End-to-End SDC Constraints Methodology

Earlier, Easier and more Accurate with Timing Constraints Manager

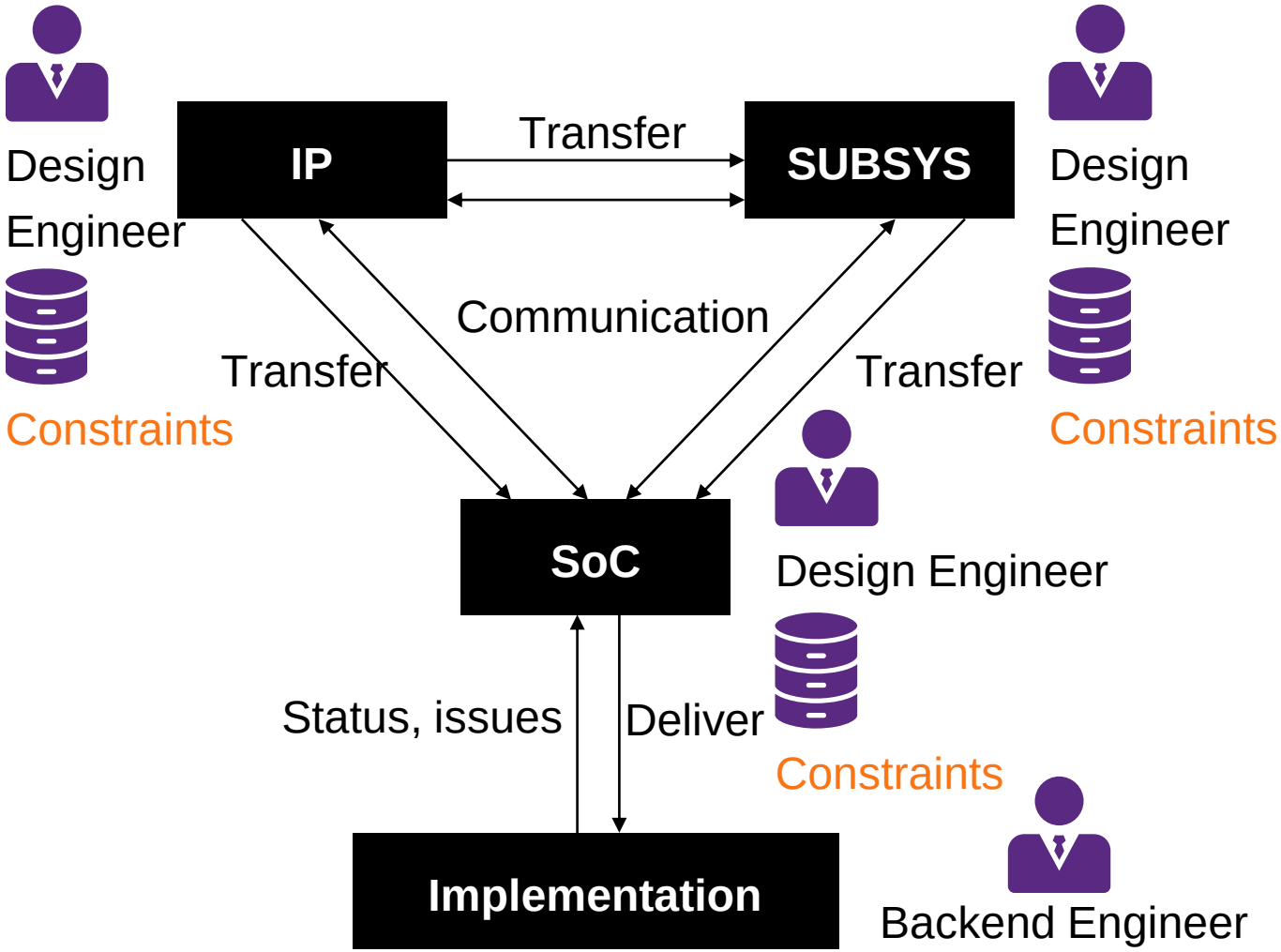
Xin An, Christian Heinrich, Yifang Wang, Infineon
James Gillespie, Synopsys

