2015 14th International Conference on Computer-Aided Design and Computer Graphics

A Fast and Simple Block-based Approach for Common Path Pessimism Removal in Static Timing Analysis

Guojie Luo[1,2,3], Baihong Jin[1*], Wentai Zhang[1]
[1] Center for Energy-efficient Computing and Applications, School of EECS, Peking University, Beijing, China
[2] PKU-UCLA Joint Research Institute in Science and Engineering
[3] Collaborative Innovation Center of High Performance Computing, National Univ. of Defense Tech., Changsha, China
e-mail: gluo@pku.edu.cn, happyucb@gmail.com, rchardx@gmail.com

***Abstract*—The "early/late split" in delay modeling is an effective approach to handle the variability in deep submicron integrated circuits. However, static timing analysis with early/late split is often too conservative due to the common path pessimism, where the common path pessimism removal (CPPR) technique is helpful to eliminate the unnecessary pessimism. In this paper, we propose a fast and simple CPPR algorithm with sub-quadratic time complexity.**

Keywords-VLSI; Static Timing Analysis (STA); Common Path Pessimism Removal (CPPR)

I. Introduction

Timing closure has always been the main concern of digital designers. A design team may spend months [1] on the design iterations to achieve the timing target. Static timing analysis (STA) is an approach to estimate the maximum frequency of a digital circuit, while an accurate estimation will reduce the design iterations and improve the design quality.

As device scaling continues, the process variation has become more and more significant. As a result, complex models are introduced to characterize the on-chip variation (OCV). OCV can be classified into chip-to-chip variation and within-chip variation. Multi-corner timing is an effective approach to handle chip-to-chip variation, and early/late split provides a simple approach to handle within-chip variation.

With early/late split, each timing edge in the timing graph has the early-mode and the late-mode delays. Independent analysis of timing paths introduces common path pessimism, since the early-mode and late-mode delays cannot occur at the same time on the common portion of two paths. Thus, common path pessimism removal (CPPR) is necessary.

In this paper, we develop a new block-based STA algorithm that can produce accurate post-CPPR results directly. Our algorithm has sub-quadratic time complexity with respect to the circuit size. As far as we know, this algorithm has not been presented in existing literature.

II. Preliminary Background

We generalize the post-CPPR timing analysis problem in the TAU 2014 Contest [3] as follows: *given the netlist and the early/late delays per timing edge of a sequential circuit, calculate the post-CPPR setup/hold slacks of the k most critical paths ending at each flip-flop (FF).*

A. Review of Pre-CPPR Timing Analysis

In this paper we use upper-case letters to represent the constant delay values that known before computation, and use the lower-case letters to represent the intermediate delay values during the computation of STA.

Assume we only concern the delay of the most critical path, we can find the worst negative slack (WNS) for the setup-time constraints using the following equation.

$$WNS = T_{cycle} - T_{setup} - \max_{t \in CFF}\{a_t - D^{early}_{clk \to t:CK}\}$$

$$\text{where} \quad \begin{cases} a_s = D^{late}_{clk \to s:CK} + D^{late}_{s:CK \to s:Q} & s \in LFF \\ a_j = \max\limits_{i \in fanin(j)}\{a_i + D^{late}_{i \to j}\} & \text{otherwise} \end{cases}$$

In this equation, the minimum clock period T_{cycle} and the setup-time T_{setup} are given, as well as the set of launching FFs (LFF) and capturing FFs (CFF) and the topology of the timing graph. The early/late modes of the FF delays and the gate/wire delays are also given, including the arrival time $D_{clk \to *:CK}$ from the central clock source to the clock pin of any FF, the clock-to-Q delay $D_{*:CK \to *:Q}$ of any FF, and the delay $D_{i \to j}$ between any neighboring timing nodes i and j.

B. Early/Late Split & Common Path Pessimism

Early/late split provides safety margins for timing analysis by assigning lower and upper bound to the delay values, but these safety margins sometimes introduce excessive and undesired pessimism if two timing paths share a common portion of the clock network. For example, in the pre-CPPR timing analysis in the last subsection, we use the late-mode for the clock signal of the launching FFs and the early-mode for the clock signal of the capturing FFs when examining the setup-time constraints. However, if the clock signals of two FFs share a common portion in the clock network, the clock signals in the common portion cannot simultaneously express two different delay modes. Such unnecessary pessimism should not be reported in the timing analysis results. The hold-time constraints can be analyzed in a similar manner.

C. Previous Work on CPPR

Although the pre-CPPR slack is a lower-bound of the real slack, there is usually no known correspondence between a pre-CPPR slack and its post-CPPR counterpart. Therefore, a complete CPPR analysis would require investigating all paths for every failing test [3], which is usually computationally

[1*] Baihong Jin involved in this project as an undergraduate student at PKU. He is now a PhD student at University of California, Berkeley.

978-1-4673-8020-1/15 $31.00 © 2015 IEEE
DOI 10.1109/CADGRAPHICS.2015.52

unaffordable. As a result, a set of filters or fast-outs are used by existing CPPR algorithms to prune the search space. References [3][2] give an excellent overview of the existing methods.

The state-of-the-art post-CPPR STA algorithms are based on path searches, using either the top-k critical path search [4] or the branch-and-bound path retrieval [5]. In contrast, our algorithm is compatible to the conventional block-based STA framework, and can directly obtain the post-CPPR without path searches.

