Ph.D. DISSERTATION

Clock Tree and Flip-flop Co-optimization for Reducing Power Consumption and Power/Ground Noise of Integrated Circuits and Systems

집적회로 및 시스템에서의 전력 소모와 파워/그라운드 노이즈 감소를 위한 클럭 트리와 플립플롭 통합 최적화

BY

Joohan Kim

August 2017

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

Clock Tree and Flip-flop Co-optimization for Reducing Power Consumption and Power/Ground Noise of Integrated Circuits and Systems

집적회로 및 시스템에서의 전력 소모와 파워/그라운드 노이즈 감소를 위한 클럭 트리와 플립플롭 통합 최적화

BY

Joohan Kim

August 2017

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Clock Tree and Flip-flop Co-optimization for Reducing Power Consumption and Power/Ground Noise of Integrated Circuits and Systems

집적회로 및 시스템에서의 전력 소모와 파워/그라운드 노이즈 감소를 위한 클럭 트리와 플립플롭 통합 최적화

지도교수 김 태 환

이 논문을 공학박사 학위논문으로 제출함

2017 년 5 월

서울대학교 대학원

전기 컴퓨터 공학부

김 주 한

김 주 한의 공학박사 학위논문을 인준함

2017 년 6 월

위 원 장 _____

부위원장 _____

위 원 _____

위 원 _____

위 원 _____

Abstract

For very-large-scale integration (VLSI) circuits, the activation of all flip-flops that are used to store data is synchronized by clock signals delivered through clock networks. Due to very high frequency of clock signal switches, the dynamic power consumed on clock networks takes a considerable portion of the total power consumption of the circuits. In addition, the largest amount of power consumption in the clock networks comes from the flip-flops and the buffers that drive the flip-flops at the clock network boundary. In addition, the requirement of simultaneously activating all flip-flops for synchronous circuits induces a high peak power/ground noise (i.e., voltage drop) at the clock boundary.

In this regards, this thesis addresses two new problems: the problem of reducing the clock power consumption at the clock network boundary, and the problem of reducing the peak current at the clock network boundary. Unlike the prior works which have considered the optimization of flip-flops and clock buffers separately, our approach takes into account the co-optimization of flip-flops and clock buffers. Precisely, we propose four different types of hardware component that can implement a set of flip-flops and their driving buffer as a single unit. The key idea for the derivation of the four types of clock boundary component is that one of the inverters in the driving buffer and one of the inverters in each flip-flop can be combined and removed without changing the functionality of the flip-flops. Consequently, we have a more freedom to select (i.e., allocate) clock boundary components that is able to reduce the power consumption or peak current under timing constraint. We have implemented our approach of clock boundary optimization under bounded clock skew constraint and tested

it with ISCAS 89' benchmark circuits. The experimental results confirm that our approach is able to reduce the clock power consumption by 7.9∼10.2% and power/ground noise by 27.7%∼30.9% on average.

Keywords: Clock tree synthesis, low power, post-placement optimization, simultaneous switching noise, peak current, power/ground noise

Student Number: 2012-30200

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Clock Signal

In a synchronous digital circuit, a clock signal is delivered through clock distribution network to sequential elements, such as flip-flops or latches. A clock signal is a periodic signal switching from 1 to 0 or from 0 to 1, which is used to maintain synchronization of the circuit. It determines when a storage element receives the input data and change its output value. This change can only happen at clock rising/falling edges so that we can assure that the synchronization of storage elements. In other words, a clock signal in a digital circuit can be likened to a heart of humans. Hence, for keeping circuit's synchronization and operating stably, the clock signal should be designed with careful consideration the following design metrics.

1.2 Metrics of Clock Design

Power consumption: Due to its incessant switching activity, a clock distribution network consumes significant amount of power. According to the works of [2, 3], power consumption in the clock distribution network accounts for up to 40% of the total circuit power. Power consumption of a clock can be represented as follows:

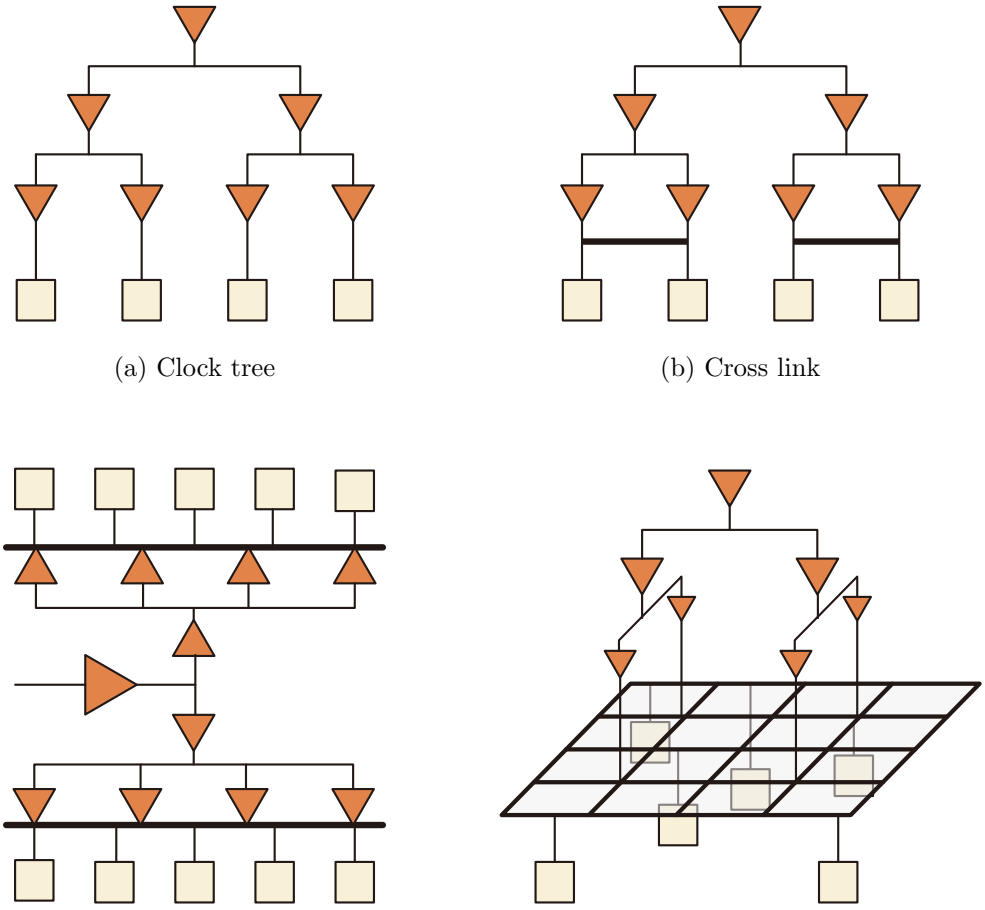
$$P = \alpha \cdot C \cdot V^2 \cdot f \tag{1.1}$$

, where α is switching activity of the circuit, C is total amount of circuit, V is supply voltage and f is clock frequency. Many techniques to minimize each factor in Eq. 1.1 have been suggested: dynamic voltage and frequency scaling[4, 5], which minimizes V of f, clock gating[6, 7], which cuts clock signal so that minimize α, and buffer insertion technique[8, 9, 10, 11], which reduces capacitive load, C. And as a special method to save power, resonant clock technique[12, 13, 14] has been suggested. In the resonant clock, the electrons are stored in the embedded inductors, not dissipated through NMOS transistor.

Slew rate: The slew rate of a clock signal refers to the transition time from V_{low} to V_{high}, or from V_{high} to V_{low}. Usually V_{low} and V_{high} is determined as 10% and 90% of supply voltage, respectively. A clock signal with high slew rate, i.e., a clock signal that switches more slowly, can cause the delay and power consumption to increase. Thus, the slew rate should be met the given constraints in clock design.

Clock skew: For proper operation of synchronous systems, the clock signal at every storage elements must appear at the same instance. Unfortunately, as the clock signal is transferred through physical metal wires from one source to the multiple clock sinks, this cannot always be guaranteed. This difference of the maximum and minimum arrival time between flip-flops is referred to the

skew. Since a large clock skew may cause the circuit to malfunction, which leads to a system failure. In this regards, lots of researches have been studied under various clock skew constraints: zero skew constraint[15, 16, 17], bounded skew constraint[18, 19, 20, 21], and useful skew constraint[22, 23].



(a) Clock tree

(b) Cross link

(c) Clock spine

(d) Clock mesh

Figure 1.1: Four clock network topologies.

1.3 Clock Network Topologies

As mentioned in the previous section, a clock signal must be delivered to every flip-flops while the clock skew is minimized or does not excess certain bound. As the CMOS technology scales down to sub-micron, however, a small change in supply voltage or temperature causes larger effect on timing, which makes it more difficult to meet timing constraints.

To mitigate the effect of variation, several clock network topologies where some redundant clock signal paths are added. Fig. 1.1 shows four types of clock network topology, according to tolerance to variations. Clock tree in Fig. 1.1(a) has been most widely used clock networks for its simplicity. Instead, since each flip-flop receives the clock signal from only one buffer, the delay variation on a driving wire or a buffer directly induces timing variation, which results in worsening the clock skew. The structure of cross link in Fig. 1.1(b) has some wires that crosses between leaf nodes offering path redundancy, by which more tolerant to the clock skew.

Clock spine[24, 25, 26] in Fig. 1.1(c) is more skew-tolerant topology. The clock spine structure has special wires, called a *spine*, to which every flip-flop is connected in the close distance. Each spine is driven by multiple buffers, which offers multiple clock signal path to sink flip-flops so that the clock network becomes more tolerant to the variation.

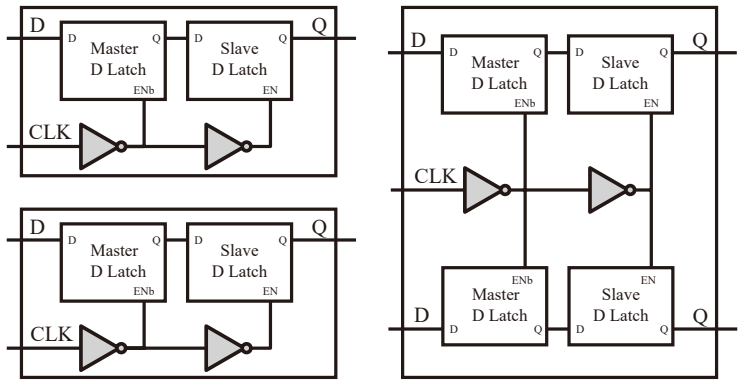
The structure in Fig. 1.1(d), having even more redundant clock path, called clock *mesh*[27, 28, 29]. A clock mesh structure consists of a grid-shaped clock mesh, sink flip-flops connected to the mesh in the close distance, buffers which directly drive the grid-shaped clock mesh, and finally a top level tree that drives the buffers. The clock mesh structure, due to its plenty of redundancy, has lowest clock skew and strong tolerance to variation. However, due to that

redundancy, large power consumption on the clock mesh is the biggest flaw of this structure.