Agenda

- Challenges with the Bottom-up Constraints Methodology
- Enhancing Current Methodology with Synopsys Timing Constraints Manager
- Conclusion

Challenges with the Bottom-up Constraints Methodology

Bottom-up Constraints Methodology



Benefits

- **Portable**: Timing info is portable between engineers to lower resource risk
- **Reusable**: Subsystem or SoC level can remove or add IP constraints
- **Time2Market**: Constraints come along with IP and are prepared for top level in early stage. It relaxes project schedule
- **High quality**: IP level design engineer provides the detailed and correct timing constraints

Challenges with Current Methodology

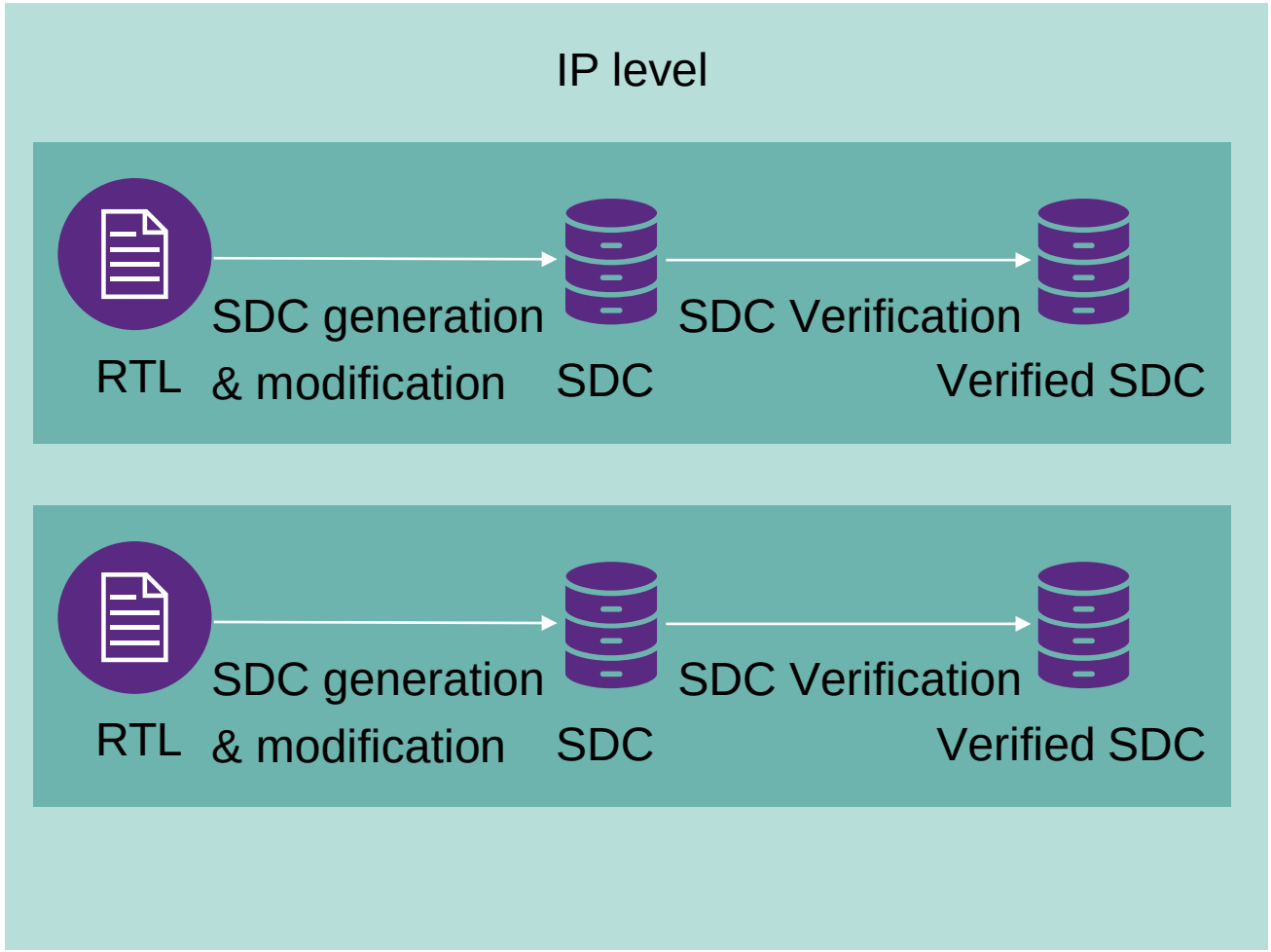
- How can IP level design engineer write constraints with higher quality more efficiently
  - Manual creation of constraints is time consuming and error prone
  - How to know all the required constraints are correctly defined and ready for handoff to SoC
- How to propagate IP constraints to top level constraints
  - Manual promotion is complex, time consuming and error prone
  - The generics, parameters and top level modes play an important role for constraints propagation
  - 3rd party IP constraints are especially challenging to integrate (different SDC styles, unfamiliarity with IP)
- How to check the constraints are correct
  - Under-constraining or over-constraining leads to unoptimized implementation
  - Timing exceptions (MCP/FP) are especially risky and error prone
  - Missing or incorrect constraints can easily lead to silicon failure

