Taking SDC Constraints to the Next Level with Timing Constraints Manager

Dustin Peterson
Robert Bosch GmBH (ME/EIS3)

Taking SDC Constraints to the Next Level with TCM

1. Problem:
   Why are constraints an issue for flat timing signoff in hierarchical designs?

2. Solution without TCM:
   What was the custom solution for constraint promotion to top-level?

3. Solution with TCM:
   Implementation of TCM-based constraint promotion flow in RadarSoC

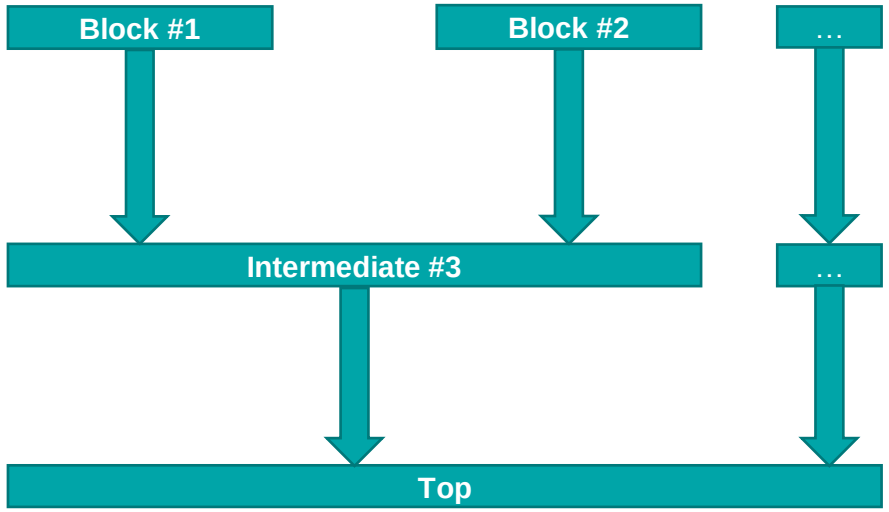
4. Results:
   Benefits and drawbacks of custom flow vs. TCM

1. Problem

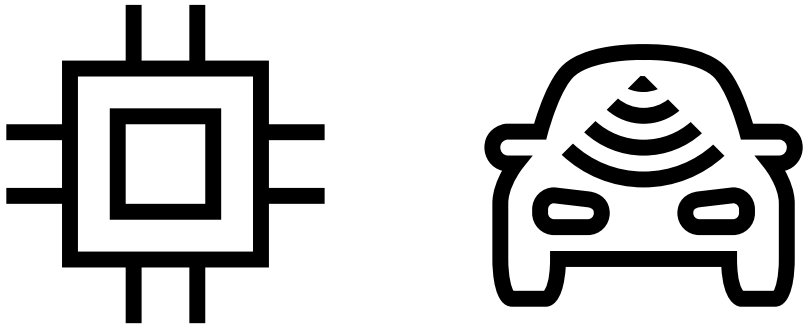
Why are constraints an issue for flat timing signoff in hierarchical designs?

Timing Signoff of Hierarchical Designs

- This talk focuses on hierarchical designs, i.e. when a chip is comprised of multiple individually hardened blocks, sub blocks and macros.



- How to signoff hierarchical designs safely to reach quality for Automotive products?
- **Our Vehicle:** GF 22FDX-based RadarSoC[1][2]



[1] P. Ritter et al., "A Fully Integrated 78 GHz Automotive Radar System-an-Chip in 22nm FD-SOI CMOS," 2020 17th European Radar Conference (EuRAD), Utrecht, Netherlands, 2021, pp. 57-60, doi: 10.1109/EuRAD48048.2021.00026

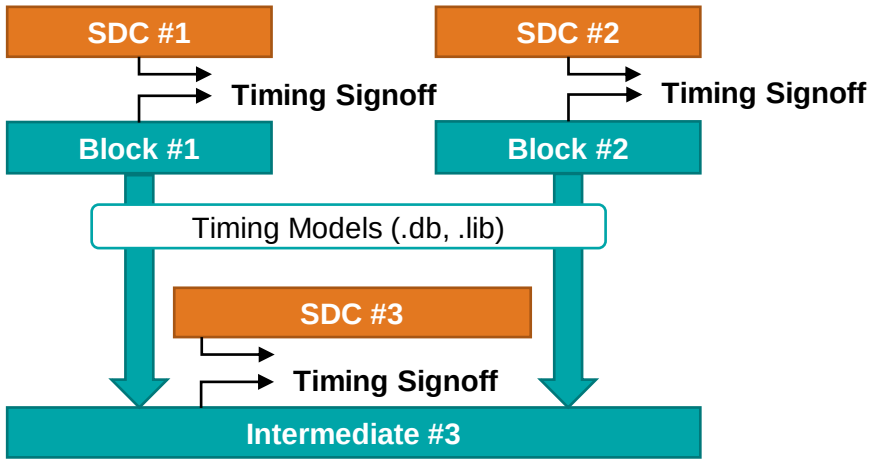
[2] https://www.all-electronics.de/markt/globalfoundries-und-bosch-entwickeln-automotive-radar-technologie.html

Hierarchical Timing Signoff

- Each block signed off individually with block-level SDC.
- Top signoff: ETMs and top-level SDC

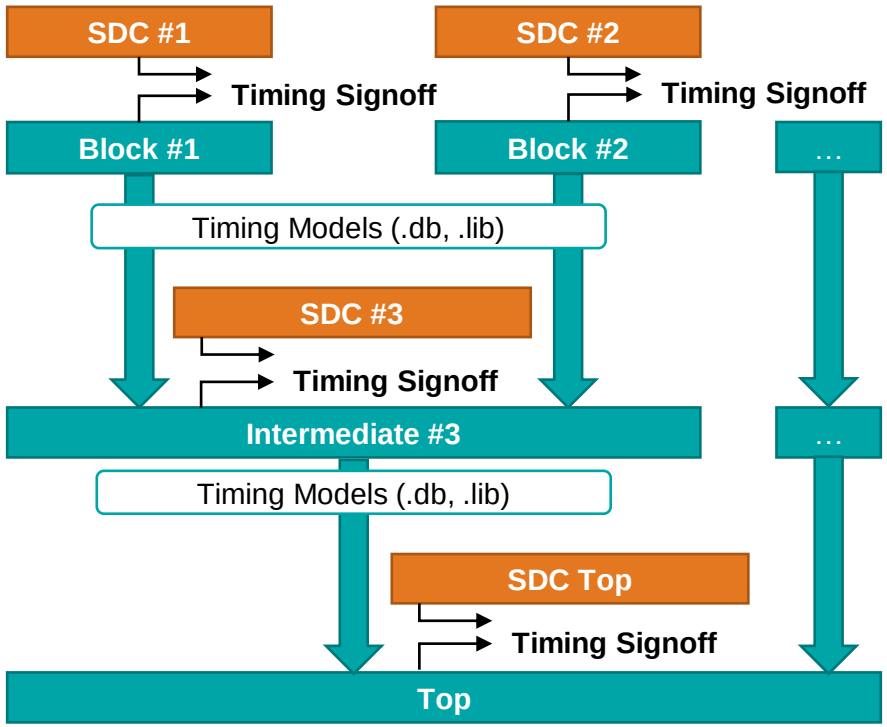
Hierarchical Timing Signoff

- Each block signed off individually with block-level SDC.
- Top signoff: ETMs and top-level SDC



