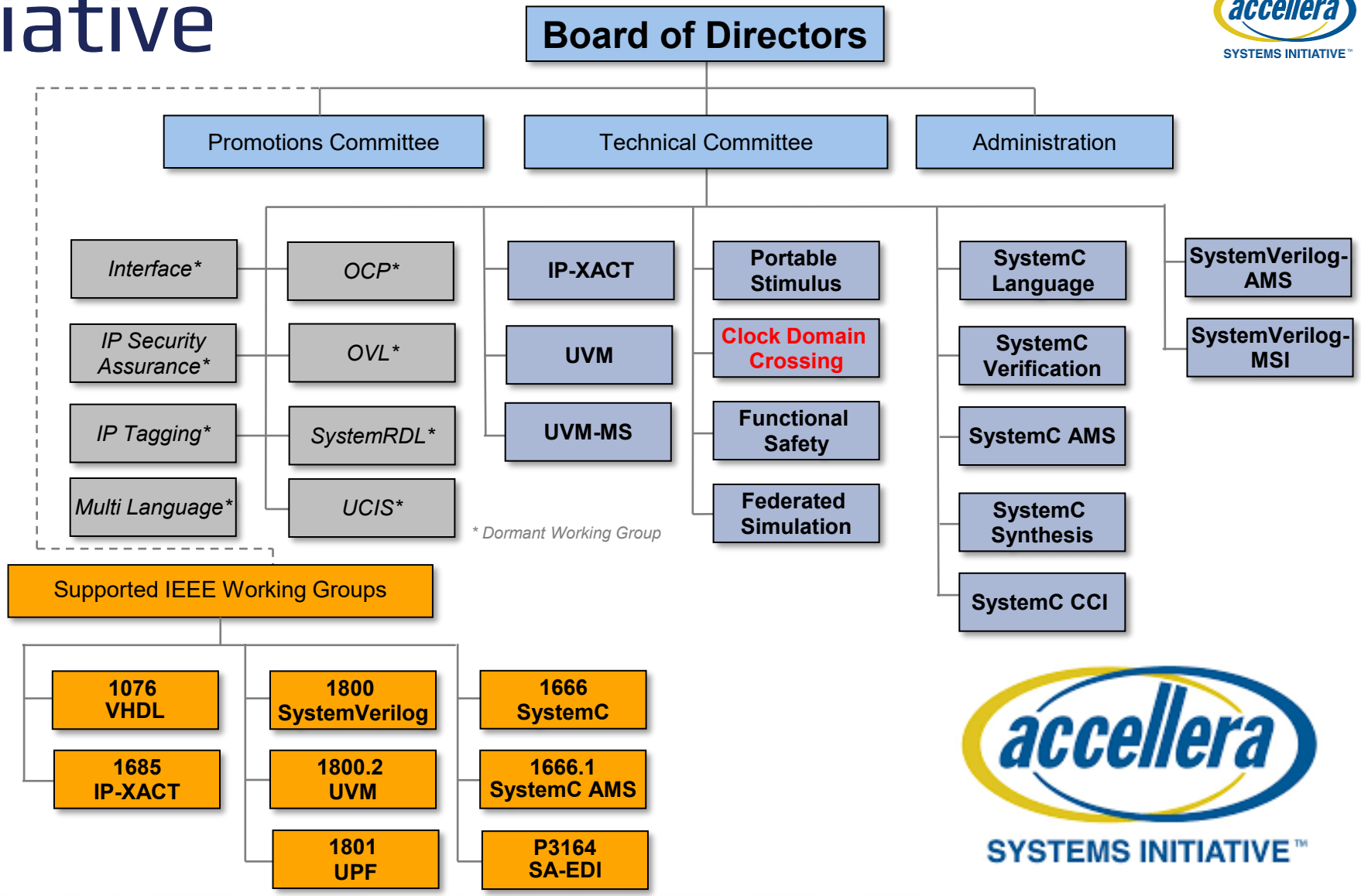
Accellera Initiative

Our Mission

To provide a platform in which the electronics industry can collaborate to innovate and deliver global Electronic Design Automation and IP standards that improve design and verification productivity for today's advanced integrated circuits and embedded systems. In addition, we strive to promote the widespread adoption of these standards.



accellera SYSTEMS INITIATIVE™

Presenter Introductions

Presenters Bio

Don Mills (arm)

- Don is a Principle Engineer at arm SE division supporting system verification
- Don has over 40 years of experience in the industry and has worked on over 35 chips during that time. Many of these chips required functional safety CDC analyses including chips for commercial avionics
- Don earned his Bachelors degree in Electrical Engineering in 1986 from Brigham Young University

Presenters Bio

Bill Gascoyne (Blue Pearl Solutions)

- Currently works at Blue Pearl Solutions in documentation, training, and software QA
- Previously worked as an AE and in technical training at LSI Logic, and in technical training at Magma Design Automation

Presenters Bio

Chetan Choppali Sudarshan (Marvell)

- Chetan is a Senior Staff RTL Design Engineer at Marvell Technology working on connectivity chips.
- He has 9 years of experience in the industry,he previously worked at Intel.
- He holds a Master in Electrical Engineering from Arizona State University.

Presenters Bio

Dammy Olopade (Google)

- Dammy is a Technical Lead at Google, where he leads Quality and Innovation.
- Previously at Intel, he built teams, managed teams and led teams as a technical contributor. His Quality work focused on the prevention of metastability (CDC, RDC, Glitches), innovated new tools/solutions with leading EDA companies for Clock/Reset/Voltage-domain-crossings
- His multi-disciplinary expertise has been essential in tackling complex challenges
- His Innovation work centered on taking key technologies from incubation to products, making sure the culture/environment supported the nurturing of creative ideas, and the corporation appropriately invested in the future, as it focused on executing on the flagship projects
- He is an avid mentor who prioritizes relationships (faith, family, friends)

Presenters Bio

Devendra Kumar Gupta (Agnisys)

- Devendra is a Principal Design/Verification Engineer at Agnisys.
- He has 30+ years of experience in the industry, he previously worked at C-DOT, Fujitsu, Multilink/Vitesse, OnSemi on Telecom ASICs/FPGAs architecture, design and verifications
- He graduated with a B.E.(EcE) degree from Jodhpur University and a M.Tech.(Microwave Electronics) from Delhi University.

Agenda

	Topic	Slide update/create	Presenter	Time
#0	**Workshop introduction**		**Don Mills (arm)**	**5m**
#1	**CDC-RDC**			**85m**
	CDC-RDC Basic Knowledge	Bill Gascoyne (Blue Pearl), Jan Hayek (Bosch Sensortec)	**Bill Gascoyne (Blue Pearl Solutions)**	20m
	Setup Constraints & Verification	Ping Yeung (Nvidia) Jebin Vijai (Intel)	**Bill Gascoyne (Blue Pearl Solutions)**	10m
	Structural CDC/RDC	Chetan Choppali Sudarshan (Marvell) Suman Chalana (Qualcomm)	**Chetan Sudarshan (Marvell)**	20m
	CDC Assertion-Based Verification	Kranthi Pamarthi (Renesas Electronics), Don Mills (arm)	**Don Mills (arm)**	15m
	CDC-RDC Hierarchical Flow	Jean-Christophe Brignone, Ashish Soni (STMicroelectronics)	**Don Mills (arm)**	5m
	Q & A		**Don Mills (arm)**	15m
	BREAK TIME			**30m**
#2	**Accellera CDC WG Reports**			**90m**
	Standard	Dammy Olopade (Google), Jebin Vijai (Intel)	**Dammy Olopade (Google)**	10m
	Output	Joachim Voges (Infineon), Nomita Goswami (Agnisys)	**Dammy Olopade (Google)**	15m
	Assertion	Kranthi Pamarthi (Renesas Electronics), Don Mills (ARM)	**Don Mills (arm)**	10m
	Testing	Farhad Ahmed (Siemens EDA), Suman Chalana (Qualcomm)	**Devendra Gupta (Agnisys)**	10m
	Format	Nomita Goswami (Agnisys)	**Devendra Gupta (Agnisys)**	10m
	Training	Jean-Christophe Brignone, Diana Kalel, Ashish Soni (ST)	**Bill Gascoyne (Blue Pearl Solutions)**	10m
	Call For Contribution	Jean-Christophe Brignone (ST), Anupam Bakshi (Agnisys)	**Don Mills (arm)**	5m
	Q & A		**Don Mills (arm)**	20m

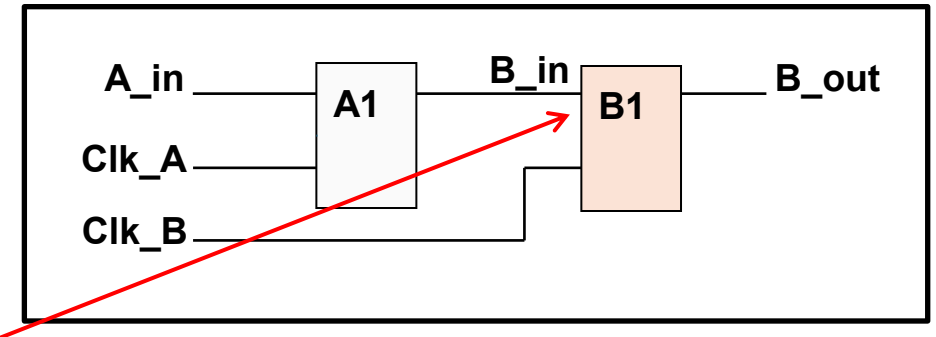
1.1 CDC-RDC Basic Knowledge:

CDC-RDC Basic Knowledge:

- Synchronous vs. asynchronous clocks
- Problems related to Clock Domain Crossing (CDC)
- CDC Synchronization
- Problems related to Reset Domain Crossing (RDC)
- RDC Synchronization

What is Clock Domain Crossing?

- What is a Clock Domain?
  - flip-flops with same clock (clock tree) or derived clocks sharing the same source
- Clock Domain Crossing
  - Data from one clock domain is captured (sampled) in another clock domain



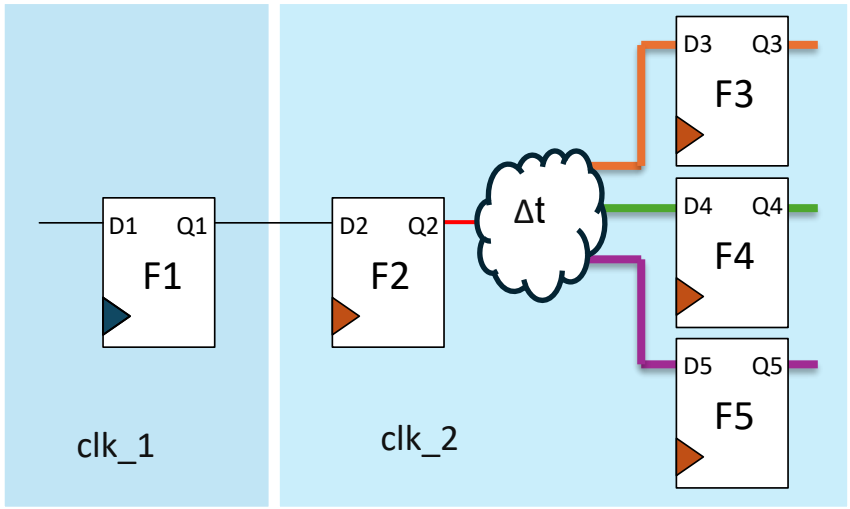
Coin Toss Analogy

- Think of a setup/hold violation result as the toss of a coin
  - Heads or Tails, but also very rarely it might just stay on its edge (metastability) before falling one way or the other
- Fixing metastability and fixing data coherency are independent
- For one bit, fixing metastability is enough
  - Coherency doesn’t matter, since either heads or tails is fine
- For multiple bits, must fix metastability AND data coherency
  - Requires all heads or all tails from multiple coins
  - A losing bet!

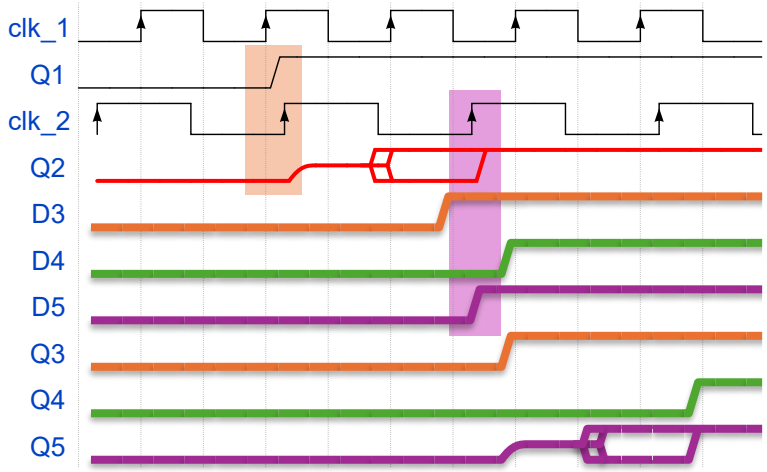
Why do CDCs need fixing?

- Metastability
  - Timing violations on registers resulting in an indeterminate state lasting long enough to violate timing assumptions of the following circuits

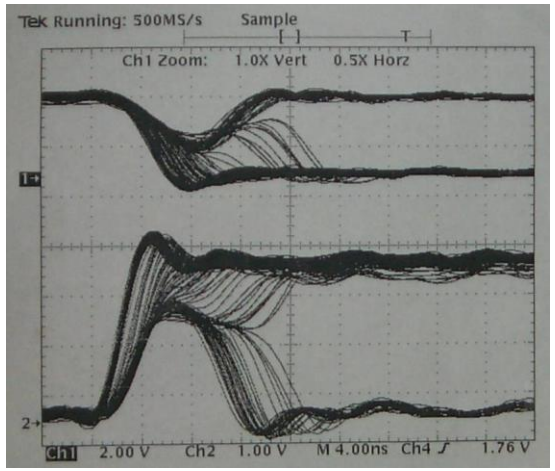
    … the coin decided too late / has not decided yet



(a) structure



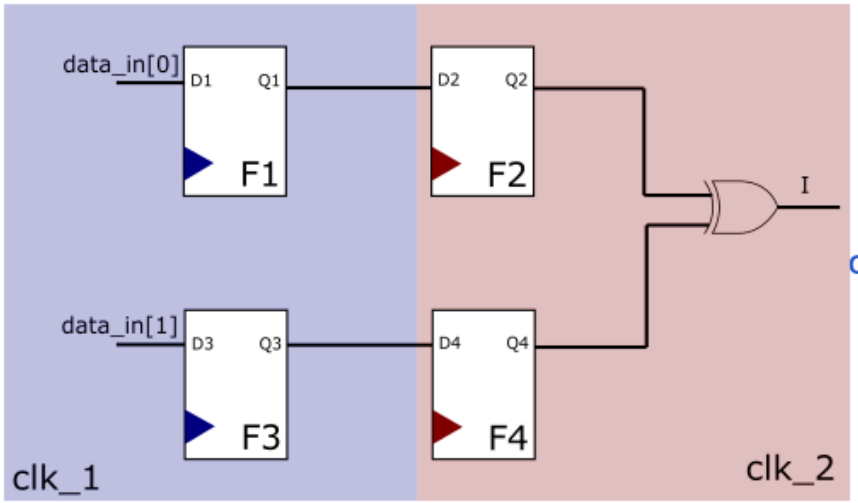
(b) waveform



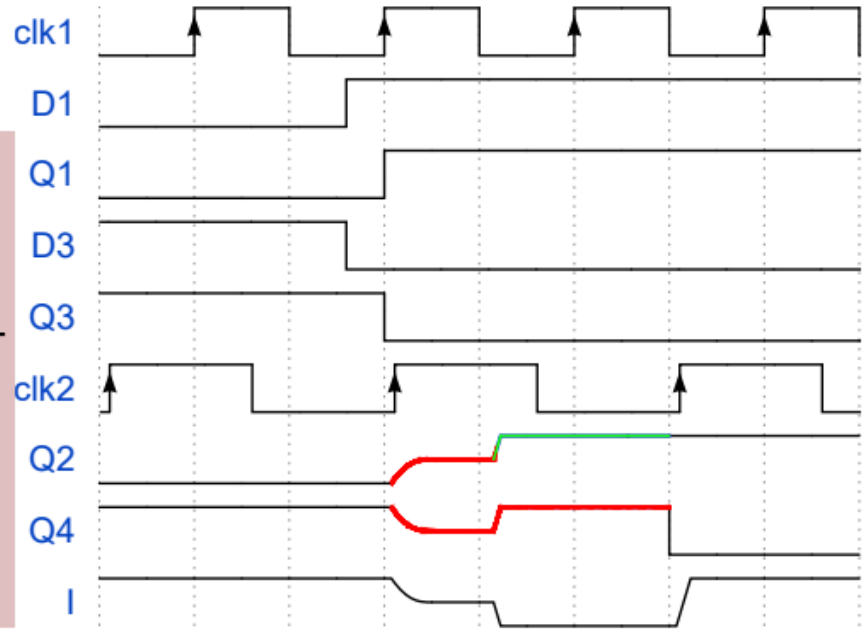
(c) Physical representation

Why do CDCs need fixing?