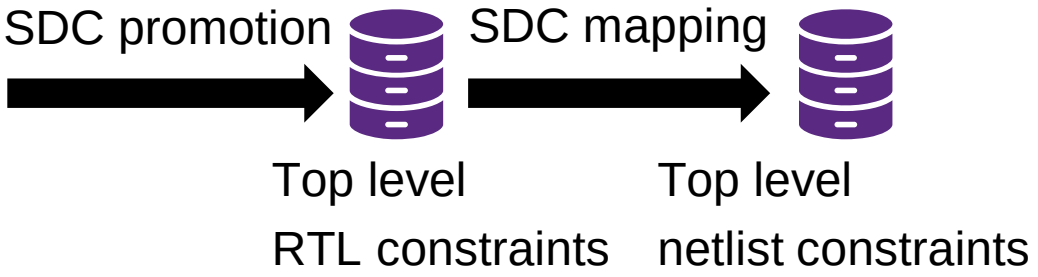
Enhancing Current Methodology with Synopsys Timing Constraints Manager (TCM)

Timing constraints manager features

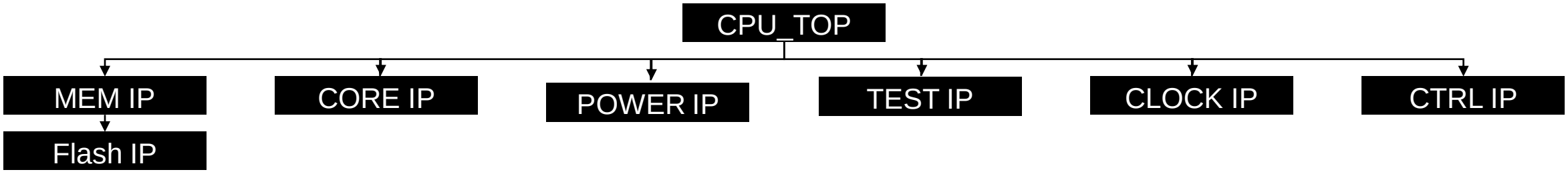
- SDC generation & modification
  - IP level constraints are initially generated by TCM and modified by IP level design engineers
  - IP level design engineers can prepare constraints with minimized efforts
- SDC verification
  - IP level constraints quality can be improved by TCM SDC verification feature
  - The quality of SoC level constraints can be signed-off before they are released to backend
- SDC promotion & demotion
  - IP and SoC constraints can be merged automatically with reduced efforts
- SDC mapping
  - RTL constraints can be mapped to the netlist hierarchy
  - Mapped constraints are compatible with implementation tools