1.4 Multibit Flip-flop

Many researches have been focused on decreasing clock power, as briefly mentioned in Section 1.2. Recently, studies on a multi-bit flip-flop have been conducted.

Fig. 1.2 represents the structure of 2-bit MBFF. The idea of MBFF is that by merging multiple flip-flops into a merged larger cell, cell power consumption can be minimized due to the removed clock inverters.



(a) 1-bit flip-flops (b) 2-bit flip-flop

Figure 1.2: Internal structure of 1-bit master-slave based flip-flops and 2-bit flip-flop in which the pairs of master-slave latches share the clock driving logic.

Obviously, an MBFF can reduce power consumption, however, have two drawbacks: (1) due to its large cell size, it may be not easy to find region to locate the MBFF. (2) Because an MBFF is a merged structure from multiple flip-flops, the original data path of the flip-flop should be routed again.

1.5 Simultaneous Switching Noise

In a synchronous digital circuits, a clock signal is delivered through clock distribution network to sequential elements. By the clock signal, all sequential elements (e.g., flip-flops) switch simultaneously at clock edges. This simultaneous switching causes a high peak current on the power/ground line, resulting in voltage fluctuation on the line. This is called as simultaneous switching noise (SSN) or power/ground bounce. The high peak current weakens the circuit performance and undermines the reliability of system [30].

To mitigate SSN, (1) distance between power/ground lines should be widen and distance from the ground line should be shortened, (2) current on a power/ground line should be dispersed, or (3) decoupling capacitors should be used to shield intrinsic inductance of power/ground line.

Among methodologies to mitigate SSN, this thesis focuses on minimizing the amplitude of peak current by dispersing peak current position of flip-flops, which will be described in Chapter 3.

1.6 Contributions of This Dissertation

In this dissertation, each chapter presents the new techniques which makes the important two energy metrics of a clock, namely power and current peak, be minimized.

In Chapter 2, clock tree and flip-flop co-optimization technique for reducing clock power is developed. Firstly, we introduce the new four types of hardware components that can implement a set of flip-flops and their driving buffer as a single unit. By exploiting these cells with buffer/inverter sizing technique, A new algorithm which can minimize power consumption in the clock boundary is proposed, while not violating clock skew and slew constraints, even with no

burden of cell relocation and net rerouting.

In Chapter 3, clock tree and flip-flop co-optimization technique for minimizing peak current noise is proposed. Firstly, we show that considering the new four types of cells for minimizing current peak is the extended version of conventional polarity assignment technique, then a new algorithm which can minimize supply noise with reducing current peak by exploiting the new four types of cells is developed. Without any burden of additional placement or rerouting, our algorithm can lower current peak and power supply noise, while meeting clock skew and slew constraints.

Chapter 2

Clock Tree and Flip-flop Co-optimization for Reducing Power Consumption

2.1 Introduction

As it was introduced in Changer , by removing the number of clock inverters, an MBFF is able to reduce clock power. several techniques of an MBFF have been proposed [31, 32, 33, 34, 35, 36, 37].

The representative flaws of an MBFF is that we should perturb the existing placement, and its data path should be detoured.

To overcome the problem, Moon and Kim [38] proposed a new flip-flop design called LC-MBFF (*loosely coupled multi-bit flip-flop*) to resolve the area and routing capacity constraints incurred by the generation and placement of MBFFs during the post-placement stage. The fundamental difference between the structures of LC-MBFF and normal MBFF is that LC-MBFF does not physically merge their constituent single-bit flip-flops. Instead, they logically

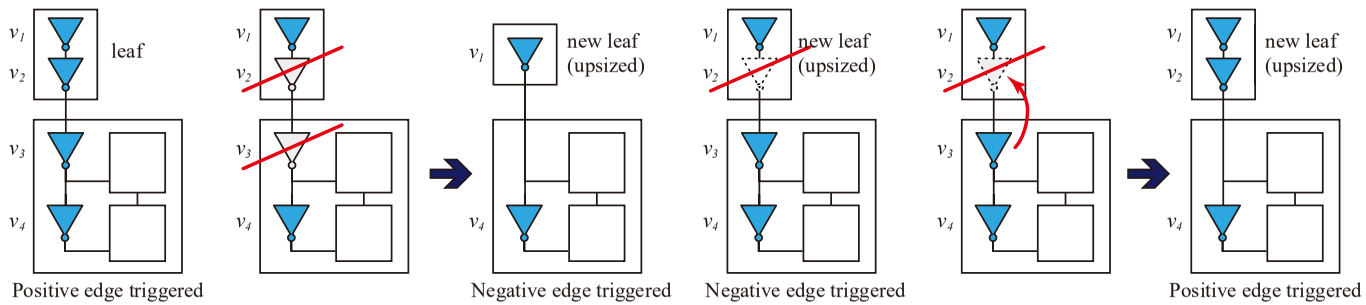
merge them by creating short (internal clock) wires to connect them without physically moving them. Our approach to the reduction of clock power including the flip-flop power is different from the conventional ones in that unlike the existing approaches in which the two tasks of buffer insertion/sizing and flip-flop optimization were performed independently with no interaction between them, our work focuses on a new (complementary) problem of optimizing the boundary between the clock tree and flip-flops, which has never been addressed by the prior works.

2.2 Types of Boundary Optimization

The problem of clock tree boundary optimization we want to solve is based on the following observation of the internal implementation of clock buffers and flip-flops.

1. *Boundary-unoptimized positive clock (FF^{+}_{2inv})*: Fig. 2.1(a) shows the logic on the clock signal path from a leaf of clock tree to a flip-flop. The leaf is represented with a buffer, made of two inverters v_1 and v_2, and the flip-flop has two clock inverters v_3 and v_4. Fig. 2.1(a) is the typical logic path of clock signal of conventional clock trees. Since a buffer (v_1 and v_2) drives the flip-flop, the polarity of clock signal to the flip-flop is positive. We use notation FF^{+}_{2inv} to refer this type of flip-flop.

2. *Boundary-optimized negative clock ($FF^{+}_{2inv} \rightarrow FF^{-}_{1inv}$)*: Fig. 2.1(b) shows an optimization of the logic on the clock signal path in Fig. 2.1(a), in which inverters v_2 and v_3 are cancelled, and inverter v_1 is up-sized to meet the clock time constraint. Our experiments show that the amount of power saved by removing the two clock inverters is more than the amount of power loss by the up-sizing of the inverter. Since inverter v_1 drives the



(a) FF^{+}_{2inv} (b) FF^{-}_{1inv} (c) FF^{-}_{2inv} (d) FF^{+}_{1inv}

Figure 2.1: Four types of flip-flops that can be exploited in our algorithm of clock tree boundary optimization. The four types exhibit different clock latencies and power consumptions.

optimized flip-flop, the polarity of clock signal to the flip-flop is changed to be negative. We use notation FF^{-}_{1inv} to refer this type of optimized flip-flop with negative polarity of input clock signal.

3. *Polarity-optimized negative clock ($FF^{+}_{2inv} \rightarrow FF^{-}_{2inv}$)*: Fig. 2.1(c) shows an optimization of the leaf buffer in Fig. 2.1(a), in which inverter v_2 is removed and the flip-flop is replaced with a negative edge triggered one. The power saving by this conversion in general would not be larger than that in Fig. 2.1(b). However, it enables to make a chance of boundary optimization if the resulting optimization does not violate the timing constraint at all. We use notation FF^{-}_{2inv} to refer this type of flip-flop which is driven by the transformed clock signal.

4. *Boundary-optimized positive clock ($FF^{+}_{2inv} \rightarrow FF^{-}_{2inv} \rightarrow FF^{+}_{1inv}$)*: Fig. 2.1(d) shows a boundary optimization of the leaf buffer in Fig. 2.1(c) and flip-flop, in which inverter v_2 is removed and v_3 in the flip-flop is grouped with v_1 to form a leaf buffer. The power saving can be achieved if the total power saved by the flip-flops with one clock inverter is larger than the power loss by the buffer that drives the flip-flop. We use notation FF^{+}_{1inv} to refer this type of optimized flip-flop with positive polarity of input clock signal.

Note that the upsizing of inverters in Fig. 2.1 may cause the white space problem. To take into account this problem, we set the size of the second inverter in every leaf clock buffer allocated in the initial clock tree to that of its one-level bigger one. For example, if an INV_X4 ($= 0.798um^2$) is allocated, its size is assumed to that of INV_X8 ($= 1.064um^2$) before placement. The rationale is that if an upsizing happened in the process of boundary optimization, in almost all cases, it was observed that it was upsized to that of its immediate

up-level. This scheme effectively solves the space problem. However, an area penalty occurs, but the area overhead is very low, which is measured to less than 0.3% increase of the total area of clock tree.

In addition, our low-power boundary optimization technique of the clock tree is performed after the placement of logic gates including flip-flops and the synthesis of the clock tree have been completed. Thus, our boundary optimization should not violate the clock skew and slew constraints, which have been preserved from the prior stages of design process.

2.3 Analysis of Four Types of Flip-flop

In this section, we analyze the power consumption characteristics of four types of flip-flop, in terms of inside and outside of a flip-flop.

2.3.1 Internal Power Comparison

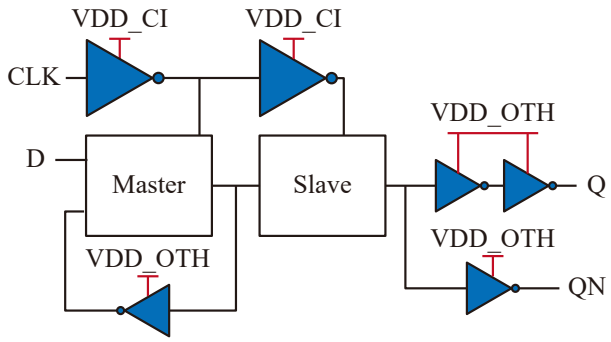


Figure 2.2: Measuring internal power of four types of flip-flops. Power consumption of inside of a flip-flop is measured by applying different supply voltage to each internal flip-flop elements. VDD_CI, VDD_ML, VDD_SL, and VDD_OTH mean the supply voltage applied to clock inverters, master latch, slave latch, and other components, like a feedback inverter, respectively.

Table 2.1: Amount of power consumption of four types of flip-flops, which are driven by a/an buffer/inverter. The input clock signal with input slew $30ps$ is passed through two inverters, INV_X4 and INV_X16 of Nangate 45nm technology, for more natural clock signal. Note that, compared with the conventional flip-flop, FF^{+}_{2inv}, the amount of power consumption in FF^{+}_{1inv} and FF^{-}_{1inv} reduces by 9.7% and 13.8% respectively when measured in only flip-flop itself, and 6.6% and 9.3% respectively when considering the driving buffer/inverter and flip-flop together.

