INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

IJARIIT

ISSN: 2454-132X

Impact factor: 4.295

(Volume3, Issue3)

Available online at www.ijariit.com

Efficient Timing Closure in SOC through Timing Quality Checks and Engineering Change Order

L. Shanthala
PG Scholar, Department of Electronics and
Communication Engineering
BMS College of Engineering, Bangalore
shanthala.l.bhat@gmail.com

Dr. R. Jayagowri
Associate Professor, Department of Electronics and
Communication Engineering
BMS College of Engineering, Bangalore
rjayagowri.ece@bmsce.ac.in

Abstract: Ensuring correct operation of design despite rising levels of design complexity has been a major focus of research and development since the dawn of digital system design. As design size and complexity increase so is the need to verify designs quickly and reliably. This, combined with the reduced design cycle of 3-6 months, makes verification an extremely challenging task. Once the chip is taped out, if the chip is not meeting timing or if the chip is consuming too much power then it is a big issue and results in a huge loss. So in order to avoid these post-silicon surprises, preliminary signoff checks are a must. This paper covers the various kinds of timing quality checks that are used in the industry including the checks related to connectivity, clock, Max slope and Max capacitance, checks for Latch etc. Also in advanced technology nodes, one has to consider the effect of PVT variation, temperature inversion, noise effect on delay, which is adding more scenarios for STA to cover. With big SOC project, one timing ECO-run normally takes 3~5 days. The turn-around-time for timing ECO and timing signoff has become a bottleneck in the later stage of the project. So there is an immense need for effective implementation of ECO with new technologies including physically aware ECO with less resource, reduced design cycle time and reduced manual effort. The proposed algorithm is tested on multiple industrial designs and found to achieve good improvement in terms of Worst Negative Slack, Total Negative Slack and Failing End Points. Also, the algorithm is physically aware meaning that the placement blockages, congestions are considered while inserting buffers. The algorithm works under Distributed Multi Scenarios Analysis (DMSA) environment and considers the effect of ECO across multiple corners and modes.

***Keywords**: Data Path Optimization, Distributed Multi-Scenario Analysis, Engineering Change Order Eco, Physically Aware, Timing Quality Checks*

I. INTRODUCTION

Quality checks are the collective name given to a series of verification steps that must pass before the design can be taped out. Quality check tools contain set of rules written in a particular scripting language. Choice of scripting language is generally decided based on the base language of the tool (Eg. tickle, Perl). So the main motto of using these quality check tool would be to analyse and debug the quality check violations and clean the design. The tool will include set of quality checks for clocking, connectivity, timing, DRC etc. There will be some standard which is known as "golden" or signoff-quality. The goal is to improve the design w.r.t. the measuring criteria to meet the sign-off quality. E.g. the clock manipulation elements can modify the pulse width of the clock signal or may change the clock waveform which might be unable to switch the state of the driven sequential. They may change the clock polarity sometimes. So these can be quality hazards and may result in problems in silicon.

Engineering Change Order is a technique how the industries incorporate last minute design changes. If there is a bug in RTL or suppose design is not meeting timing, it is possible to incorporate those changes through ECO, without the need of resynthesize and entire physical design flow. Thus ECO plays a major role in SOC design cycle time reduction. The majority of the ECO techniques are freeze silicon based and focuses on the usage of available spare cells in the design to accomplish the required functional changes/ incorporating timing fix. Many kinds of literature are already available for ECO using spare cells. In [1], the author speaks about efficiently using the available spare cells in the design by technology remapping based on the wiring cost. In [2], a unified approach for functional ECO considering the timing constraints is discussed. In [3], the author discussed hold violation removal problem for today’s industrial designs by linear programming based methodology to model the setup and hold-time constraints. Then based on the solution to the linear programming, buffers are inserted as delay elements to solve hold violations. In [4], the concept of skew scheduling by adjusting skew to Flip Flop to improve the performance is discussed.