Design Development Methodology with TCM



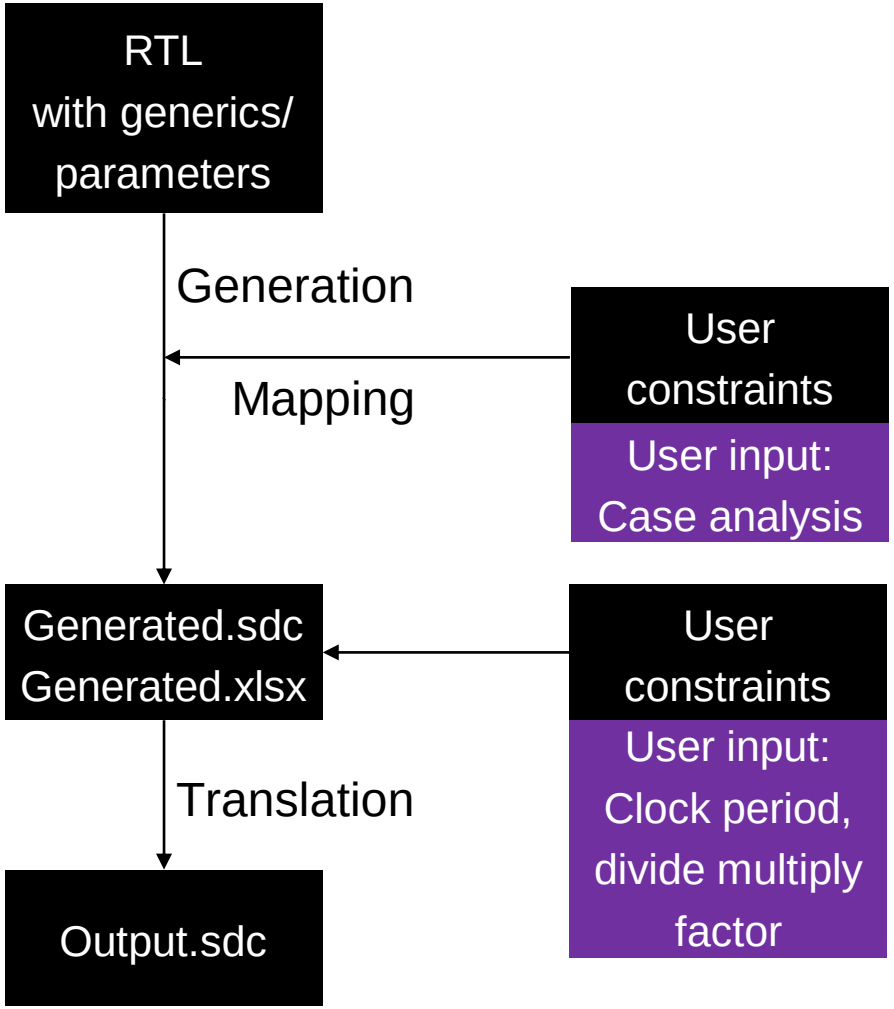


TCM evaluation on CPU_TOP



Flow IP	Flash IP	MEM IP	CORE IP	POWER IP	TEST IP	CLOCK IP	CTRL IP	CPU_ TOP
Generation+Modification	Y	Y	Y	Y	Y	Y	Y	
Promotion		Y						Y
Equivalence check (Promoted vs IP)		Y						Y
Validation (Lint & MCP)			Y					Y
RTL2Netlist Mapping								Y

TCM: Constraints generation and modification



Features

- Clock definition
  - The principle for identifying the clock is based on tracing the path backwards from the clock pin of a Flip-Flop
  - User defined clock definitions can optionally be provided as input
  - The relationship of generated clock with master clock shall be maintained by users
- Generics / Parameters can be specified in source file list
- Constraints can be managed using Excel
  - Converter from SDC to XLS, XLS to SDC
- Clock visualisation allows for easy review of clock propagation and relevant control logic

TCM: Constraints XLS format

- Clocks definition in XLS (Constraints generation)

