IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 7, Issue 3, Ver. I (May. - June. 2017), PP 35-41
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Simultaneous Data Path and Clock Path Engineering Change Order for Efficient Timing Closure in Complex SOC

L. Shanthala [1], Dr. R. Jayagowri[2]
[1]*PG Scholar, Department of Electronics and Communication Engineering, BMS College of Engineering, Bangalore,*
[2]*Associate Professor, Department of Electronics and Communication Engineering, BMS College of Engineering, Bangalore,*

Abstract: *With ever increasing IC complexity and aggressive technology scaling towards cutting edge technologies, the SOC timing closure is becoming a tedious, time consuming and challenging task. 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. In this work, we have proposed an algorithm for simultaneous usage of data path ECO and clock path ECO for efficient timing closure in SOC. The algorithm here tries to fix multiple failing end points through simple clock path optimization, instead of performing data path optimization across multiple paths, thus reducing area and power overhead. The proposed algorithm is tested on multiple industrial designs and found to achieve 30.98% improvement interms of Worst Negative Slack, 63.63% interms of Total Negative Slack, 58.19% interms of 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.*

Keyword: *Engineering Change Order ECO, Clock rescheduling / Clock push pull, physically aware, Distributed Multi Scenario Analysis*

I. Introduction

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.

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 literatures are already available for ECO using spare cells. In [1], 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], author discussed about 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 out placement 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 higher metal layer, changing cell to low Vt cell also accomplishes the purpose. But one can think of touching the clock path and experiment on Clock Tree Synthesis CTS. Clock path rescheduling means performing clock push/ pull by inserting or removing clock buffers. The idea is to utilize the positive slacks at the fanout cone of the clock tree element as a safe margin and then fix the negative slacks on other paths incrementally through clock tree restructuring/ resizing. [5]

Major challenges with the proposed method:

1. Finding a common point on the clock tree where the ECO patch can be placed such that it will help in fixing multiple failing end points.

2. Checking the fanout cone of the clock tree restructuring / resizing element, so that restructuring doesn't result in negative slack across the paths which were not failing before. i.e Fixing negative margin for some path shouldn't create new paths with negative slack.
3. Fix should be MCMM aware (Multi Corner Multi Mode). Before inserting/removing any buffers, we should check its impact on setup/ hold across multiple corner including max, min, typical and across multiple modes including scan, functional.
4. If clock rescheduling is not possible or not feasible, then performing regular data path optimization by gate sizing and buffer insertion.

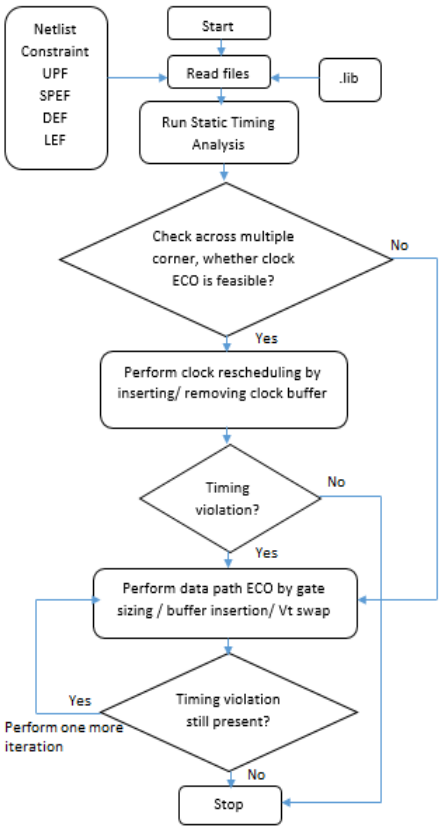
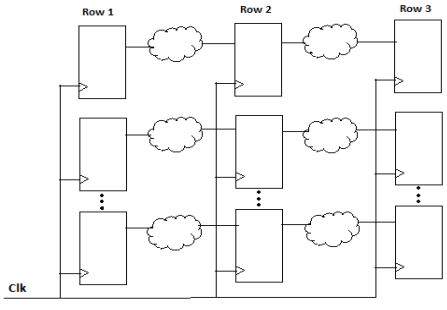


Fig.I Flow chart of the proposed algorithm

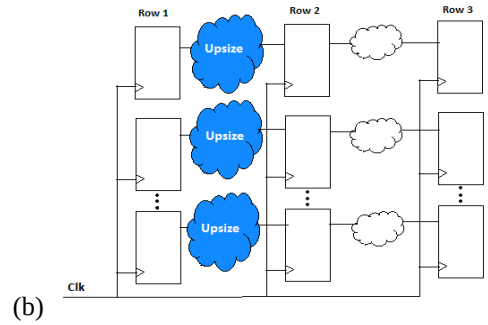
The rest of paper is organized as follows. Section II explains the Clock path ECO approach. Section III explains the Data path ECO approach. Finally Section IV discusses results obtained and conclusion.

