Accelerating CDC Verification Closure on Gate-Level Designs

Anwesha Choudhury, Mentor Graphics
Ashish Hari, Mentor Graphics
Agenda

- Why CDC Verification on Gate-Level Designs
- Traditional Methodology and Challenges
- Proposed Gate-CDC Verification Methodology
- Experiments & Results
Clock Domain Crossing (CDC)

- What is CDC
  - Signal originating in one clock domain sampled in another asynchronous clock domain
- CDC issues are #2 reason for silicon respins
Need for Gate-Level CDC Verification

- Traditionally CDC verification done on RTL

Possible CDC issues introduced during synthesis
- Glitch-prone combinational logic in CDC path
- New CDC paths due to insertion of Power/DFT logic
- CDC synchronizer template may break due to logic optimizations

Silicon Respin
Gate-CDC Example

- CDC path introduced due to insertion of logic

```upf
upf_version 2.0
set_design_top top
create_power_domain TOP
create_power_domain PD_TX -elements (Tx)
create_power_domain PD_RX -elements (Rx)
create_power_domain PD_RX2 -elements (Rx2)
create_supply_port VDD_HIGH
create_supply_net VDD_HIGH -domain TOP
connect_supply_net VDD_HIGH -ports VDD_HIGH
set_isolation PD_TX_ISO_OUT -domain PD_TX -clamp_value 1 "
- applies_to outputs -isolation_power_net VDD_HIGH "
- isolation_ground_net VSS -isolation_signal (x) -isolation_sense low -location parent
set_isolation PD_TX_ISO_OUT -domain PD_RX -clamp_value 1 "
- applies_to outputs -isolation_power_net VDD_HIGH "
- isolation_ground_net VSS -isolation_signal (x) -isolation_sense low -location parent
```
Gate-CDC Glitch Example

• Combination logic that can reduce to:

\[ x \& \sim x \]

\[ x \mid \sim x \]
Gate-CDC Glitch Example

RTL Logic: Mux based synchronizer

Mux implementation after synthesis

Glitch-prone: sel | ~sel when A = 1 and B = 1

Glitch blocked: Output = 1 when A = 1 and B = 1
Gate-CDC Glitch Example

RTL Logic: Mux based synchronizer

Combo-logic implementation after synthesis

For given constants, logic reduces to (tx0|~tx0) which causes glitch
Gate-CDC Glitch Example

RTL Logic: Mux based synchronizer

Combo-logic implementation after synthesis
• CDC verification is necessary on gate-level netlist
Traditional Methodology

Setup

Design

Setup Constraints: clocks, resets, constants, …

Run CDC analysis

Refine clocks

Analyze CDC results

Debug

yes

noise

Add constraints: stable, constant

Fix violations

no

Add waivers

CDC sign-off
Traditional Methodology

Setup
- Design
  - Setup Constraints: clocks, resets, constants, ...
  - Run CDC analysis
  - Refine clocks
  - Analyze CDC results
  - noise
    - yes: Add constraints: stable, constant
    - no: Fix violations
  - Add waivers
  - CDC sign-off

Huge setup effort
- Manually creating constraints setup not feasible

Noise
- bit-blasted CDC paths
- Scan inserted CDC path
- New hierarchies introduced
- Too many iterations needed
- Crossings multiplied by number of bits of vector signals in RTL
- False crossings due to inserted scan logic

Debug
- Difficult to debug
  - RTL waivers do not work
  - No correlation with RTL
- What about glitch?
Proposed Methodology

- **Setup**
  - CDC Analysis
  - Gate Design
  - Constraints at RTL

- **CDC Analysis**
  - Run CDC analysis
  - CDC results
  - Glitch results

- **Debug**
  - Fix CDC & glitch issues

- **Transformation engine**
  - Automated Setup
  - Reduced Noise
  - Debug Eased
  - Glitches Detected

- **Waivers at RTL**
  - Constraints at RTL reused
  - No iteration
  - Waivers at RTL honored
  - Correlation with RTL signals
  - Automated
  - Reduced Noise
  - Glitches detected
  - Debug eased