Delete	Name	Pin	Period (ns)	Min Input Delay Budget	Max Input Delay Budget	Min Output Delay Budget	Max Output Delay Budget	Virtual Clock	Comment
	sx_dft_scan_clk	sx_dft.scan_clk	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_dft_scan_clk	
	sx_pmu2scu_ad_reg5v_tm_clk	sx_pmu2scu.ad_reg5v_tm_clk	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_pmu2scu_ad_reg5v_tm_clk	
	sx_pmu2scu_mclk	sx_pmu2scu.mclk	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_pmu2scu_mclk	
	sx_pmu2scu_refclk	sx_pmu2scu.refclk	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_pmu2scu_refclk	
	sx_scu2tcu_test_clk	sx_scu2tcu.test_clk	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_scu2tcu_test_clk	
	sx_srcclk_clksrc_ext_0_	sx_srcclk.clksrc_ext[0]	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_srcclk_clksrc_ext_0_	
	sx_srcclk_clksrc_ext_1_	sx_srcclk.clksrc_ext[1]	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_srcclk_clksrc_ext_1_	
	sx_srcclk_clksrc_int_0_	sx_srcclk.clksrc_int[0]	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_srcclk_clksrc_int_0_	
	sx_srcclk_clksrc_int_1_	sx_srcclk.clksrc_int[1]	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_srcclk_clksrc_int_1_	
	sx_srcclk_clksrc_int_2_	sx_srcclk.clksrc_int[2]	10,000	50,0%	50,0%	50,0%	50,0%	virtual_sx_srcclk_clksrc_int_2_	

- Generated clocks definition in XLS (If the clock signal does not propagate to any Flip-Flop within the system, it won't be recognized)

Delete	Name	Pin	Period (ns)	(M)ultiply (D)ivide (E)dges	Other Options	Source Pin	Master Clock	Min Input Delay Budget	Max Input Delay Budget	Min Output Delay Budget	Max Output Delay Budget	Virtual Clock	Comment
0	sx_clk_fsys0_clk_1	i_inst/tck_retime_reg/Q	20.000	D 2		sx_clk.fsys0_clk	sx_clk_fsys0_clk	50.0%	50.0%	50.0%	50.0%		
0	sx_paddft_spclk_1	i_inst/tck_retime_reg/Q	20.000	E 2 4 6		sx_paddft.spclk	sx_paddft_spclk	50.0%	50.0%	50.0%	50.0%		

- Timing exception, case analysis, clock groups, etc

TCM: Constraints .sdc

- Variable definitions: Collection of project-dependent variables

```
 7 # Variable Definitions
 8 variable ft_setup_uncertainty_factor 0.10
 9 variable ft_hold_uncertainty_factor 0.00
10 variable ft_input_delay_factor 0.30
11 variable ft_output_delay_factor 0.50
12 set FT_CLK_ATTR(cpu_clk:period) { 25 }
13 set FT_CLK_ATTR(hclk:period) { 25 }
14 set FT_CLK_ATTR(sx_dft_scan_clk:period) { 100 }
15 set FT_CLK_ATTR(sx_pmu2scu_ad_reg5v_tm_clk:period) { 50 }
16 set FT_CLK_ATTR(sx_pmu2scu_mclk:period) { 50 }
17 set FT_CLK_ATTR(sx_pmu2scu_refclk:period) { 10000 }
```

