

# Build Reliable and Efficient Reset Networks with a Comprehensive Reset Domain Crossing Verification Solution

Wanggen Shi, Big Fish Semiconductor Ltd., China

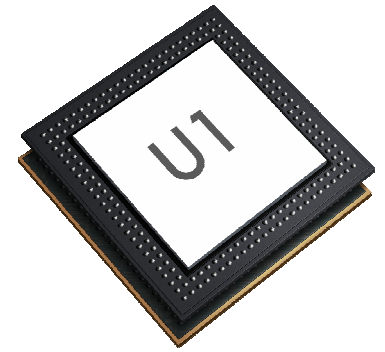
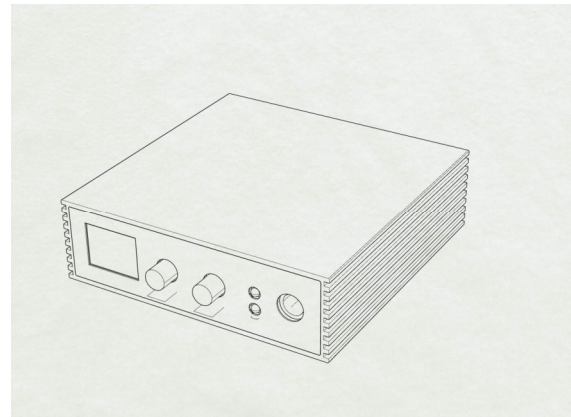
Yuxin You, Mentor, a Siemens Business, China

Kurt Takara, Mentor, a Siemens Business, USA



# BigFish Overview

- Focus on AI & IoT and chip solution
- Capability including SoC, sw Dev.& OS, Modem tech, Software&Hardware system integration and Design of 2C&2B products
- Products including mobile, UAV, super Ethernet ,IoT...

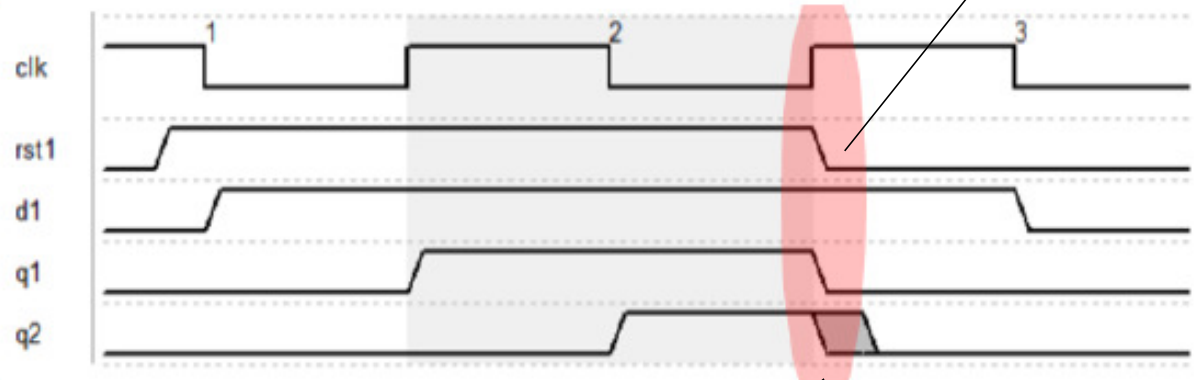
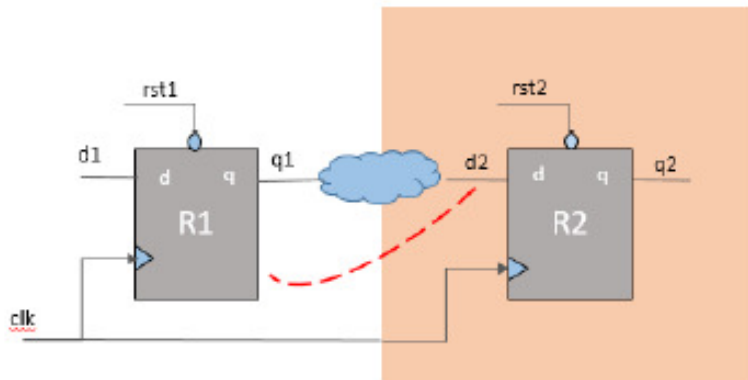