Hierarchical Timing Signoff

- Each block signed off individually with block-level SDC.
- Top signoff: ETMs and top-level SDC

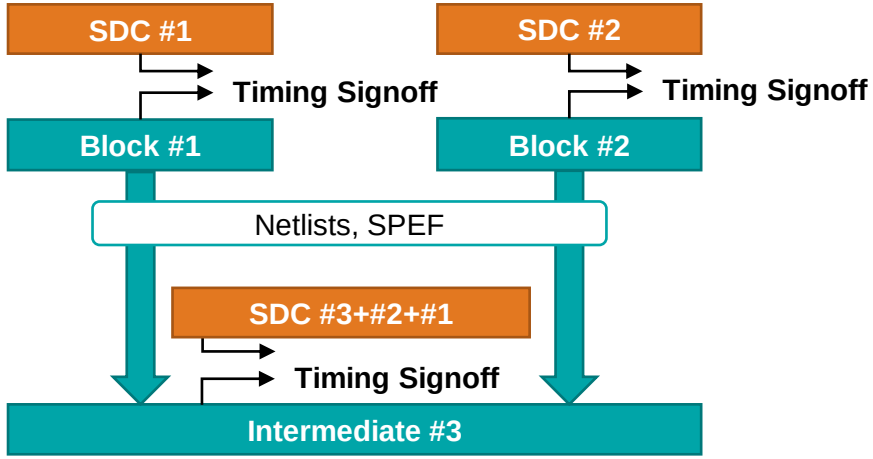


- Benefits:
  - Lower runtimes.
  - Block-level constraints to be ignored mostly
- Drawbacks:
  - Applicable only for coarse nodes with neglectable crosstalk impacts!

- **For larger technology nodes reasonable signoff approach!**

Flat Timing Signoff

- Each block signed off individually with block-level SDC.
- Top signoff: Netlists and all SDCs

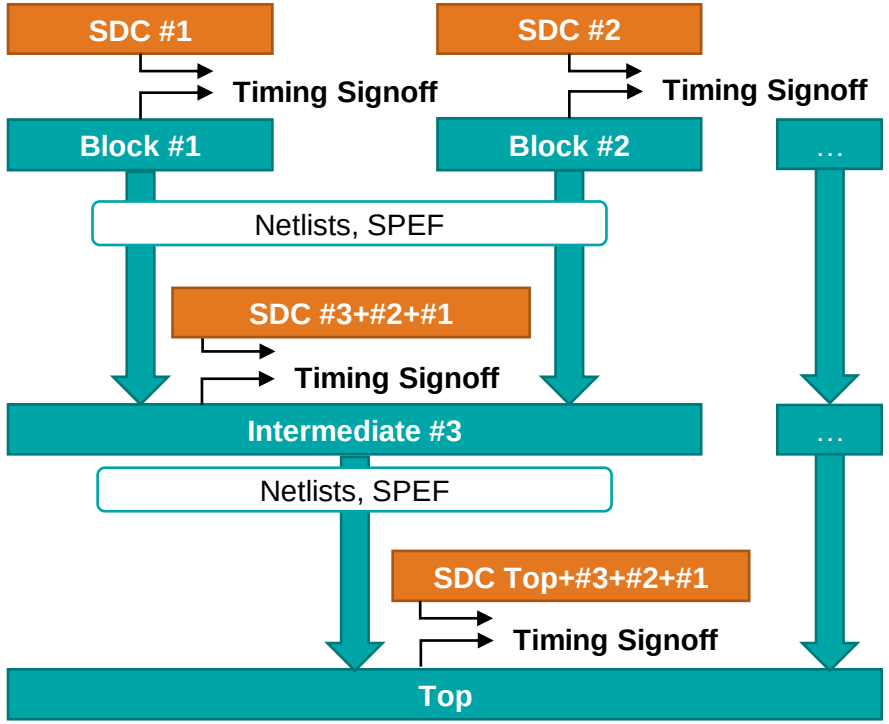


Flat Timing Signoff

- Benefits:
  - Most accurate signoff:
    - Modeling of clock crosstalk effects on block-level reg2reg timing.
    - Modeling of arrival curves from top- to block-level could also unveil noise violations.
- Drawbacks:
  - Larger runtimes.
  - Block-level constraints need to be promoted to top-level (clock grouping, exceptions, case settings, …)

- **Indispensable for smaller technology nodes like for our 22nm case**

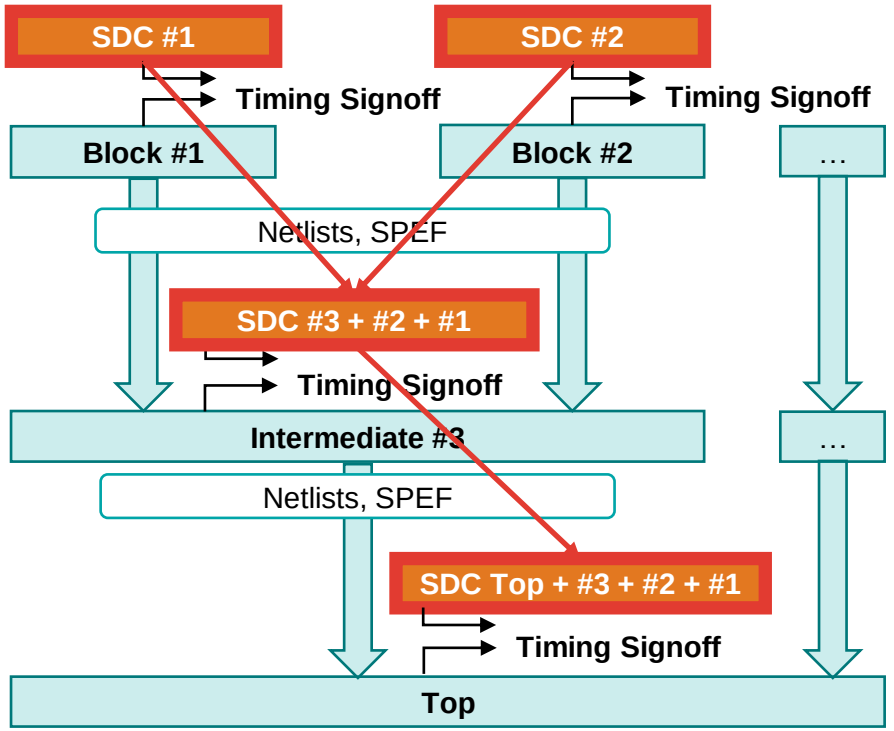
- Each block signed off individually with block-level SDC.
- Top signoff: Netlists and all SDCs