- Loss of Data Coherency (Reconvergence)
  - The indeterminate state settles to a random value



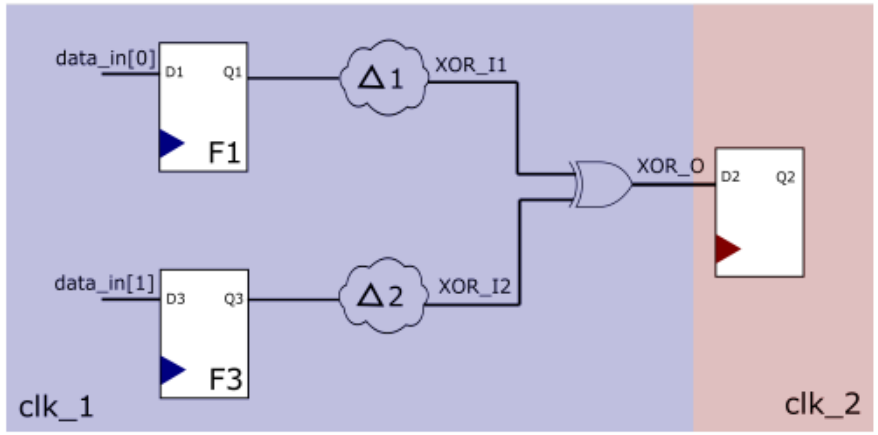
(a) Structure



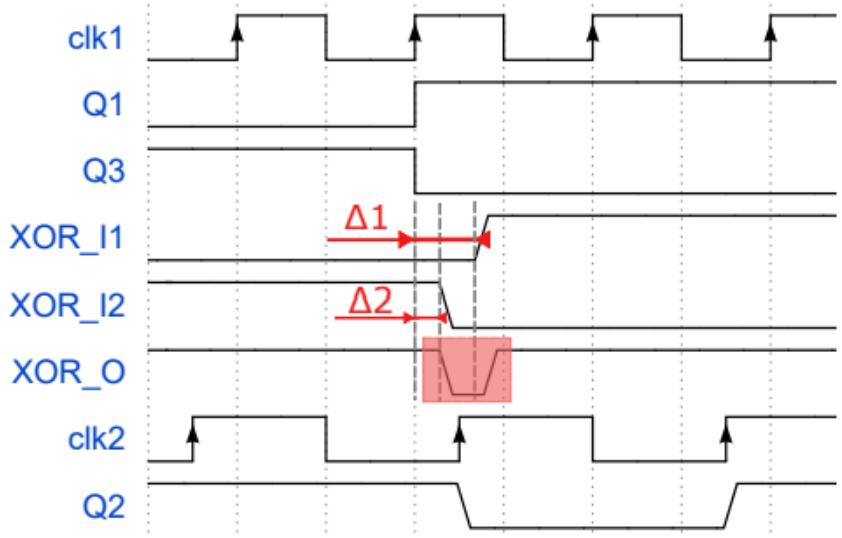
(b) Waveform

Why do CDCs need fixing?

- Glitches
  - Multiple synchronized paths reconverge to cause unexpected momentary transitions



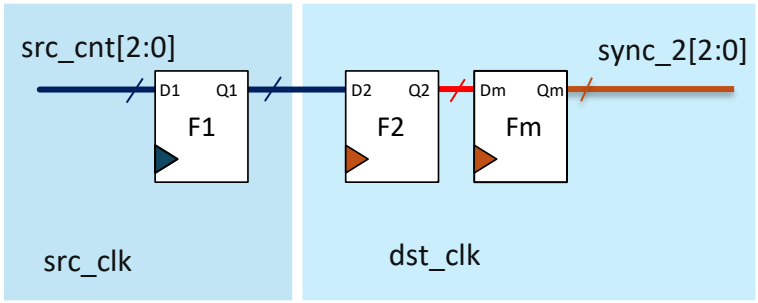
(a) Structure

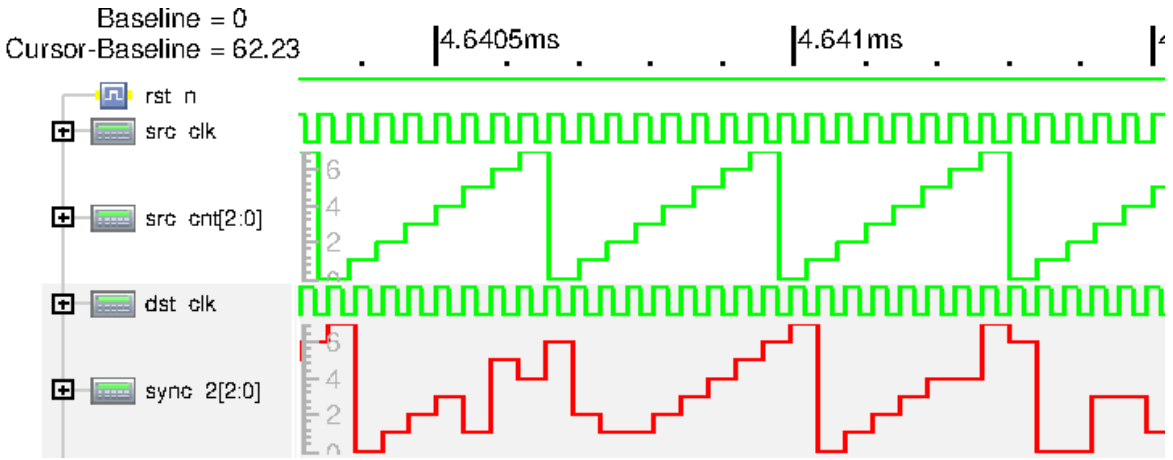


(b) Waveform

Synchronization

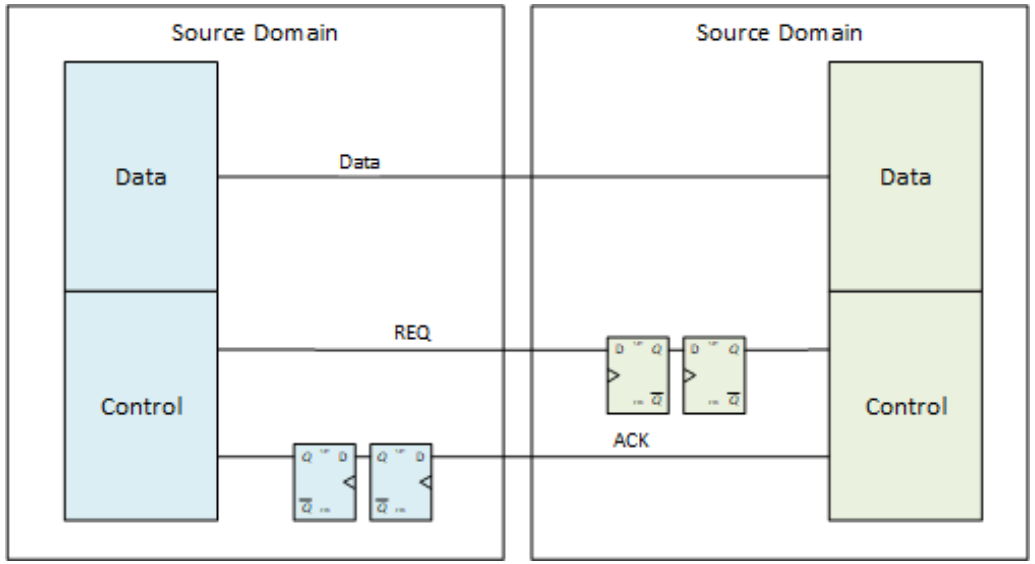
- With multiple bits, metastability is still addressed but data coherency is a problem!
  - If multiple bits change on the same cycle, the result of each bit is random
  - This synchronization works only if the data is “gray” (only one bit changes)



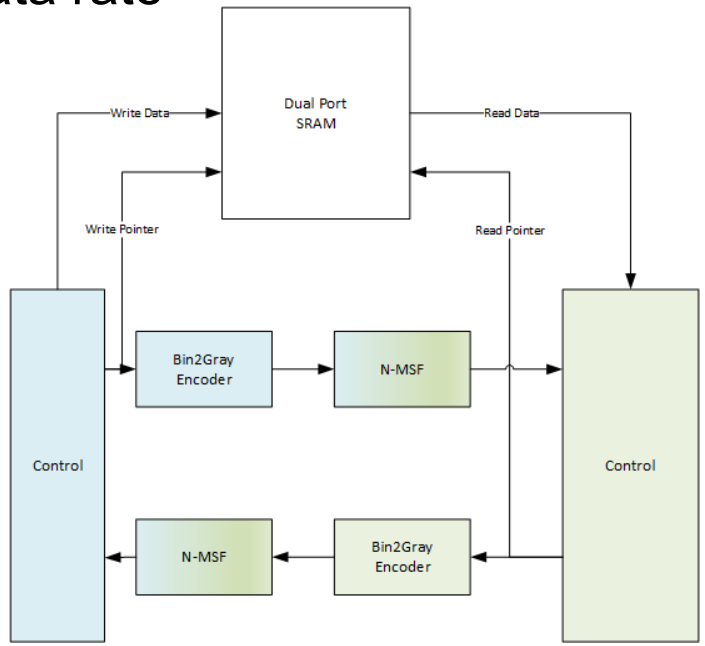


More Complex Synchronization

- Handshake Protocol

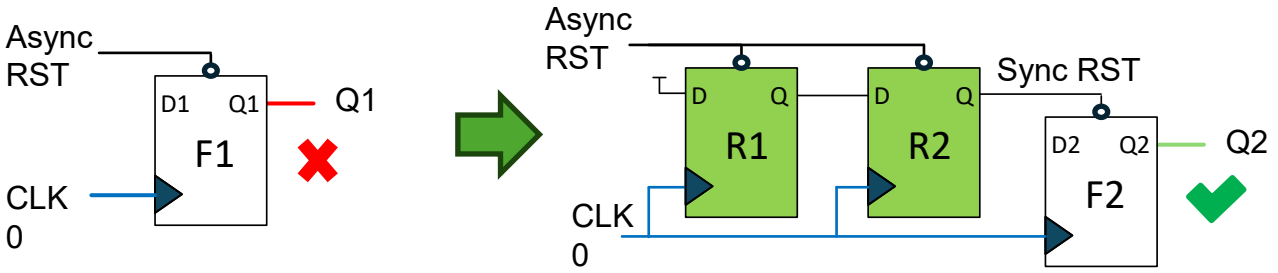


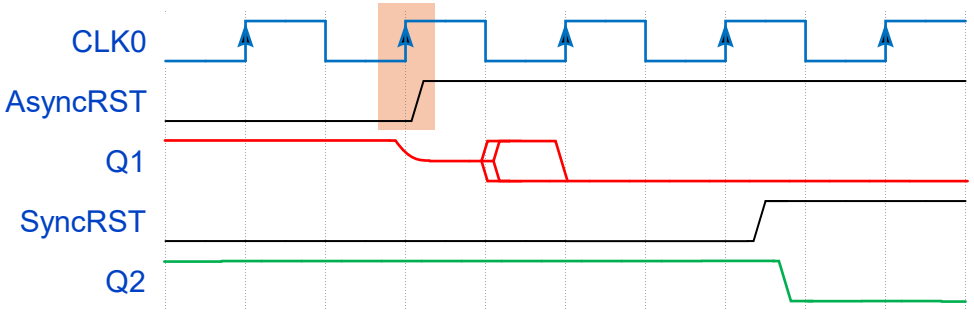
- FIFO
  - Increased bandwidth
  - Throttling
    - Handles intermittent peaks of incoming data rate



Asynchronous Reset Release

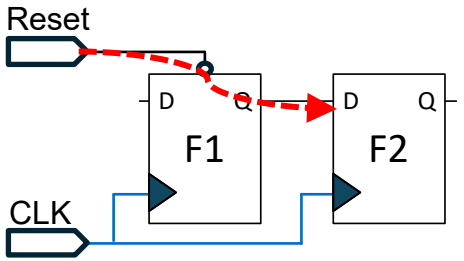
- In addition to setup- and hold-, DFF models also have **recovery-time**
  - Time between asynchronous Set/Reset release and clock when data and output are different
  - Violating recovery time is no different than violating setup/hold
- Possible to synchronize asynchronous reset on release edge only
  - Static analysis is sufficient to make this determination

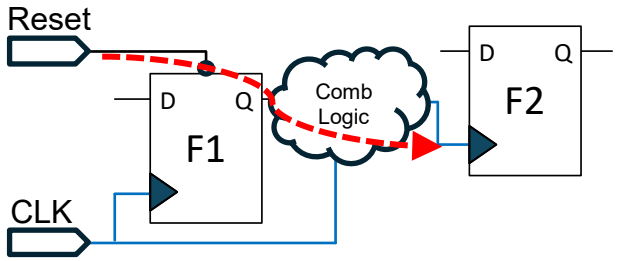




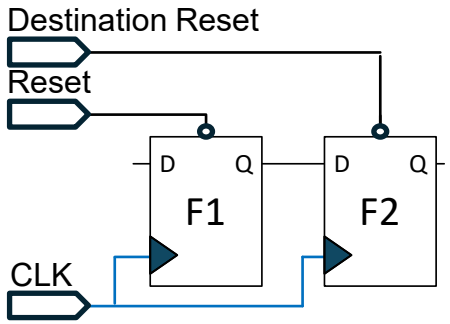
RDC problem: Async Reset Applied

- Paths passing from CLR to Q are usually not timing closed

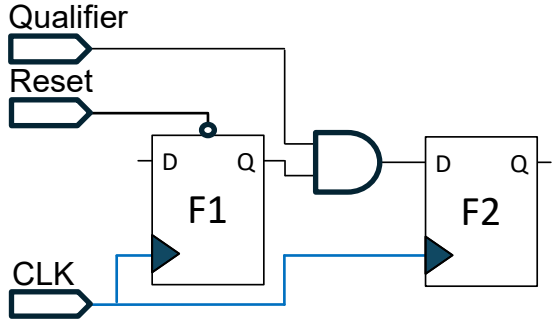




- Using reset applied ordering



- Using a CDC Control
  CDC Control must be synchronous with target domain



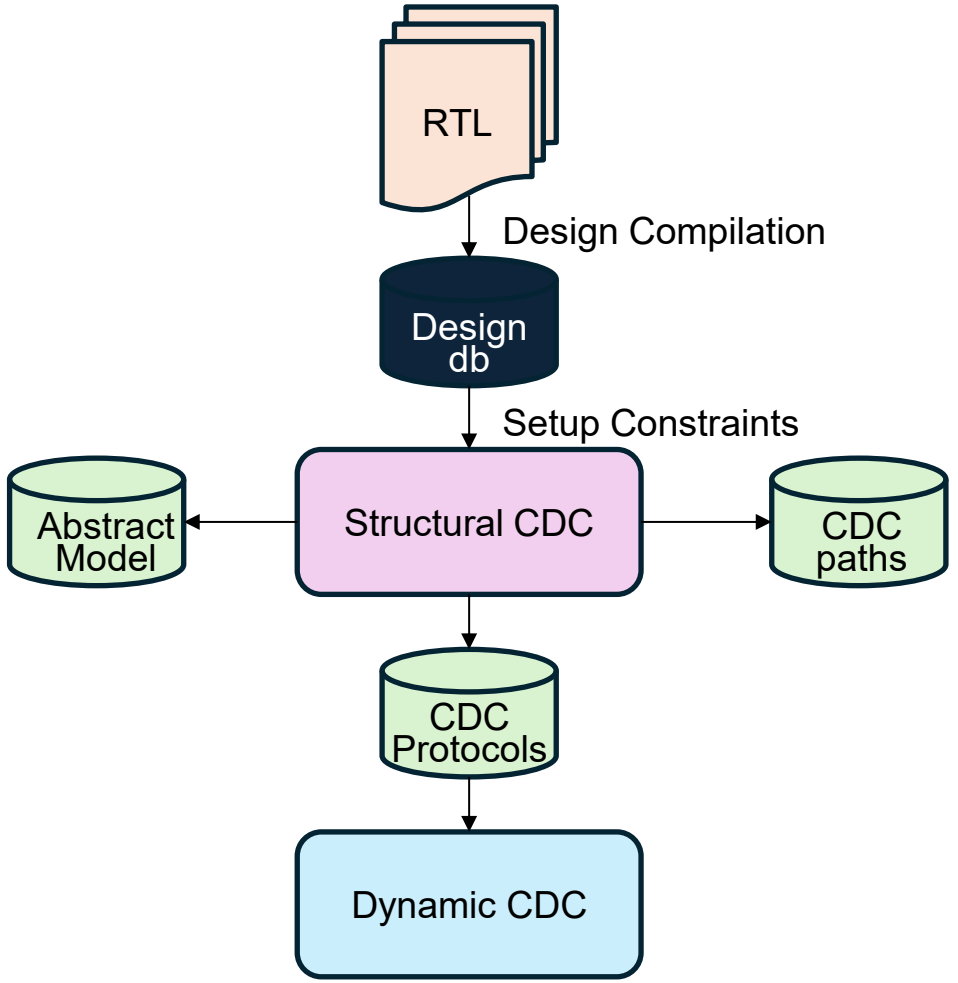
1.2 CDC Setup & Constraints

CDC Setup & Constraints

- CDC Verification flow
- Setup constraints
- Challenges

CDC Verification flow

- Design Compilation
  - Parameters, defines
  - SV packages, SV configuration, SV interfaces
- Setup Constraints
  - Clock, reset, and IO signals
  - Configuration: stable, constant inputs
- Structural CDC Check
  - CDC schemes validation and debugging
- Abstract Model Generation
- Dynamic/Formal CDC Verification
  - CDC constraints and protocols