Compared to freeze silicon-based ECO, pre-silicon based ECO are also very important and plays a significant role in SOC timing closure. Once the design is placed and routed, if there is a bug in the design which is causing the functional failure or timing violation, resynthesizing the design and carrying outplacement and routing again is not a feasible as it is time-consuming. So there is an immense need of efficient ECO technique to handle these last minute changes. Most of the timing ECO techniques which are already proposed concentrates on the data path optimization for fixing setup/ hold. These data path optimization are done either by cell sizing (increasing the drive strength) or inserting buffers. In some cases, routing with a higher metal layer, changing cell to low Vt cell also accomplishes the purpose.

The rest of paper is organized as follows. Section II explains various timing quality checks. Section III explains the ECO through Data path optimization. Finally, Section IV discusses results obtained and conclusion.

II. TIMING QUALITY CHECKS

This section list out the various types of timing checks and its importance in timing closure.

A. *Data cells on clock path:* Clock cells are more variation robust. On other hand data, cells are not designed to toggle every cycle, so they fail reliability verification if used in a clock context. Also, data cells are skewed having different rise and fall delay. Hence they may affect the pulse width of the clock signal when used in clock path. This will result in inaccurate timing results.

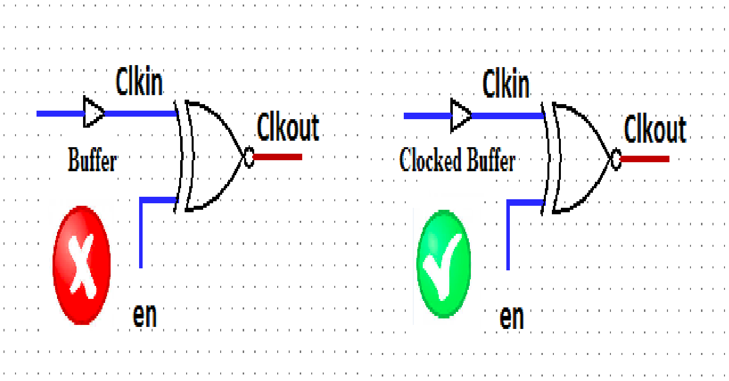


Fig.1 Data cells on clock path

The fix can be done by removing the bad stamping or swapping out “data” cells for “clock” version.

B. *Minimum pulse width constraint:* Minimum pulse width checks are done in order to make sure that width of the clock signal is wide enough for the cell’s internal operations to complete. Pulse width it is the interval between the rising edge of the signal crossing 50% of VDD and the falling edge of the signal crossing 50% of VDD. In terms of low signals, it is the interval between falling edge of the signal crossing 50% of VDD and the rising edge of a signal crossing 50% of VDD. Thus the input pin transition or slew will affect the actual pulse width. We can specify the min pulse width constraint either by specifying timing type: min_pulse_width in .lib file or by using the set_min_pulse_width command.

C. *Flops connected to constant clocks:* It is necessary to find the clock pins of registers and IP that do not have propagated clocks (the clock is not recognized/ stamped). The timing for sequential cells will not be checked if it is unlocked/ unstamped. Also, the check_timing command does not return flip-flops whose clock pin is tied to a constant value (0/1). These can happen because of the RTL bug, where clock pin is permanently tied to logic 0/1. So it is important to define clocks and generated clocks in the block properly.

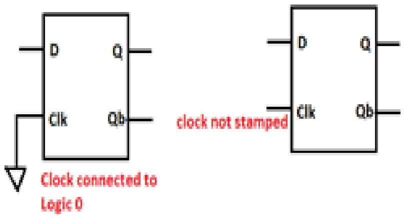


Fig 2. Flops with constant clock/ missing clock definition

D. *Combinational timing loop:* It is not preferred to have combinational timing loops in the design. Timing tool breaks the loop, but it might not be timed as expected. Therefore, timing can be too optimistic and critical path will not be shown. Also, loops will cause the slew degradation and results in bad timing picture. So careful analysis over the RTL is required to prevent timing loops.