Constraints in Hier. vs. Flat Timing Analysis

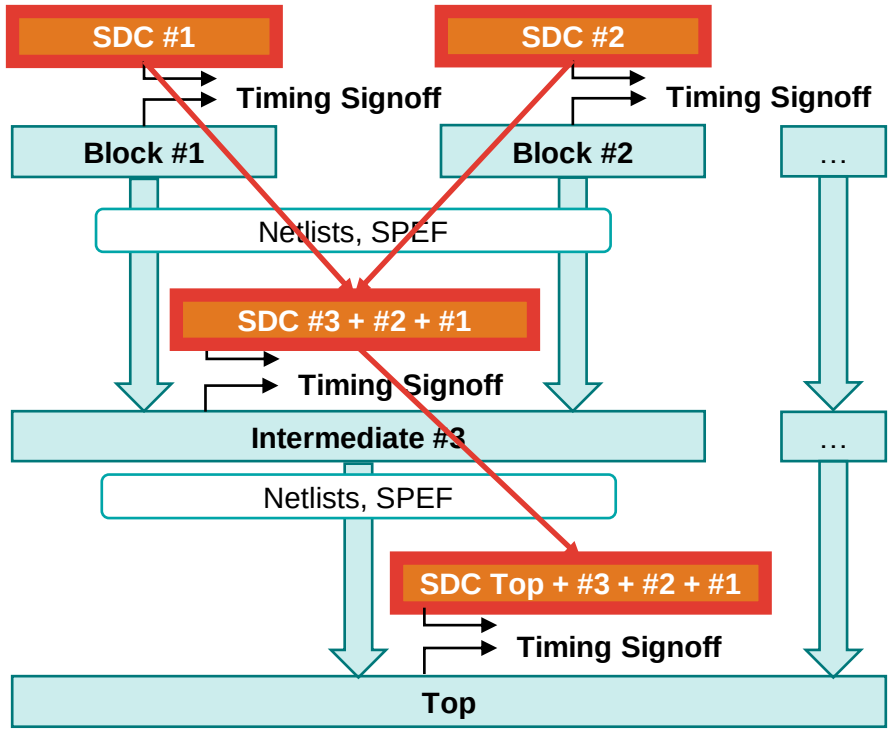
Each block is a „blackbox“ so no need for block-level constraints at top-level signoff.

Hierarchical STA



We need block-level constraints for top-level signoff – how do we achieve that?

Flat STA

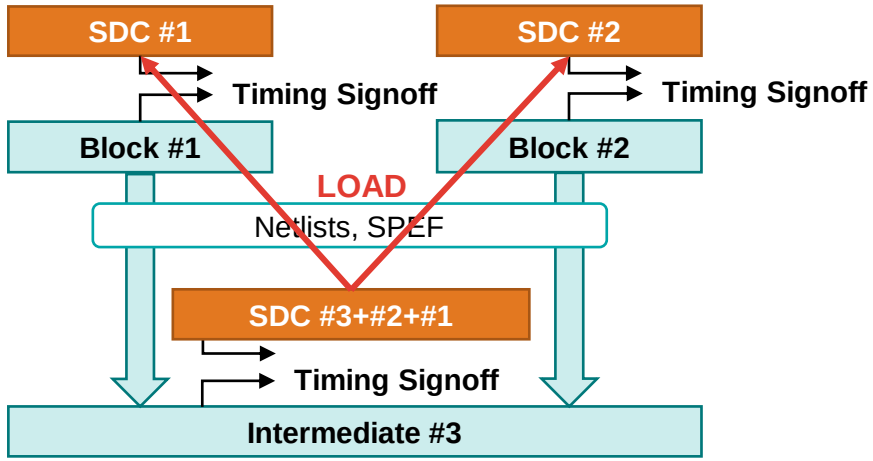


How do we take SDC constraints to the next level?

2. Solution without TCM

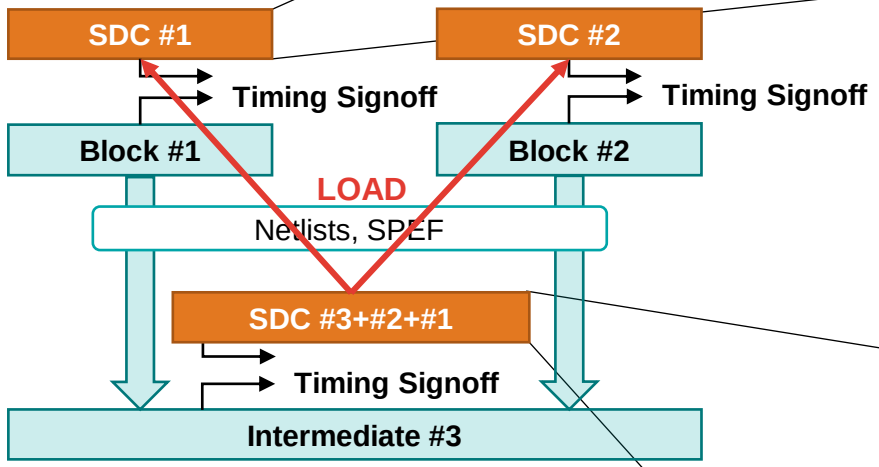
Custom solution for constraint promotion to top-level

Tcl-based Constraining



- One main Tcl constraints file per block.
- Constraints used for block- and top-level signoff!
- Definition of pseudo Tcl API to define:
  - Instance prefix of block-level
    `$design_setup(<block>,prefix) <path/to/instance/>`
  - Timing mode of block-level
    `$design_setup(<block>_3,mode) <mode_name>`
  - Clock mapping from block-level to top-level clocks
    `$design_setup(<block>,clock_map) [list <block_level_clock_1> <top_level_clock_1> ... <block_level_clock_n> <top_level_clock_n>]`
- Top-level Tcl constraints file loads block-level Tcl constraints files