Setup Constraints

- The set of constraints used to guide CDC verification

- Clocks
- Resets
- Configuration signals
- Black boxes
- Primary inputs/outputs

Don’t rely blindly on timing constraints

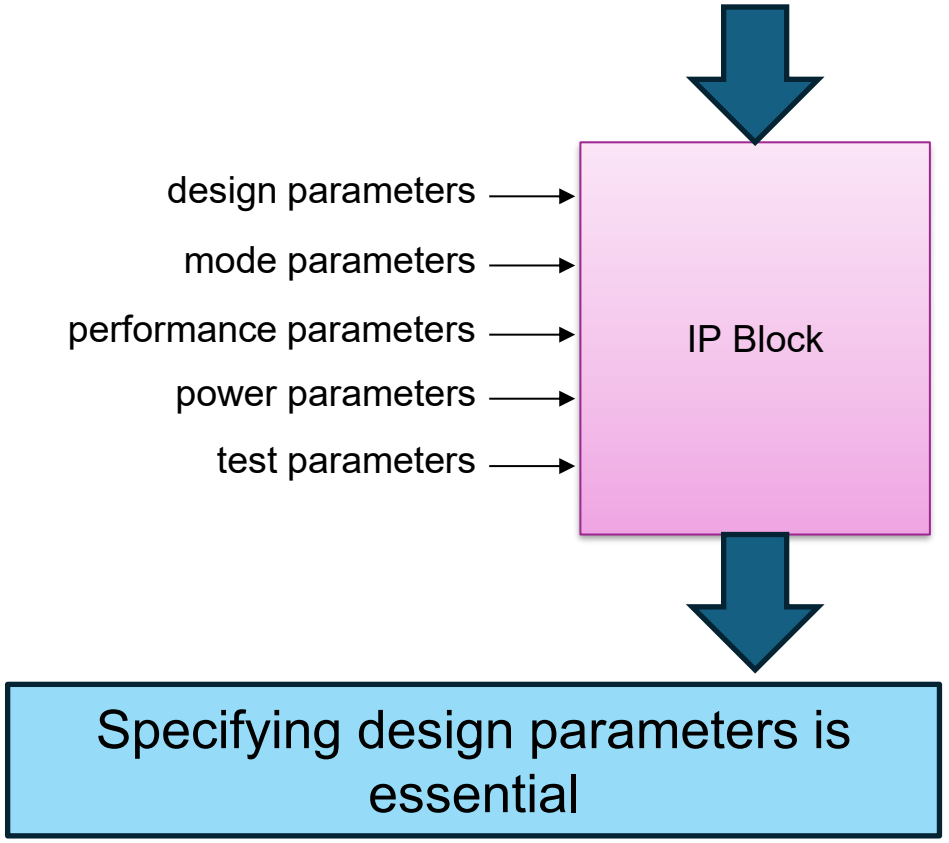
Reusing timing constraints is risky

- Pseudo-static signals
- Exclusive signals
- Gray coded buses
- Custom synchronizers
- False path

Clock groups for timing analysis ≠ Clock groups for CDC analysis
Signal paths waived for timing analysis ≠ Signal paths waived for CDC analysis

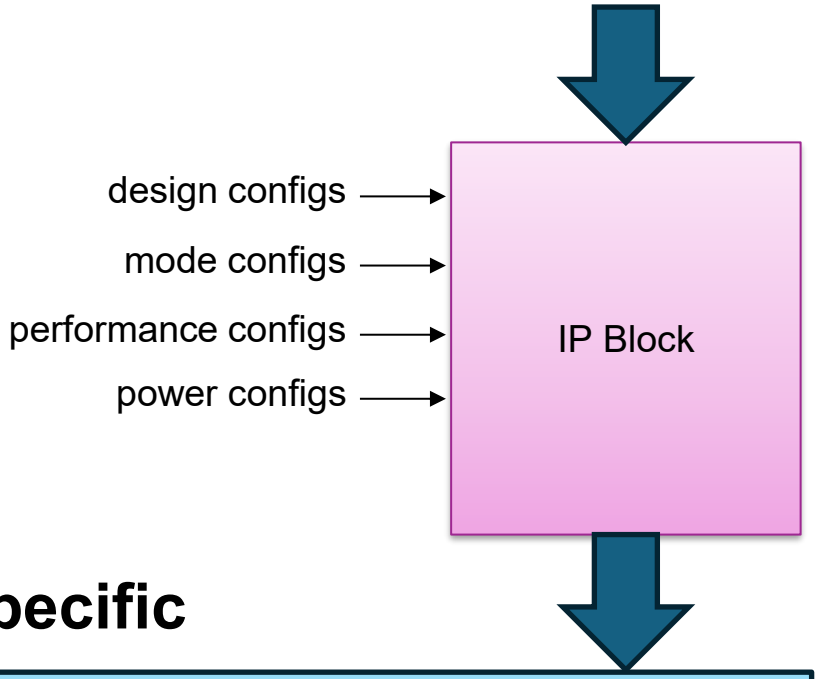
Challenge #1: Design Parameters

- Blocks/IPs have many parameters
  - Some used for design, performance, optimization
  - Some used for integration, mode
  - Some used for DFT, DFP, DFM, etc
- Most parameters will affect CDC results
- Some parameters may not affect CDC results
  - Data_width, Addr_width
  - RAM_size
- **The Abstract Model may become parameter-specific!**



Challenge #2: Configuration Signals

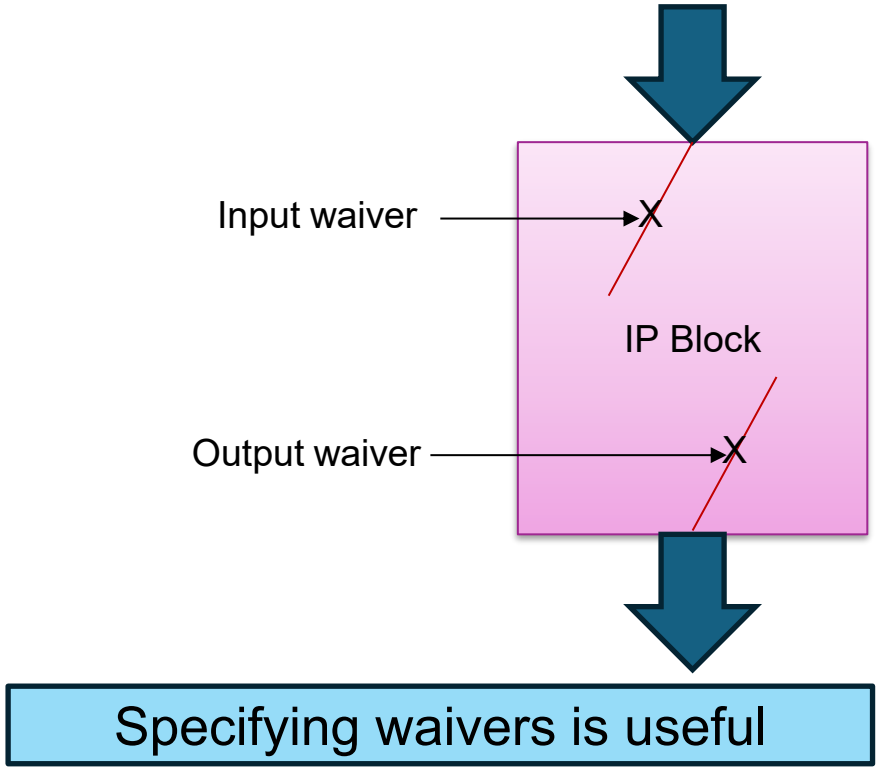
- Blocks/IPs have many configuration signals
  - Some used for design, performance, optimization
  - Some used for integration, mode
  - Some used for DFT, DFP, DFM, etc
- Most configuration signals will affect CDC results
  - Clock select, gating signals
- **The Abstract Model will become configuration-specific**



Specifying constraints on configuration signals is essential

Challenge #3: CDC Waivers

- Blocks/IPs have many CDC violations
  - Some are on the input signals
  - Some are on the output signals
- Some of the input violations can be waived
  - Pseudo-static input signals
  - Output signals
- Some of the input violations must not be waived
- **The Abstract Model will become waiver-specific**

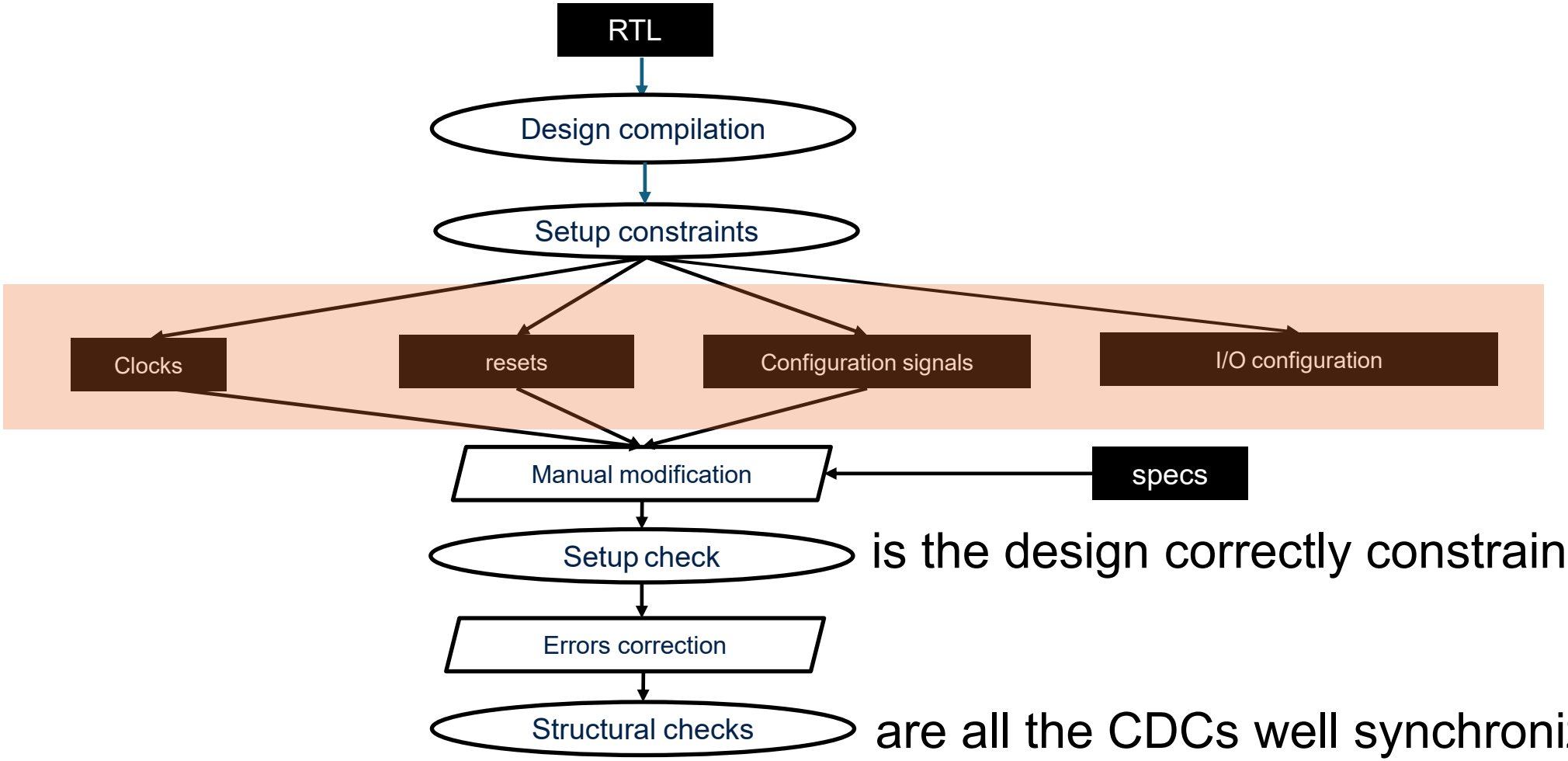


1.3 Structural CDC/RDC

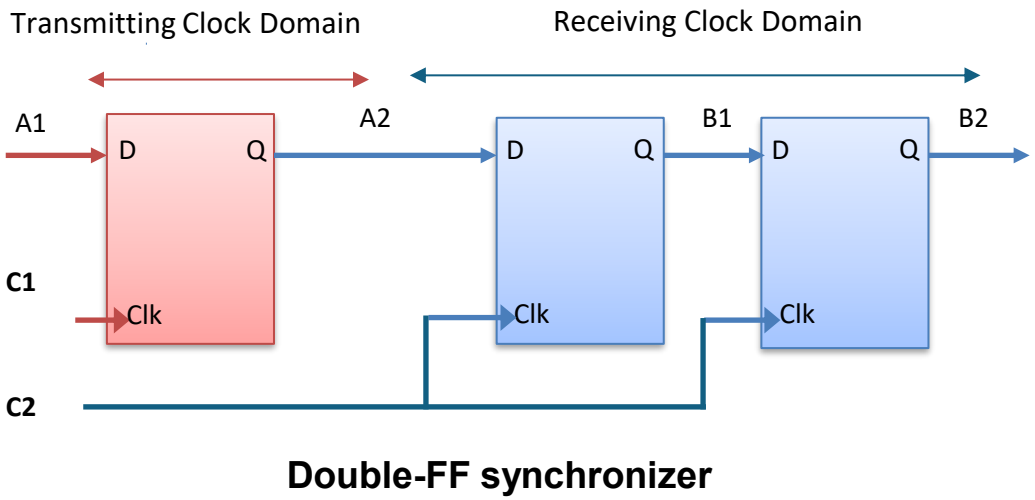
Structural CDC/RDC

- Structural CDC
- User defined synchronization modules
- CDC constraints
- Reset Domain Crossings

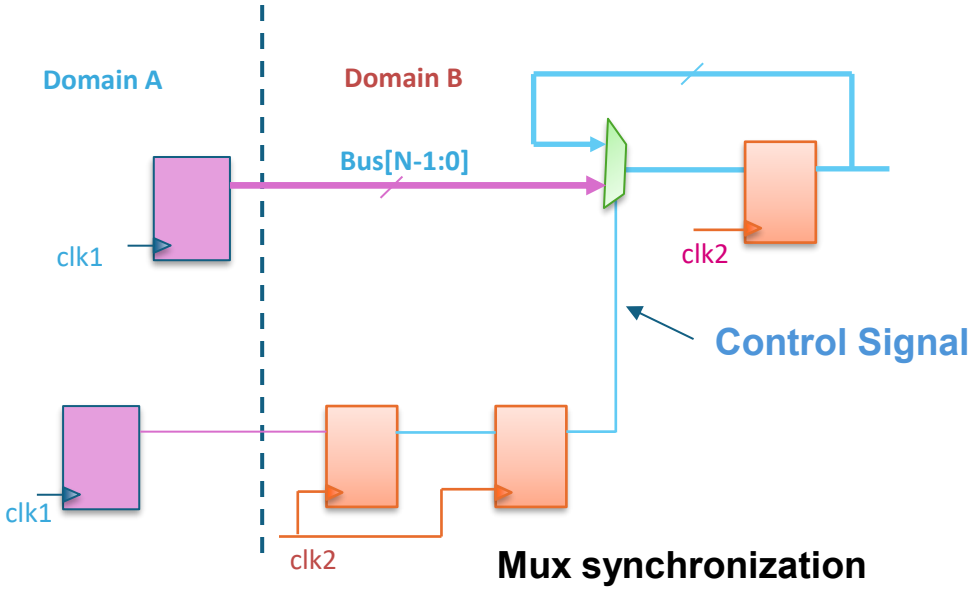
Structural CDC Analysis Process



Structural CDC - Commonly Used Synchronization Schemes (1)

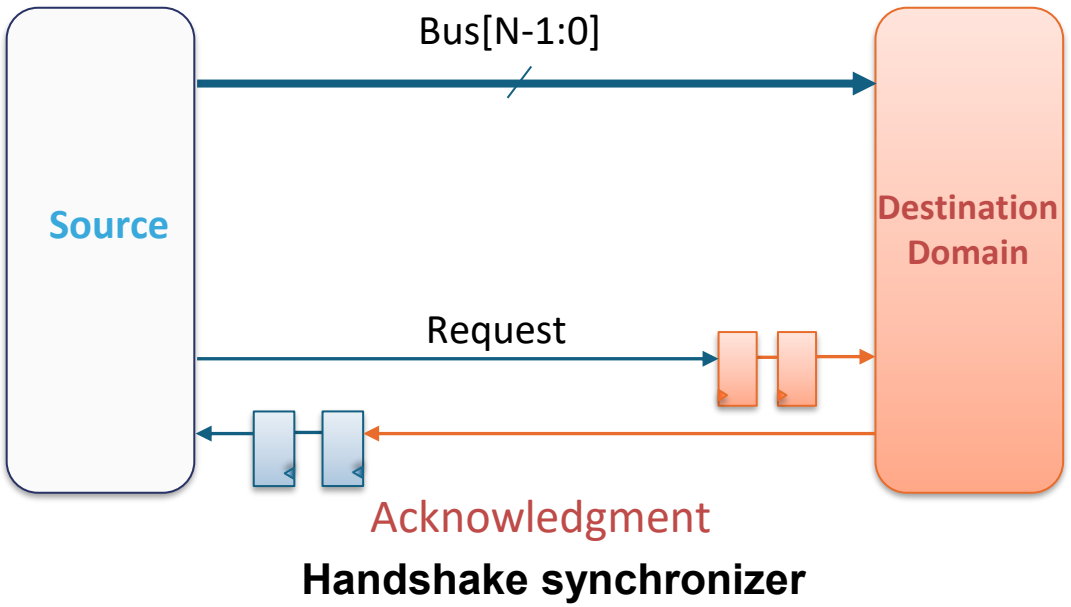


Double-FF synchronizer

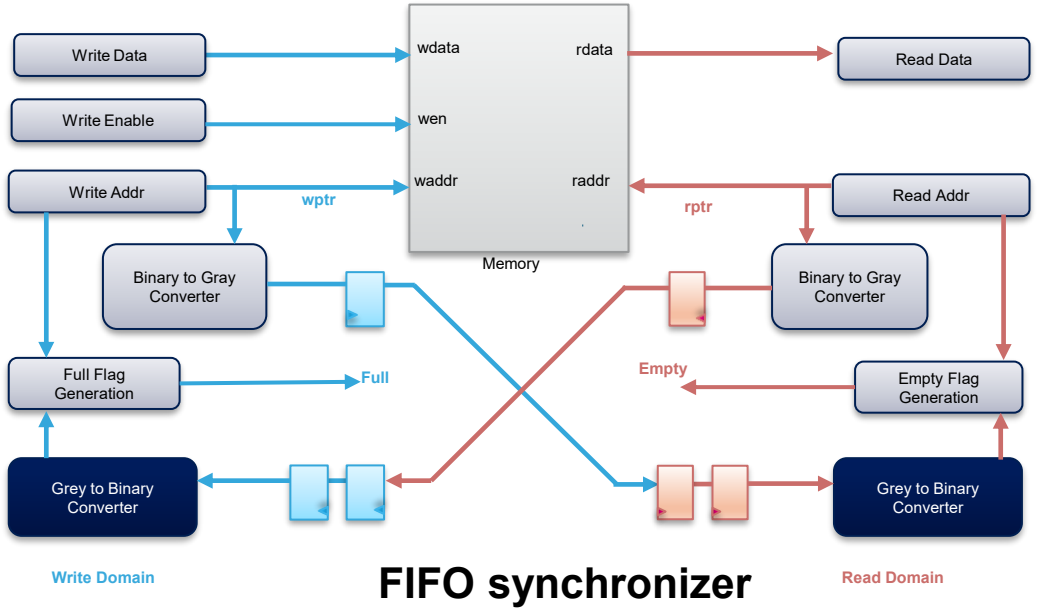


Mux synchronization

Structural CDC - Commonly Used Synchronization Schemes (2)