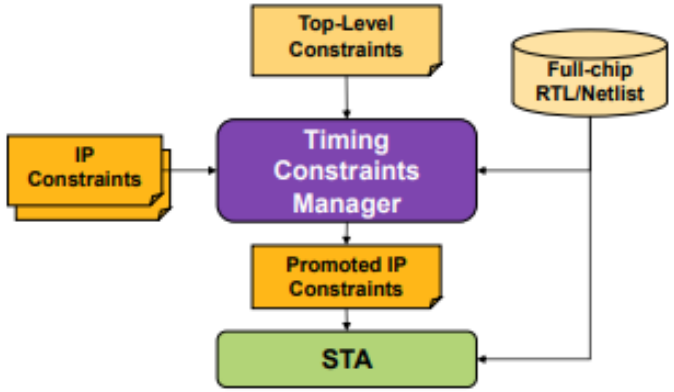
- Generated clock definitions

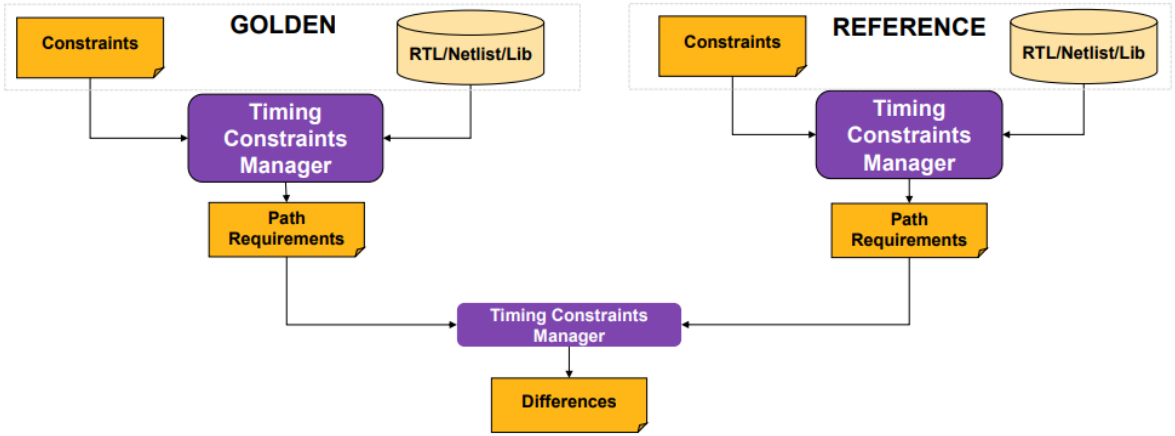
```
174 create_clock -name sx_pmu2scu_mclk -period $::fishtail::FT_CLK_ATTR(sx_pmu2scu_mclk:period) [get_ports sx_pmu2scu.mclk]
175 set_clock_uncertainty -setup $::fishtail::FT_CLK_ATTR(sx_pmu2scu_mclk:setup_uncertainty) [get_clocks sx_pmu2scu_mclk]
176 set_clock_uncertainty -hold $::fishtail::FT_CLK_ATTR(sx_pmu2scu_mclk:hold_uncertainty) [get_clocks sx_pmu2scu_mclk]
177 create_clock -name virtual_sx_pmu2scu_mclk -period $::fishtail::FT_CLK_ATTR(sx_pmu2scu_mclk:period)
```

- Mapped clock definitions

```
195 create_clock -name sx_pmu2scu_mclk -period $FT_CLK_ATTR(sx_pmu2scu_mclk:period) sx_pmu2scu.mclk
196 create_clock -name virtual_sx_pmu2scu_mclk -period $FT_CLK_ATTR(sx_pmu2scu_mclk:period)
```

TCM: Constraints promotion & equivalence check





Features

- SDC promotion
  - Propagates standalone IP level constraints to top level
  - Multimode (mission, scan etc.) constraints promotion in parallel is available
  - 3rd party IP constraints can also be promoted
- SDC equivalence check
  - Ensures integrity of constraints transformation
  - Issues are captured in early phase
  - Conflict between two clock constraints is captured. Quality is improved

TCM: Constraints promotion & equivalence check

- SDC promotion: Clock period is reported as conflicts between IP level and top level. The messages benefit design engineers to analyse the issues.

Waive this message
Warning: The period of the top-level clock 'hclk' is different from the block-level clock 'sysclk' that it is mapped to on instance ' nvm_top_i' of block '74k_bst_top'. (REFOCUS-070)'(Top-to-Block Clock Mapping)

- SDC equivalence check: original IP constraints vs. promoted IP constraints. Design engineers manage the constraints with GUI easily.