Tcl-based Constraining



block_1_constraints.tcl

```
set_false_path -to $design_setup(block_1,prefix)endpoint
if { [string length $design_setup(block_1,prefix)] > 0 } {
  set_case_analysis 0 [get_ports scan_mode]
} else {
  set_case_analysis 0 [get_pins $design_setup(block_1,prefix)/scan_mode]
}
…
```

block_2_constraints.tcl

```
set_false_path -to $design_setup(block_1,prefix)endpoint
if { [string length $design_setup(block_1,prefix)] > 0 } {
  set_case_analysis 0 [get_ports scan_mode]
} else {
  set_case_analysis 0 [get_pins $design_setup(block_1,prefix)/scan_mode]
}
…
```

intermediate_3_constraints.tcl

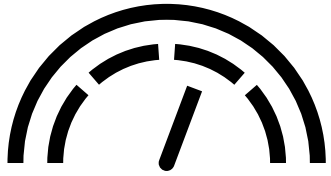
```
# Intermediate #3 Constraints …

set design_setup(block_1,mode)    $design_setup(intermediate_3,mode)
set design_setup(block_1,prefix)  block1_i1
source path/to/block_1_constraints.tcl

set design_setup(block_2,mode)    $design_setup(intermediate_3,mode)
set design_setup(block_2,prefix)  block2_i1
source path/to/block_2_constraints.tcl

# clock grouping
```

LOAD

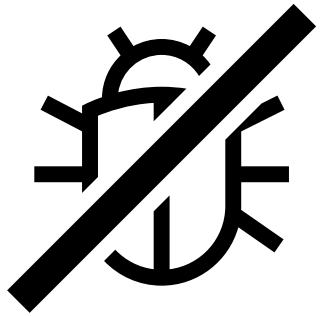
LOAD

Pros and Cons of Tcl Constraining

- Changes can be implemented fast
- Supports full set of constraints (also non-SDC)



- High effort for integration of 3rd party IPs
- Error-proneness increases
- Wildcarding in Tcl constraints dangerous and runtime-intense
- Clock grouping needs to be redone individually on each level



- Quality of Constraints for Safety (ISO26262)

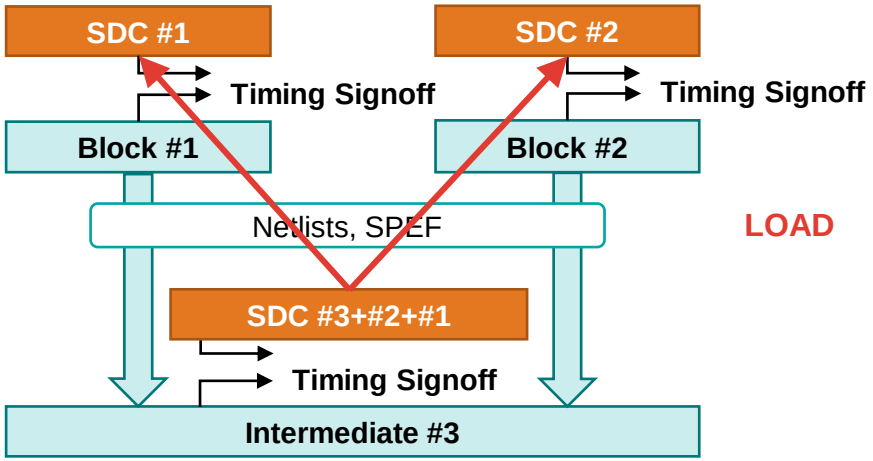
3. Solution with TCM

Implementation of TCM-based constraint promotion flow in RadarSoC

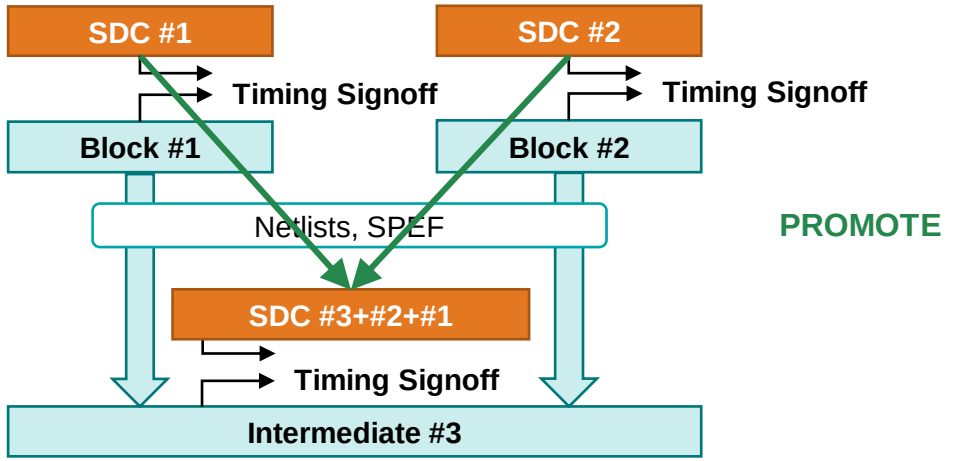
TCM Basics

Constraining Methodology

- Similar constraining methodology like for hierarchical STA.
- Block constraints are written individually without any preparation for integration.
- Intermediate- and top-level constraints are also written in the same way.
- For flat STA TCM is taking the netlists and SDCs of all blocks and promotes them to the top-level context, i.e. we end up with a merged static SDC.



Custom Tcl approach



TCM approach

TCM Basics

Design Environment

- Definition of designs and sub designs
  - Verilog files
  - Constraint modes
  - Constraint files (SDC, non-SDC) per mode
  - Liberty/DB files for macros and standard cells

- Configuration of promotion
  - Force pushing constraints to leaf pins instead of hierarchical pins?
  - What to do with block-internal clocks that are not promoting to a block‘s interface?
  - Do you want to promote case analysis settings?
  - Threshold for matching clock periods

TCM Basics

Design Environment | Design Definition

```
# List of blocks to exclude
set EXCLUDE_BLOCKS {}

# name of design
set design_name {rfe_sys_top}

# files containing netlists
set netlist_file {top.v sub_block/block1.v sub_block/block2.v}

# list of mode names
set mode_names {func shift atspeed}

# combinations of modes and top-level constraint files
set top_level_clocks_for_promotion {{func func.sdc} {shift shift.sdc} {atspeed atspeed.sdc}}

# list of combinations of blocks, modes, and constraint files
set block_mode_sdc_list {{block1 {func {tmp/block1_func.sdc}} {shift {tmp/block1_shift.sdc}} {atspeed {tmp/block1_atspeed.sdc}}}
                         {block2 {func {tmp/block2_func.sdc}} {shift {tmp/block2_shift.sdc}} {atspeed {tmp/block2_atspeed.sdc}}}}

# this variable determines whether
# html should be generated for block- and top-level setup runs so that
# view_fishtail_results may be invoked in those directories
set generate_html_for_setup_runs true

# split generated sdc files into header and body
set partition_generated_sdc_into_header_and_body false

# list of directories containing Verilog simulation models to represent standard cells (not memories or hard macros).
# the simulation models should instantiate Verilog primitives or udps; they should not be behavioral Verilog models
# unless the purpose of verification is to create assertions for functional simulation.
set simulation_file_dirs {/tools/gatelib/cmos22fdsoi/pdk-22FDX-EXT_V1.0_4.1_20230731/verilog
                          /tools/gatelib/cmos22fdsoi/pdk-22FDX-EXT_V1.0_4.1_20230731/verilog/3.0}

# list of directories containing Liberty (.lib) models for memories and hard macros
set lib_file_dirs {memory1.db}
```