# The Need for RDC Verification

- Why reset issues are a problem?
  - Asynchronous reset domain crossing cause metastability
  - Reset issues result in unreliable functionality or possible silicon damage
  - Functional simulation detection is probabilistic
- Reset domain crossing (RDC) verification
  - Static and formal methods detect RDC issues in RTL designs

# What is RDC?

- Data crossing from one async reset domain to another
- Transmitting(Tx) flop async-reset assertion close to clock edge can cause metastability on receiving(Rx) flop

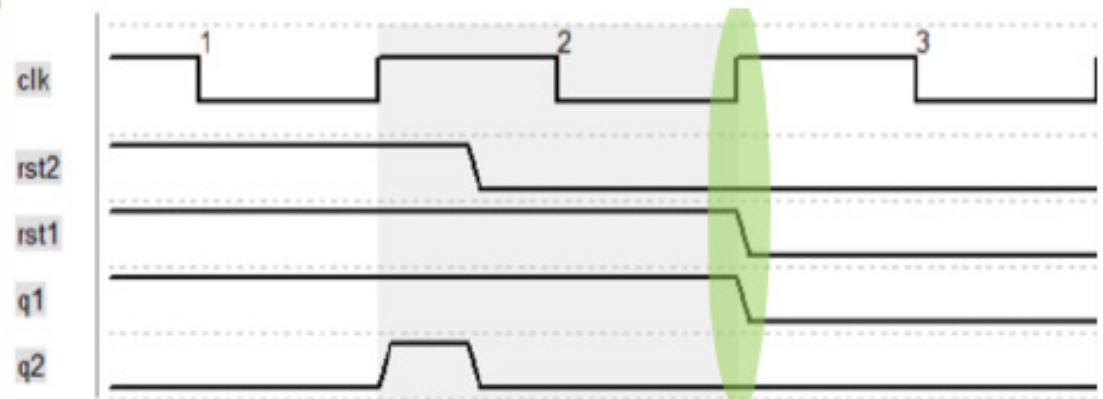
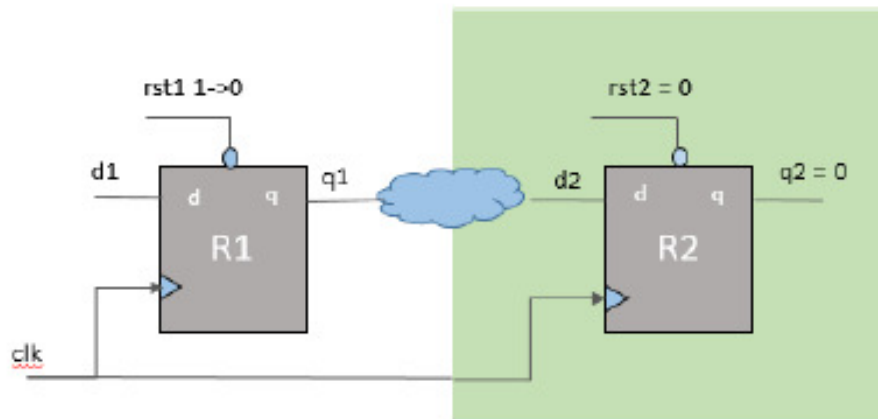


'rst1' asserted very close to 'clk' posedge

'R2' flop output goes metastable due to setup/hold time violation

# Techniques to Address RDC issues

- Reset Sequencing
  - Async-reset on Rx flop always asserts before async-reset on Tx flop
  - Rx flop already in reset state, so any change on Rx D-pin will not cause metastability

