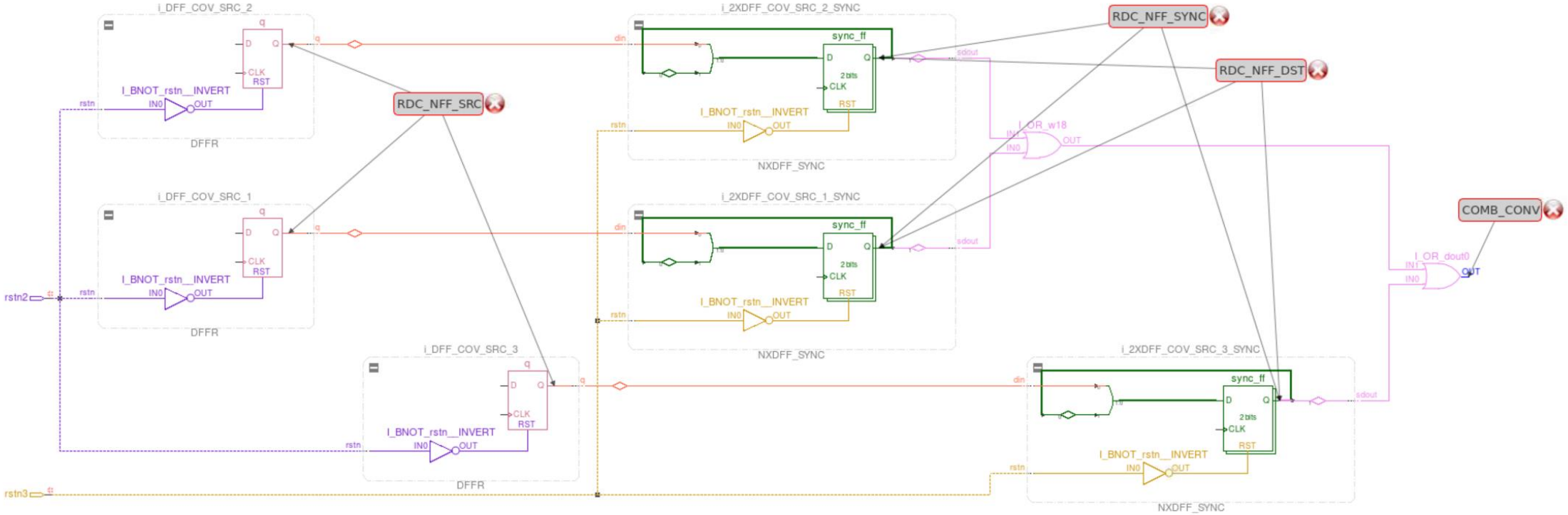
RDC Glitch



Data gating is used in an RDC path which in upstream combined with another RDC path.
If DFF sync is used, metastability will not fully resolve since potential glitches can throw the destination flip-flop to metastability.



RDC Convergence



Separately synchronized (using flip-flop synchronizers) RDC paths from same or different reset domains converged in downstream logic.

Erroneous due to uncertain one clock cycle delay of flip-flop synchronizers.

Can cause data integrity issues.