Netlist files

Top-level constraints

Block-level constraints

Macros / Standard cells

TCM Basics

Design Environment | Promotion Configuration

```
# tcl variable definition overrides
set variable_definitions {
"## Mapping Variables"
  "#set case_analysis_sequential_propagation 0;          # determines whether constants propagate thru sequential
  "set move_hier_pins_to_leaf_pins 0;                    # in mapping, promotion, and demotion runs, write constra
  "#..."


"## Constraint Promotion Variables"
  "set clock_mapping_period_threshold 5;                 # tolerance value (as a percentage) for whether a warning
  "set create_async_clock_groups 1;                      # create an async clock group that specifies that clocks
  "set create_clocks_on_hier_pins 0;                     # when 0, if a divide-by 1 generated clock cannot be move
  "set keep_unpropagated_clocks 1;                       # in promote-without-push, preserve create_clock commands
  "set promote_case_analysis 1;                          # when 0, block-level set_case_analysis commands are ignc
  "set promote_clock_attributes 0;                       # when 0, set_clock_latency and set_clock_uncertainty cor
  "set promote_clock_groups 1;                           # when 1, clock groups and clock-to-clock false paths are
  "set promote_interface_exceptions 1;                   # when 1, exceptions that are specified on primary inputs

"## Equivalence Checking Variables"
  "#set compare_clock_uncertainty 0;                     # when 1, clock uncertainty and clock latency are conside
  "#set compare_internal_timing 1;                       # when 0, during top-vs-block equivalence checking, do nc
  "#set compare_io_timing 1;                             # when 0, during top-vs-block equivalence checking, only
  "#set optimism_threshold 0.1;                          # minimum optimism difference in timing requirement for p
  "#set pessimism_threshold 0.1;                         # minimum pessimism difference in timing requirement for
  "set report_path_timing_requirement 1;                 # compare both endpoints and paths during equivalence che
}
```

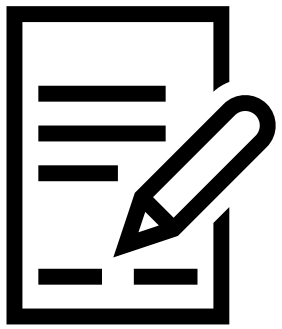
Mapping Configuration

Promotion Configuration

Equivalence Check Configuration

TCM Basics

Promotion Flow



- Change ports with pins, -from/to with -through etc.
- Update prefixes of objects accordingly.

```
set_false_path -from [get_ports scan_enable]
```

Prefix update

```
set_false_path -through [get_pins path/to/instance_1/scan_enable]
```

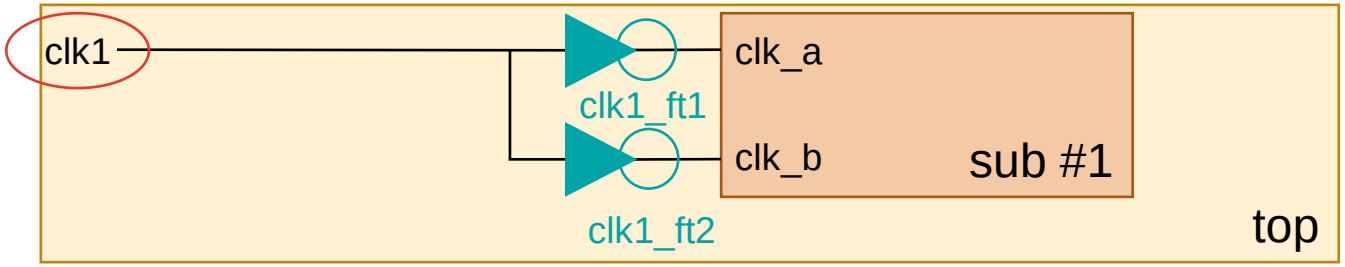
- Remove non-applicable constraints (e.g. I/O delays)
- Promote case settings, min/max delays, data checks, …
- Special handling of clocks, see next slide

- Check for:
  - case conflicts
  - clock period matches
  - …

TCM Basics

Promotion Flow | Clock handling

- Root clocks at block interfaces are replaced by generated clocks
- For each clock input, a generated clock is generated on the leaf driver:



- The clock grouping for those generated clocks is taken from block-level SDC:

```
set_clock_groups -logically_exclusive
                 -group clk_a
                 -group clk_b
```

Block-level SDC to promoted SDC

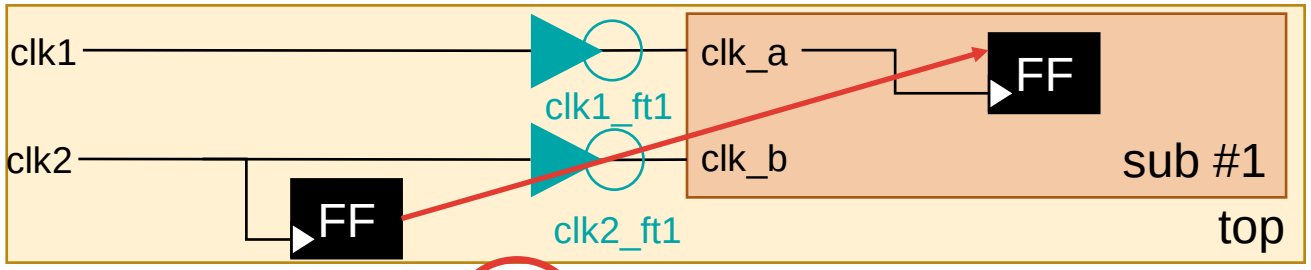
```
set_clock_groups -logically_exclusive
                 -group clk1_ft1
                 -group clk1_ft2
```

- **Please note:** Clock mapping between block and top is configurable!