II. Approach & Algorithm For Clock Path Eco

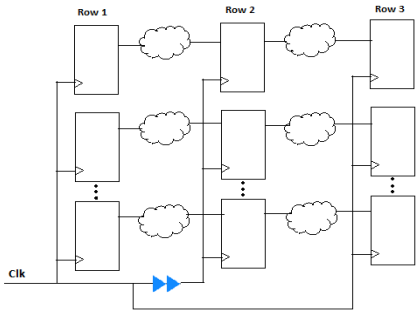
The algorithm here tries to find out a point (cell) in the clock tree, which is common to multiple failing end points in the design. Inserting an ECO patch, by clock rescheduling at this common point helps in fixing the timing violation across multiple failing end points, instead of performing data path ECO on all paths.



(a)



(b)



(c)

Fig.IIa. All FlipFlop in row2 are violating setup b. Datapath ECO by gate upsizing c. Clock path ECO by clock push

The figure 2 shows the main advantage of clock path ECO. Suppose there are multiple paths which are violating setup as shown in row 2. Performing data path ECO will tries to upsize the cells on all the failing paths as shown in Fig 2b, which will unnecessarily results in area and power overhead. In such case, finding a common point in clock tree which is driving all these failing end points and inserting few buffers on clock path will fix the setup violation across all these failing paths as shown in Fig 2c.If there is no such common point in clock tree, then the algorithm performs usual data path fixing by gate sizing or insert buffer.

There are some prerequisites before performing clock ECO:

1. CTS done and is stable with balanced skew.
2. Clock transition violations should be fixed prior to clock rescheduling. Because clock transition creates unnecessary skew and results in inaccurate delay calculations.

Algorithm 1: Clock path ECO

```
1: procedure Clockpath_ECO (FEP_list, corner, mode)
2: Foreach endpoint in FEP_list
3:    get_timing_path –through CLK pin of FEP
4:    Foreach i in $path
5:       list_i <- get_points in the clk path
6:    end
7: end
8: sort all list. Backtrace each list to find common point
9: if (common_point)
10:    check slack for clock push/ pull in multiple corner
and mode
11:    if (safe_slack_margin)
12:       insert / remove buffers
13: else
14: proc (Datapath_ECO)
15: end procedure
```

The clock ECO algorithm takes a set of Failing End Points as input. The clock pins of each FEP is backtraced and all points in clock path are stored in separate lists. Each list is sorted and first common clock buffer amongst all lists is greped. Then the slack checking is done across multiple corners and multiple modes before pushing / pulling clock. The slack checking is done to ensure the positive safe margin at the fanout cone of clock tree restructuring element and exploit this safe margin (useful skew) to fix negative slack on violating path. If anyone path at the fanout cone has negative margin, then it is not preferred to do clock push/ pull. Regular data path ECO is preferred in such case. Figure 3 and 4 shows how the slack checking is done before clock push / pull.

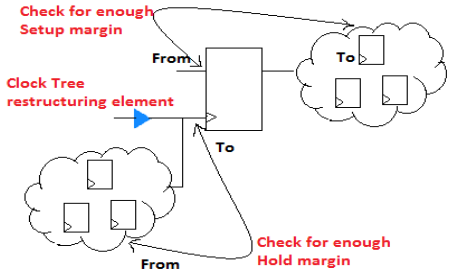


Fig.III slack checking before clock push (inserting clock buffer)

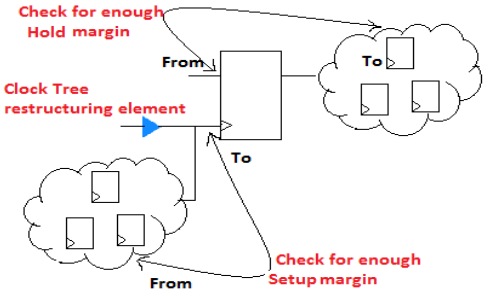


Fig.IV slack checking before clock pull (removing clock buffer)

III. Approach & Algorithm For Data Path Eco

Based on the drive strengths either VT swapping is chosen or upsizing (upsize to the maximum size available except for don'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 and the value equivalent to path slack value is updated for the next iteration. Iteration continues till either the slack of the path significantly improves or the list gets exhausted. In such a case, the next path of a group is taken into consideration. During ECO flow we can also specify the margin for setup/ hold fix. This is called margin based setup / hold fixing. For example, if majority of the paths are violating hold by margin of0 to -500ps, we can fix negative hold paths in the range 0 to -500ps, while ignoring those with negative margin worse than -500ps.