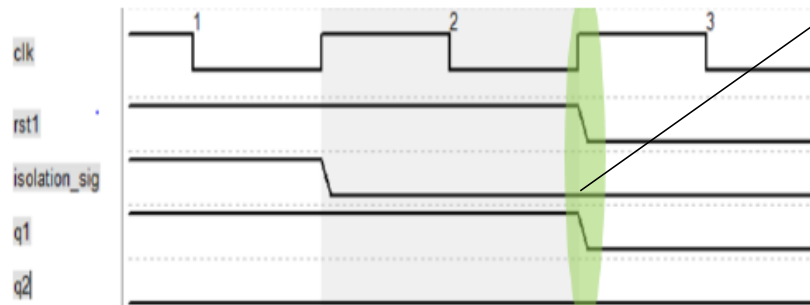
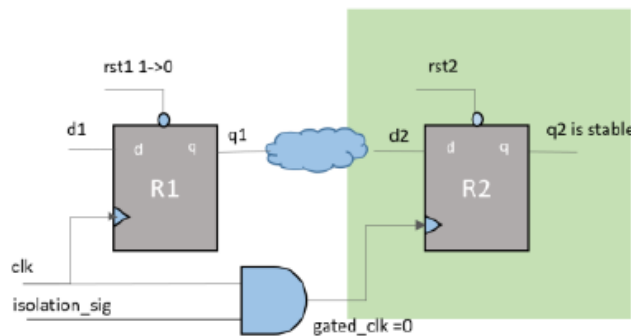


# Techniques to Address RDC issues

- Isolation Techniques

- Clockgate isolation

- Turn off clock of Rx flop before Tx reset asserts
    - If clock is off, then any change on Rx D-pin will not cause metastability



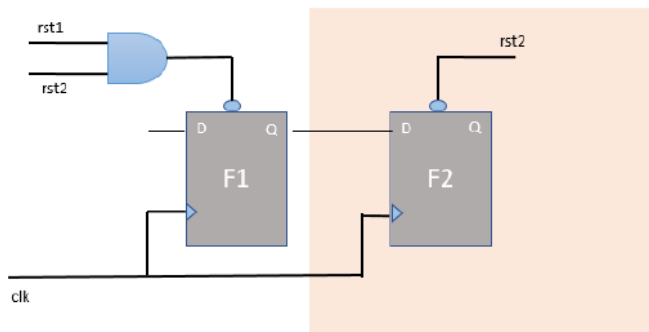
Isolation signal is 0, turning clock of 'R2' to constant 0 when 'rst1' asserts

- Data Isolation

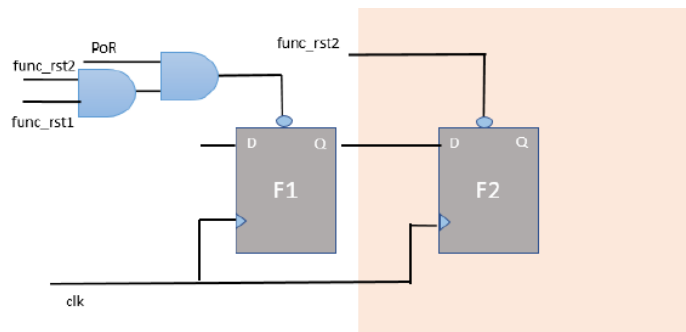
- Block Tx to Rx data transmission through isolation signal before Tx reset asserts