III. Our Proposed Algorithm

Aware of the common path pessimism, we find the WNS for the setup-time constraints using the following equation.

$$WNS = T_{cycle} - T_{setup} - \max_{t \in CFF}\{a_{t,t}\}$$

$$\text{where } \begin{cases} a_{s,t} = Skew^{max}_{s:CK,t:CK} + D^{late}_{s:CK \to s:Q} & s \in LFF \\ a_{j,t} = \max\limits_{i \in fanin(j)}\{a_{i,t} + D^{late}_{i \to j}\} & \text{otherwise} \end{cases}$$

The equation is very similar to the equation in Section II.A, except that there are more intermediate delay terms. In the pre-CPPR timing analysis, every timing node j has only one intermediate delay term a_j; but in our post-CPPR timing analysis, every timing node j has p intermediate delay terms $a_{j,t}$, where p is the total number of capturing FFs in the downstream paths of this timing node.

By setting $Skew^{max}_{s:CK,t:CK}$ to $D^{late}_{clk \to s:CK} - D^{early}_{clk \to t:CK}$, this equation generates exactly the same WNS as in Section II.A. As discussed earlier, the maximum delay for the timing path ending at $s{:}\,CK$ and the minimum delay for the timing path ending at $t{:}\,CK$ will not happen at the same time when they share common clock paths. Specifically, assume timing node r is the most recent common ancestor of $s{:}\,CK$ and $t{:}\,CK$, we set $Skew^{max}_{s:CK,t:CK}$ to $D^{late}_{r \to s:CK} - D^{early}_{r \to t:CK}$ in the post-CPPR timing analysis by removing the common path from clk to r.

Although this equation consumes more computation and memory resources, it is an accurate solution for the post-CPPR timing analysis. The experimental results in the next section will also demonstrate its efficiency. It can be easily extended to analyze the top k critical paths in addition to the most critical path.

IV. Experimental Results

To validate our proposed algorithm, we implemented it in C++ and performed the experiments on a Linux server with dual Intel Xeon E5-2430 2.2GHz CPUs with 32GB RAM. We use the benchmarks provided by the TAU 2014 Contest.

UI-Timer [4] and iTimerC [5] are the top timers in the TAU 2014 Contest, and we compare the runtime consumption between our timer and theirs. Though the runtime of UI-Timer and iTimerC is extracted from [5], the machine configuration is similar to ours. Please note that those two timers are multi-threaded, and our timer runs with a single thread. The results are shown in Table 1. We can see that our timer is as efficient as iTimerC, and outperforms the both timers in some cases.

Table 1. Runtime comparisons with data extracted from [5]

Bench	type	#test	#path	UI-Timer (s)	iTimerC (s)	ours (s)
Combo2	setup	10000	15	15.3	13.69	11.70
		20000	1	8.95	6.44	5.93
	hold	10000	15	12.5	11.71	9.81
		20000	1	7.46	5.87	5.54
Combo3	setup	6000	20	6.89	7.73	6.78
		8000	1	2.92	3.60	3.16
	hold	6000	20	5.95	6.28	5.32
		8000	1	2.18	3.41	2.99
Combo4	setup	15000	15	82.88	90.03	57.99
		25000	1	43.55	43.71	28.66
	hold	15000	15	67.52	37.79	44.86
		25000	1	42.3	17.14	27.37
Combo5	setup	20000	15	222.87	169.23	120.56
		35000	1	150.78	89.33	69.38
	hold	20000	15	167.84	87.25	90.30
		35000	1	134.84	48.97	64.64
Combo6	setup	35000	15	584.83	244.05	253.71
		50000	1	433.9	116.48	165.49
	hold	35000	15	500.47	173.08	208.93
		50000	1	378.29	81.72	154.64
Combo7	setup	35000	20	484.15	364.2	237.26
		50000	1	299.6	136.31	135.85
	hold	35000	20	383.47	174.92	190.26
		50000	1	260.65	66.81	126.61
geomean	-			63.46	38.84	38.79

V. Conclusions and Future Work

In this paper, we proposed an algorithm for performing post-CPPR timing analysis. We show that, with a proper transformation, CPPR can be implemented within the block-based STA framework with sub-quadratic complexity. We will work on parallelizing our algorithm on multi-core CPUs or GPUs for further accelerating post-CPPR timing analysis.

Acknowledgment

This work is partly supported by National Natural Science Foundation of China (NSFC) Grant 61202073, Research Fund for the Doctoral Program of Higher Education of China (MoE/RFDP) Grant 20120001120124, and Beijing Natural Science Foundation (BJNSF) Grant 4142022.

References

[1] J. Bhasker and R. Chadha. “Static Timing Analysis for Nanometer Designs.” Springer, 2009.

[2] V. Garg, “Common path pessimism removal: An industry perspective: Special Session: Common Path Pessimism Removal,” Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 592–595, 2014.

[3] J. Hu, D. Sinha, and I. Keller. “Tau 2014 contest on removing common path pessimism during timing analysis.” Proceedings of the 2014 International symposium on physical design, pp. 153-160, 2014.

[4] T. Huang, P. Wu, and M. D. F. Wong, “Fast path-based timing analysis for CPPR,” Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 596–599, 2014.

[5] Yu-Ming Yang, Yu-Wei Chang, and Iris Hui-Ru Jiang. “iTimerC: common path pessimism removal using effective reduction methods.” Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 600-605, 2014.