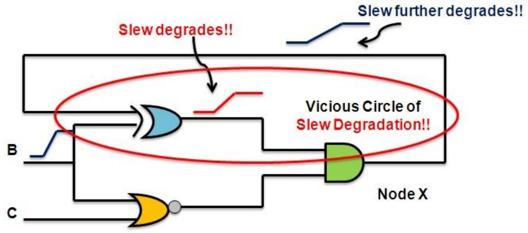


Fig.3 Timing loop in design

E. *Black Box Check:* Checks if there is a black box cell in the design without any library cells. These cells are linked with black boxes instead of real library cells. No timing information will be available for these cells. So before proceeding towards timing closure, one should check whether the design is linked properly or not and resolve the unresolved references.

F. *Max slope/ transition check:* Defined for an input pin. Max transition is the maximum time taken by the net in order to change its logic value. Cell delay = f (Input transition, Output load). So if the transition of a net is more/ bad, the delay also will be huge. To fix transition violations one should upsize the driver cell or decrease the net length by moving cells nearer or splitting loads through buffers to reduce the fan-out number or route using higher metal layer.

G. *Max capacitance check:* Defined for output pin. It denotes the maximum capacitive load that the output pin can drive. This comprises of both pin and interconnects net capacitance. More the capacitive load, more the time to charge/discharge to/from a particular logic level. So more the delay and dynamic power consumption. Increasing the drive strength of the drive cell is the solution to fix Max cap.

H. *Max fan-out check:* Defined for output pin. This is the max number of the input pins, which the particular output pin can drive. ie it denotes the load driving capacity of that pin.

III.ECO BY DATA PATH OPTIMIZATION

The majority of the timing ECO techniques tries to optimize data path instead of clock path, as touching clock path can sometimes proves risky. The ECO is either done by buffer insertion, cell sizing, adding or removing cells, vt swapping, net renaming, manual netlist editing, routing layer change etc.

A. *VT Swapping:* Switching to LVT cell, having low oxide thickness, ensures that the time taken by the cell to switch its output is less, increasing its speed. However, decreasing the threshold voltage also results in leakage related problems.

B. *Driver Upsizing:* Upsizing the driver cell results in improving the output drive strength of cell. Upsizing the cell means, increasing the width of the transistor which effectively decreases the resistance of the cell, resulting in increased current. Upsizing the cell improves the output transition, reduces the propagation delay time of the cell.

C. *Buffer Insertion:* This is an effective technique for interconnect-driven timing optimization. The interconnect delay is directly proportional to the square of the interconnect length. Inserting buffers in smaller segments makes interconnect delay linearly proportional to the wire length. It reduces the elmore delay of the nets by driving net capacitances through smaller resistances, thereby improving the slack of the path. However, it increases the area drastically along with requiring modifications in the rerouting stage.

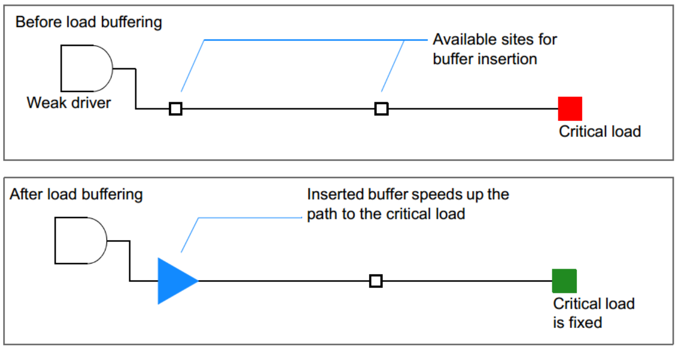


Fig. 4 Load buffering – Inserting buffers to strengthen weak drivers [8]

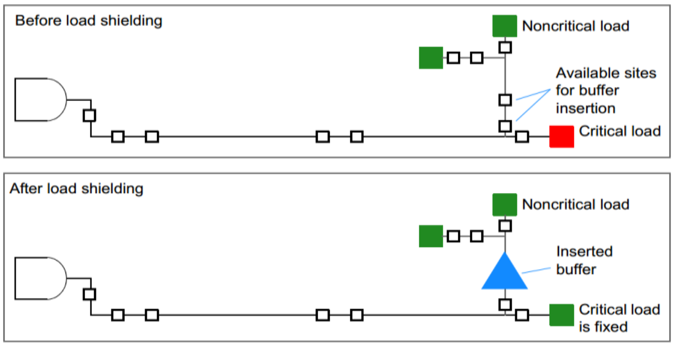
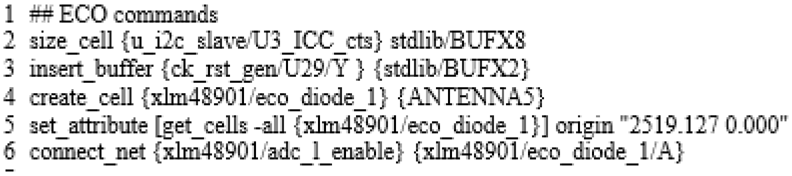