Waive	Definition point	Golden Clock	Period	Reference Clock	Period	Block
	i_nvm_top/gen_nvm/fsm_rd_asbuf_s_reg/Q	fsm_rd_asbuf_x_clk	47.619	fsm_rd_asbuf_x_clk	50.000	mem_top_top
	i_nvm_top/gen_nvm/fsm_rd_phase0_o_reg/Q	fsm_rd_phase0_x_clk	47.619	fsm_rd_phase0_x_clk	50.000	mem_top_top
	i_nvm_top/gen_nvm/fsm_rd_phase1_o_reg/Q	fsm_rd_phase1_x_clk	47.619	fsm_rd_phase1_x_clk	50.000	mem_top_top
	i_nvm_top/gen_nvm/fsm_rd_phase2_o_reg/Q	fsm_rd_phase2_x_clk	47.619	fsm_rd_phase2_x_clk	50.000	mem_top_top
	i_nvm_top/gen_nvm/fsm_rd_phase3_s_reg/Q	fsm_rd_phase3_x_clk	47.619	fsm_rd_phase3_x_clk	50.000	mem_top_top
	i_nvm_top/gen_nvm/fsm_wr_asbuf_s_reg/Q	fsm_wr_asbuf_x_clk	47.619	fsm_wr_asbuf_x_clk	50.000	mem_top_top
	i_nvm_top/gen_nvm/sysclk_x_i	sysclk_x	23.810	nvm_clk_ft_sysclk_x_ft_I_gen_nvm_gen_nvm_nvm_top_i	25.000	mem_top_top

TCM: Formal SDC Lint

Clock constraints

- Clock definition
  - Unclocked registers
  - Missing clock definitions
- Generated clock definition
  - Missing generated clocks
  - Incorrect edge specification
  - No combinational path from master clock to generated clock
- Clock Propagation
  - Clock reconvergence
  - False paths in clock propagation network
- Clock Crossing
  - Missing or incorrect clock groups, clock-to-clock false paths

I/O delays and Exceptions

- Case Analysis
  - Conflicting case analysis
- Exception Issues
  - Setup MCPs with missing hold MCP
  - Overlapping timing exceptions
  - Timing exceptions that apply to large numbers of paths
- Input Output Delays
  - Missing I/O constraints

TCM: Multi-Cycle Path Verification

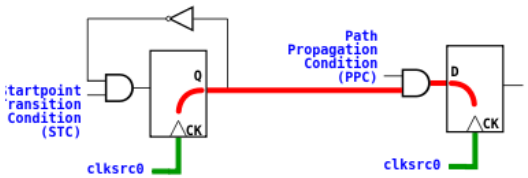
- Formal engine analyses MCP
- If MCP failed, SVA assertion is generated
- Functional regression can integrate SVA assertion for checking the MCP during simulation
- TCM does not perform MCP verification on the hold path. Instead, it checks whether the number of clock cycles on the hold path is less than the number of clock cycles on the setup path

TCM: MCP verification GUI

Launch Clock Information
Name: clksrc0
Edge: Rise
Time: 100.000
Pin: clk_i

Capture Clock Information
Name: clksrc0
Edge: Rise
Time: 125.000
Pin: sx_clk.cpu_clk

Startpoint: crcctrl_norm_do_reg[3:0]
The startpoint is static
Endpoint: result_reg[*]
Failures to this endpoint can be ignored
Failures between this start/end pair can be ignored
The endpoint is allowed to go metastable



Ignore STC
Ignore PPC

Failure State: (=)

The path is not multi-cycle because when the startpoint transition condition is true, then in the next clock cycle it is possible for the path propagation condition to also be true. As a result, a transition on the startpoint will propagate to the endpoint in the same clock cycle.