Flip-flop	Power (uW)					Sum (uW)		%	
	BUF/INV	CI	ML	SL	OTH	p(FF)	p(total)	p(FF)	p(total)
FF^{+}_{2inv}	2.02	1.70	0.18	0.89	6.29	9.06	11.08	-	-
FF^{-}_{2inv}	1.78	1.70	0.17	0.92	6.28	9.07	10.85	**0.1**%	**-2.0**%
FF^{+}_{1inv}	2.17	0.80	0.17	0.92	6.29	8.18	10.35	**-9.7**%	**-6.6**%
FF^{-}_{1inv}	2.24	0.45	0.18	0.89	6.29	7.82	10.05	**-13.8**%	**-9.3**%

As shown in Fig. 2.1, the proposed four types of flip-flops have different power consumption characteristics due to their variant structure. To measure and compare the amount of internal power consumption of four types of flip-flops, we added additional VDD/VSS input pins in the flip-flop spice netlist. Each type of flip-flops is driven by a buffer or an inverter, depending on triggering edge of them, i.e., FF^{+}_{2inv} and FF^{+}_{1inv} are driven by BUF_X4, and FF^{-}_{2inv} and FF^{-}_{1inv} are driven by INV_X8, respectively. The clock input signal entering to these buffers/inverters are generated through two inverters from the primitive pulse with 30ps of input slew. The measured power are arranged in Table 2.1. As expected, FF^{-}_{2inv} consumes similar amount of power with FF^{+}_{2inv}, as they have the same number of clock inverters. In the other hand, FF^{+}_{1inv} and FF^{+}_{2inv} shows that they spend less amount of power by $9.7 sim 13.8\%$ than FF^{+}_{2inv}. With considering the driving buffers/inverters together, reduction decreases upto by $6.6 sim 9.3\%$, however, it still has possibilities to reduce power consumption if we allocate them properly.

2.3.2 Characterization of Power Consumption

Figs. 2.3(a) and (b) show the changes of the power consumption of the four types of flip-flop introduced in Fig. 2.1 as the output capacitance and transition time constraints of the driving leaf buffer (or inverter) change, respectively. For FF^{+}_{2inv} and FF^{-}_{2inv}, we used DFFRS_X1 in Nangate Open Cell library. For FF^{+}_{1inv} and FF^{-}_{1inv}, we removed one internal clock inverter in DFFRS_X1. We performed HSPICE simulation for the flip-flops by varying the output capacitance and transition time values. Note that besides each flip-flop itself, the power consumption in Figs. 2.3(a) and (b) includes that of the driving leaf buffer or inverter (i.e., the blue triangles of leaf in Fig. 2.1). Thus, the gap from each power curve of FF^{-}_{2inv}, FF^{+}_{1inv}, and FF^{-}_{1inv} to FF^{+}_{2inv} in Fig. 2.3 directly

indicates the amount of power saving achieved if the initial type of flip-flop (FF^{+}_{2inv}) is to be replaced with the type of flip-flop corresponding to the power curve. The gaps between curves indicate that the clock tree boundary power can be reduced by up to 9% if the clock tree boundary optimization is effectively performed. Our goal is to find a set of flip-flops of FF^{+}_{2inv} in an input circuit and replace them with the flip-flops of FF^{-}_{2inv}, FF^{+}_{1inv}, and FF^{-}_{1inv} with sink buffer optimization so that the resulting saving of clock power consumption should be maximized.

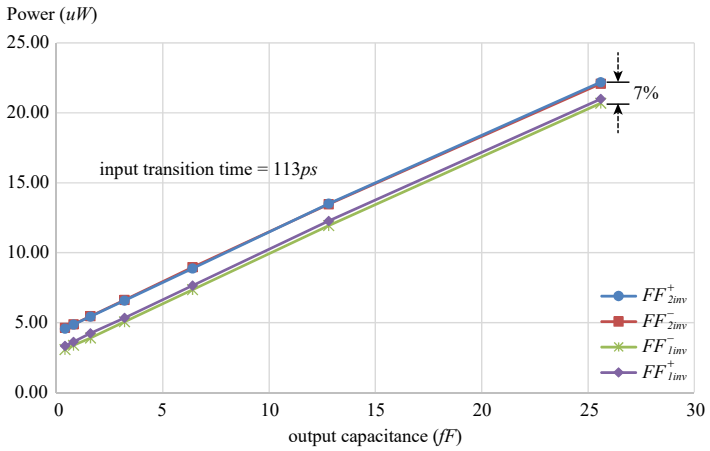
2.4 Problem Formulation

We formally describe the clock tree boundary cell optimization problem as follows:

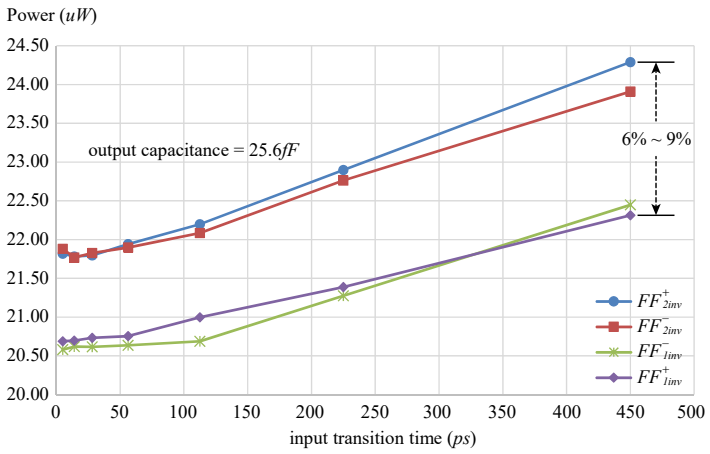
BoundaryMin problem: (**Clock tree boundary optimization for clock power minimization**) *Given a buffered clock tree $\mathcal{T}$ with a set E of (already allocated) leaf buffering elements, a library B of buffers, a library I of inverters, a set $\mathcal{R}$ of (already allocated) flip-flops driven by the cells in E, clock skew bound constraint δ, and clock slew rate constraint κ, replace the cells in E and $\mathcal{R}$ by finding mapping functions $\phi_1 : E \mapsto B \cup I$ and $\phi_2 : \mathcal{R} \mapsto \{FF^{+}_{2inv}, FF^{+}_{1inv}, FF^{-}_{2inv}, FF^{-}_{1inv}\}$ that*

$$\begin{aligned} minimize \quad & \sum_{e_i \in E} power(\phi_1(e_i)) + \sum_{f_j \in R} power(\phi_2(f_j)) \qquad (2.1) \\ s.\ t.\ & \max_{f_i \in \mathcal{R}}\{t_i\} - \min_{f_i \in \mathcal{R}}\{t_i\} < \delta, \\ & \max_{e_j \in E}\{s_j\} < \kappa \end{aligned}$$

where t_i and s_i are the clock arrival time at flip-flop $\phi_2(f_i)$ and the output slew rate of the driving buffer or inverter $\phi_1(e_i)$, respectively.



(a) Power curves with respect to output capacitance



(b) Power curves with respect to transition time

Figure 2.3: Comparison of power consumed by FF^{+}_{2inv}, FF^{-}_{2inv}, FF^{+}_{1inv}, and FF^{-}_{1inv}. (a) Power curves produced as the output driving capacitance changes. (b) Power curves produced as the driving transition time changes.

Note that BoundaryMin considers all possible flip-flop mappings, including the mapping corresponding to the input clock tree. Thus, BoundaryMin may return the input clock tree as a solution if its power cost is the least.

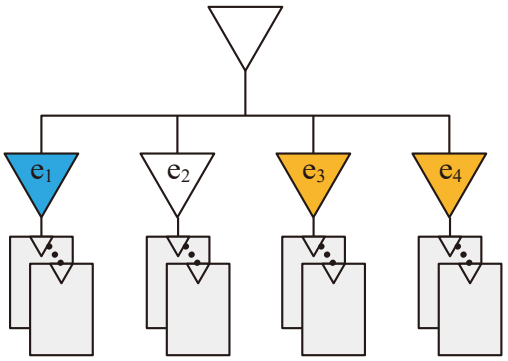
2.5 The Proposed Algorithm

2.5.1 Independence Assumption

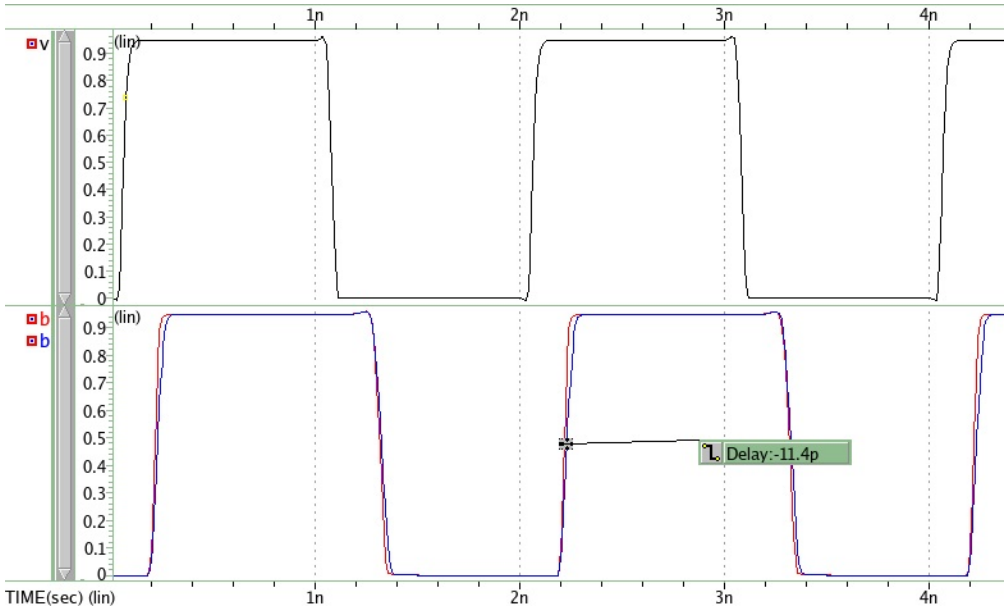
To simplify the approach, we assume that changing the cell type of a leaf node has little impact on its sibling nodes, as in [1]. However, the impact on delay variation is not negligible amount. To verify this, HSPICE simulation modelling ISCAS'89 benchmark circuit s35932 was executed. Among several leaf nodes, we choose one leaf node with 4 buffers as in Fig. 2.4(a). The leaf buffers $e_1 \sim e_4$ are initially mapped to BUF_X16. After replacing e_3 and e_4 into INV_X16, HSPICe simulation was executed again. This is one of the worst case in in terms of skew where half of the siblings have opposite polarity from the other half. Fig. 2.4(b) shows delay difference of clock signal measured on input of a buffer, e_1. The delay difference is roughly 11ps, when the clock period is 2ns. This is only 0.5% of the clock period. Change of flip-flop types in the lower level has little impact on the timing variation in the upper level. It is because the capacitance of flip-flop clock pin is not seen in the upper level, blocked by the input capacitance of the buffer. Hence, we assume that a cell type of each sink group can independently changed.