Fig. 5 Load shielding – Inserting buffers to shield critical loads from noncritical [8]

If the design has timing, design rule, or noise violations, or we want to optimize area or power, we use Engineering Change Order (ECO) flow to fix the violations, implement the changes, and rerun timing analysis. The figure shows the flow chart of the timing ECO flow. The algorithm tries to fix the timing violations for all path in the design by using the above-mentioned techniques like upsizing, Vt swap, buffer insertion.

Based on the drive strengths either VT swapping is chosen or upsizing (upsize to the maximum size available except for doingn't touch and don't use cell list). The alternative cell with the best input slew is chosen for replacement. Its slew value is compared against the path slack value. If it's lesser than that, the driver cell is successfully replaced with this alternative cell. This iteration continues till either the slack of the path significantly improves or the list gets exhausted. In such a case, the next path is taken into consideration. And after all the paths of a group have been analysed, the next group is considered. The result of ECO flow is simply an ECO patch/ change list which can be sourced in the implementation tool during the P&R stage to accomplish the changes. The figure6 shows an example of ECO patch.



```
## ECO commands
size_cell {u_i2c_slave/U3_ICC_cts} stdlib/BUFX8
insert_buffer {ck_rst_gen/U29/Y } {stdlib/BUFX2}
create_cell {xlm48901/eco_diode_1} {ANTENNA5}
set_attribute [get_cells -all {xlm48901/eco_diode_1}] origin "2519.127 0.000"
connect_net {xlm48901/adc_1_enable} {xlm48901/eco_diode_1/A}
```

Fig. 6 Example ECO patch

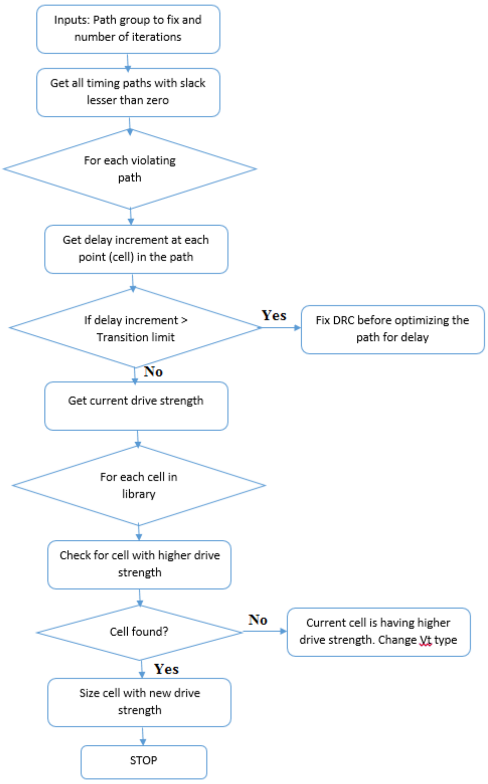

Inputs: Path group to fix and number of iterations
Get all timing paths with slack lesser than zero
For each violating path
Get delay increment at each point (cell) in the path
If delay increment > Transition limit
Yes
Fix DRC before optimizing the path for delay
No
Get current drive strength
For each cell in library
Check for cell with higher drive strength
Cell found?
No
Current cell is having higher drive strength. Change Vt type
Yes
Size cell with new drive strength
STOP

Fig. 7 Flowchart of ECO flow

Physically aware ECO

With the available placement information, ECO flow can consider placement congestion and blockages and provide precise ECO guidance with location. Accurate ECO timing estimation can also be achieved by separating the original net parasitic based on the target location and recalculating cell and net delays as well as crosstalk effects. ECO guidance with location and accurate timing estimation ensures predictable signoff timing closure after implementation. Also ECO can utilize available space along the net route to increases success rates for ECO fixing in congested regions. The main use of physically aware ECO is it prevents cell displacement by constraining cell upsizing to the available neighbouring free space and recognizes placement blockage and inserts an ECO buffer on the route.