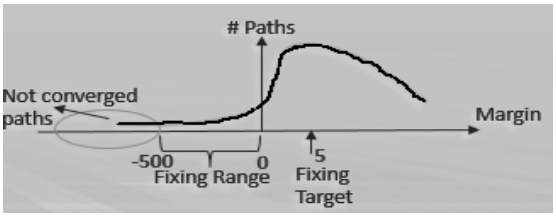


Fig.V margin based setup/ hold fixing

Algorithm 2: Data path ECO

```
1: procedure Datapath_ECO (path_group,
2: No.of iterations, corner, mode)
3: Foreach path in design
4:    if (slack < 0) then
5:    get_timing_path through FEP
6:    Foreach point in violating path
7:        get delay increment
8:        if (delay increment > tran limit)
9:        Input tran is bad. Do DRC fix
10:       else
11:       current_drive <- get_drive_strength (cell)
12:       Foreach cell in library
13:           if (get_drive_strength($cell) > current_drive)
14:           perform size_cell
15:           else perform Vt swap
16:       end
17:   end
18: end
19: update_timing in all corners and modes
20: perform next iteration
21: end procedure
```

For making the ECO physically aware, DEF file (Design Exchange Format) is taken as input. The DEF file contains information about,

- Physical aspects of a design: Die size, Connectivity, 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.

Even with the routed design, we can perform ECO by adding buffers on route. Thus exact location where the buffer needs to be added is specified in the ECO script. After sourcing the ECO script legalization of placement must be done. To verify the correct operation of design under different operating conditions (extreme temperature & operating voltages) and different operating modes (functional & scan), we require several parallel timing runs. But by using DMSA (Distributed Multi Scenario Analysis), a utility of synopsys primetime tool, instead of analyzing each scenarios in sequence, we can analyze several scenarios in parallel. Different scenarios are remotely executed in different hosts running in parallel thus reducing the overall turnaround time / run time, resources and effort. Also the effect of on chip variation can be analyzed in DMSA environment by providing timing derate values. Final algorithm is a wrapper written on top of clock path ECO and data path ECO algorithms. It first calls the clock path ECO procedure. If all paths are not fixed by clock rescheduling, then data path ECO procedure is called.

Algorithm 3: Clock path + Data path ECO

```
1: procedure Simultaneous_ECO (path_group, FEP_list,
Corner, mode)
2: Execute Clockpath_ECO (FEP_list, corner, mode)
 for all clock, corners and modes
3: set n report_timing –slack_lesser_than 0
4: size_of_collection n not_equal_to 0
5: Execute Datapath_ECO (path_group, No.of iterations,
 Corner, mode) for all path_group, corners and modes
6: end procedure
```

IV. Results And Conclusion

The presented algorithm is implemented as a tcl script (Tool Command Language) in a UNIX machine with i5 core, 4GB RAM, 2.5 GHz CPU. Several industrial designs are taken for experimental purpose. Synopsys Primetime tool version k-2015 is used as a main tool for performing timing analysis. Table 1 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 Figure 6 shows the respective fix rate interms of WNS, TNS, and FEP. The proposed algorithm results matches pretty good as compared with the existing primetime utility.

	Before ECO			After Primetime_ECO				Data PathECO 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

Table. I WNS, TNS and FEP comparison between PT_ECO and DataPath ECO using proposed algorithm