2.5.2 BoundaryMin Algorithm

Fig. 2.6 shows the positioning and steps of our optimization algorithm which solves the BoundaryMin problem in four steps: (Step 1) *Generating mapping candidates of flip-flops*, from which the resulting clock latency, clock transition time, and (local) clock power are extracted; (Step 2) *Sifting out the flip-flop*



(a)

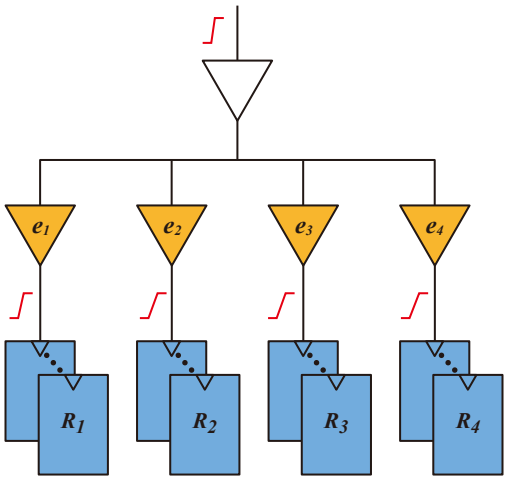


(b)

Figure 2.4: The effect of cell type change on sibling nodes. Simulation was performed on HSPICE with an ISCAS'89 benchmark circuit s35932, where P&R and clock tree synthesis was finished in IC compiler. We measured delay and slew values on one of the leaf nodes of s35932, shown in (a), all of which had initial cell type of BUF_X16. After replacing e_3 and e_4 with INV_16, HSPICE simulation was executed again. The clock signal at the input of e_1 before and after replacing e_3 and e_4 are plotted in (b).

Table 2.2: A summary of results produced by **Step 1**: clock latency, clock transition time, and clock power related data for each mapping candidate of flip-flops in set R_1 of Fig. 2.5(a). For simplicity, the table includes partially the resulting data of the mappings of FF^{+}_{2inv} and FF^{-}_{1inv} only.

FF type	B/I sizing	Timing extraction			Power-related data extraction		
		latency (ps)		max tran. (ps)	rise tran(ps)	fall tran. (ps)	output cap. (fF)
		fast corner	slow corner				
FF^{+}_{2inv}	BUF_X1	836	924	595	45	80	14.0
	BUF_X2	542	599	290	46	80	
	...	...	...	...	...	...	
	BUF_X16	428	474	84	46	80	
	BUF_X32	374	414	61	46	80	
FF^{-}_{1inv}	INV_X1	543	601	306	46	82	33.4
	...	...	...	...	...	...	
	INV_X8	279	309	70	75	90	
	INV_X16	277	307	45	60	86	
	INV_X32	270	298	25	53	83	
...	...	...	...	...	...	...	...



(a) A simple clock tree

worst skew		$8.4ps$
maximum transition		$65.1ps$
mapping	(e_1, R_1)	(BUF_X16, FF^{+}_{2inv})
	(e_2, R_2)	(BUF_X16, FF^{+}_{2inv})
	(e_3, R_3)	(BUF_X16, FF^{+}_{2inv})
	(e_4, R_4)	(BUF_X16, FF^{+}_{2inv})
clock power		$424.9\mu W$

(b) Initial data for (a)

Figure 2.5: A small example for illustrating our four-step algorithm. (a) A clock tree with four subsets R_1, R_2, R_3 and R_4 of flip-flops, each of which is driven by distinct buffering elements (sinks) e_1, e_2, e_3, and e_4, respectively. (b) Clock skew bound and slew rate constraints, and the clock power of the initial mapping of ϕ_1 and ϕ_2. 'Initial' mapping represents the types of clock buffers and flop-flops at the four clock leaves of the initial clock tree.

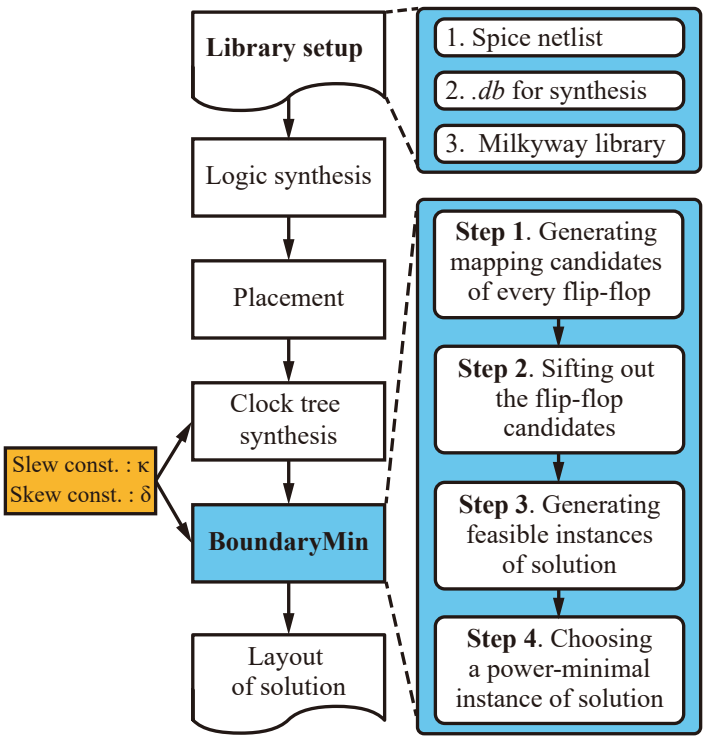


Figure 2.6: The positioning and steps of the proposed optimization of clock tree boundary. The library setup is described in detail in Sec.2.6.1.

Table 2.3: A summary of results produced by **Step 2**: The candidate mappings of flip-flops in the clock tree in Fig. 2.5(a) under $\kappa = 100ps$ are listed. The flip-flops that are driven by the same buffering element are mapped uniformly.

Flip-flop group	Flip-flop type	Buffer/Inverter sizes meeting trans. constraint
R_1	FF^{+}_{2inv}	BUF_X16, BUF_X32
	FF^{-}_{1inv}	INV_X8, INV_X16, INV_X32
	FF^{+}_{1inv}	BUF_X32
	FF^{-}_{2inv}	INV_X8, INV_X16, INV_X32
R_2	FF^{+}_{2inv}	BUF_X16, BUF_X32
	FF^{-}_{1inv}	INV_X4, INV_X8, INV_X16, INV_X32
	FF^{+}_{1inv}	BUF_X32
	FF^{-}_{2inv}	INV_X8, INV_X16, INV_X32
R_3	FF^{+}_{2inv}	BUF_X16, BUF_X32
	FF^{-}_{1inv}	INV_X4, INV_X8, INV_X16, INV_X32
	FF^{+}_{1inv}	BUF_X32
	FF^{-}_{2inv}	INV_X16, INV_X32
R_4	FF^{+}_{2inv}	BUF_X16, BUF_X32
	FF^{-}_{1inv}	INV_X4, INV_X8, INV_X16, INV_X32
	FF^{+}_{1inv}	BUF_X16, BUF_X32
	FF^{-}_{2inv}	INV_X4, INV_X8, INV_X16, INV_X32

candidates that violate the transition time (i.e., clock skew) constraint; (Step 3) *Enumerating feasible instances of solution from the combinations of flip-flop candidates* that satisfy the clock skew constraint; (Step 4) *Selecting a power-minimal instance* among those obtained from Step 3. We describe the four steps of our algorithm in the following, using a simple illustrative clock tree in Fig. 2.5(a).

Step 1 (*Generating mapping candidates of every flip-flop*): Table 2.2 partially summarizes the timing and power-related data extracted for the mapping candidates of the flip-flops in set R_1 of the clock tree in Fig. 2.5(a). Each mapping is described by the first and second columns of the table. For example, the first row corresponds to the mapping of flip-flop type FF^{+}_{2inv} to every flip-flop in R_1 and buffer type BUF_X1 to buffering element e_1. The three *Timing extraction* columns indicate the longest and shortest clock latencies, and the longest clock transition time to the flip-flops, respectively, from which we can compute the local clock skew in R_1 and the clock slew rate on e_1 to R_1 for the corresponding mapping. The last three columns are the data required for the calculation of clock power consumption for the mapping. (We used HSPICE to calculate the power consumption based on the parameter values.) The table shows, for brevity, only the mapping results for the mappings of FF^{+}_{2inv} and FF^{-}_{1inv}. We simulated process variation by expanding the gap between the best and the worst latencies to each flip-flop by -15% and +15% of its nominal latency, respectively. We mark the flip-flop candidates that meet the skew constraint for every combination of latencies.

Step 2 (*Sifting out the flip-flop candidates*): This step examines each of the mapping candidates obtained in Step 1 and filters out those that violate the clock slew constraint. For example, when the clock slew rate constraint $\kappa =$

Table 2.4: The list of feasible mappings and their clock power consumption for the clock tree in Fig. 2.5(a). **Step 4** selects the mapping with the least power consumption, which corresponds to the flip-flop allocation and buffer/inverter sizing in the first row.

no.	R_1		R_2		R_3		R_4		power
	FF type	B/I sizing	FF type	B/I sizing	FF type	B/I sizing	FF type	B/I sizing	(μW)
1	FF^{-}_{1inv}	INV_X8	FF^{-}_{1inv}	INV_X4	FF^{-}_{1inv}	INV_X4	FF^{+}_{1inv}	BUF_X4	**379** ($\surd$)
2	FF^{-}_{1inv}	INV_X8	FF^{-}_{1inv}	INV_X4	FF^{-}_{1inv}	INV_X4	FF^{+}_{1inv}	BUF_X8	382
3	FF^{-}_{1inv}	INV_X4	FF^{-}_{1inv}	INV_X4	FF^{-}_{1inv}	INV_X4	FF^{+}_{2inv}	BUF_X8	385
4	FF^{-}_{2inv}	INV_X8	FF^{-}_{2inv}	INV_X4	FF^{-}_{2inv}	INV_X8	FF^{-}_{2inv}	INV_X4	389
5	FF^{-}_{1inv}	INV_X4	FF^{+}_{2inv}	BUF_X8	FF^{-}_{1inv}	INV_X4	FF^{+}_{2inv}	BUF_X8	394
6	FF^{+}_{2inv}	BUF_X8	FF^{+}_{2inv}	BUF_X8	FF^{+}_{2inv}	BUF_X8	FF^{+}_{2inv}	BUF_X8	401
7	FF^{+}_{1inv}	BUF_X8	FF^{+}_{1inv}	BUF_X8	FF^{+}_{1inv}	BUF_X8	FF^{+}_{1inv}	BUF_X8	409
8	FF^{+}_{1inv}	BUF_X16	FF^{+}_{1inv}	BUF_X16	FF^{+}_{2inv}	BUF_X8	FF^{+}_{1inv}	BUF_X8	414
...	...	...	...	...	...	...	...	...	...

$100ps$, the mapping candidates (FF^{+}_{2inv}, BUF_X1), (FF^{+}_{2inv}, BUF_X2), and (FF^{-}_{1inv}, INV_X1) in Table 2.2 will be excluded from consideration. The mapping candidates of flip-flops in Fig. 2.5(a) under $\kappa = 100ps$ are shown in Table 2.3.