TCM Basics

Promotion Flow | Clock handling

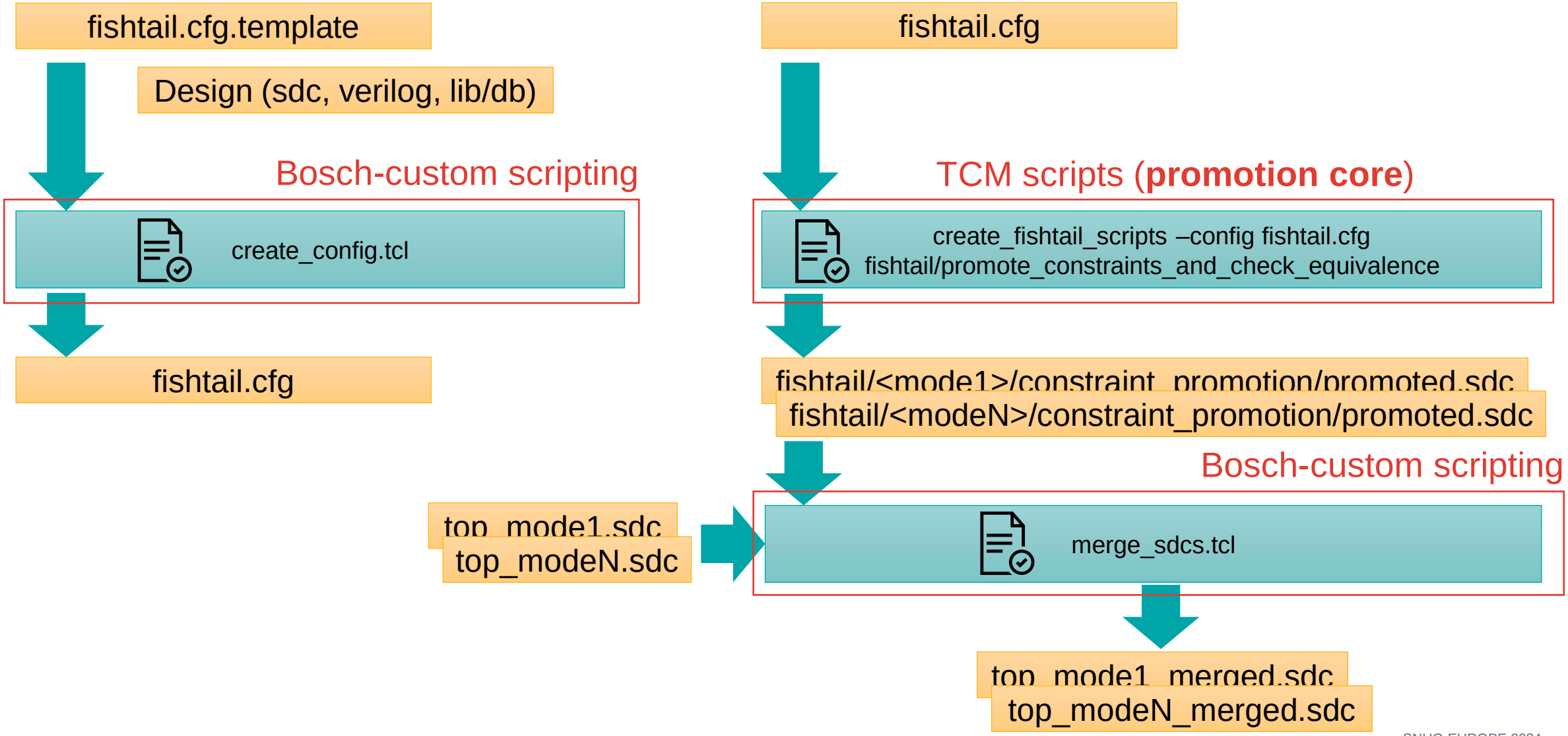
- Paths from root or block-level clocks to asynchronous block-level clocks of other blocks is not implicitly set to false:



Pitfall: clk1 is asynchronous to clk2,
but clk1_ft* are not implicitly asynchronous to clk2!

- TCM supports runtime-intense second promotion run for further clock group investigation to put a false path also on those crossing paths. (not default)

Flow Implementation



Promotion Results

Database of the design, can be used for debugging purposes

```
RH7[414] ls `pwd`

fishtail/func/constraint_promotion:
../                             ./
check_block_list.log            clock_mapping.tcl
design.rdb                      design_setup_issues.html
design_setup_issues_files/      lite_mem.log
map_mem.log                     map_sdc.log*
mapping.rpt                     promote_constraints*
promoted.sdc                    refocus_interactive.tcl
refocus_mem.log                 report_setup_issues.log*
report_setup_issues.tcl         rmodules.rpt.gz
```

HTML Reports

Script that executes the constraint promotion

Promoted Constraints

```
# SDC file generated by FishTail Refocus version U-2022.12-SP3

set_units -time ns
set remove_extraneous_combinational_clocks 1

# set_case_analysis on
                    /tmp/mode_func_0.tcl line 26 for instance
set_case_analysis \
   1 \
   {                                                }

# set_case_analysis on
                    tmp/mode_func_0.tcl line 38 for instance
set_case_analysis \
   0 \
   {                                                }
```

Promoted constraints file documents origin of each constraint

Promotion Reports

HTML Reports

Design Setup Summary for rfe_sys_top

Design Information

Constraint Mapping

Clock Mapping

Promoted Constraints

Issues

Total Runtime: 1:23:02

Memory Usage: 42.1 gB

Promotion Reports

HTML Reports

Design Setup Summary for rfe_sys_top

Design Information

Constraint Mapping

Clock Mapping

Promoted Constraints

Issues

Total Runtime: 1:23:02

Memory Usage: 42.1 gB

Constraint Mapping Reports

Mapped Constraints for module [redacted] **(instance** [redacted] **)**

Name	Mapped	Unmapped	Waived
Unapplied	0	1	0
create_clock	7	0	0
create_clock (virtual)	1	0	0
create_generated_clock	21	0	0
group_path	12	0	0
set_case_analysis	726	0	0
set_clock_groups	1	0	0
set_disable_timing	5012	0	0
set_dont_touch	0	403	0
set_false_path	21	0	0
set_false_path (hold)	3	0	0
set_max_delay	2	0	0
set_min_delay	1	0	0
set_multicycle_path (hold)	4	0	0
set_multicycle_path (setup)	4	0	0
set_sense	53	0	0

Promotion Reports

HTML Reports

Design Setup Summary for rfe_sys_top

Design Information

Constraint Mapping

Clock Mapping

Promoted Constraints

Issues

Total Runtime: 1:23:02

Memory Usage: 42.1 gB

Clock Mapping Reports

Clock Report for Module rfe_sys_cpll

Clock Name	Phase Shift	High Pulse	Low Pulse	Period	Master Clock
	0.000	5.600	5.600	11.200	
	0.000	6.250	6.250	12.500	
	0.000	10.000	10.000	20.000	
	0.000	10.000	10.000	20.000	
	0.000	6.250	6.250	12.500	
refclk_ft_1	0.000	6.250	6.250	12.500	refclk
	0.000	1.120	1.120	2.240	
	0.000	1.120	1.120	2.240	
	0.000	1.120	1.120	2.240	
	0.000	0.390	0.390	0.780	
	0.000	0.390	0.390	0.780	