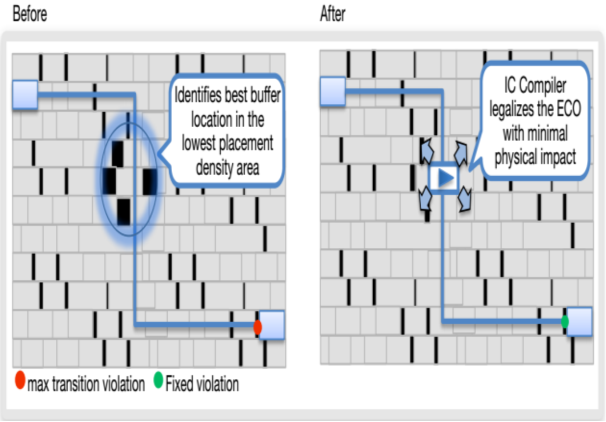

Before
After
Identifies best buffer location in the lowest placement density area
IC Compiler legalizes the ECO with minimal physical impact
max transition violation
Fixed violation

Fig. 8 Congestion-aware ECO

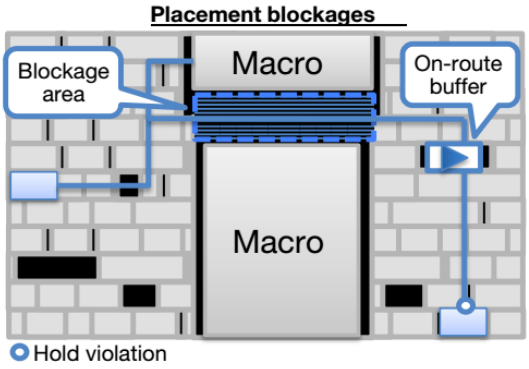


Fig. 9 Placement-aware ECO

To make the algorithm physically aware, DEF (Design Extraction Format) and LEF (Library Exchange Format) files should be given as input while performing ECO. The DEF file contains the following information.

- Physical aspects of a design: Die size, Connectivity, the Physical location of cells and macros on the chip.
- Floorplan information: Standard cell rows, Placement and routing blockage, Placement constraints, Power domain boundaries.
- Routing information: Metal layer used, Routing congestion.

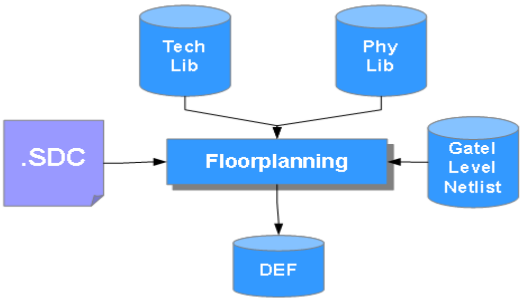


Fig 10. DEF file format

ECO in DMSA environment:

Verifying a chip design requires several parallel timing runs to check correct operation under different operating conditions (such as extreme temperatures and operating voltages) and different operating modes (such as mission or test mode). A specific combination of operating conditions and operating modes for a given chip design is called a scenario.
Scenarios = [Sets of operating conditions] X [Modes]

With distributed multi-scenario analysis (DMSA) we can analyze several scenarios in parallel. Instead of analyzing each scenario in sequence, DMSA uses a master process that sets up, executes, and controls multiple worker processes one for each scenario thus reducing the overall turnaround time to analyze the timing results. So performing ECO in DMSA environment helps us in understanding the effect of ECO patch in different scenarios simultaneously with less resource, runtime, and effort.