Hierarchical RDC using VC-Spyglass

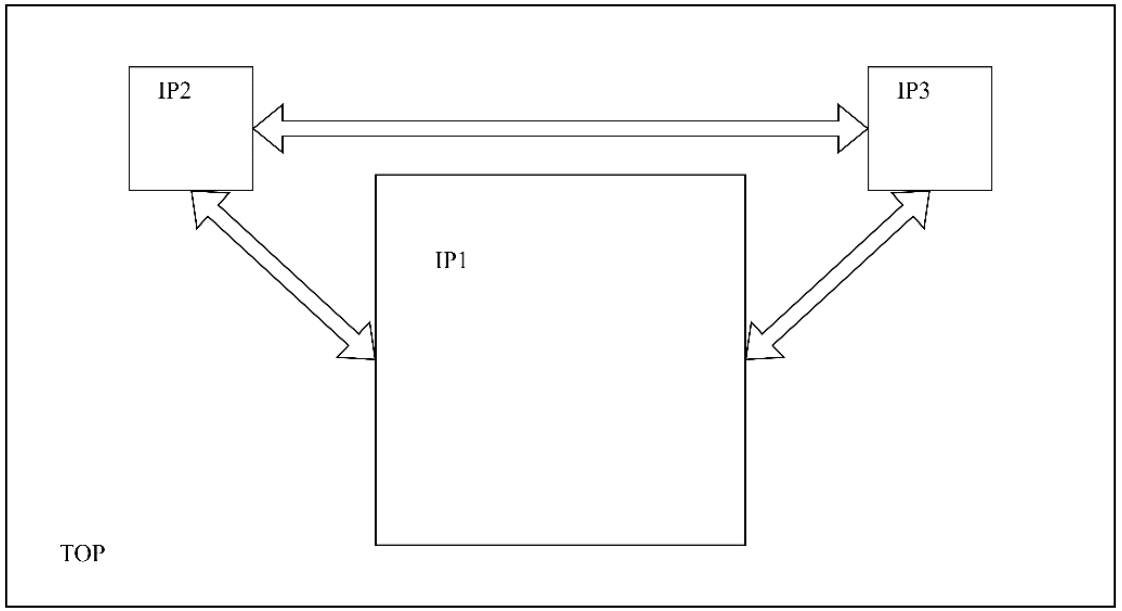
Block-level Approach

- Fixing and completing design-constraints
  - `create_clock, create_reset, set_case_analysis, clock and reset groups`
- Check for reset order optimizations using reset assertion re-ordering and possible reset mergers.
  - `set_rdc_define_assertion_sequence`
- Selecting a suitable mechanism to resolve an RDC considering,
  - Complexity of the required design change to implement the solution.
  - Added path delays
  - Implementation of controlling mechanism for blocking scheme.
  - Impact on static/dynamic power utilization
  - `set_rdc_qualifier`
- Deliver the sign-off abstract model to the top-level

Top-level Approach

- Utilizing RDC hierarchical flow with SAM
  - To reduce unnecessary analysis of internal logic inside subsystems.
  - Avoid reporting violations that does not concern at the top level
  - Avoid over utilization of computational resources.
- Address Constraints mismatches between the top-level and block levels abstract models.
- Analysis of best RDC mechanism to resolve inter-subsystem violations. Considering,
  - Possible reset-assertion order changes and reset domain mergers.
  - Possibility to resolve the violation internally in subsystem.
- Automating violation report(s) post-processing, categorizing and distribution.

DUT – Subsystem



	Type	IP1	IP2	IP3	TOP
Clocks	Total	21	19	8	33
	Primary	5	13	3	14
	Generated	16	5	5	19
Clock Domains		6	4	3	7
Resets	Total	7	7	3	15
	Primary	5	3	3	9
	Generated	2	4	0	6
Reset Domains		2	1	2	4

Block-level and Top-level results

Violation	Initial	Constraint fix	Clock-gating fix
RDC Corruptions	G:4, C:79447	G:2 , C:32322	G:2 , C:2558
RDC Potential Corruptions	-	-	G:1 , C:294
RDC Safe by Assertion Order	-	G:2 , C:47125	G:2 , C:47125
RDC Corruptions Blocked	-	-	G:1 , C:29470

Violation	FLAT level		HIER – Sign-off Abstract Model flow	
	Initial	Assertion Order Fix	Initial	Assertion Order Fix
RDC Corruptions	G:22 , C:127770	G:11 , C:72799	G:16 , C:48331	G:4, C:34621
RDC Clock Corruptions	G:24 , C:5392	G:17 , C:526	G:3 , C:16	G:2 , C:12
RDC Potential Corruptions	G:7 , C:6581	G:4 , C:4189	-	-

Conclusion

- We have outlined a series of methodologies and VC-Spyglass tool that not only reduce the risks related to RDC but also streamline the verification process at both the block-level and the top-level.
- The strategic implementation of blocking techniques, such as clock and data gating, along with proper utilization of VC SpyGlass, has played a crucial role in tackling RDC issues and ensuring the functionality of the SoC.
- This study has emphasized the significance of adopting advanced RDC verification methods like hierarchical verification, glitch analysis, convergence etc.


THANK YOU
Our Technology, Your Innovation™