Clear highlighting | Show All | Show STC | Show PPC | Show Combinational Signals

View Waveform

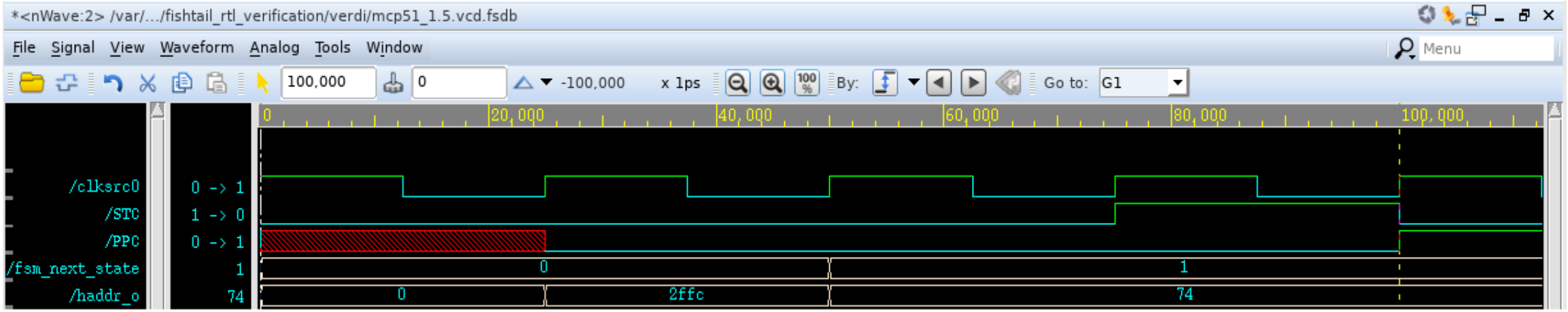
Hide	Ignore	Static	Constant 0	Constant 1	Illegal Stimulus / Failure Stimulus	0.000	25.000	50.000	75.000	100.000
-					clksrc0	r	r	r	r	r
-					STC	0	0	0	1	0
-					PPC		0	0	0	1
-					fsm_next_state[1:0] (1 0) (= =)	'00'	'00'	'01'	'01'	'01'
-					haddr_o[13:0] (13 12 11 10 9 8 7 6 5 4 3 2 1 0) (= = = = = = = = = = = = = =)	'00000000000000'	'10111111111100'	'00000001110100'	'00000001110100'	'00000001110100'

- How does the tool find the failure?
  - Propagation condition from startpoint to endpoint (PPC) is true one clock cycle after the startpoint transitions(STC)
- Add waivers
  - Startpoint is static or is held stable for multiple cycles
  - Endpoint is allowed to go metastable

TCM: MCP verification debug via VERDI

- Waveform debugging

Hide	Ignore	Static	Constant 0	Constant 1	Illegal Stimulus / Failure Stimulus	0.000	25.000	50.000	75.000	100.000
-					clksrc0	⌜	⌜	⌜	⌜	⌜
-					STC	0	0	0	1	0
-					PPC		0	0	0	1
-					fsm_next_state[1:0] (1 0) (= =)	'00'	'00'	'01'	'01'	'01'
-					haddr_o[13:0] (13 12 11 10 9 8 7 6 5 4 3 2 1 0) (= = = = = = = = = = = = = =)	'00000000000000'	'10111111111100'	'00000001110100'	'00000001110100'	'00000001110100'



TCM: RTL constraints to netlist constraints mapping

- Ports name mapping: sx_paddft.spclk (rtl) -> sx_paddft_spclk (netlist)

```
create_clock \
  -waveform { 0 5 } \
  -period 10 \
   [get_ports { sx_paddft_spclk }] \
  -name spclk
```

Conclusions

Summary

- TCM supports design engineers with varying constraints skills to build comprehensive and correct constraints by **verifying, generating, and managing** SDC constraints

- The multiple functions of TCM can be **independently** utilized, offering **flexibility** tailored to the specific design development methodology

- **Earlier, Easier and more Accurate** timing constraints with timing constraints manager

Outlook

- Seamless integration through scripting
  - Within the specific design methodology, TCM varying functions can be seamlessly converged through scripting
  - This harmonized approach empowers efficient and cohesive design flow

- Enhance TCM performance
  - Long runtime for the SDC promotion and verification can be improved
    - SDC promotion (6 IP constraints with 4 modes to CPU_TOP): Runtime 10 minutes
    - SDC verification (lint, MCP, FP verification) for CPU_TOP (gate count: 44k) under function mode: Runtime 30 minutes
  - The optimized performance improves the operation efficiency


THANK YOU
YOUR
INNOVATION
YOUR
COMMUNITY