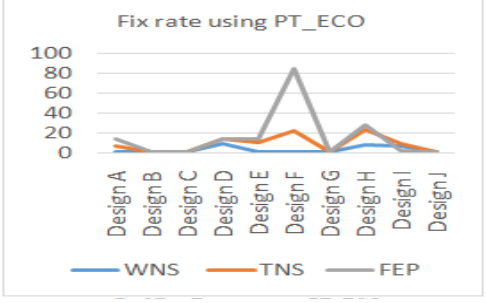


Fig.VI a. Fix rate using PT_ECO

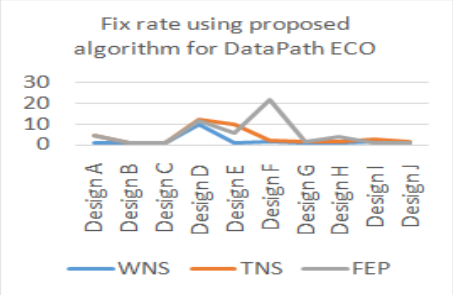


Fig.VI b. Fix rate using proposed algorithm for DataPath ECO

Two sets of experiments are done on the same designs. One by performing ECO only using data path fixing. Other by performing clock ECO followed by data path ECO. The corresponding results are shown in table 2 and the fix rate is shown in Figure 7. The Figure 8 shows the size of ECO patch before and after using clock rescheduling. It is clearly observed that using clock path ECO significantly reduce the size of ECO patch i.e less number of resize and insert buffer commands. Thus the overall amount of area increased after ECO is 59.42 % less in clock + data ECO as compared to data ECO alone. This highlights the advantage of clock path rescheduling.

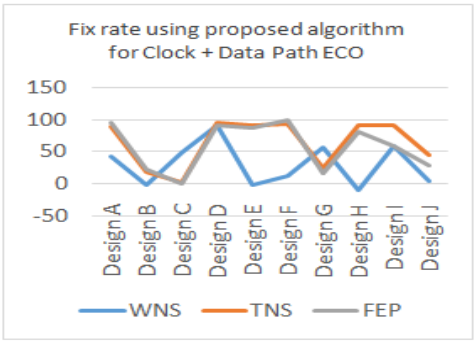


Fig.VII Fix rate using proposed algorithm for Clock + DataPath ECO

	DataPath ECO using proposed Algorithm				ClockPath + DataPath ECO using proposed algorithm			
	WNS (-)	TNS (-)	FEP	ECO patch size (size cell)	WNS (-)	TNS (-)	FEP	ECO patch size (insert buffer + size cell)
Design A	144	3848	102	14880	98	2090	26	1095
Design B	354	22802	93	84	354	18509	72	24
Design C	2108	75477	180	583	1078	73670	180	408
Design D	1810	21189	240	16883	1500	14334	242	10290
Design E	235	589	10	559	235	555	8	128
Design F	502	19553	90	2421	526	3357	20	571
Design G	282	18006	608	4008	120	18102	705	2991
Design H	124	764	15	2943	124	108	11	433
Design I	10533	334731	980	7333	6344	100155	480	3040
Design J	4058	35078	180	724	3850	25455	146	446

Table. II WNS, TNS and FEP comparison between only DataPath ECO&Clock+ Data Path ECO using proposed algorithm

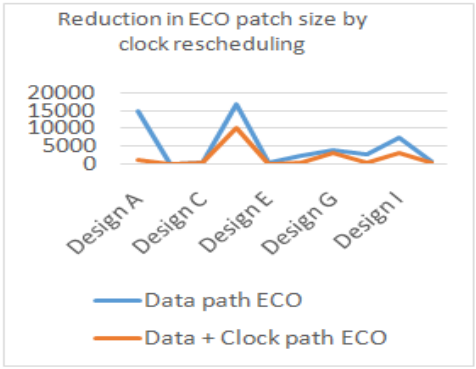


Fig.VIII Reduction in ECO patch size by clock + data path ECO

To conclude, the proposed algorithm shows a better fix rate of 30.98%interms of Worst Negative Slack, 63.63% interms of Total Negative Slack, 58.19% interms of Failing End Points. However the main disadvantage of clock path ECO is that, it creates skew in clock tree and results in imbalanced CTS. So prefer clock tree ECO only if it targets fixing a bunch of failing end points. Performing clock rescheduling to fix one / two paths is not advisable.

Acknowledgment

Our grateful thanks to Mr. Gaurav Kaushik, Miss.Ashraf Harshana, Miss.Chandrani Saha and Mr. Deepak Gupta from Intel technologies India Pvt. Ltd. for the valuable discussions and technical assistance.

Reference

[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, IvailoNedelchev, Sarvesh Bhardwaj, VidyamaniParkhe, "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] Subhendu Roy, Pavlos M. Mattheakis, Laurent Masse-Navette, and David Z. Pan, *"*Clock Tree Resynthesis for Multi-Corner Multi-Mode Timing Closure*,"in* IEEE transactions on computer-aided design of integrated circuits and systems, vol. 34, no. 4, April 2015
[6] S. Huang and Y. Lin, "Utilizing Clock Skew For Timing Reliability Improvement," IEEE, 2007
[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.