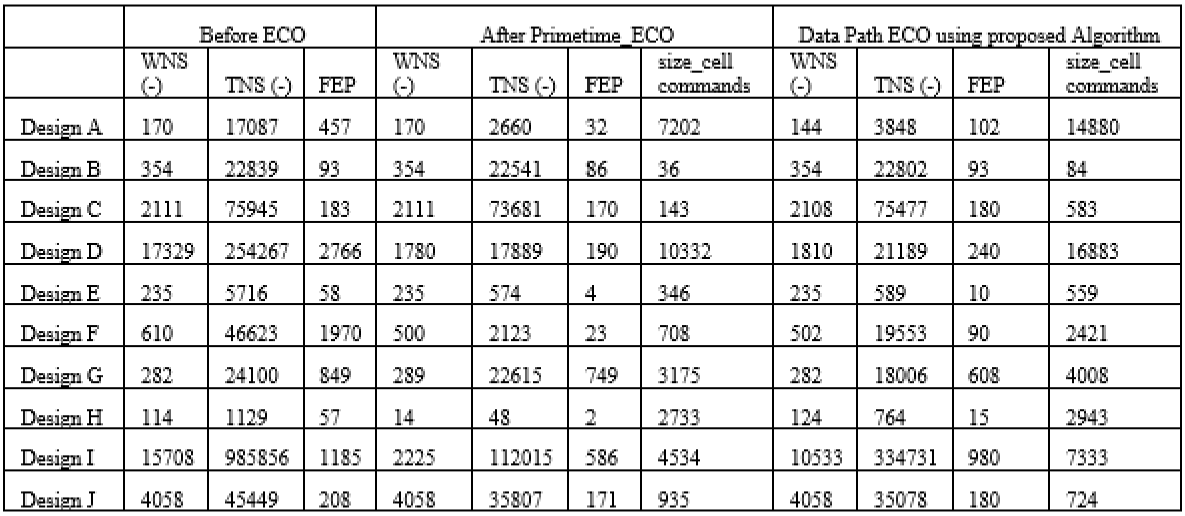
IV.RESULTS & CONCLUSIONS

It is observed that performing timing quality checks before timing analysis greatly helps in reducing the timing closure effort and it aids to timing closure activity by cleaning up the netlist. Thus timing analysis can now be performed on the good quality netlist which is free from missing clock stamping, unwanted timing loops, unresolved references and DRC violations.

The presented algorithm is implemented as a tcl script (Tool Command Language). Several industrial designs are taken for the experimental purpose. Table I shows the comparison of Worst Negative Slack WNS, Total Negative Slack TNS, Failing End Points FEP and number of size cell commands between data path ECO using proposed algorithm and ECO using existing Primetime ECO utility (fix_eco_timing). The proposed algorithm results match pretty well as compared with the existing primetime utility.

Two sets of experiments are done on the same designs. One by performing ECO using normal data path fixing. Other by performing physically aware ECO and in DMSA environment. The corresponding results are shown in figure 11, 12, 13. It is observed that using physically aware ECO gives an efficient result compared to normal ECO, but however, the run time is pretty high. This can be compensated by performing runs in DMSA environment which reduces the resource usage and manual effort.

TABLE I
WNS, TNS AND FEP COMPARISON BETWEEN PRIMETIME ECO AND DATAPATH ECO USING PROPOSED ALGORITHM



	Before ECO			After Primetime_ECO				Data Path ECO using proposed Algorithm			
	WNS (-)	TNS (-)	FEP	WNS (-)	TNS (-)	FEP	size_cell commands	WNS (-)	TNS (-)	FEP	size_cell commands
Design A	170	17087	457	170	2660	32	7202	144	3848	102	14880
Design B	354	22839	93	354	22541	86	36	354	22802	93	84
Design C	2111	75945	183	2111	73681	170	143	2108	75477	180	583
Design D	17329	254267	2766	1780	17889	190	10332	1810	21189	240	16883
Design E	235	5716	58	235	574	4	346	235	589	10	559
Design F	610	46623	1970	500	2123	23	708	502	19553	90	2421
Design G	282	24100	849	289	22615	749	3175	282	18006	608	4008
Design H	114	1129	57	14	48	2	2733	124	764	15	2943
Design I	15708	985856	1185	2225	112015	586	4534	10533	334731	980	7333
Design J	4058	45449	208	4058	35807	171	935	4058	35078	180	724