Constraint Mapping Reports

Mapped Constraints for module **(instance)**

Name	Mapped	Unmapped	Waived
Unapplied	0	1	0
create_clock	7	0	0
create_clock (virtual)	1	0	0
create_generated_clock	21	0	0
group_path	12	0	0
set_case_analysis	726	0	0
set_clock_groups	1	0	0
set_disable_timing	5012	0	0
set_dont_touch	0	403	0
set_false_path	21	0	0
set_false_path (hold)	3	0	0
set_max_delay	2	0	0
set_min_delay	1	0	0
set_multicycle_path (hold)	4	0	0
set_multicycle_path (setup)	4	0	0
set_sense	53	0	0

Promotion Reports

HTML Reports

Design Setup Summary for rfe_sys_top

Design Information

Constraint Mapping

Clock Mapping

Promoted Constraints

Issues

Total Runtime: 1:23:02

Memory Usage: 42.1 gB

Clock Mapping Reports

Clock Report for Module rfe_sys_cpll

Clock Name	Phase Shift	High Pulse	Low Pulse	Period	Master Clock
	0.000	5.600	5.600	11.200	
	0.000	6.250	6.250	12.500	
	0.000	10.000	10.000	20.000	
	0.000	10.000	10.000	20.000	
	0.000	6.250	6.250	12.500	
refclk_ft_1	0.000	6.250	6.250	12.500	refclk
	0.000	1.120	1.120	2.240	
	0.000	1.120	1.120	2.240	
	0.000	1.120	1.120	2.240	
	0.000	0.390	0.390	0.780	
	0.000	0.390	0.390	0.780	

Constraint Mapping Reports

Mapped Constraints for module **(instance)**

Name	Mapped	Unmapped	Waived
Unapplied	0	1	0
create_clock	7	0	0
create_clock (virtual)	1	0	0
create_generated_clock	21	0	0
group_path	12	0	0
set_case_analysis	726	0	0
set_clock_groups	1	0	0
set_disable_timing	5012	0	0
set_dont_touch	0	403	0
set_false_path	21	0	0
set_false_path (hold)	3	0	0
set_max_delay	2	0	0
set_min_delay	1	0	0
set_multicycle_path (hold)	4	0	0
set_multicycle_path (setup)	4	0	0
set_sense	53	0	0

Clock Mapping Editor

Clock Mapping for

Apply all tool clock mappings

Reset all tool clock mappings

Block Clock	Definition Pin	Propagated Top-Level Clocks	Mapped Top-level Clock	User Defined Clock Mapping
blc_clk	blc_clk	clk_rfe	clk_rfe	☑ blc_clk ☐ clk_rfe

Promotion Reports

HTML Issue Overview

Issues

- Setup Issues
- SDC Issues
- Exception Issues
- Promotion Issues

Setup Issues

Issues	Severity	Message Count
Parser Issues		
Design Issues		
Design object does not exist. (RTL-002)		492
Module has no definition (RTL-004)		32

RTL-004 (32 messages)

```
Warning: Module '                ' has no definition. (RTL-004)
Warning: Module '                ' has no definition. (RTL-004)
Warning: Module '                ' has no definition. (RTL-004)
```

Promotion Reports

HTML Issue Overview

Issues

Setup Issues

SDC Issues

Exception Issues

Promotion Issues

SDC Issues

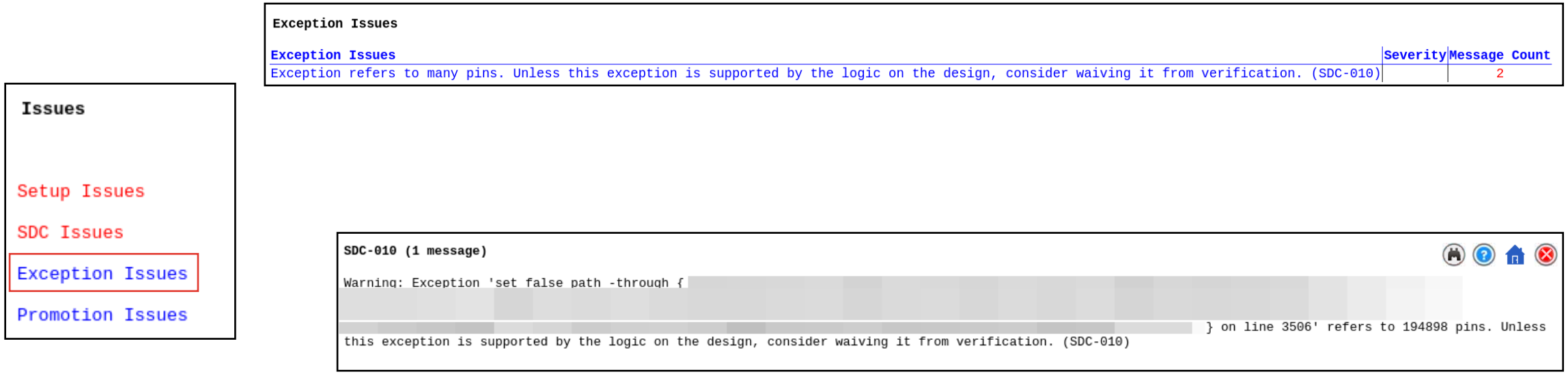
SDC Issues	Severity	Message Count
SDC command could not be applied. (SDC-038)		406
The command is not supported. (SDC-059)		538

SDC-038 (405 messages)

```
Warning: SDC command 'set_dont_touch [get_lib_cells {                                                    }] false'
     on line 594 of file                                                                                 /design.tcl
     could not be applied. (SDC-038)

Warning: SDC command 'set_dont_touch [get_lib_cells {                                                    }] false'
     on line 595 of file                                                                                 /design.tcl
     could not be applied. (SDC-038)
```

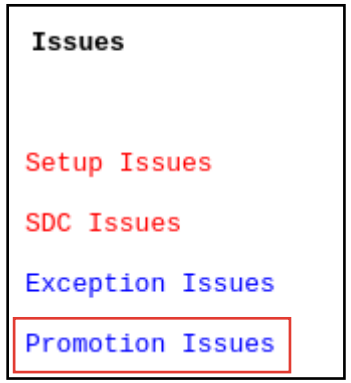
Promotion Reports

HTML Issue Overview



Promotion Reports

HTML Issue Overview



Promotion Issues

Promotion Issues	Severity	Message Count
Multiple top-level clocks propagate to a block-level clock. All but one of them are dropped. (REFOCUS-064)		1
The period of the top-level clock is different from the block-level clock that it is mapped to. (REFOCUS-070)		2

REFOCUS-064 (1 message)

Warning: Multiple top-level clocks propagate to clock '[redacted]' defined on instance '[redacted]' of block '[redacted]'. The following top-level clocks are dropped. (REFOCUS-064)

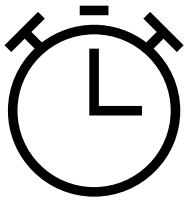
4. Results

Benefits and drawbacks of custom flow vs. TCM

Results

- Application of TCM in the latest Bosch RadarSoC tapeout
  - GlobalFoundries 22FDX technology
  - ~10 million leaf cells in total (combo, sequential, memories)
  - ~40 root clocks, ~100 generated clocks in TCM-generated constraints
  - Multiple AMS macros with nested digital blocks (individually hardened)
  - Complex 3rd party IP integration
  - 3 modes @ 40 corners (120 timing scenarios to signoff)

- Constraint promotion is run automatically within AMS macro preparation and at the end of top-level synthesis subsequently.