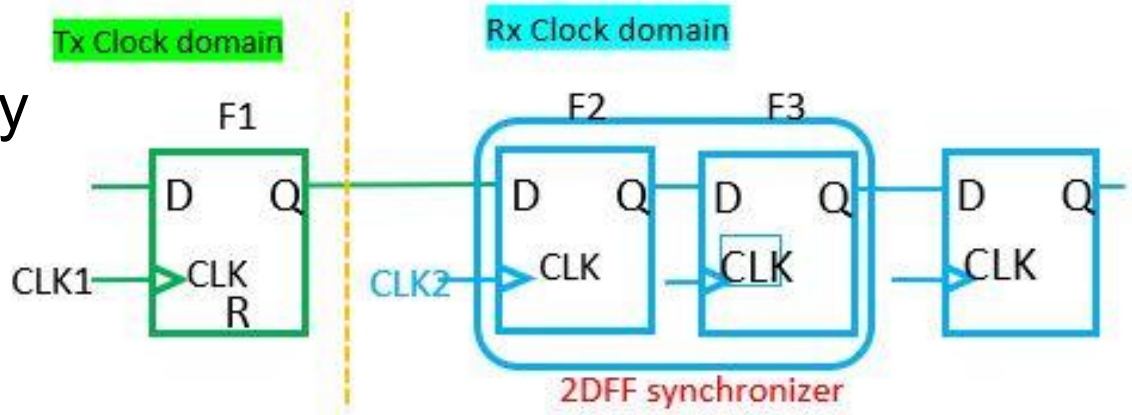
Handshake synchronizer



FIFO synchronizer

Structural CDC – User defined sync modules

- Double FF Synchronizer
  - Most design houses prefer to use their own CDC components
    - Disable automatic detection of the specific synchronizer type that you don't want the tool to recognize automatically
    - Declare your own scheme as user-defined synchronizer (before scheme detection)
  - Example: Use my own 2DFF only
    - Disable auto-detection of 2DFF
    - Declare your own module as 2DFF



Structural CDC

Design Constraints

- Purpose
  - Define signal behavior that can help to **reduce CDC analysis noise**
    - Exclude certain paths which may not have any standard synchronizer but safe for CDC
    - Helps to speed up CDC analysis
- Various Signal Configurations possible for structural CDC Analysis
  - Constants
  - Static signals
  - Mutually exclusive / Gray code
  - Externally synchronized
  - CDC False paths
    - *Not recommended (avoid using it to mask real CDCs)*

Structural CDC

Design Constraints

- CDC Constraints – Constant Declaration
  - It can be applied on a port or on an internal signal
  - A constant signal does not change in a given mode and hence does not cause a CDC issue

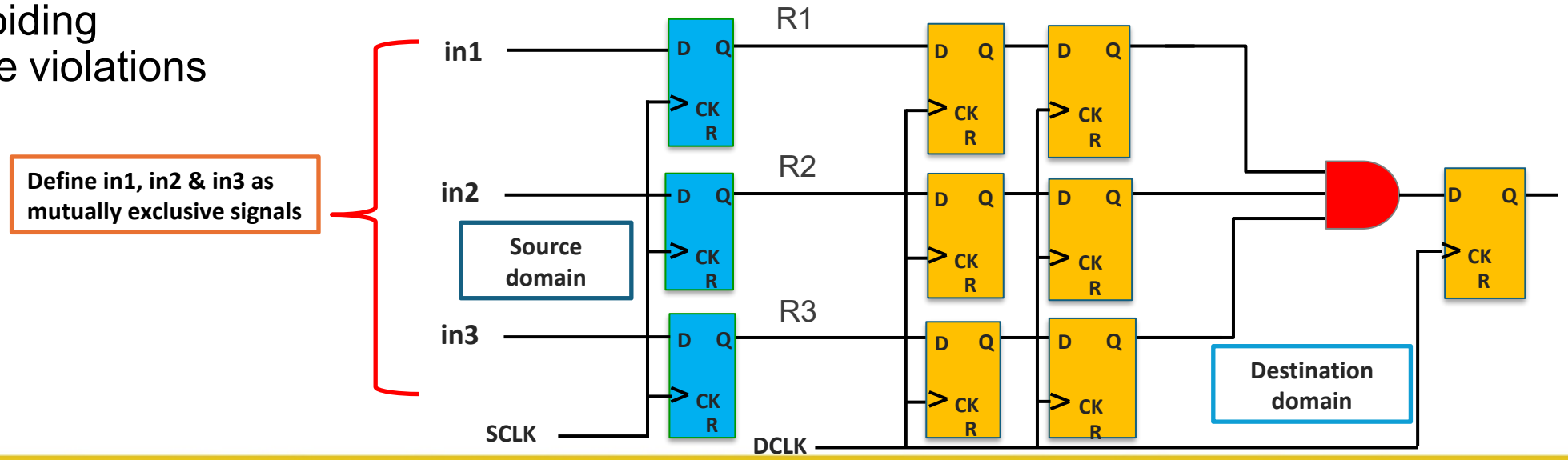
Structural CDC

Design Constraints

- CDC Constraints – Static Declaration
  - Any signal that does not change while the destination is active
  - Same as quasi-static or pseudo-static
  - A static signal does not cause CDC issues because:
    - The receiver clock is not active
    - The receiver is under reset
    - The receiver has an enable for receiving signal

Structural CDC / Design Constraints

- CDC Constraints – Gray Coded Declaration
  - A bus can be specified as gray coded – Only one bit can toggle at a time
- CDC Constraints – Mutually Exclusive Toggle Declaration
  - A set of independent signals that can toggle only one at a time can be defined as mutually exclusive toggle
    - Helps in avoiding convergence violations

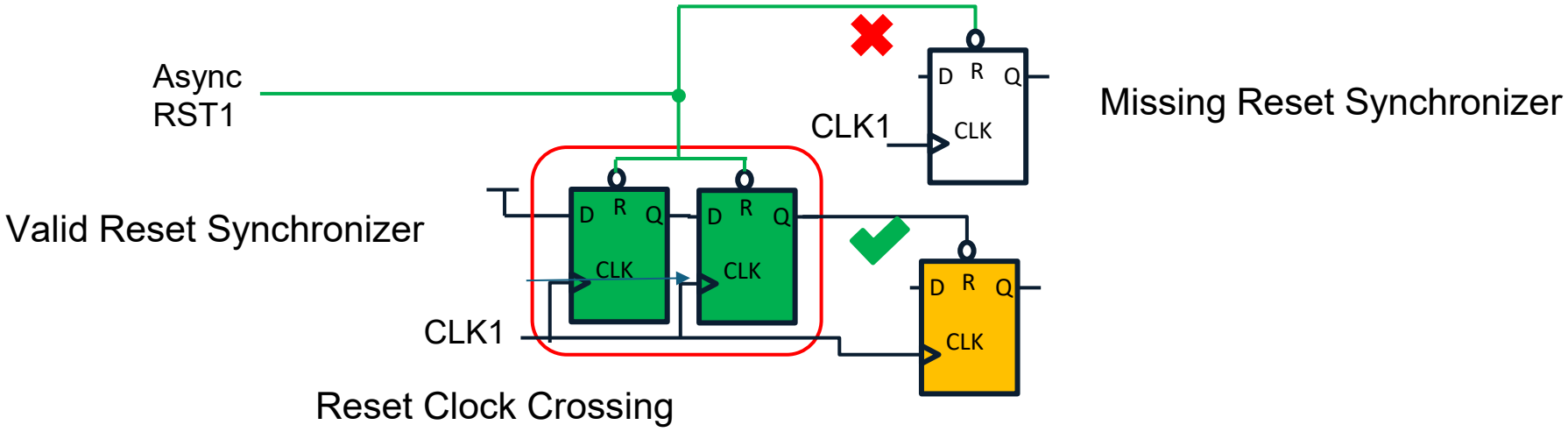


Structural CDC

Design Constraints

- CDC Constraints - Externally Synchronized
  - A block level input/output port can be declared as *externally synchronized*
    - Represents the output of a control synchronizer (2DFF/Edge/Pulse)
    - Can be used as the control path for complex synchronizers (MUX Synchronizer, Glitch Protector)
    - Helps in auto-detection of the above composite synchronization scheme types
- CDC Constraints - CDC False Path Declaration
  - CDC Checks can be disabled on certain paths by user-defined constraints
  - User can set a constraint to let the tool automatically identify a functionally false path and hence reports the path as a safe path

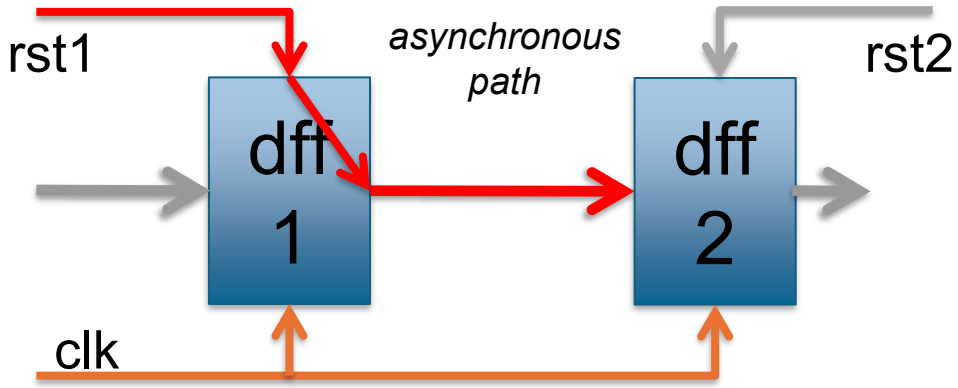
Asynchronous Reset De-assertion



- Reset signal crossing from one clock domain to another
  - The asynchronous de-assertion of the reset signal at the destination flop can cause the signal to become metastable
  - Reset signals are required to be synchronized to destination domains for synchronous de-assertions

Reset-Domain Crossing

- Asynchronous reset domains causes meta-stability
  - Contain registers whose resets are asserted asynchronously
  - Originate in one asynchronous reset domain
  - Sampled by register(s) in a different reset domain
  - Reset ordering of different resets in the design

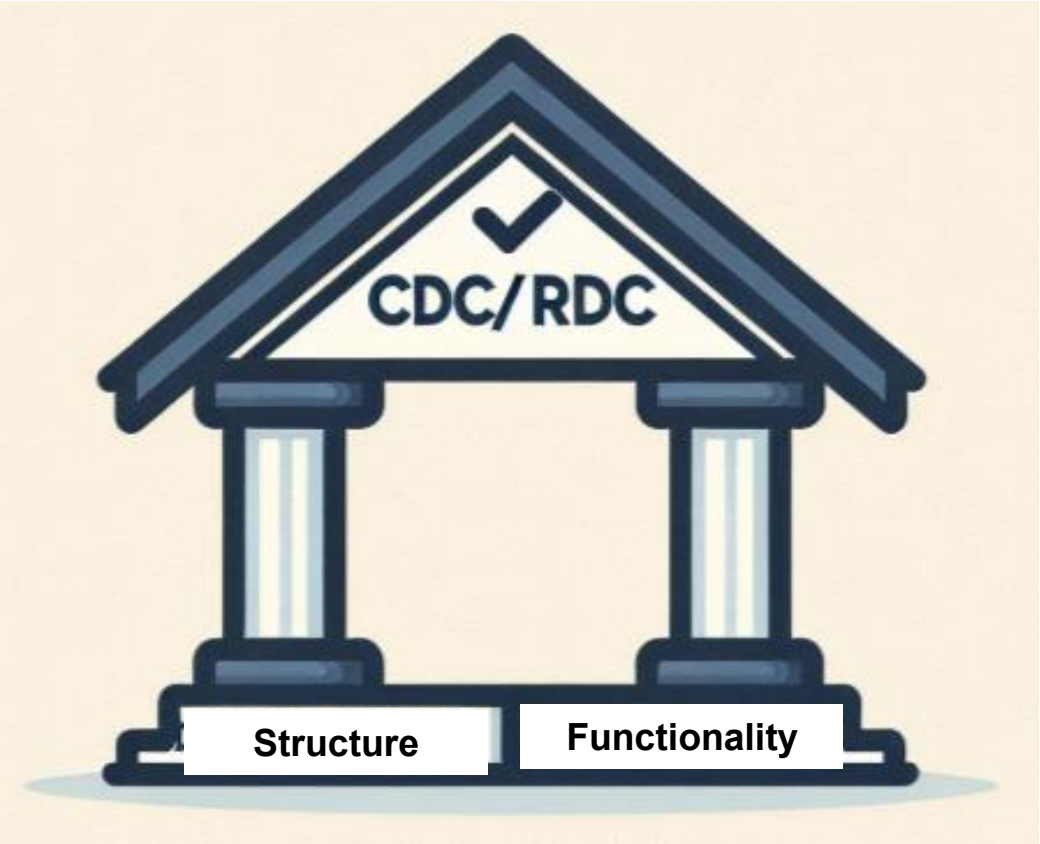


1.4 CDC Assertions

CDC Assertions

- Assertion Based Verification
- Overcoming Limitations

Structural Verification vs Assertion



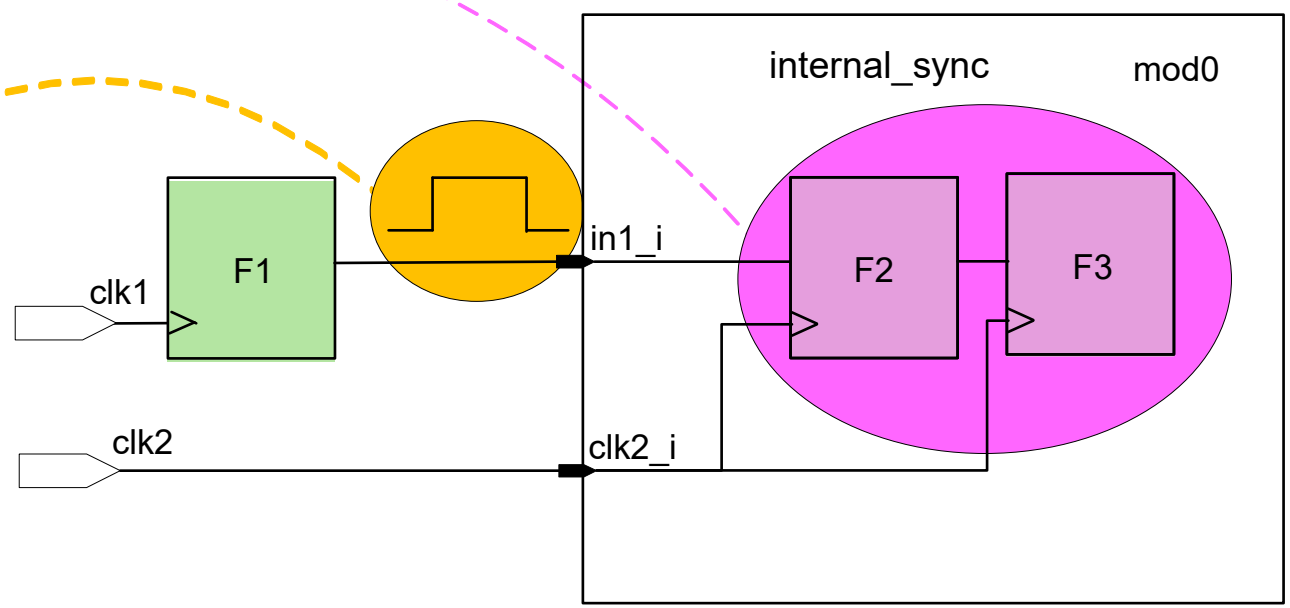
- ***Structural CDC/RDC Verification*** analyze the design structure. It doesn't analyze the design functionality. As long as the structure is valid, the verification pass.
- ***Assertions*** covers design functionality. They can be used in Dynamic Simulation and/or Formal Verification.

Valid CDC/RDC =
Correct Structure +
Correct Functionality

A Simple CDC Example

Correct structure:
There is a 2-DFF synchronizer between clock domains.

Correct functionality:
Pulse width of in1_i must be wide enough to be sampled by clk2_i.