# Real-world RDC Issues Examples

- Combination of resets on Tx or Rx flop



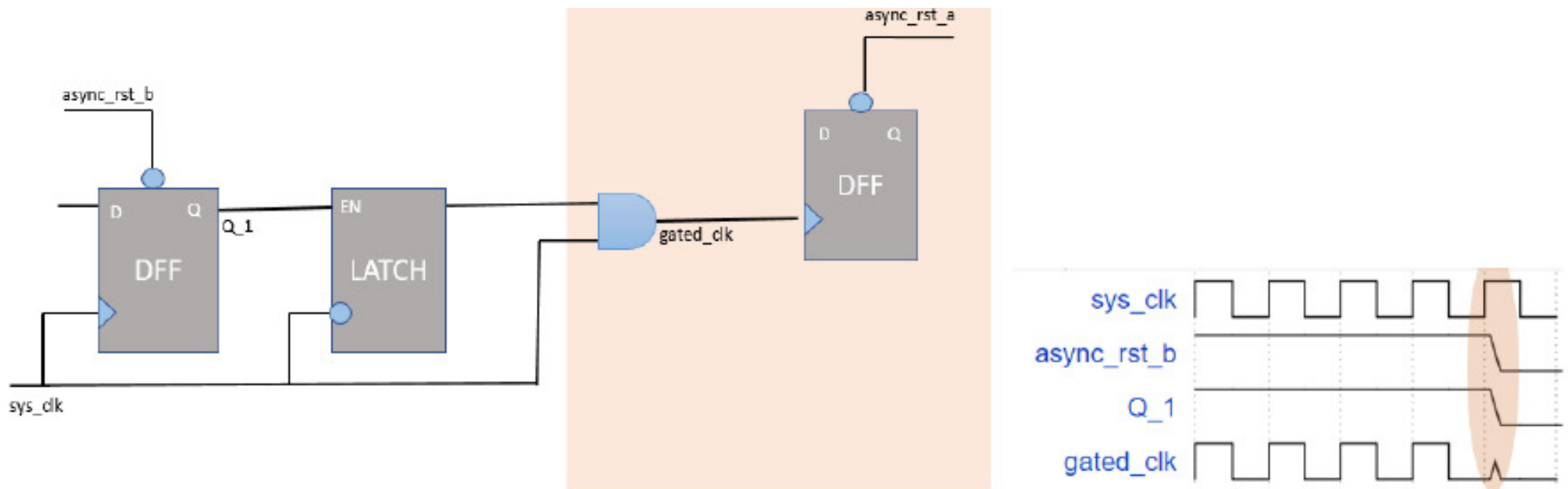
If rst1 asserts before rst2, metastability can occur on F2



If func\_rst1 or PoR asserts before func\_rst2, metastability can occur on F2

# Real-world RDC Issues Examples

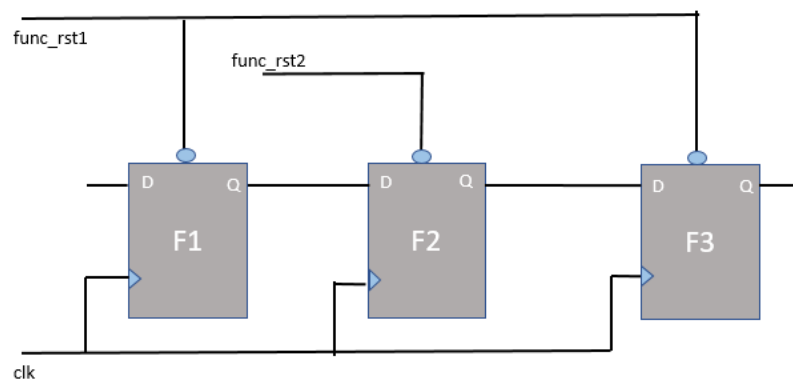
- Glitch in gated clock output due to different asynchronous reset



