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

'R2' flop output is already 0, when Tx reset asserts
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
  – Data Isolation
    • Block Tx to Rx data transmission through isolation signal before Tx reset asserts

Isolation signal is 0, turning clock of ‘R2’ to constant 0 when ‘rst1’ 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

1. RTL
   - Run Static Checks
     - Clean ?
       - Constraints
         - Verify Constraints
           - Clean ?
             - N
             - N
       - N
   - N

2. RDC Complete
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

```verilog
assign work_flag_rstn = test_mode ? rstn : rstn && (c_st==PD);

always @(posedge clk or negedge work_flag_rstn) begin//
    if(work_flag_rstn) begin//
        work_flag <= 0;
        active_r <= 0;
    end //
    else begin//
        active_r <= {active_r[0],active};
        if (work_en)
            work_flag <= 1;
    end //
end //always @
```
FIFO RDC Detection Case
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