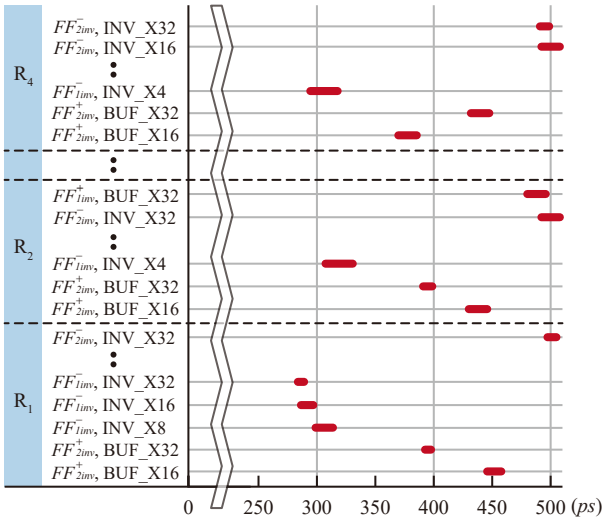
Step 3 (*Generating feasible instances of solution*): This step explores all possible solution instances that satisfy the clock skew constraint as well as the clock slew rate constraint. The exploration is performed by preparing the latency intervals of all candidate mappings produced in Step 2. Fig. 2.7(a) shows a *chart of latency intervals* of all candidate mappings where the x-axis represents the clock latency and the y-axis represents the mapping candidates. For example, the red short interval at (FF^{+}_{2inv}, BUF_X16) is [428, 474], the values of which are respectively the latencies of earliest and latest corners in the row of (FF^{+}_{2inv}, BUF_X16) in Table 2.2. Then, we slide the interval chart from right to left by using *clock skew window*, which is a rectangular box whose width (i.e., horizontal interval) equals the clock skew bound constraint δ. Fig. 2.7(b) shows the clock skew window that starts scanning the latency interval chart. We sort the latency intervals on the chart according to their max latency values of the intervals. Let $(t_1, t_2, \cdots, t_M)$ be the sorted max values of the latency intervals. Then, the clock skew windows to be checked are those windows of intervals $[t_1 - \delta,\ t_1]$, $[t_2 - \delta,\ t_2]$, $\cdots$, $[t_M - \delta,\ t_M]$.

The window of an interval $[t_i - \delta,\ t_i]$ is called *feasible* if the window contains at least one latency interval on the rows corresponding to each flip-flop set driven by the same buffer or inverter. For example, the starting window in Fig. 2.7(b) is feasible since each of $R_1, \cdots, R_4$ has at least one latency interval. Then, we compute, for every feasible clock skew window, a *feasible instance of mapping solution* that minimizes the quantity in Eq.(1) while considering the ranges of mappings ϕ_1 and ϕ_2 for each flip-flop set R_i are the pairs of

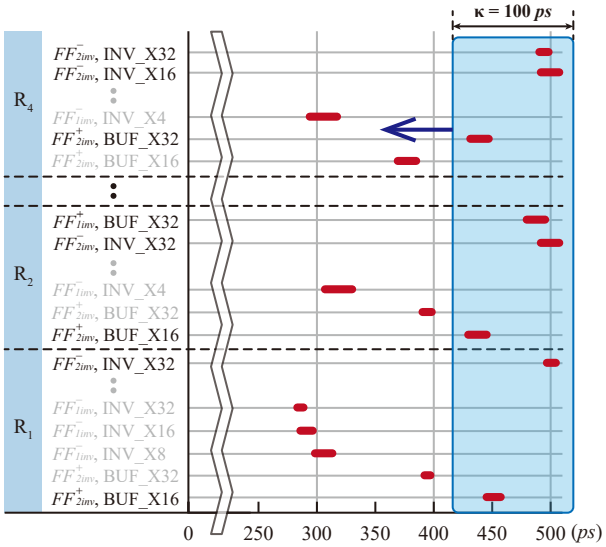
buffers/inverters and flip-flops corresponding to the latency intervals on that window.

Step 4 (*Choosing a power-minimal instance of solution*): From the feasible instance of solution, we choose the solution instance which has the least power consumption. Thus, the boundary optimization result corresponding to the chosen one makes sure that the clock tree with the mapped types of flip-flops and driving buffers/inverters consumes a minimal (boundary) clock power while still meeting the clock skew and slew constraints. Table 2.4 lists the feasible mappings obtained from Step 3 and their clock power consumption for the clock tree in Fig. 2.5(a). This step selects the mapping on the first row which consumes the least power. The corresponding mapping is (FF^{-}_{1inv}, INV_X8) for R_1, (FF^{-}_{1inv}, INV_X4) for R_2 and R_3, and (FF^{+}_{1inv}, BUF_X4) for R_4. Thus, the clock power is reduced from $424.9uW$ (in Fig. 2.5(b)) to $379uW$, which amounts to 10.8% power reduction.

Time complexity: Let $R_1, \cdots, R_K$ be the input flip-flop sets, each of which is driven by the same buffering element, and B and I are the types of buffers and inverters to which a buffering element can be mapped. Since the timing and power numbers will be extracted for every possible mapping of input flip-flop sets, the number of applications of extraction tool (we used Synopsys' IC compiler in this work) is bounded by $K \cdot 4 \cdot (|B| + |I|)$, which equals the maximal number of rows in Table 2.2. Thus the run time is $O(K \cdot (|B| + |I|) \cdot H)$ where H indicates the design tool's timing/power extraction time run for a section of the clock sub-tree only containing a flip-flop set with the same driving buffering element. Since checking if a mapping candidate (i.e., each row in Table 2.2) violates the slew constraint or not can be done in a constant time, Step 2 takes $O(K \cdot (|B| + |I|)$.



(a) Latency interval chart of Table 2.2



(b) Enumeration of solution instances by using clock skew window.

Figure 2.7: An illustration of **Step 3** which exhaustively explores solution instances using concept of latency interval chart and clock skew window.

```
1: function BoundaryMin(T, F, B, I, δ, κ)
2:     // T: input clock tree with flip-flip sets R_1, ··· , R_K
3:     // F: { FF^+_{2inv}, FF^-_{1inv}, FF^-_{2inv}, FF^+_{1inv} }   ▷ used in
   Step 1
4:     // B, I: buffer and inverter libraries   ▷ used in Step 1
5:     // δ, κ: clock skew bound and slew rate constraints
6:     D ← Mapping candidates for R_1, ··· , R_K;   ▷ Step 1
7:     L ← Candidates in D satisfying slew < κ;   ▷ Step 2
8:     L ← Latency intervals in L in decreasing order;
9:     P_min ← ∞;
10:    Sol ← ∅;
11:    // Explore feasible solutions using clock skew window ▷
   Step 3
12:    for [t_i − δ, t_i] ∈ L do
13:        l_i ← [t_i − δ, t_i];
14:        if exist_feasible_mapping(l_i, {R_1, ··· , R_K}) then
15:            P_{l_i} ← min. power on [t_i − δ, t_i];
16:            if P_{l_i} < P_min then   ▷ Step 4
17:                P_min ← P_{l_i};
18:                Sol ← mapping of min-power on l_i;
19:            end if
20:        end if
21:    end for
22:    return Sol;
23: end function
```

Figure 2.8: Algorithm of synthesizing a power-minimal clock tree boundary cells.

In Step 3, the number of windows to be checked is bounded by $K \cdot 4 \cdot (|B|+|I|)$ since it amounts to the maximal number of rows a latency interval chart can have. Since we can find, for a feasible window, a power minimal assignment of the flip-flops together with compatible buffers/inverters for each input flip-flop set in a window in $O(K \cdot (|B| + |I|)$, the run time of Step 3 is bounded by $O(K^2 \cdot (|B| + |I|)^2)$. The last step can be done in constant time. Thus, the complexity of our clock tree boundary optimization takes $O(K^2 \cdot H \cdot (|B|+|I|)^2)$ where K is the input flip-flop sets and H is the (local) timing/power extraction time for a section of the clock subtree containing an input flip-flop set. The pseudo code of the algorithm is shown in Fig. 2.8.

2.6 Experimental Results

2.6.1 Experimental Setup

Our proposed algorithm BoundaryMin has been implemented in tcl scripts and Python language on a Linux machine and tested on ISCAS'89 benchmark circuits. The benchmarks were synthesized using Synopsys's Design compiler and clock trees were synthesized with Synopsys' IC compiler, using Nangate 45nm Open Cell library [39].

To make the benchmarks synthesizable with the new types of flip-flop FF^{+}_{1inv}, FF^{-}_{2inv}, and FF^{-}_{1inv} in the interaction with Synopsys' Design compiler and IC compiler, we have prepared the following libraries: we created a Milkyway reference library from Synopsys to import them into IC compiler. We also modified liberty library and generated *.db* file with Synopsys' Library compiler to be synthesizable by Design compiler and to be timing available by IC compiler. In addition, for HSPICE simulation, we created spice netlist files for the new types of flip-flop from the original D flip-flops in Nangate Open Cell library.

Table 2.5: Boundary optimization results by our BoundaryMin for clock trees synthesized using buffer library {BUF_X1, BUF_X2, BUF_X4, BUF_X8, BUF_X16, BUF_X32}. The three values in the parentheses of the columns labelled *Power*, from left to right, indicate respectively the total power consumed by the leaf clock buffers/inverters, by flip-flops, and by the rest of clock tree.

Circuits	#FF	#Lev.	$\#\mathcal{R}$	Input clock tree						Clock tree by BoundaryMin						power saving	power saving
				skew	slew	power (uW)				skew	slew	power (uW)					
			FF group	(ps)	(ps)	L.bufs	L.ffs	others	sum	(ps)	(ps)	L.bufs	L.ffs	others	sum	(total)	(pure)
s1423	74	3	**37**	12.9	22.19	78.05	640.8	44.44	763.29	19.2	37.99	68.41	571.6	49.66	689.67	**-9.65 %**	**-10.97%**
s15850	134	3	**66**	13.9	22.38	136.1	1156	83.67	1375.77	22.5	36.85	116.2	1031	92.87	1240.07	**-9.86 %**	**-11.21%**
s5378	163	3	**91**	17.7	24.38	199.7	1502	93.21	1794.91	26.5	46.91	156.3	1360	100.6	1616.9	**-9.92 %**	**-10.89%**
s13207	330	3	**160**	13.2	25.06	335.1	2842	174.5	3351.6	23	38.98	300.4	2533	195.4	3028.8	**-9.63 %**	**-10.82%**
s38584	1168	3	**618**	28.2	24.55	1321	10070	678	12069	35.1	39.87	1127	8980	752.8	10859.8	**-10.02%**	**-11.27%**
s38417	1154	3	**781**	22.6	25.9	1669	13480	794.5	15943.5	34.4	40.92	1458	12040	900.1	14398.1	**-9.69 %**	**-10.90%**
s35932	1728	2	**875**	20.3	23.6	1863	14860	986.9	17709.9	31.5	39.45	1615	13260	1089	15964	**-9.86 %**	**-11.05%**
											Avg.	**↓25.38%**	**↓10.14%**	**↑17.06%**	**↓11.28%**	**-10.17%**	**-10.84%**

Table 2.6: Boundary optimization results by our BoundaryMin for clock trees synthesized using buffer library {BUF_X8, BUF_X16, BUF_X32}. The three values in the parentheses of the columns labelled *Power*, from left to right, indicate respectively the total power consumed by the leaf clock buffers/inverters, by flip-flops, and by the rest of clock tree.