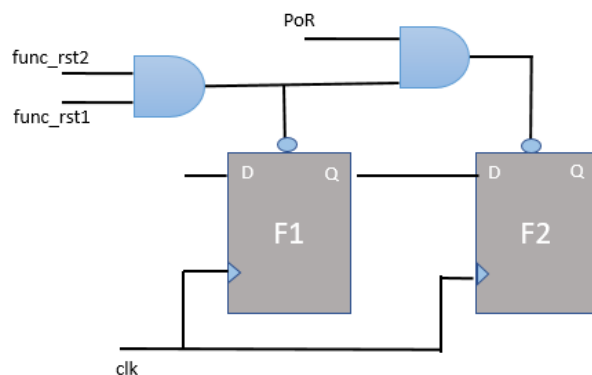


# Safe-RDCs Examples

- Some scenarios which may look to be RDC issue, but actually safe paths

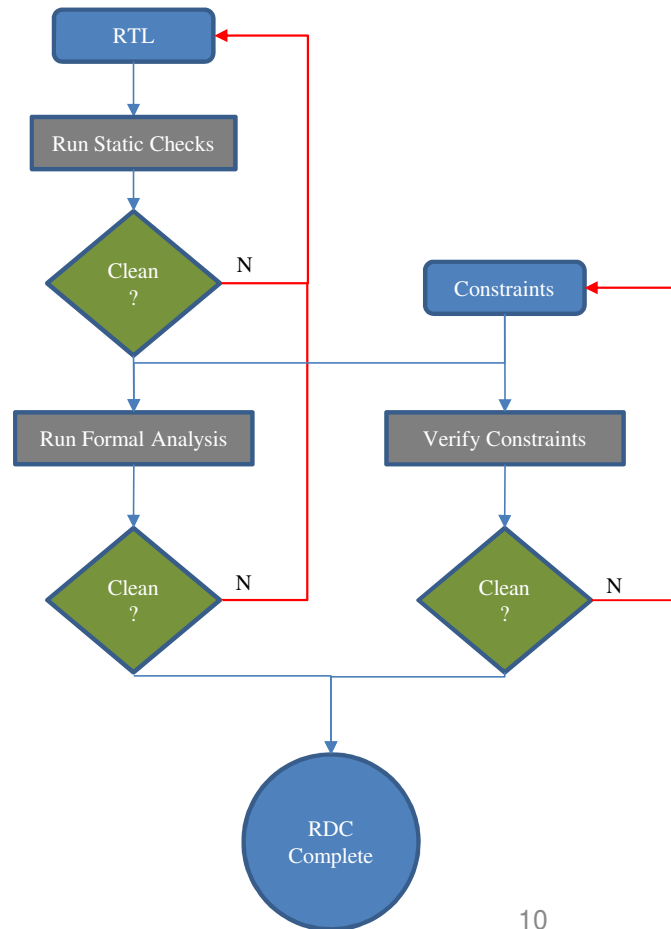


Fanouts of Rx Flop(F2) transmits to Tx Reset Domain (func\_rst1)  
Tx reset assertion blocks metastability transmission



When Tx reset is asserted (through func\_rst1/func\_rst2) it always asserts Rx reset

# Recommended RDC Verification Methodology



# RDC Verification Preparation

- Static reset checks
  - Validate reset tree integrity and RDC verification readiness
  - Assist in developing constraints
- Quality constraints
  - Defines reset sequencing and safe isolation enables
  - Isolation enables validated by functional checks

# Case Study of Project Usage

- Glitches found by static checks
  - Static reset check identified potential reset glitch
  - Reset glitch would occur for specific state of state machine
- Identified asynchronous FIFO issue
  - Asynchronous FIFO used in a synchronous application
  - TX and RX resets on asynchronous resets
  - RDC verification identified a violation due to missing constraint

# Reset Glitch Detection Case

Questa Verify (Output\_Results/resetcheck.db)