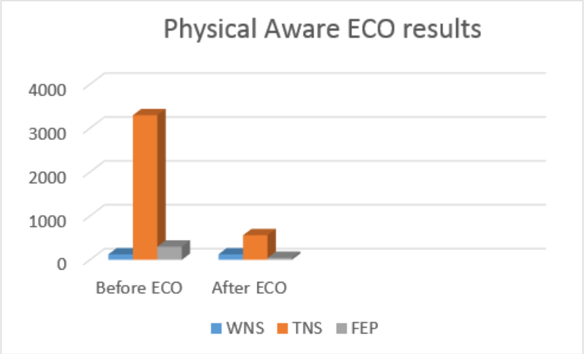


Fig. 11 Physically aware ECO results

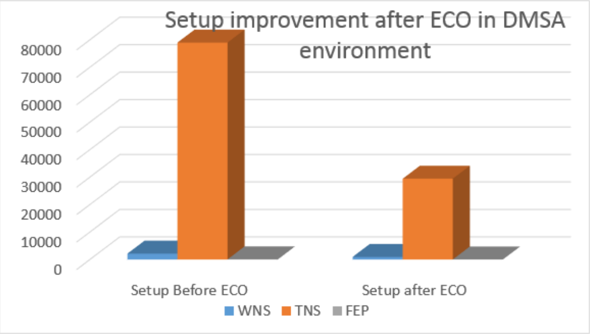


Fig. 12 Setup improvement after ECO in DMSA environment

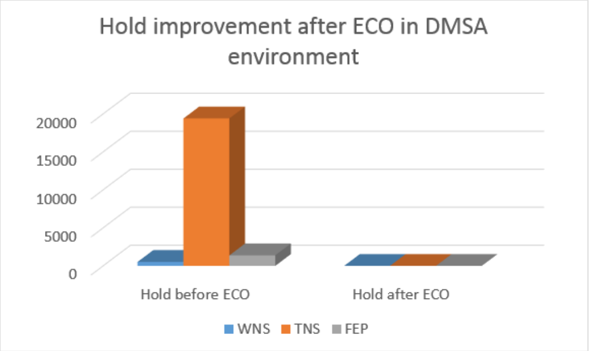


Fig. 13 Hold improvement after ECO in DMSA environment

REFERENCES

[1] Y.P. Chen, J.W. Fang, Y.W. Chang, “ECO timing optimization using spare cells,” in Proceedings of International Conference on Computer-Aided Design, pp. 530-535, 2007.
[2] Jui-Hung Hung, Yu-Cheng Lin, Wei-Kai Cheng, Tsai-Ming Hsieh, “Unified approach for simultaneous functional and timing ECO,” in IET Circuits Devices Syst, Vol. 10, Iss. 6, pp. 514–521, 2016.
[3] Pei-Ci Wu, Martin D. F. Wong, Ivailo Nedelchev, Sarvesh Bhardwaj, Vidyamani Parkhe, “On Timing Closure: Buffer Insertion for Hold-Violation Removal,” ACM 978-1-4503-2730-5/14/06, 2014.
[4] C. Lin and H. Zhou, “Clock skew scheduling with delay padding for prescribed skew domains,” in Design Automation Conference, 2007. ASP-DAC’07, Asia, and South Pacific, pages 541–546, IEEE, 2007
[5]K. H. Tsai and S. Sheng *“Design rule check on the clock gating logic for testability and beyond"*, 2013 IEEE International Test Conference (ITC), Anaheim, CA, 2013.
[6]Welch, Brent B. *“Practical Programming in Tcl & Tk"*, Upper Saddle River, New Jersey: Prentice Hall, 2000.
[7] Bhasker, Jayaram, and Rakesh Chadha “Static Timing Analysis for Nanometer Designs: A Practical Approach", Springer Science & Business Media, April 3, 2009.
[8] Synopsys “PrimeTime User Guide", Version L-2016.06.
[9] Synopsys “PrimeTime Suite Variables and Attributes", Version K-2015.06-SP2, June 2015.