Covered by Assertion

A Simple CDC Example: Pulse width check

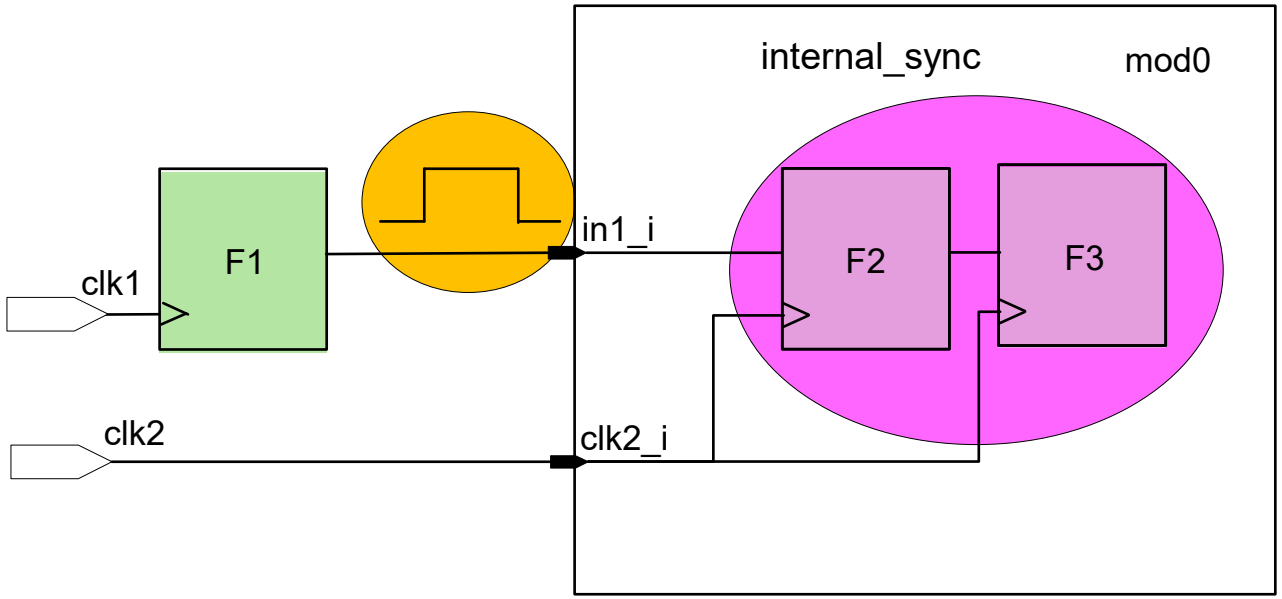
mod0's collateral:

```
cdc_set_port in1_i                    \
  -direction input                    \
  -type data                          \
  -logic internal_sync                \
  -associated_to_clocks clk2_i
```

Assertion property generated from the information

Example:

```
property pulse_width;
  @(posedge clk2_i)
    $changed(in1_i) |=> $stable(in1_i)[*2];
endproperty
```

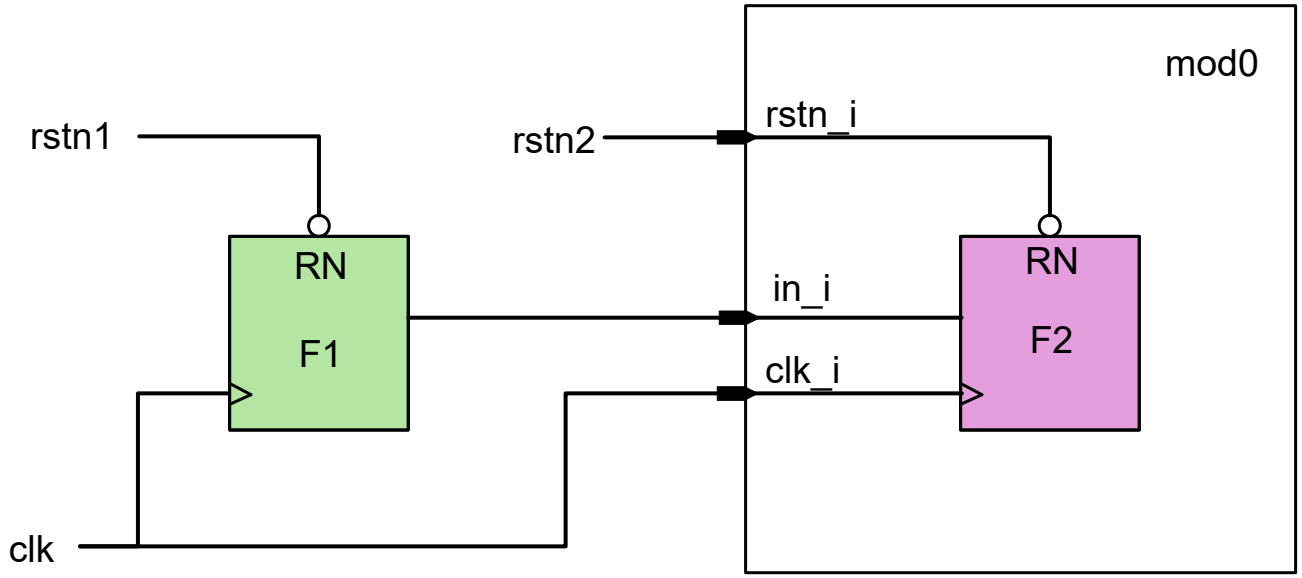


NOTE: The SVA here is just an example.
It may vary in different EDA tools.

A Simple RDC Example

Structure:
mod0 doesn't provide any RDC handling.

Correct Functionality:
rstn2 must be already 0 or going to 0 when rstn1 is going to 0.



A Simple RDC Example: Reset sequence

mod0's collateral:

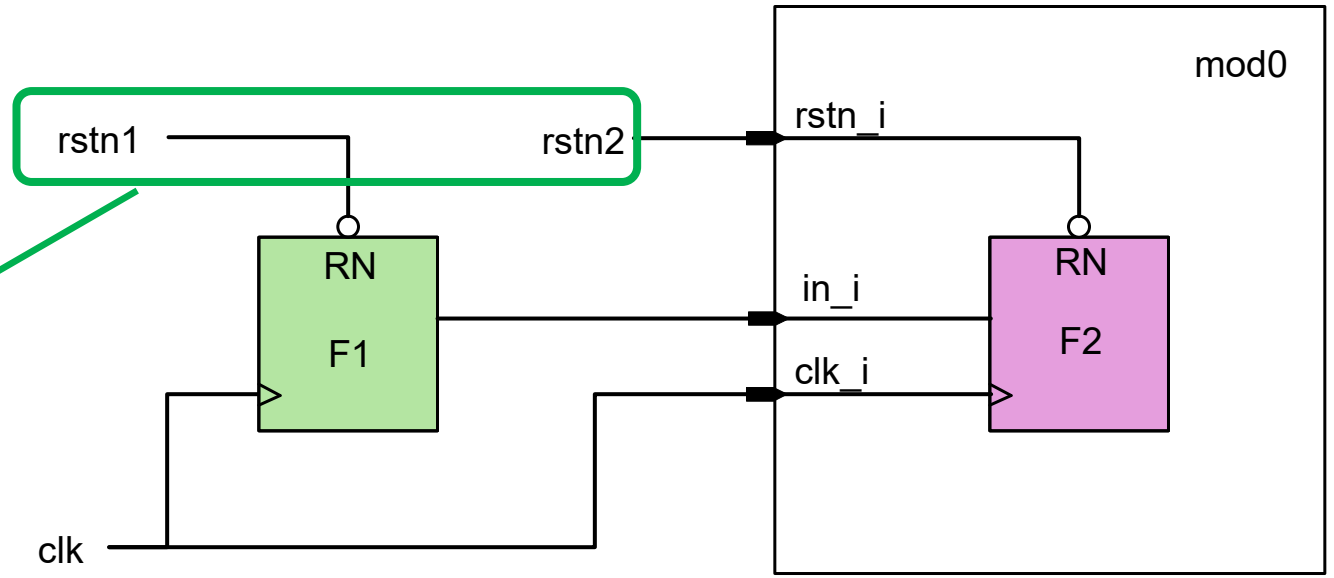
```
cdc_set_port in_i                    \
  -direction input                   \
  -type data                         \
  -associated to clocks clk_i        \
-associated_to_reset rstn_i
```

Assertion property generated from the information

Example:

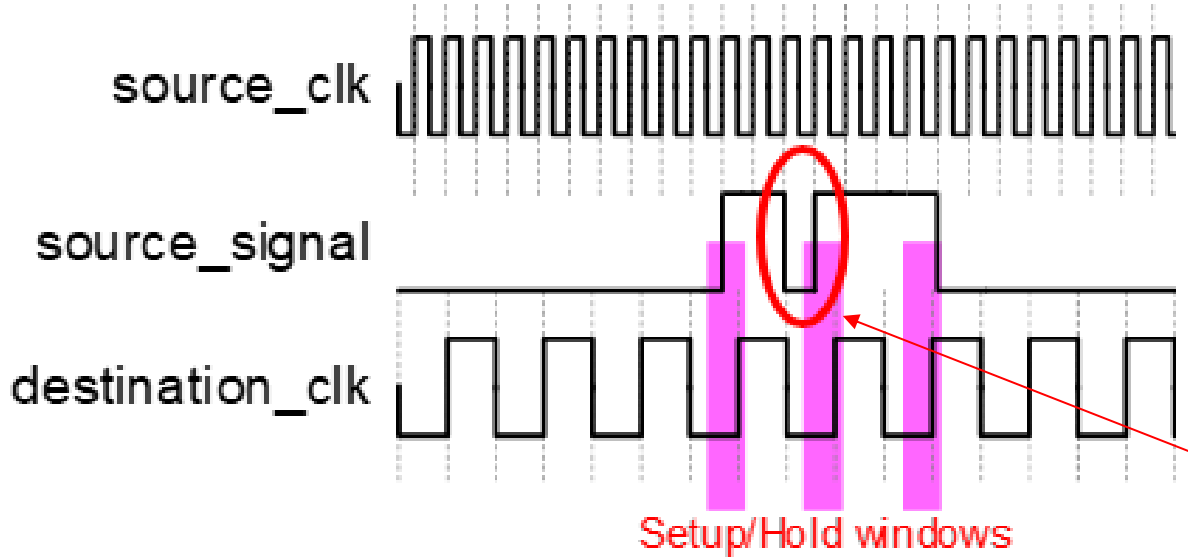
```
property reset_order;
  @(negedge rstn1)
    (rstn2 === 1'b0);
endproperty
```

NOTE: The SVA here is just an example.
It may vary in different EDA tools.




```
property reset_order_sync;
 @(posedge clk)
    $fell(rstn1) |-> (rstn2 === 1'b0);
endproperty
```

Assertion Clocking Concern



- **Pulse width assertion pass**
- **But silicon fail as it is a false positive in RTL for this example**

Solution:

- **Use fastest clock in the system shown on next slide**

Do not use this property as this misses the glitch on source_signal

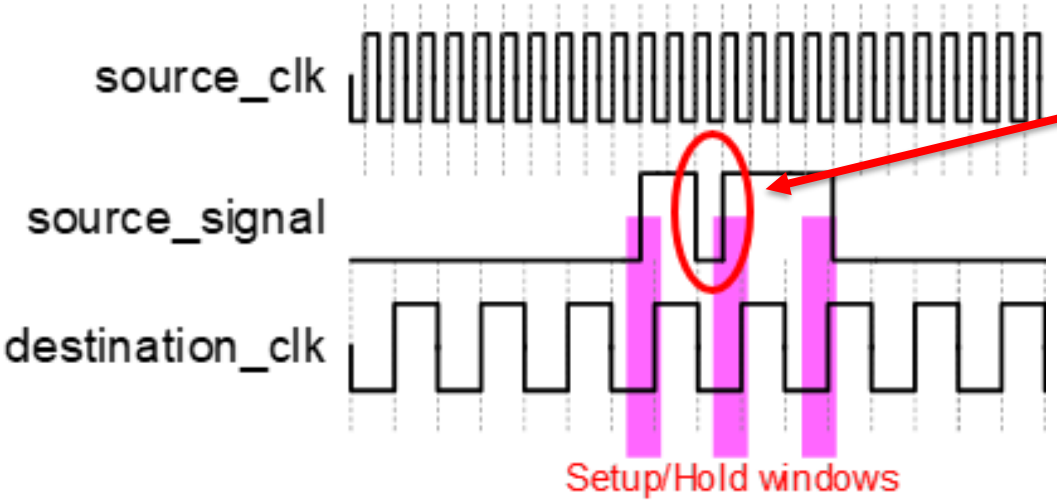
```
property pulse_width;
  @(posedge destination_clk)
    $changed(source_signal) |=> $stable(source_signal)[*2]
endproperty
```

NOTE: The SVA here is just an example. It may vary in different EDA tools.

Assertion Clocking Concern – Solution

Solution:

- **Use fastest clock in the system**



- **Pulse width assertion fail**

- **Source_clk is the fastest clock in this example.**
- **We can create a virtual faster clock depending on need.**

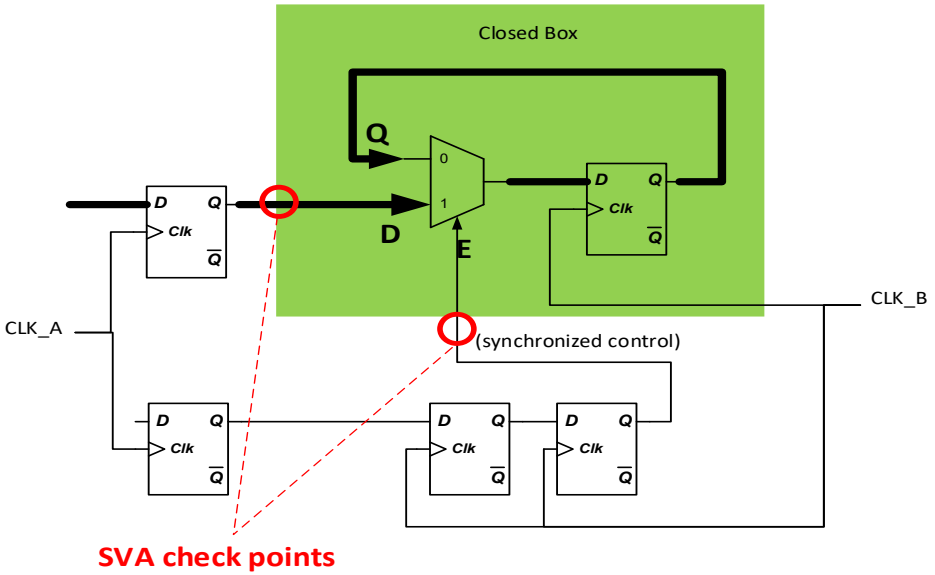
$$N = \text{ceil}[2 * (^{period\ of\ destination\_clk}/_{period\ of\ fast\_clk})]$$

```
property pulse_width(fast_clk,source_signal,N);
  @(posedge fast_clk)
    $changed(source_signal) |=> $stable(source_signal)[*N]
endproperty
```

NOTE: The SVA here is just an example. It may vary in different EDA tools.

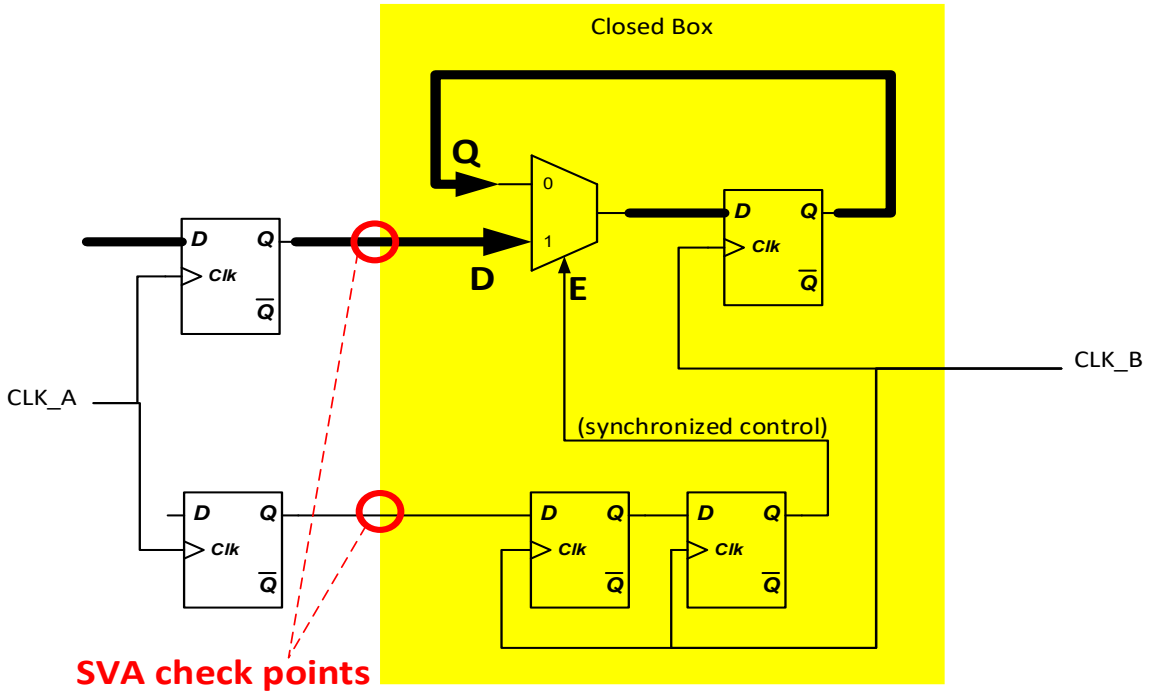
SVA limitation: Closed Box Complexity

The assertion is applied at the closed box boundary, leading to complexity in some cases



Less complexity in this scenario:

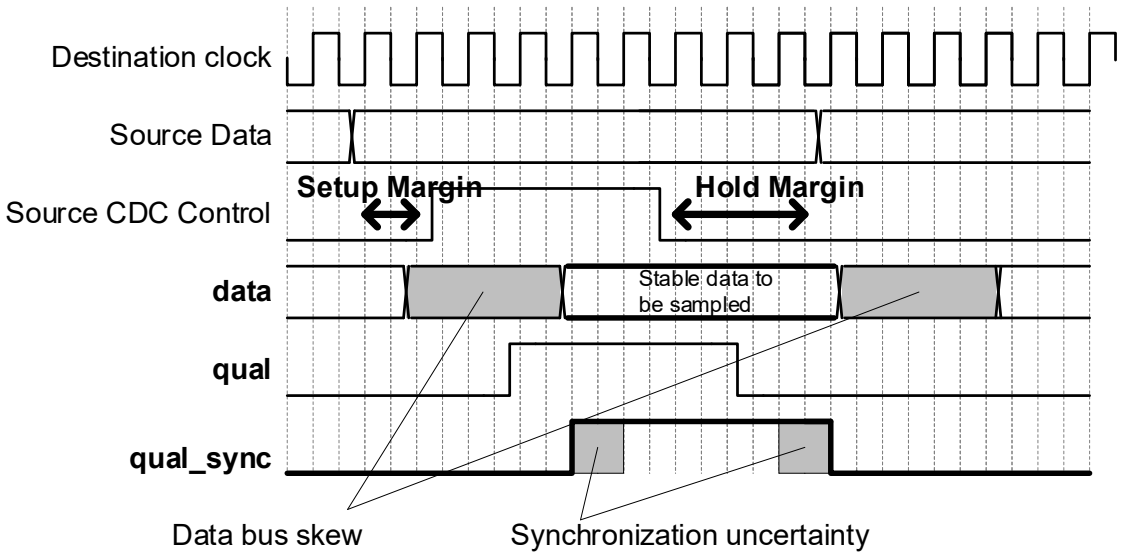
1. Signal path delay due to physical implementation



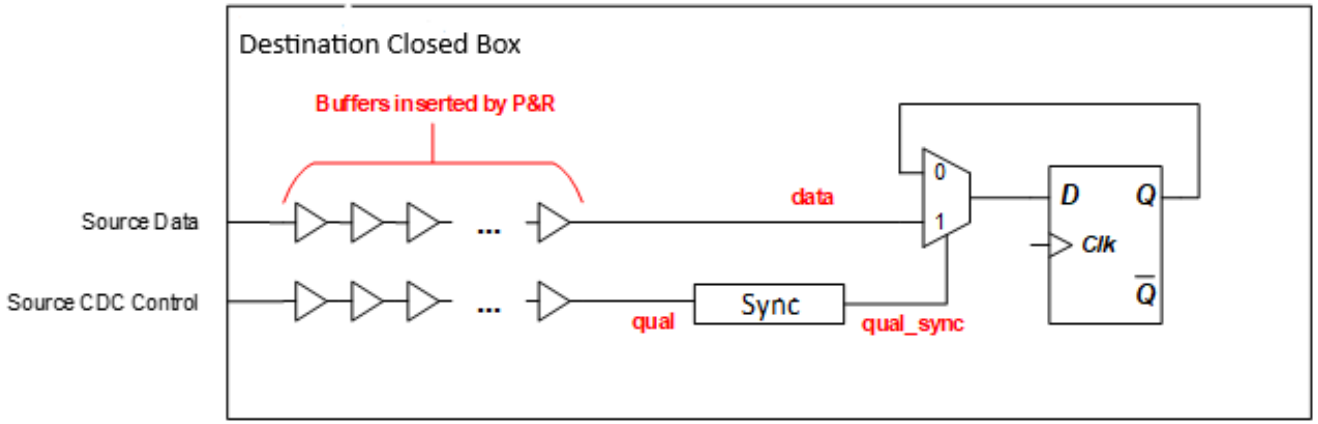
Complexity in this scenario:

1. We might not know the synchronizer's structure since it is in the closed box:
   a. Number of DFFs
   b. Active clock edges of each DFF
2. Signal path delay due to physical implementation

SVA limitation: Physical Implementation

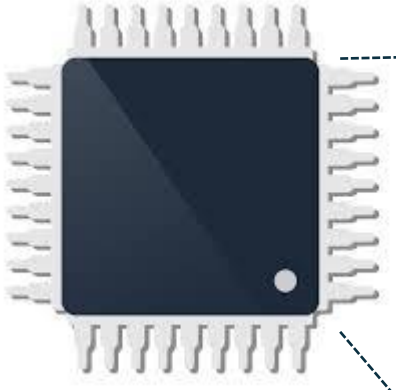


Failed silicon if we don't consider physical implementation up-front!

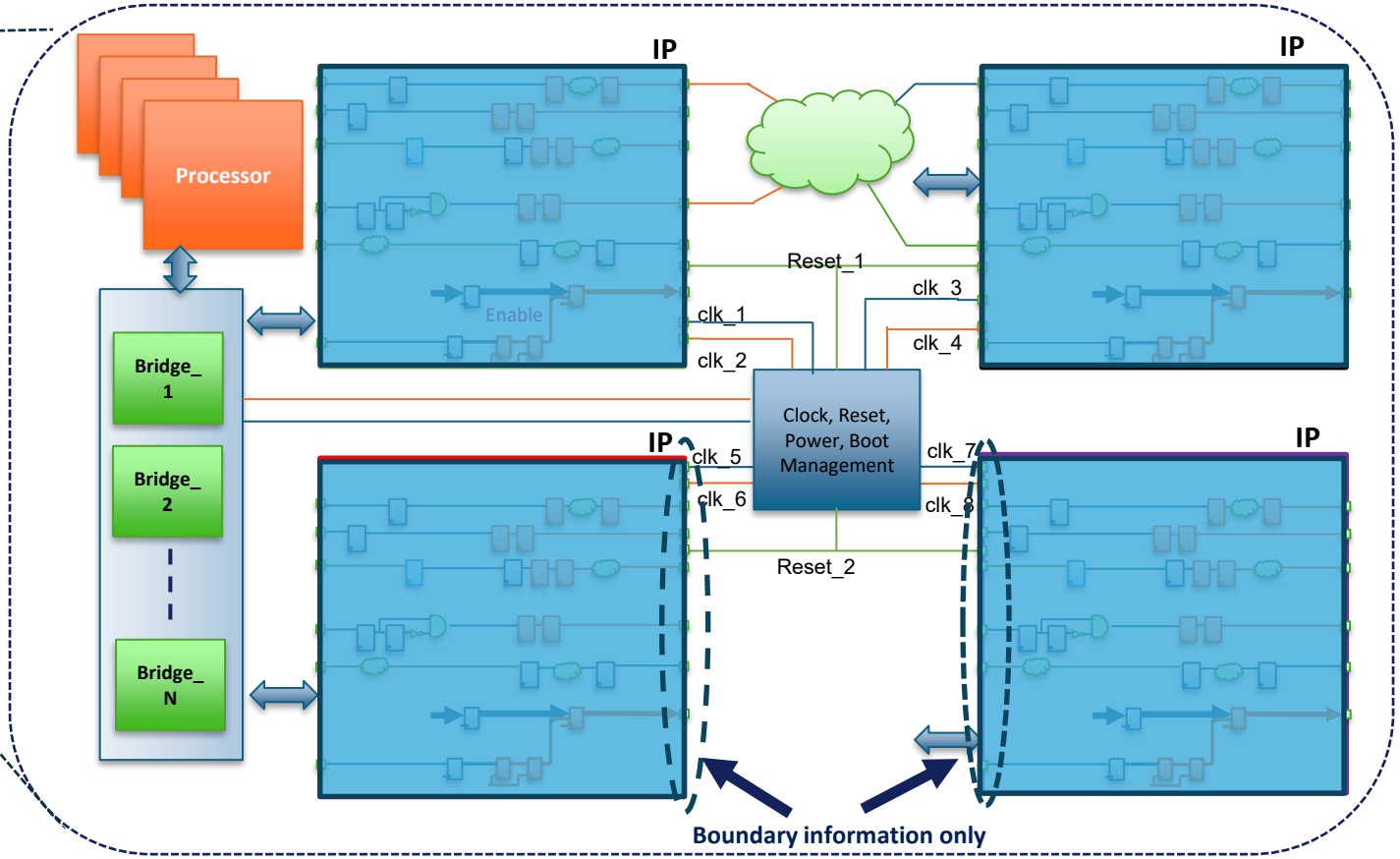


1.5 Hierarchical CDC/RDC

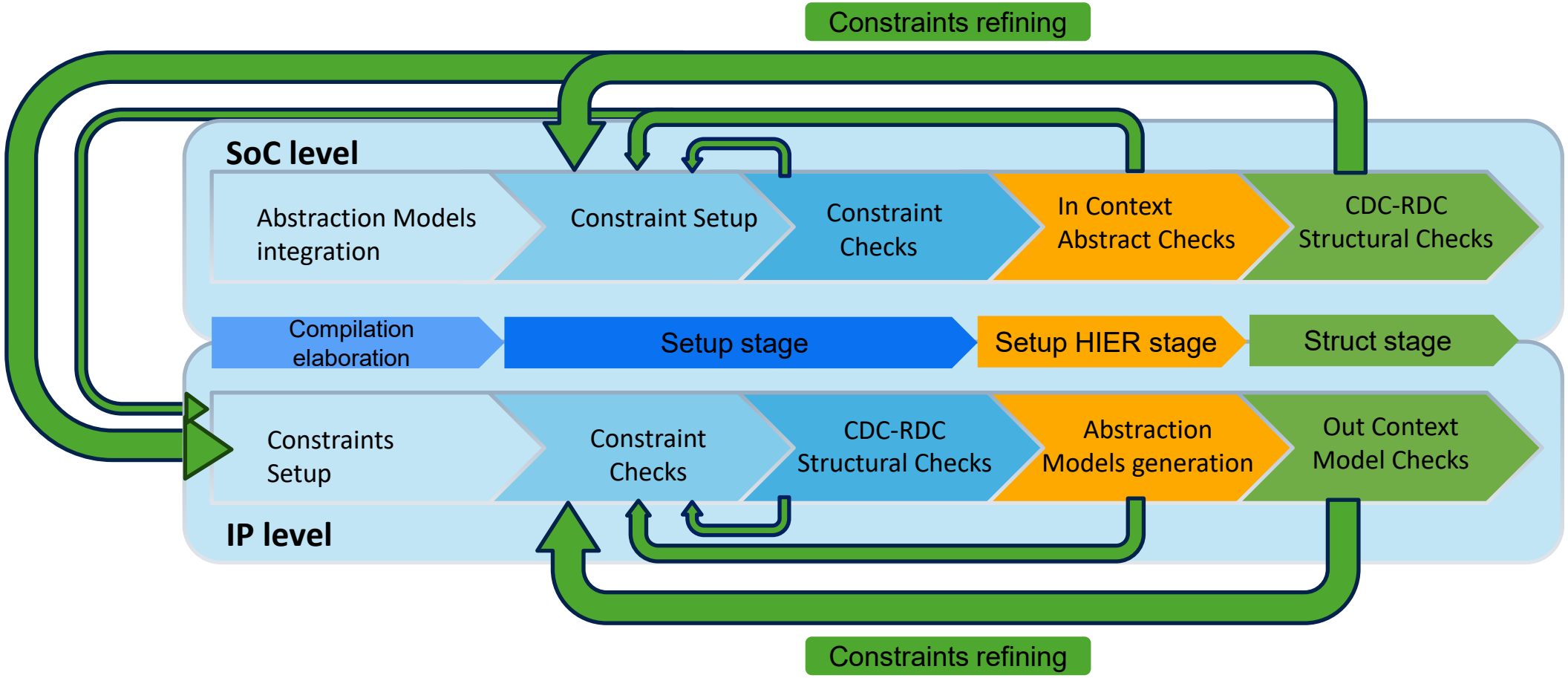
CDC-RDC Hierarchical Flow



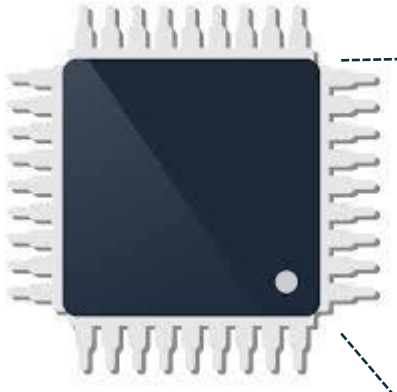
- Abstraction Models enable all Boundary related CDC-RDC info required at SoC integration Checks
- Much better approach compared to Black Box



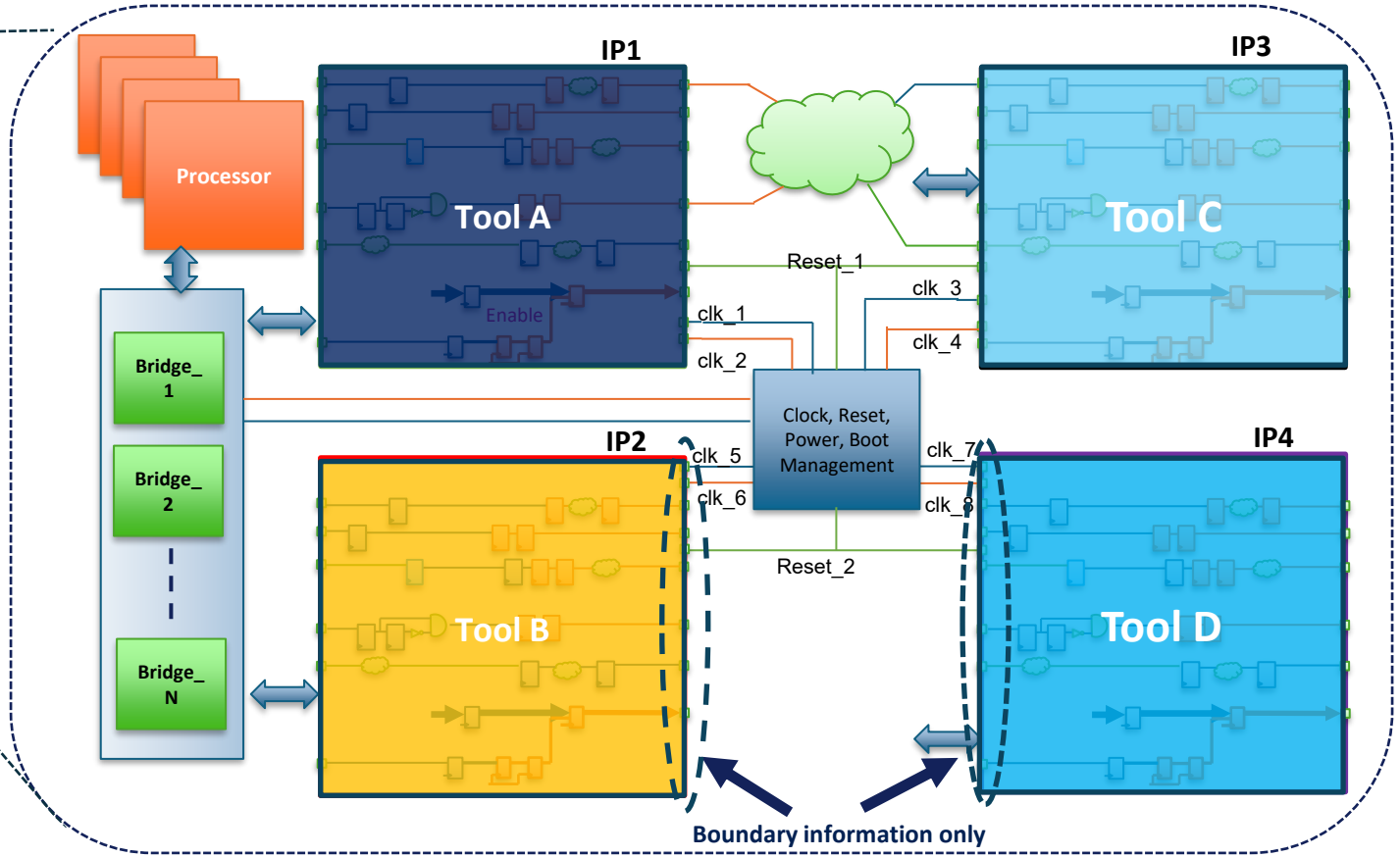
CDC-RDC Hierarchical Flow



Hierarchical Model Standardization



- CDC models are currently lacking standardization
- CDC models from different tools are not compatible



2.1 Accellera CDC Working Group

Accellera CDC Working Group

- Presentation
- The five sub working groups
- Call for contribution

What was the problem?