- **Additional analysis**
  - Additional logic for glitch
  - Focused glitch debug
RTL Constraints Reuse

- Constraints not applied due to name and topology changes post synthesis

- No black-boxing
  - Redundant processing inside the module

- Custom synchronizer not detected
  - False missing synchronizer - Noise
RTL Constraints Reuse

- **RTL Constraints**
  - `netlist blackbox cm_cdc_master`
  - `netlist blackbox cm_cdc_master_0_1`
  - `netlist blackbox cm_cdc_master_0_2`
  - `cdc custom sync qctlib_edge_detect_async_rs_ctrl`
  - `cdc custom sync qctlib_edge_detect_async_rs_ctrl_16`

- **Auto-generated gate constraints**

---

3/1/2022

Anwesha Choudhury, Mentor Graphics
RTL Waivers Reuse

Reduces noise and verification effort

- cdc report crossing -from {*io_cal_top.pcnt_reg_reg_0_.iq} -to {*ioc_pcnt_set_0_.iq} -severity waived
- cdc report crossing -from {*io_cal_top.pcnt_reg_reg_0_.iq} -to {*io_cal_top.pcnt_reg_reg_0_.iq} -severity waived

Auto-generated gate constraints

3/1/2022

Anwesha Choudhury, Mentor Graphics
Auto Infer Test Mode Settings

• Scan mux inserted CDC path

• Traditional methodology
  – Synchronizer not detected
Auto Infer Test Mode Settings

• Scan mux inserted CDC path

![Diagram showing scan mux insertion into CDC path]

• Proposed methodology
  – Structural analysis detects scan enable settings
  – Two dff synchronizer detected
Regroup Bit-blasted Crossings

**RTL: 16-bit data path**

**Gate-level: 16 separate 1-bit paths**

- Traditional methodology
  - 16 separate 1-bit paths reported
  - \( r_{\text{reg}_0}.iq \rightarrow q_{\text{reg}_0}.iq, r_{\text{reg}_1}.iq \rightarrow q_{\text{reg}_1}.iq \), ...

Crossing count multiplied by number of vector signal bits
Regroup Bit-blasted Crossings

- Proposed methodology
  - 16 separate paths regrouped to 1 CDC path
  - \( r_{reg}[15:0].iq \rightarrow q_{reg}[15:0].iq \)

Crossing count same as RTL for vector signals

Anwesha Choudhury, Mentor Graphics
Proposed Methodology

- **Setup**: Gate Design, Constraints at RTL
- **CDC Analysis**: Run CDC analysis, CDC results, Glitch results
- **Debug**: Fix CDC & glitch issues, CDC sign-off
- **Waivers at RTL**: Transformation engine
- **Automated Setup**: Reduced Noise, Debug Eased, Glitches Detected

3/1/2022
Anwesha Choudhury, Mentor Graphics
Experiment Results

- Average 1.6x improvement in runtime
- Average 10% improvement in memory consumption
- No iterations required for setup
• Upto 80% reduction in noise
  – Bit-merging
  – Constraints & waivers reuse
  – Test logic removed
Summary

• Gate-CDC verification is necessary

• Proposed methodology addresses gate-level CDC verification challenges
  – Automatic RTL constraints transformation engine
  – Auto detection of test logic
  – Detect glitches introduced in synthesis
  – Regroup bit-blasted CDC paths
  – Correlates gate-CDC results with RTL
  – Waiver reuse

• Benefits
  – Seamless setup and reduced noise
  – Accelerates verification closure
Summary

• Gate-CDC verification closure in now possible

• Proposed methodology addresses gate-level CDC verification challenges
  – Automatic RTL constraints transformation engine
  – Auto detection of test logic
  – Detect glitches introduced in synthesis
  – Regroup bit-blasted CDC paths
  – Correlates gate-CDC results with RTL
  – Waiver reuse

• Benefits
  – Seamless setup and reduced noise
  – Accelerates verification closure
Thank You