Circuits	#FF	#Lev.	$\#\mathcal{R}$	Input clock tree						Clock tree by BoundaryMin						power saving	power saving
				skew	slew	power (uW)				skew	slew	power (uW)					
			FF group	(ps)	(ps)	L.bufs	L.ffs	others	sum	(ps)	(ps)	L.bufs	L.ffs	others	sum	(total)	(pure)
s1423	74	3	**8**	17.6	42.44	59.52	640.8	14.46	714.78	8.17	35.78	67.27	571.8	22.38	661.45	**-7.46%**	**-8.75 %**
s15850	134	3	**16**	16.1	41.95	124	1156	19.06	1299.06	19.14	35.28	118.8	1031	34.75	1184.55	**-8.81%**	**-10.17%**
s5378	163	3	**19**	24.1	38.43	168	1501	28.24	1697.24	13.8	40.22	146.8	1360	47.69	1554.49	**-8.41%**	**-9.72 %**
s13207	330	3	**41**	26.9	41.97	280.9	2841	56.8	3178.7	19	36.98	291.8	2535	93.86	2920.66	**-8.12%**	**-9.45 %**
s38584	1168	3	**145**	19.4	41.71	976.9	10070	266.7	11313.6	23.2	38.53	1067	8985	396.6	10448.6	**-7.65%**	**-9.01 %**
s38417	1154	3	**190**	30.2	42.38	1247	13480	270.7	14997.7	31.2	42.42	1403	12040	447.8	13890.8	**-7.38%**	**-8.72 %**
s35932	1728	2	**208**	28.3	42.05	1389	14850	377.3	16616.3	22.3	38.46	1553	13250	565.8	15368.8	**-7.51%**	**-8.84 %**
											Avg.	**↑9.48%**	**↓10.70%**	**↑55.71%**	**↓7.60%**	**-7.91%**	**-9.24%**

Table 2.7: Boundary optimization results by our BoundaryMin for clock trees synthesized using buffer library {BUF_X16, BUF_X32}. The three values in the parentheses of the columns labelled *Power*, from left to right, indicate respectively the total power consumed by the leaf clock buffers/inverters, by flip-flops, and by the rest of clock tree.

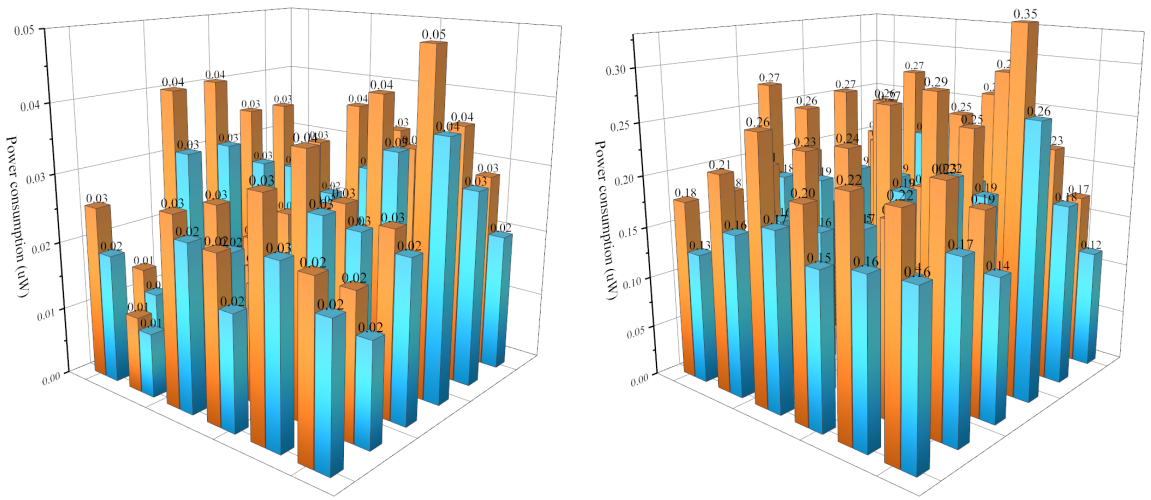
Circuits	#FF	#Lev.	$\#\mathcal{R}$	Input clock tree						Clock tree by BoundaryMin						power saving	power saving
				skew	slew	power (uW)				skew	slew	power (uW)					
			FF group	(ps)	(ps)	L.bufs	L.ffs	others	sum	(ps)	(ps)	L.bufs	L.ffs	others	sum	(total)	(pure)
s1423	74	3	**6**	6.4	49.27	94.99	642.2	27.21	764.4	10.4	97.66	64.06	578.1	30.09	672.25	**-12.06%**	**-12.89%**
s15850	134	3	**8**	9.8	47.48	99.81	1157	28.18	1284.99	25.09	66.23	113	1035	35.01	1183.01	**-7.94%**	**-8.66%**
s5378	163	3	**11**	11.8	44.96	138.3	1503	31.62	1672.92	23.6	81.84	137.2	1366	38.35	1541.55	**-7.85%**	**-8.41%**
s13207	330	3	**21**	15	47.9	250.9	2847	60.17	3158.07	30.49	70.41	276	2545	78.74	2899.74	**-8.18%**	**-8.94%**
s38584	1168	3	**76**	36	51.53	1436	10100	293.8	11829.8	36.4	96.29	983.2	9094	327.9	10405.1	**-12.04%**	**-12.65%**
s38417	1154	3	**97**	37.6	51.38	1681	13520	276.7	15477.7	36.8	78.07	1323	12090	352.5	13765.5	**-11.06%**	**-11.76%**
s35932	1728	2	**101**	40.4	52.13	2092	14900	372.1	17364.1	39.2	97.62	1426	13430	413.1	15269.1	**-12.07%**	**-12.57%**
											Avg.	**↓13.58%**	**↓10.72%**	**↑11.39%**	**↓9.83%**	**-9.80%**	**-11.02%**

All benchmark circuits were implemented with operating clock frequency of 500MHz and VDD of 0.95V. RC information of the clock trees before and after applying our algorithm was extracted in SPEF format from IC compiler for HSPICE simulation.

2.6.2 Clock Tree Boundary Optimization Results

Experiments are performed on three versions for a set of benchmark clock trees by varying the content of buffer library available to use. The forth columns (labeled by #R) in Tables 2.5, 2.6, and 2.7 indicate the numbers of flip-flop groups driven by the same clock buffering elements produced by the IC compiler when the buffer libraries are set to {BUF_X1, BUF_X2, BUF_X4, BUF_X8, BUF_X16, BUF_X32}, {BUF_X8, BUF_X16, BUF_X32}, and {BUF_X16, BUF_X32} from Nangate Open Cell library, respectively. The fifth, sixth, and seventh columns of the tables show the (global) clock skew, maximum clock slew, and the power consumption of the initial clock trees. For all testcases, we set the clock skew constraint (δ) to 100ps and slew constraint (κ) to 100ps.

The columns labelled as *power* represent the amount of power consumption before and after the application of our BoundaryMin, in which the three values labelled as *L.buf*, *L.ffs* and *others*, indicate respectively the total power consumed by in the parentheses of the columns labelled *Power*, the leaf clock buffers/inverters, by flip-flops, and by the rest of clock tree. The last two columns show the power saving by total power consumption of clock tree and by power consumption of the leaf clock buffers/inverters and flip-flops only, respectively. All measurements were done by HSPICE with SPEF-formatted file from IC compiler for the clock trees before and after the application of BoundaryMin. In summary, BoundaryMin is able to reduce the clock power by 7.9∼10.2% on average. It should be noted that the two advantages of Bound-

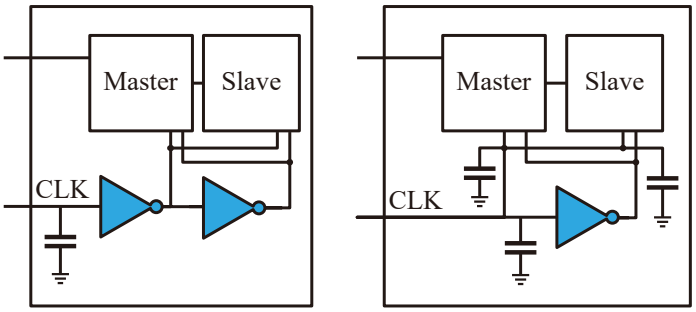


Power distribution for s5378 Power distribution for s38417

Figure 2.9: Distribution of flip-flop power consumption for (a) s5378 in Table 2.5 and (b) s38417 in Table 2.7. It shows that by properly utilizing our BoundaryMin algorithm to minimize the peak power, rather than to minimize the total power, the power/ground noise caused by clock tree will be controllable.

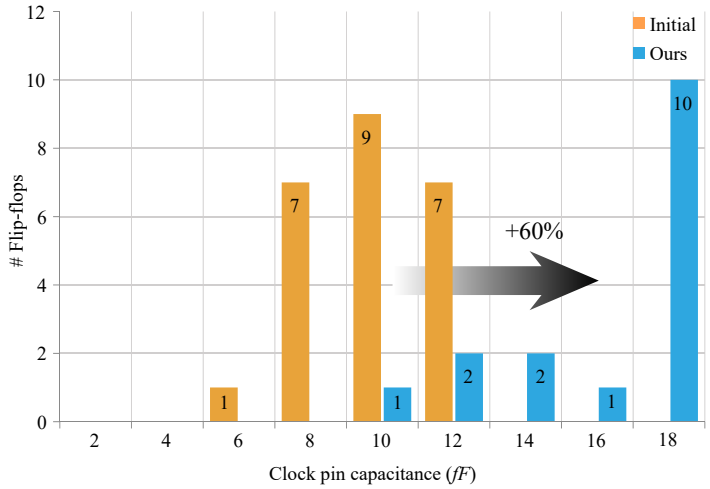
Table 2.8: Boundary optimization results by our BoundaryMin under restricted timing and power constraints for clock trees synthesized using buffer library {BUF_X16, BUF_X32}. Each column indicates the number of used flip-flop types and their percentage. Under more restricted conditions, flip-flop types can diversely selected by our BoundaryMin.

Circuits	# FF^{+}_{2inv}	# FF^{-}_{2inv}	# FF^{-}_{2inv}	# FF^{-}_{1inv}	# Total	FF^{+}_{2inv} (%)	FF^{-}_{2inv} (%)	FF^{-}_{2inv} (%)	FF^{-}_{1inv} (%)	Total (%)
s1423	1	2	0	3	6	17	33	0	50	100
s15850	1	3	0	4	8	13	38	0	50	100
s5378	0	3	1	7	11	0	27	9	64	100
s13207	2	6	0	13	21	10	29	0	62	100
s38417	9	28	0	60	97	9	29	0	62	100
s38417	2	41	0	33	76	3	54	0	43	100
s38417	7	42	0	52	101	7	42	0	51	100



FF^{+}_{2inv}, FF^{-}_{2inv} FF^{+}_{1inv}, FF^{-}_{1inv}

(a) Input capacitances on clock pin



(b) Distribution of clock pin capacitances for s5378

Figure 2.10: (a) Input capacitances seen at the front of clock pin before and after one of the clock inverters is removed. (b) The distribution of input capacitances at clock pin of all flip-flops in s5378 before and after the application of BoundaryMin.