As we move from monolithic designs … to IP/SOC with IPs sourced from a small/select providers … to sourcing IPs globally (to create differentiated products) …

We must maintain quality as we drive faster **time-to-market**

In areas where we have standards (SystemVerilog, OVM/UVM, LP/UPF), the integration is able to meet the above (**quality, speed**)

But in areas where we don't have standards (in this case, CDC), most options trade-off either quality, or time-to-market, or both :-(

Creating a standard for inter-operable collateral addresses this gap

Accellera CDC WG initiative

Pre-WG launched Sep ‘22 to evaluate need. WG launched Jan ’23

154 members from 24 companies (as of Sept 23 ‘25)

5 active sub-groups:

Output-Collateral, Format, Assertions, Testing, Training

V0.5 available for public review in April 2025

Ver	Focus	Timeline
v0.1	CDC	Oct 2023
v0.3	RDC & Assertions	July 2024
v0.5	Complexities & Extensions	April 2025
v1.0	Final LRM release	Nov 2025

Agnisys	AMD	Analog Devices	Apple	ARM	Arteris	Blue Pearl Solutions	Cadence
Google	Huawei	IBM	Infineon	Intel	Marvell	Microsoft	NVIDIA
NXP	Qualcomm	Renesas	Robert Bosch	Samsung	Siemens EDA	ST Microelectronics	Synopsys

Achievements 2024

Jan	Feb	March	April	May	June	July	Aug	Sep	Oct	Nov	Dec
WG meeting	WG meeting	WG meeting	WG meeting	WG meeting		WG meeting		WG meeting	WG meeting	WG meeting	WG meeting
		LRM 0.2				**LRM 0.3**					LRM 0.4
		DVCON US					DVCON Japan	DVCON India	DVCON Europe		IP SoC Design conference Europe

accellera SYSTEMS INITIATIVE

Achievements 2025

Jan	Feb	March	April	May	June	July	Aug	Sep	Oct	Nov	Dec
WG meeting	WG meeting	WG meeting	WG meeting	WG meeting	WG meting	WG meting	WG meting	WG meeting			
			LRM 0.5		LRM 0.6						LRM 1.0
	DVCON US		DVCON China		DAC US		DVCON Japan	DVCON India	DVCON Europe		
Promotional Video							DVCON Taiwan				

Standard for IP Abstraction for Clock and Reset Domain Crossing Integration draft proposal; Subject to change

The Five Sub Working Groups

Output Collateral

Training

Format

Testing

Assertion

Structure
- WG Leads / co-Leads
- Members

Meetings
- Leads / co-Leads monthly
- main WG monthly
- Sub WG weeklies

2.2 Assertion Subgroup

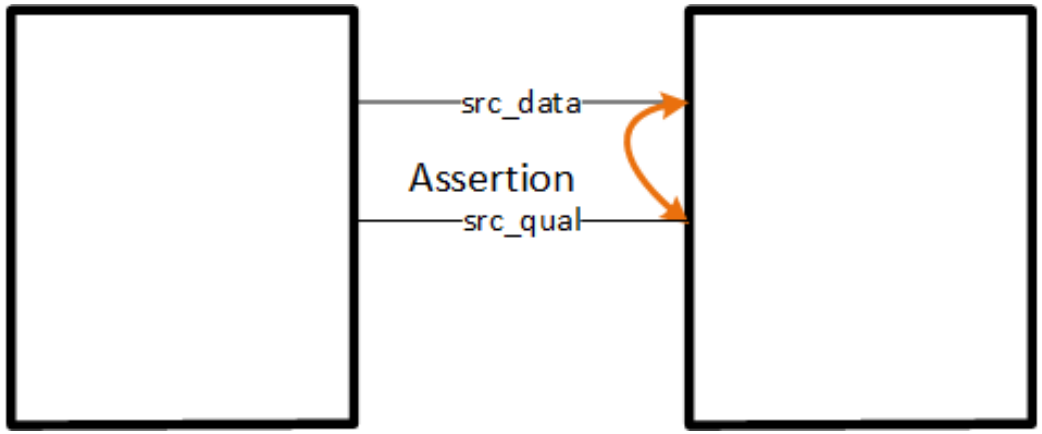
Assertion Subgroup Mission

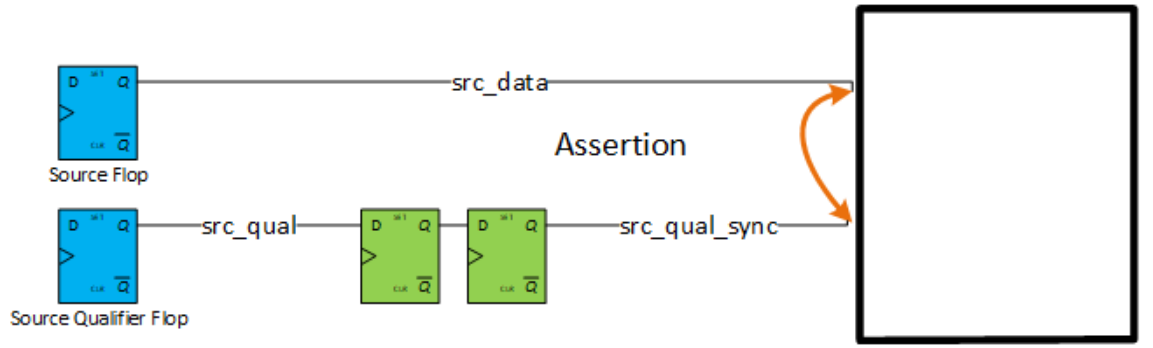
- **Goal**

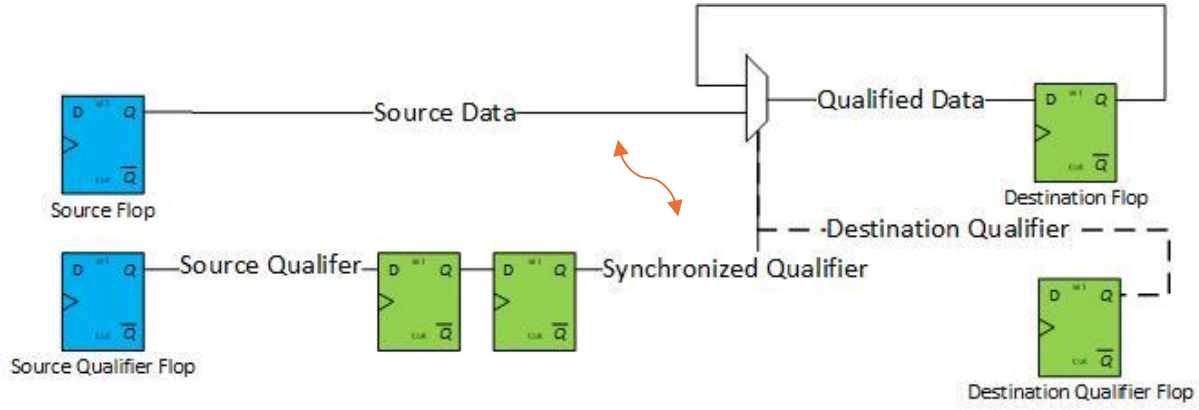
  1. Produce Language Reference Manual (LRM) addendum for Assertions.
  2. Enable all EDA vendors in developing tools that meet specification for generating System Verilog Assertions (SVA) along with collateral.
  3. Enable Intellectual Property (IP) companies to generate SVA along with collateral using various vendors/tools.
  4. Enable System On Chip (SOC) companies to consume generated SVA from any vendor/tool into their tool of choice.

Assertion Subgroup

- Integration Scenarios under consideration:
  - Blackbox IP to Blackbox IP at SOC level.
  - SOC level glue logic to Blackbox IP.
  - Blackbox to SOC level glue logic.



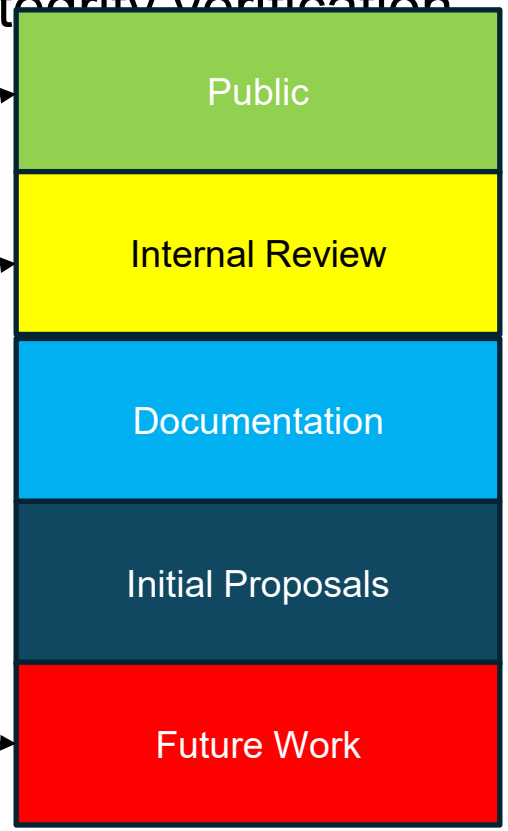




Assertion Subgroup

- LRM Clause 8 : SVA Requirements for black box CDC integrity verification

8.1 : Overview
8.2 : Sampling Edge Requirement
8.3 : Implementation Headroom for a Crossing
8.4 : Verification Clock for a Crossing
8.5 : Verification of Parameters
8.6 : Verification of Constant Signals
8.7 : Verification of Open Loop Crossing
8.8 : Verification of Closed Loop Crossing – Handshake
8.9 : Verification of Mutually Exclusive Buses
8.10 : Asynchronous FIFO
8.11 : Reset Domain Crossing



2.3 Output Collateral Subgroup

Output Collateral Subgroup

Address most industry standard interfaces

Identify limitations and extensions for the attributes

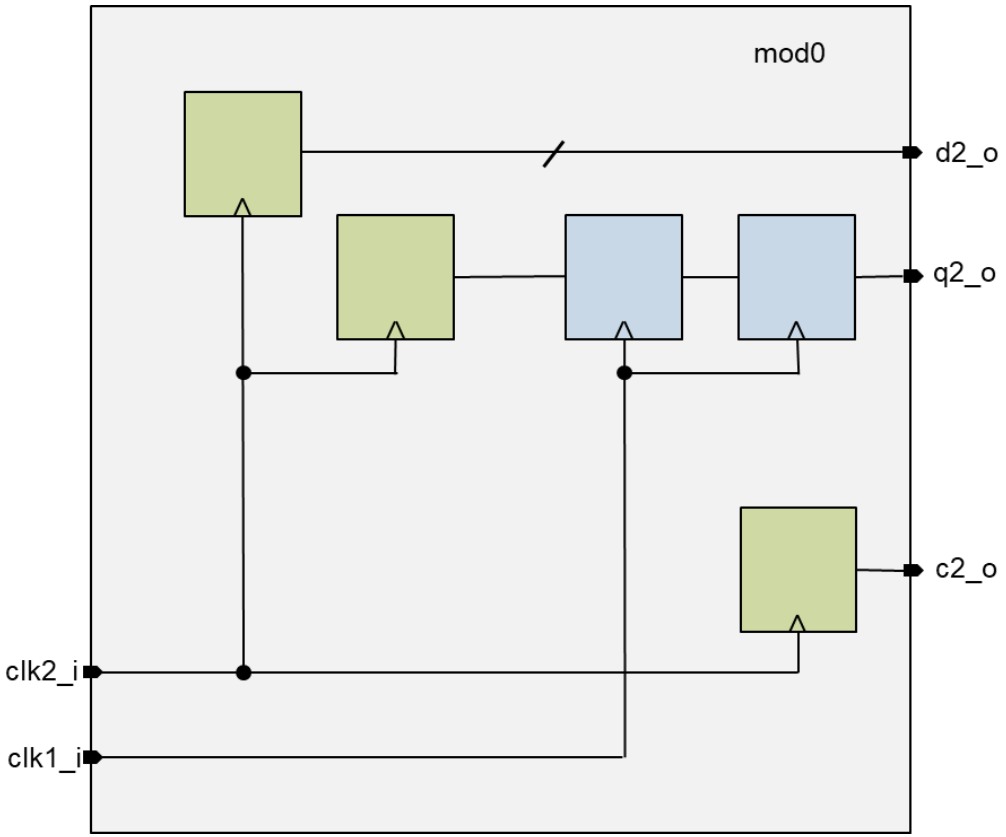
Attributes for CDC/RDC block model

Attributes Table in LRM v0.5

- Attributes split into domains
  - Indicated by the color

Domain	Attribute	Type	Values	Mandatory
module	name	string	{module name}	Yes
parameter	name	string	{parameter name}	Yes
parameter	value	range-list	{values}	Optional
parameter	type	defined set	{string, Boolean, number (hex, decimal, oct, binary)}	Optional
parameter	ignore	Boolean	{true, false}	Optional
port	name	string	{port name}	Yes
port	direction	defined set	{input, output, inout}	Yes
port	type	defined set	{data, clock, virtual_clock, async reset, cdc_control, rdc_control, virtual_reset}	Yes
port	logic	defined set	{combo, buffer, inverter, glitch-free-combo, internal-sync}	Optional
port	cdc_data_from _clock	; separated list	{clock-names}	Optional
port	associated_from_clocks	; separated list	{clock-names}	Yes
port	associated_to_clocks	; separated list	{clock-names}	Optional
port	associated_inputs	; separated list	{ports}	Optional
port	associated_outputs	; separated list	{ports}	Optional
port	cdc_control	; separated list	{associated-ports}	Optional
port	polarity	defined set	{high, low, low_high}	Yes
port	ignore	defined set	{blocked, hanging}	Optional
port	cdc_static	; separated list	{clock-names}	Optional
port	constant	; separated list	{binary, hex, and of any length}	Optional
port	gray_coded	Boolean	{true, false:default}	Optional
port	clock_period	string	{clock period}	Optional
port	associated_from_reset	; separated list	{reset-names}	Optional
port	associated_to_reset	; separated list	{reset-names}	Optional
port	rdc_data_from_reset	; separated list	{reset-names}	Optional
port	rdc_data_to_reset	; separated list	{reset-names}	Optional
port	rdc_data_to_clock	; separated list	{clock-names}	Optional
port	rdc_clock_gate_location	defined set	{external or internal}	Optional
tool	name	string	{tool name}	Yes
tool	version	string	{tool Version}	Yes
design	version	string	{design milestone}	Optional
design	date	string	{collateral generation date}	Yes
design	username	string	{user/tool that generated the collateral}	Optional
design	description	string	{description}	Optional
set_cdc_clock_group	clocks	; separated list	{clock-names}	Yes
set_cdc_clock_group	name	string	{group-name}	Optional
set_reset_group	reset	; separated list	{clock-names}	Yes
set_reset_group	name	string	{group-name}	Optional

Port Attribute Modelling



clk1_i clock domain

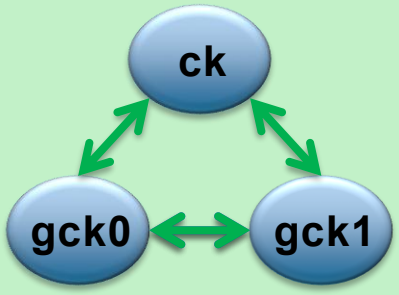
clk2_i clock domain

example in Tcl-format for output interface

```
cdc_set_module mod0

   cdc_set_port d2_o                    \
      -type                    data     \
      -direction               output   \
      -associated_from_clocks  clk2_i   \
      -cdc_control             q2_o

   cdc_set_port q2_o                        \
      -type                    cdc_control  \
      -direction               output       \
      -cdc_data_from_clock     clk2_i       \
      -associated_from_clock   clk1_i       \
      -associated_outputs      d2_o

   cdc_set_port c2_o                     \
      -type                    data      \
      -direction               output    \
      -associated_from_clock   clk2_i
```

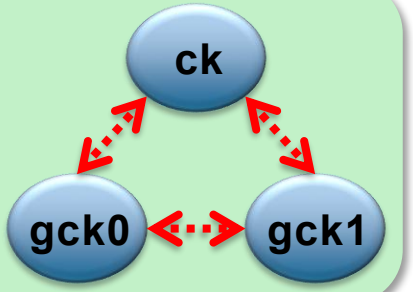
cdc_set_clock_group (Tcl format)

1 domain



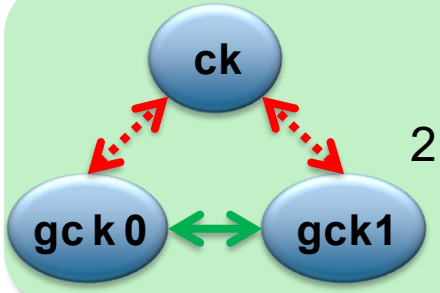

```
cdc_set_clock_group -name common_domain -clocks {ck
gck0 gck1}
```

3 domains



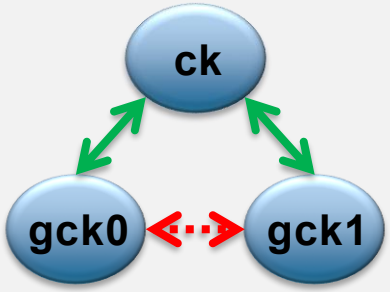

```
cdc_set_clock_group -name domain_C -clocks {ck}
cdc_set_clock_group -name domain_0 -clocks {gck0}
cdc_set_clock_group -name domain_1 -clocks {gck1}
```

2 domains




```
cdc_set_clock_group -name small_domain -clocks {ck}
cdc_set_clock_group -name large_domain -clocks {gck0 gck1}
```

2 clock branches



compatibility sets: common members allowed

```
cdc_set_clock_group -name left_branch  -clocks {ck gck0}
cdc_set_clock_group -name right_branch -clocks {ck gck1}
```

2.4 Testing Subgroup

Testing Subgroup Mission

- **Goal**

  1. **Evaluate** the set of Accellera CDC attributes and protocols for completeness using multiple tools from multiple vendors.
  2. **Demonstrate** the use of the complete set of attributes and protocols as defined and formatted by other sub-groups.
  3. **Provide RTL design examples** within which the defined attributes and protocols can be further qualified and evaluated
     - Review small examples provided in the latest LRM to create RTL test cases
     - Analyze the RTL test cases using various EDA tools
  4. **Make the test cases available** for EDA vendors as first level QA testing

2.5 Format Subgroup

Format Subgroup Mission

- **Goal:**
  - Defined a formal methodology for specifying CDC constraints using **IP-XACT** and **TCL**

- **Methodology:**
  - Defined a formal methodology for specifying CDC constraints using **IP-XACT** and **TCL**

  - **IP-XACT**: Adopted accellera-CDC vendor extensions to represent CDC constraints in a structured, machine-readable format.