~5h total promotion runtime for top-level

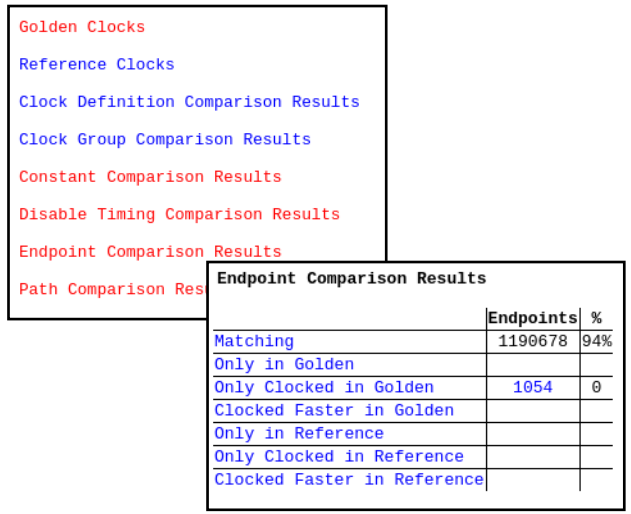
~2h total promotion runtime for largest AMS block

~7h worst-case runtime when fixing a constraint in an AMS sub block

Results

- Manual Review of promoted.sdc for critical items:
  - set_max_delay –from/-to from block was neither promoted with –through nor with –probe (non-SDC) so it would split data paths which could be wrong.
  - set_max_transition/set_max_capacitance –force was ignored, but did not show up in ignored commands.
  - No further major issues found.

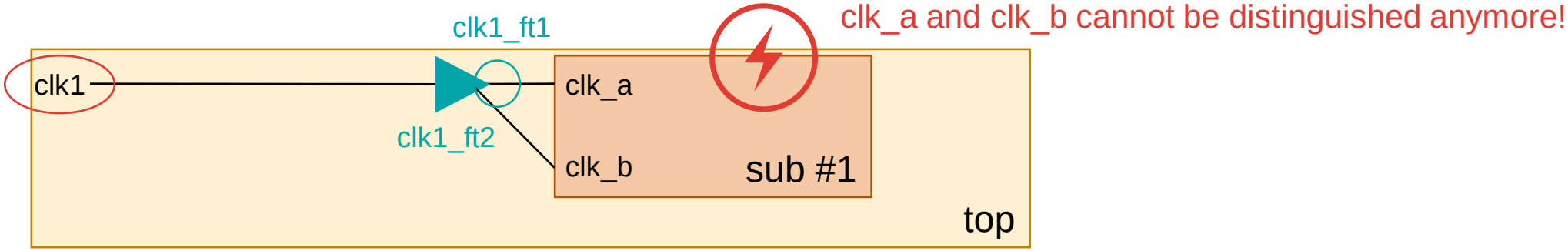
- Validation with TCM:
  - Reviewing all reported issues and promotion/mapping reports. No severe issues found.
  - Equivalence check reports of TCM



- Validation with GCA:
  - Lots of waivable errors caused by promoted constraints (incomplete clock grouping,…)

Pitfalls

- Need for an individual driver on each clock input caused issues:
  - Missing individual drivers on test clocks were found late in the flow, promotion based on synthesis netlist was not feasible anymore as it broke the clock propagation in Primetime.

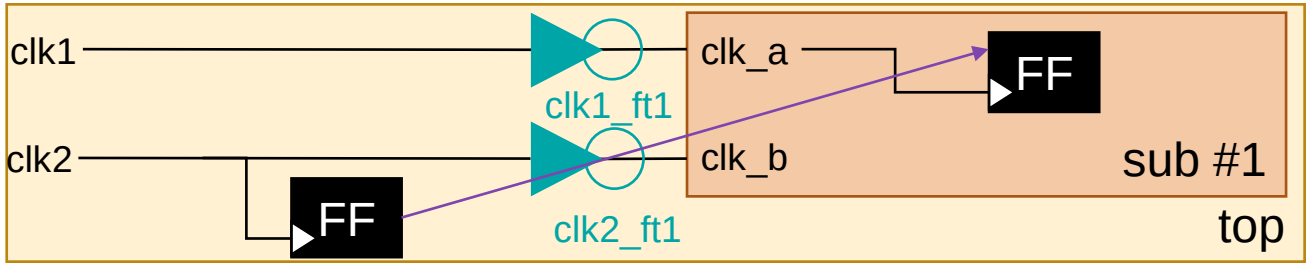


  - **Workaround:** Doing promotion based on layout netlist as we had individual drivers there.

- Each generated clock maps to one top-level clock – could be an issue to cover timing scenarios properly.

Pitfalls

- False timing violations because of paths between top- and block-level clocks:
  - Even though clk1/clk2 are defined logically exclusive as well as clk_a/clk_b and clk1_ft1/clk2_ft1 there is for safety reasons no implicit logical exclusivity between clk1 and clk2_ft1!



  - TCM supports runtime-intense second promotion run for further clock group investigation
  - Alternative: Use TCM output as baseline and add false paths / clock groups as needed
  - In Radar SoC we used a mixture of both:
    The second promotion run covered most of the issues, the rest was fixed by a patch script.

Benefits and Drawbacks

TCM vs. Custom Tcl-based Constraining

- Benefits of TCM:
  - Safer approach, generated SDCs can be reviewed.
  - No impact on block-level hardening, proper work split between block-level and top-level impl.
  - Enables 3rd party IP integration.
- Drawbacks of TCM:
  - Design requirements (individual drivers)
  - Runtime significantly higher, debugging turnaround times usually higher
  - TCM is a blackbox tool, while for Tcl it‘s WYSIWYG (ignoring script side effects)
    - set_max_cap/trans were just ignored…

- Benefits of Tcl-based Constraining:
  - Faster turnaround times when ramping up and debugging flat constraints
    - Adaption of one Tcl line in a sub block does not cause multiple runtime-intense promotion steps.
  - Not necessarily causes design requirements such as explicit buffers on clock inputs
  - Number of generated clocks needed is lower
- Drawbacks of Tcl-based Constraining:
  - Script side effects in dynamic interpretation
  - No isolated block-level constraint development
  - Need for use of many wildcards.
  - 3rd party IP integration difficult


THANK YOU
YOUR
INNOVATION
YOUR
COMMUNITY