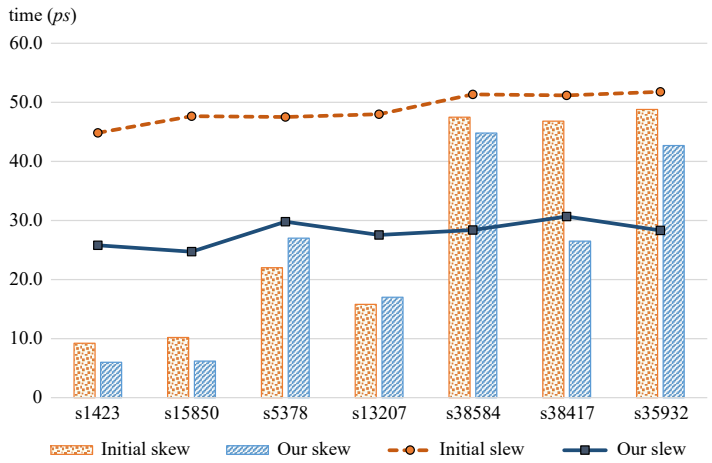


Figure 2.11: Comparison of the clock skew and maximum slew between the input clock trees and the optimized clock trees by BoundaryMin. The initial clock trees are generated with buffers BUF_X8, BUF_X16, and BUF_X32. It reveals that the differences are well controlled by BoundaryMin under the clock skew and slew constraints.

aryMin in design process is that it does not change the placement of flip-flops as well as routing of the input clock trees at all and BoundaryMin is also able to cope with process variation.

Table 2.9: More restricted timing and power constraints.

Constraints	FF^{+}_{2inv} (%)	FF^{-}_{2inv} (%)	FF^{+}_{1inv} (%)	FF^{-}_{1inv} (%)
slew	-17	-3	-49	30
latency_shift	-5	-5	0	10
latency_expand	-10	-10	-10	0
power_increase	-80	-10	-41	40
p_select_sink	80	40	55	80

Figs. 2.9(a) and (b) show the comparison of the distribution of flip-flop power consumption before and after the boundary optimization by our BoundaryMin for s5378 in Table 2.5 and s38584 in Table 2.7, respectively. We divided the layout of each circuit into 6×6 grids and measured the power consumption on those grids. It shows that the peak powers (the yellow bars) in the initial clock trees are considerably reduced, which implies that the power/ground noise will be well-controllable if our BoundaryMin algorithm can be exploited properly.

2.6.3 Capacitance Analysis on Flip-flops

The modified flip-flops FF^{+}_{1inv}, FF^{-}_{2inv}, and FF^{-}_{1inv} of FF^{+}_{2inv} used in our work have different net connections inside the HSPICE netlist. Specifically, the capacitance seen at the clock pin which was *shielded* by a clock inverter in FF^{+}_{2inv} is revealed after the removal of the inverter in the modified flip-flops FF^{+}_{1inv} and FF^{-}_{1inv}, which may cause to increase the clock pin capacitance, as shown in Fig. 2.10(a). It shows that the pin capacitances after optimization increase by about 60%, which is not trivial, as shown in Fig. 2.10(b). BoundaryMin

takes into account this effect by carefully resizing buffers and inverters in order not to violate the timing constraints.

2.6.4 Slew and Skew Analysis

Fig. 2.11 shows the changes of the worst skew and maximum slew before and after the application of BoundaryMin. The target skew and target slew each is set to $100ps$. It shows the worst skew after applying our algorithm slightly increases. The increase is mainly due to the clock pin capacitance increase as discussed in Sec.2.6.3 i.e., removing one of the clock inverters increases the clock pin capacitance. With a delicate buffer/inverter sizing by BoundaryMin under the clock skew and maximum slew constraints, no timing violation is observed even though it causes a slight increase of clock skew.

2.6.5 Window Width Analysis

Table 2.10 and Fig. 2.12 show the amount of reduced power consumption while narrowing skew window width κ in Fig. 2.7 from 100 (ps). BoundaryMin selects one set of a leaf buffer and leaf flip-flops types with least power consumption which are covered by the skew window with width of κ. As κ scales down, there exist the fewer mapping candidates, thus, the amount of power consumption we can reduce will be less. As shown in Table 2.10, the percentage of reduced power consumption is from 12.07% to 3.89%. Below window width of 70 ps, there is no solution due to skew violation.

The reason why the worst skew does not reduce while narrowing skew window width is that when measuring latency and transition values in IC compiler, the input capacitance of sibling clock pins will be able to be changed in final mapping. To cope with this matter, we iteratively solve BoundaryMin to skip the solutions which violate skew constraint.

Table 2.10: Reduced power variation along with window width scaling. L.buf and L.ffs mean the leaf buffering elements and leaf flip-flops, respectively. As the window width scales down from 100 *ps*, the amount of reduced power consumption shrinks together. Below the window width of 65 *ps*, there is not any solution which satisfies the given window width.

width	timing (ps)		power (mW)				reduced	reduced
	skew	slew	others	L.buf	L.ffs	sum	(total)	(pure)
base	40.4	52.13	0.372	2.09	14.90	17.36	-	-
0.1	39.2	97.62	0.413	1.43	13.43	15.27	-12.07%	-12.57%
0.095	53.7	78.97	0.585	1.59	13.28	15.46	-10.99%	-12.49%
0.09	53.8	78.98	0.573	1.58	13.29	15.44	-11.08%	-12.51%
0.085	33.5	97.62	0.391	1.28	14.03	15.70	-9.60%	-9.92%
0.08	42.3	97.62	0.382	1.19	14.36	15.93	-8.28%	-8.52%
0.075	41.5	97.62	0.382	1.17	14.43	15.98	-7.95%	-8.18%
0.07	40.5	93.67	0.369	1.92	14.40	16.69	-3.89%	-3.96%
0.065	No Solution							

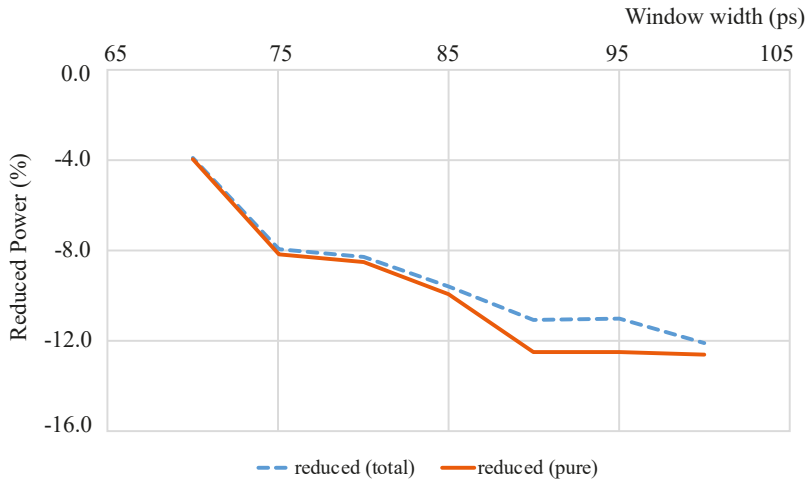


Figure 2.12: Graphical representation of Table 2.10.

2.7 Conclusions

This work solved a new problem of optimizing the boundary of buffered clock trees, which has never been addressed as yet. We showed that the clock buffering elements that directly drive flip-flops should not necessarily be buffers. Specifically, we showed that by co-optimizing the cells in both buffering elements and flip-flops, the boundary of clock tree could be optimized to save clock power further. Through experiments with benchmark circuits, it was shown that that our boundary optimization algorithm for clock trees was able to reduce the clock power by 7.9%∼10.2%on average with no timing violation, even with no burden of cell relocation and net rerouting.

Chapter 3

Clock Tree and Flip-flop Co-optimization for Reducing Power/Ground Noise

3.1 Introduction

In a synchronous digital circuits, a clock signal is delivered through clock distribution network to sequential elements. By the clock signal, all sequential elements (e.g., flip-flops) switch simultaneously at clock edges. This simultaneous switching causes a high peak current on the power/ground line, resulting in voltage fluctuation on the line. This is called as simultaneous switching noise (SSN) or power/ground bounce. The high peak current weakens the circuit performance and undermines the reliability of system [30].

Since clock buffers consume the current at the clock edges, a large amount of current is generated around the clock edges, which lets the clock buffers be one of the major sources of power/ground noise. For this reason, many researches have made efforts to divert peak current by exploiting clock skew

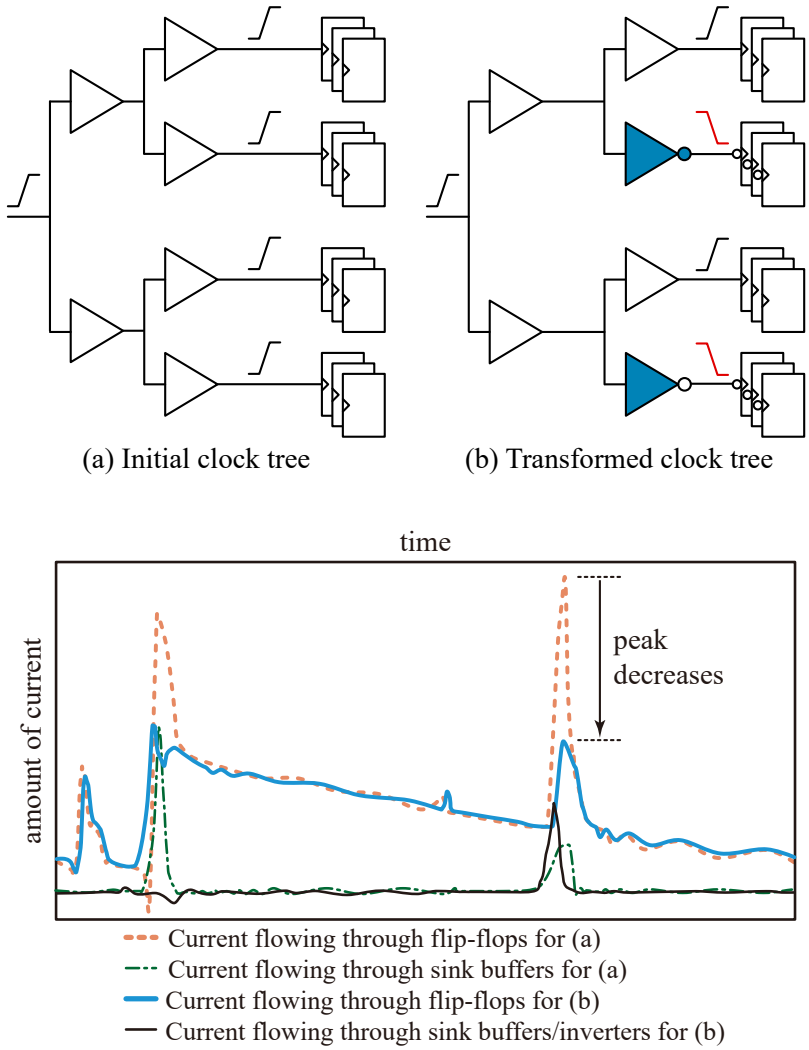


Figure 3.1: Peak current profile for a buffered clock tree of circuit s5378. (a) An initial clock tree. (b) A clock tree produced by replacing two buffers in (a) with inverters. (c) The current (charging) flows for (a) and (b) caused by sink buffers/inverters and flip-flops.