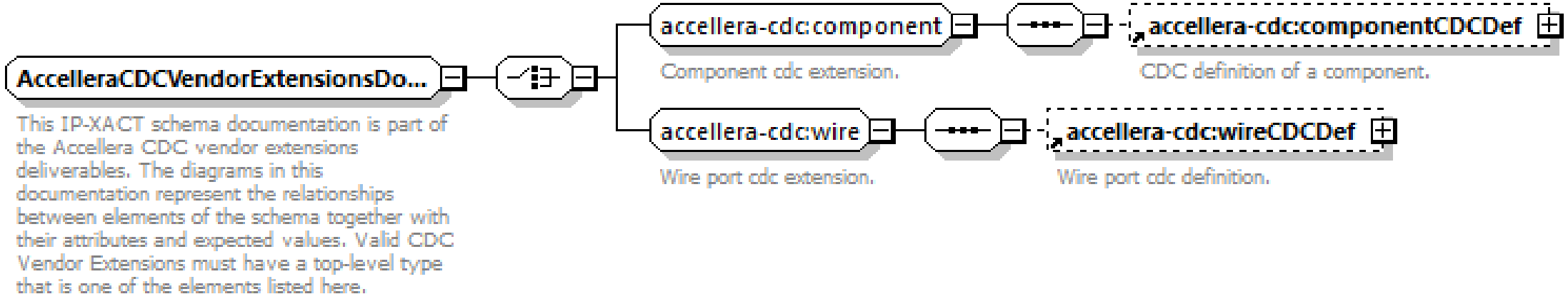
  - **TCL:** Adopted the **accellera_cdc namespace** for CDC constraint specification
    Utilized the **TCL APIs** defined last year to describe CDC intent and constraints

  - **2026 focusing on**: Formalizing the semantics and usage rules of the **TCL APIs**

- **Key Outcome**
  - Standardized TCL-based CDC constraint usage, aligned with IP-XACT representation, enabling consistent interpretation across tools

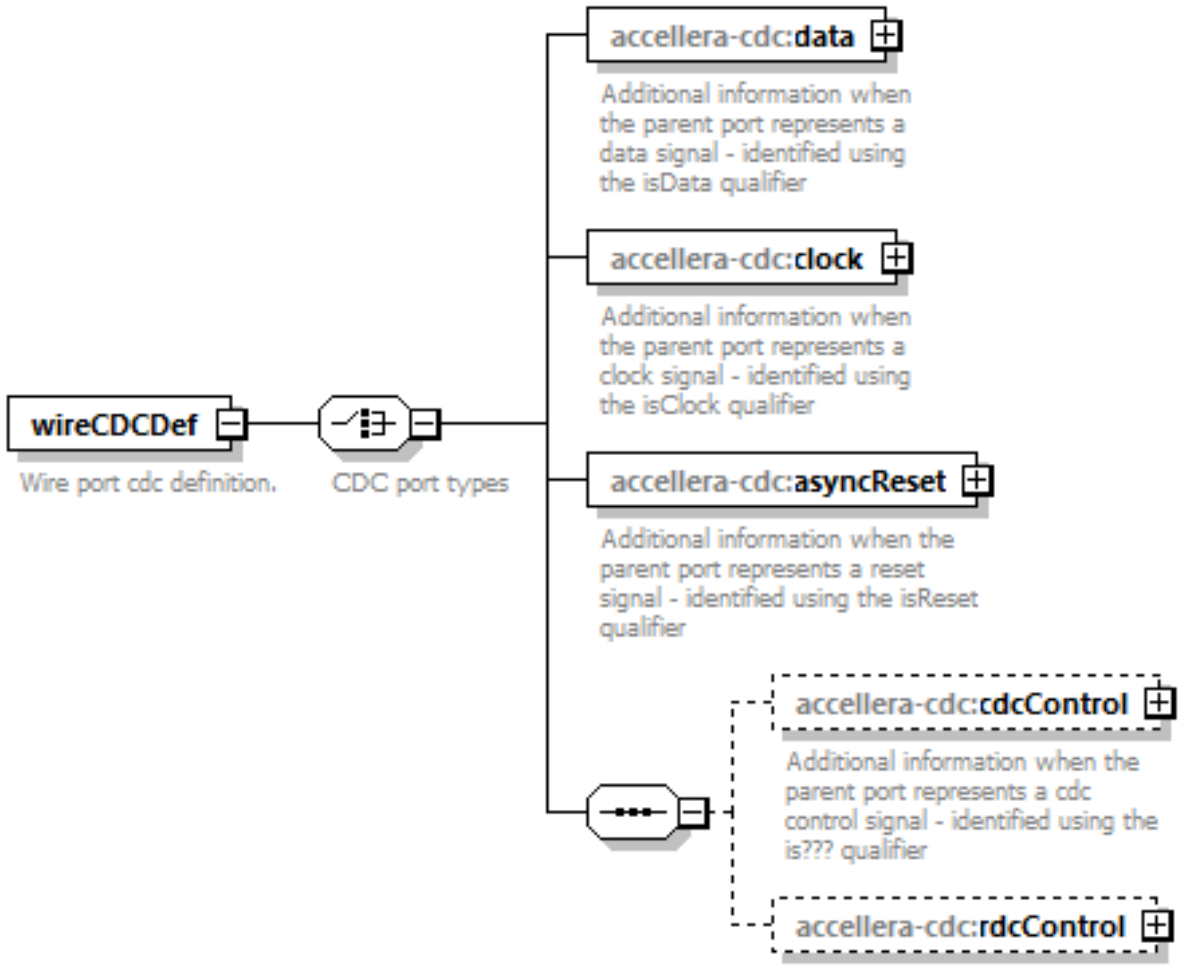
IP-XACT Format for CDC (1/4)

- The IP-XACT standard already captures some attributes related to clock and reset
- IP-XACT allows extending the standard using vendor extensions
- Vendor extensions for CDC are defined as **accellera-cdc** Vendor extensions at different elements in the design hierarchy
- Top level elements for the Accellera vendor extensions for CDC are:
  - **accellera-cdc:componentCDCDef**
  - **accellera-cdc:wireCDCDef**



IP-XACT Format for CDC (2/4)

- **accellera-cdc:wireCDCDef** is used to define an accellera CDC specific wire port extension
- It links every signal's value (data) to its source clock and control signals, providing all information needed for CDC analysis.
- Captures all information required for CDC analysis, including:
  - **accellera-cdc:data**
  - **accellera-cdc:clock**
  - **accellera-cdc:asyncReset**
  - **accellera-cdc:cdcControl**
  - **accellera-cdc:rdcControl**



IP-XACT Format for CDC (3/4)

- When a port represents a data signal, all attributes required for CDC are captured in **accellera-cdc:data** extension

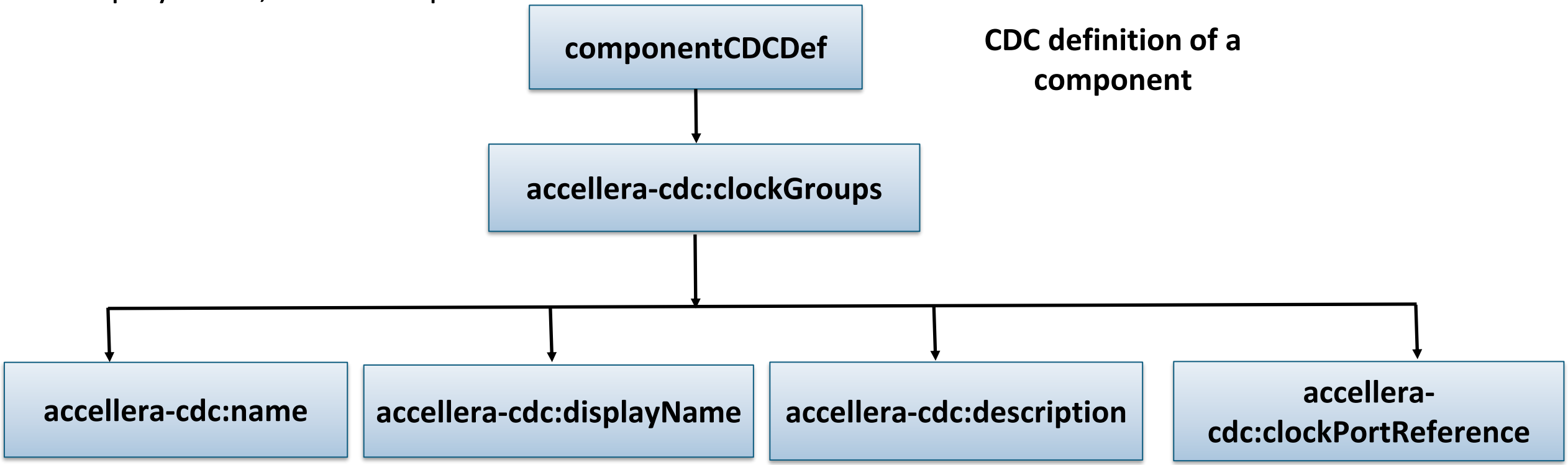
- asyncReset, cdcControl, and rdcControl define resets and CDC control signals; their schema is defined similarly to data.

- Key attributes for CDC analysis are:
  - data : signal value
  - asyncReset : associated resets
  - cdcControl / rdcControl : CDC control signals

- Full schema details are available in reference manual

IP-XACT Format for CDC (4/4)

- The extension **accellera-cdc:componentCDCDef** is defined to capture the clock groups of a component using the accellera-cdc:clockGroup extensions
- It contains references to clock ports or phantom ports (virtual clock) in addition to name, displayName, and description.



TCL Format for CDC(1/6)

- TCL Format for CDC specification is a set of API commands
- One cdc file per block/module
  - users specify the CDC attributes for ports of that module
  - users can also specify clock groups
- EDA tools should process the CDC specification in TCL format and generate corresponding IP-XACT collateral
- Similarly, there could be a requirement for generating TCL specification of CDC from IP-XACT

API Commands

- **accellera_cdc::set_module** : indicates the block/module name for which cdc specification is being provided in a file
- **accellera_cdc::set_port** : sets attributes of a port
- **accellera_cdc::set_clock_group** : set clock groups of synchronous clocks
- **accellera_cdc::set_param** : defines parameters within the scope of a module

TCL Format for CDC(2/6): accellera_cdc Namespace

- EDA tools shall support the use of an **accellera_cdc namespace**, so these commands can be used with the namespace as shown on the RHS

- NOTE: The **accellera_cdc namespace** cannot include non-standard extensions. If custom extensions are needed, a different namespace name should be used.

```
# Defines a namespace
#(within the EDA tool - transparent to the user)
namespace eval accellera_cdc {
        proc set_module {} {
                ...
        }
        proc set_port {} {
                ...
        }
        proc set_clock_group {} {
                ...
        }
        proc set_param {} {
                ...
        }
  }
# Users Tcl for CDC
accellera_cdc::set_module -name ALU
accellera_cdc::set_port CLK -type
```

TCL Format for CDC(3/6): accellera_cdc::set_module

- This command allows users to indicate the module/block name for which CDC specification is being provided in a file
- The subsequent commands are applicable to the module that is specified as module-name in the **accellera_cdc::set_module** command
- There can be an RTL file defining the specified module, but it is not mandatory
- There shall be an error if module-name is not specified.

```
# Set a module for CDC specification
accellera_cdc::set_module -name ALU
*ALU is the name of the module defined in RTL (but it is not necessary for RTL to exist
```

Example 39—accellera_cdc::set_module

TCL Format for CDC(4/6): accellera_cdc::set_param

- The **accellera_cdc::set_param** command allows users to define parameters within the scope of a module
- There shall be an error if:
  a. parameter-name is not specified.
  b. If parameter is ignored and still referred to in subsequent commands..

```
# Set a parameter for the module
accellera_cdc::set_param -name MSB -type int -value 15
accellera_cdc::set_param -name LSB -type int -value 0
                                        *MSB & LSB are parameters
```

Example 40—accellera_cdc::set_param

TCL Format for CDC(5/6): accellera_cdc::set_port

- Single **accellera_cdc::set_port** command defines all CDC and RDC attributes for a port
- Multiple **accellera_cdc::set_port** commands with the same port name within a single module override the previous values of attributes
- There shall be an error if:
  a. a module has not been set using the **accellera_cdc::set_module** command.
  b. an attribute is specified for a particular type of port that is not applicable to that type of port.
  c. range is not specified correctly for a vector or multidimensional port.
- **accellera_cdc::set_port** command can be used to define following:
  - Scalar Ports
  - Vector Ports
  - 2D (Multi-dimensional) Ports
  - Concatenated Ports
  - Negative or reverse Indexing
- Specific details on how to use it are provided in the reference manual

TCL Format for CDC(6/6): accellera_cdc::set_clock_group

- The accellera_cdc::set_clock_group command allows users to set clock groups of synchronous clocks
- A clock group can be specified as a Tcl list having one or more clock names that are synchronous
- There shall be an error if one or more specified clock names do not exist in the design

```
# set cdc clock group
accellera_cdc::set_clock_group -clocks {clk1 clk2}
*clocks is name of the clock group and clk1,clk2 are name of the clocks in clock group that are synchronous to each other
```

Example 42—accellera_cdc::set_clock_group

2.6 Training Subgroup

Training Subgroup Mission

- **Goal**
  1. **Raising awareness of** the importance of defining **a standard CDC-RDC model**
  2. **Provide generic documentation** to let the CDC-RDC IP model user understand :
     - 1.1 CDC-RDC basic knowledge
     - 1.2 List of attributes & definition (related to IP CDC-RDC features/properties) as defined and agreed by the main CDC WG
  3. Presentation of the **hierarchical flow**
     - 2.1 tool dependency issue
     - 2.2 necessity to create an inter operational CDC-RDC model
  4. Inter operational CDC-RDC model **integration manual**

Conferences

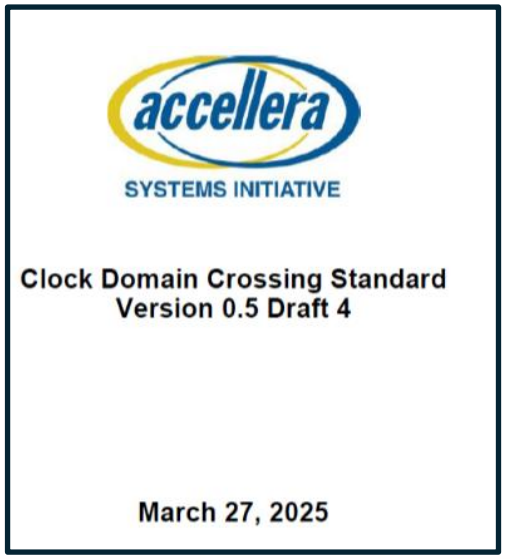
- Accellera CDC WG work promotion through conferences
  - Past/current conferences
    - DVCON Europe 2023, 2024, 2025
    - DVCON US 2024, 2025, 2026
    - DVCON Japan 2024, 2025
    - DVCON India 2024, 2025
    - IP-SoC Design Reuse conference Europe 2024, 2025
    - DVCON China 2025
    - DAC 2025
    - DVCON Taiwan 2025
  - Targeted conferences (To Be Confirmed)
    - DVCON China /Japan / India / Europe / Taiwan 2026
    - IP-SoC Design Reuse conference Europe 2026
    - DATE 2026
    - VLSI 2026
    - DAC 2026

2.7 Call For Contribution

Language Reference Manual (LRM)

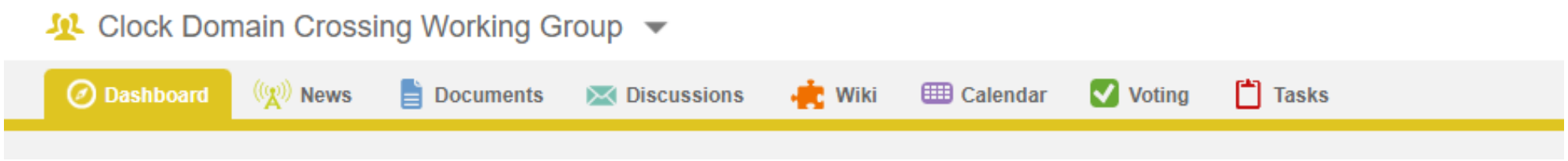
Version 0.5 of the CDC LRM has been released for public review on 14 April 2025

- CDC 0.5 Public Review Draft 2025.04.14
- Download the CDC Draft Standard 0.5: https://bit.ly/3tdwMpe
- Forum: https://bit.ly/42vYdb4



Call for Contribution !

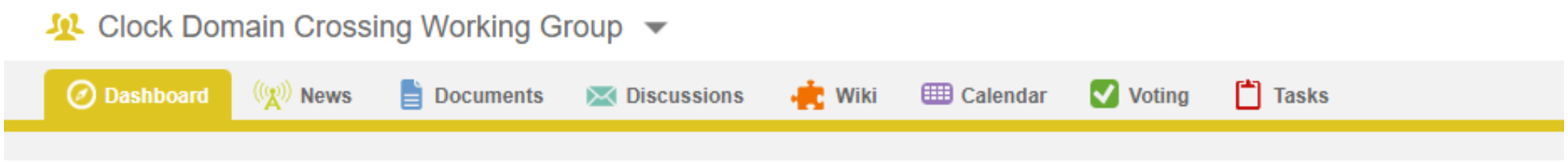
Accellera CDC Working Group



https://workspace.accellera.org/wg/CDC

Call for Contribution !

Accellera CDC Working Group



accellera.org/activities/working-groups/clock-domain-crossing

Non-Accellera members can join and provide feedback on the standard: forums.accellera.org

Accellera and IEEE

- **Accellera will release 10 standards for 10 years under an extended IEEE Get program**
- **IEEE standards Access at No Charge**
- **More than 221,000 downloads to date!**

IEEE GET EDA Standards

Download IEEE Standards

http://ieeexplore.ieee.org/Xplore/home.jsp

- or find links to specific standards at -

www.accellera.org/downloads/ieee

Questions?