File View Compile Verify Tools Reports Logs Layout Window Help

h:/zint/testcases2/ktakara/fishsemi/test\_reset\_glitch/fsm\_combo\_rst.v - Default

```
Ln# 32 //===== fsm combo rst
33
34 assign work_flag_rstn = test_mode ? rstn : rstn && !(c_st==PD);
35
36 always @(posedge clk or negedge work_flag_rstn) begin//{
37   if(!work_flag_rstn) begin//{
38     work_flag <= 0;
39     active_r <= 0;
40   end //}
41   else begin//[
42     active_r <= {active_r[0],active};
43     if (work_en)
44       work_flag <= 1;
45   end //]
46 end //always @}
```

reset\_combo\_glitch\_1454232 - Default

Reset Static Checks

☐ Waived ☒ Fixed ☒ Pending ☒ Uninspected ☒ Bug ☒ Verified

Severity	Status	Check	Reset	Rst Usage1	Reg Usage1	Rst Usage2	Reg Usage2
Caution	Uninspected	Potential Glitch In Reset Pa...	work_flag_rstn	async/reset/low			active_r

Transcript x Clocks x Resets x Reset Static Checks x Reset Checks x

# FIFO RDC Detection Case

Questa Verify (Output\_Results/resetcheck.db)

File View Compile Verify Tools Reports Logs Layout Window Help

ResetCheck

Layout Last Settings

Design

Instance Design Unit Design Unit Type

demo\_top (3) demo\_top Top Module

rdc\_aset\_6255538 - Default

fifo\_1\_d

almost\_empty\_d  
almost\_empty\_s  
almost\_full\_d  
almost\_full\_s  
clr\_cmplt\_d  
clr\_cmplt\_s  
clr\_in\_prog\_d  
clr\_in\_prog\_s  
clr\_sync\_d  
clr\_sync\_s  
data\_d  
empty\_d  
empty\_s  
error\_d  
error\_s  
fifo\_empty\_s  
fifo\_word\_cnt\_s  
full\_d

ae\_level\_d  
ae\_level\_s  
af\_level\_d  
af\_level\_s  
clk\_d  
clk\_s  
clr\_d  
clr\_s  
data  
data\_s  
init\_d\_n  
init\_s\_n  
pop\_d\_n  
push\_s\_n  
rst\_d\_n  
rst\_s\_n

data

RST

Reset

Project Library Directives Design Modules Reset Policy Checks

Reset Checks

Severity Status Check Tx Signal Rx Signal Tx Reset Tx Reset Type Rx Reset Rx Reset Type Tx Clock

Violation...	Uninspected	Reset Cdc Has No Synchronizer (69)						
Violation...	Uninspected	Reset Domain Crossing From Areset To Areset (6)						
Violation...	Uninspected	Reset Domain Crossing From Areset To Areset	fifo_1_d.<protected>.inu...	pass_valid	clr	user/asynresetflow	rst	user/asynresetflow
Violation...	Uninspected	Reset Domain Crossing From Areset To Areset	header	tx_wcnt	clr	user/asynresetflow	rst	user/asynresetflow
Violation...	Uninspected	Reset Domain Crossing From Areset To Areset	data	tx_mask	clr	user/asynresetflow	rst	user/asynresetflow
Violation...	Uninspected	Reset Domain Crossing From Areset To Areset	fifo_1_d.<protected>.dat...	crc_1.scramble	clr	user/asynresetflow	rst	user/asynresetflow
Violation...	Uninspected	Reset Domain Crossing From Areset To Areset	data	fifo_1_d._zi_cne_w08...	clr	user/asynresetflow	rst	user/asynresetflow
Violation...	Uninspected	Reset Domain Crossing From Areset To Areset	mask	rx_masked_data[0]	rst	user/asynresetflow	clr	user/asynresetflow
Violation...	Uninspected	Combinational Logic Before Rdc Synchronizer (2)						

Transcript Clocks Resets Reset Setup Checks Reset Static Checks Reset Checks

Filters in use: 0 <No Context>

# Summary

- Increase in SoC reset architecture complexity requires RDC verification
- RDC verification methodology provides completeness and efficiency
- RDC verification metrics demonstrate verification completion
- Achieved our goal of reliable silicon operation!

Thanks you