scheduling (e.g., [40, 41, 42, 43, 44]), in which they tried to disperse peak current by manipulating delay under clock skew constraint. However, for designs with bounded clock skew constraint, the applicability is strictly limited. Clock buffer polarity assignment (e.g., [45, 46, 47, 1]) is another technique used for reducing power/ground noise, which exploits the fact that a current peak of buffer (i.e., positive polarity) and inverter (i.e., negative polarity) appears at different clock triggering edge (rising and falling). An illustrative example is shown in Figs. 3.1(a) and (b), which are an initial clock tree of circuit s5378 with four sets of flip-flops and a clock tree obtained by simply replacing the two sink buffers in Fig. 3.1(a) with inverters. Mixing buffers and inverters at the boundary of clock tree is intended to disperse the power/ground noise from/to I_{DD}/I_{SS} at rising/falling edge of clock signal. (It also requires to replace the flip-flops driven by inverters with negative-edge triggered flip-flops.) To see how much the current is charged over time around clock tree boundary, we have conducted an HSPICE simulation for the clock trees in Figs. 3.1(a) and (b). The yellow and green dotted curves in Fig. 3.1(c) show the changes of the charging current i.e., power noise over time at the flip-flops and sink buffers in Fig. 3.1(a), respectively. It shows a very high current at the flip-flops. The blue and black solid curves in Fig. 3.1(c) show the changes of the charging current over time at the flip-flops and sink buffers/inverters in Fig. 3.1(b), respectively. In comparison with the dotted curves, it is shown that the peak current of the solid curves is considerably reduced.

Note that even though the technique can be applied to clock skew bounded designs as well as designs with clock skew scheduling, it is not able to control the current caused by the flip-flops on the boundary of clock tree. This work overcomes the limitation by extending the concept of polarity assignment on the clock tree boundary to further reduce the peak current at the clock tree

boundary including flip-flops.

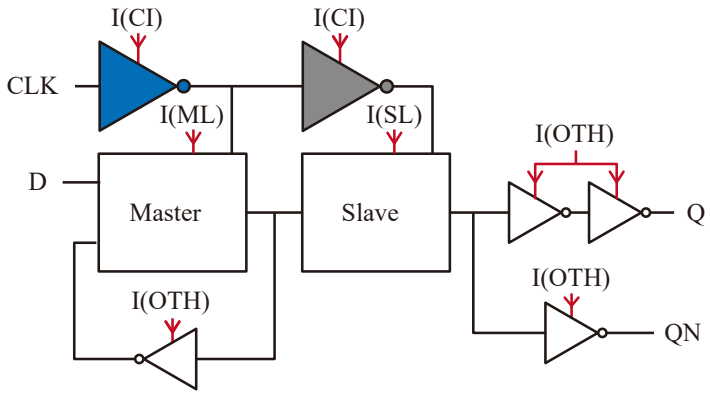
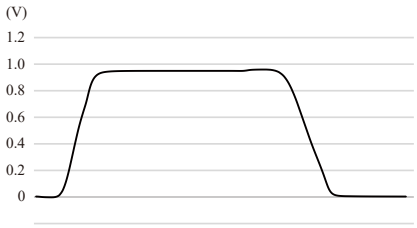


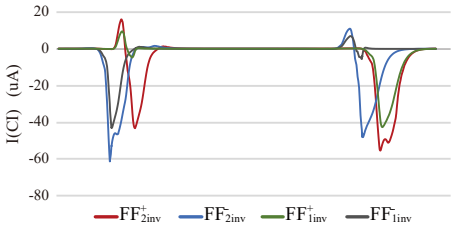
Figure 3.2: Proving currents flowing in a flip-flop. I(CL), I(ML), I(SL), and I(OTH) means current which flows into clock inverters, a master latch, a slave latch, and other components, respectively. The first clock inverter, marked as blue color, is absent in FF^{+}_{1inv} and FF^{-}_{1inv}.

3.2 Current Characteristic of Four Types of Flip-flop

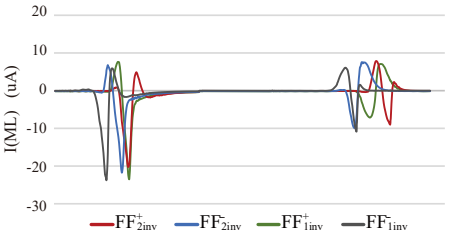
To compare current characteristic of four types of flip-flop, we added additional VDD pins to the spice netlist and measured currents, as shown in 3.2. Fig. 3.3 shows current profiles each of which is measured clock inverters, a master latch, slave latch, and other components, like a feedback inverter, respectively. Firstly, current flowing to clock inverters in (Fig. 3.3(a)) shows the biggest difference between four types of flip-flop. Since the inverter marked with blue color in 3.2 is absent in FF^{-}_{2inv} and FF^{-}_{1inv}, currents flowing clock inverters in FF^{+}_{2inv} and FF^{-}_{2inv} have the largest peak. In Fig. 3.3(b) and (c), currents flowing to the master and slave latch shows little difference among them. Note that, the master and slave latch in FF^{-}_{1inv} draws current more earlier that other flip-flops on account of a driving inverter, slightly faster than a buffer,



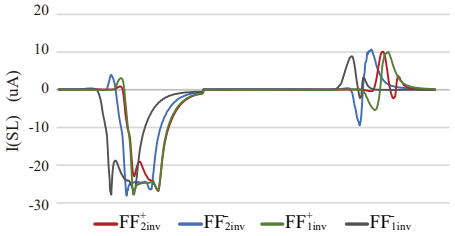
(a) A clock signal



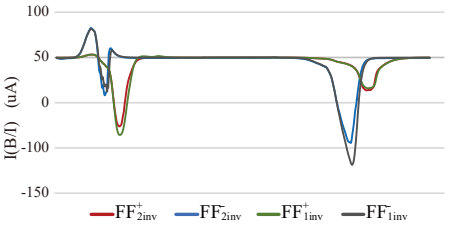
(b) Currents flowing into internal clock inverters



(c) Currents flowing into internal master latch



(d) Currents flowing into internal slave latch



(e) Currents flowing into buffers and inverters driving four types of flip-flops

Figure 3.3: Currents flowing from power supply into (b) internal clock inverters($I(CI)$), (c) master latch($I(ML)$), (d) slave latch($I(SL)$), and (e) driving buffers/inverters of flip-flop($I(B/I)$). Left and right fluctuation from (b) to (c) is derived at clock rising edge and falling edge, respectively. Since the four types of flip-flops exhibit different amount of peak current and time when peak current occurs, can be exploited in our algorithm to disperse the peak current in a clock tree.

and one clock inverter. Meanwhile, Fig. 3.3(d) shows the currents of a driving buffer/inverter of four types of flip-flops. The inverter driving FF^{-}_{1inv} has the largest peak. That is, even if current peak can be lowered by replacing other types of flip-flops, a driving buffer can show even larger peak. In this regards, to minimize the current peak, we carefully choose cell types of buffers/inverters and flip-flops.

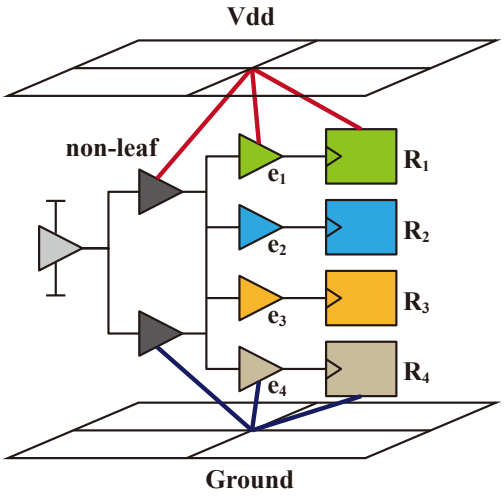


Figure 3.4: An example circuit for a motivational example.

3.3 Motivational Example

The sink flip-flops, due to their large numbers, are the most contributor to the current peak, even than leaf buffers. In the previous works until now only consider leaf buffers level and apply polarity assignment technique. However, since we have more freedom to select, e.g. allocate the buffers/inverters and flip-flop types with extended flip-flop structures, we can further minimize the current peak than only leaf buffers are considered. In this section, we demonstrate the limitation of previous works on polarity assignment, and the chance to further

Table 3.1: All possible mapping candidates of R_1 through R_4 and the maximum current peak of I_{DD} and I_{SS} when each mapping is applied. The last column is the amount of reduction in peak current.

				Peak		max	
R_1	R_2	R_3	R_4	Idd (uA)	Iss (uA)	(uA)	%
FF^{+}_{2inv}	FF^{+}_{2inv}	FF^{+}_{2inv}	FF^{+}_{2inv}	322.4	359.7	**359.7**	-
FF^{+}_{2inv}	FF^{+}_{2inv}	FF^{+}_{2inv}	FF^{-}_{2inv}	282.4	333.3	333.3	**-7.34**
FF^{+}_{2inv}	FF^{+}_{2inv}	FF^{-}_{2inv}	FF^{-}_{2inv}	307.2	347.7	347.7	-3.34
FF^{+}_{2inv}	FF^{-}_{2inv}	FF^{-}_{2inv}	FF^{-}_{2inv}	279	346.1	346.1	-3.78
FF^{-}_{2inv}	FF^{-}_{2inv}	FF^{-}_{2inv}	FF^{-}_{2inv}	319	358.0	358.0	-0.47
FF^{-}_{2inv}	FF^{-}_{1inv}	FF^{-}_{1inv}	FF^{-}_{1inv}	287.7	291.2	291.2	**-19.04**
FF^{-}_{2inv}	FF^{+}_{1inv}	FF^{-}_{1inv}	FF^{-}_{1inv}	232.2	291.6	291.6	-18.93
FF^{+}_{1inv}	FF^{-}_{1inv}	FF^{-}_{1inv}	FF^{-}_{1inv}	256.3	294.3	294.3	-18.18
...	...	...	...	...	...	...	...
FF^{+}_{2inv}	FF^{+}_{1inv}	FF^{+}_{1inv}	FF^{+}_{1inv}	235.3	343.6	343.6	-4.48
FF^{-}_{1inv}	FF^{-}_{1inv}	FF^{-}_{1inv}	FF^{-}_{1inv}	292.4	357.2	357.2	-0.70

reduce current peak when our algorithm is applied.

Fig. 3.4 shows an example clock tree with four *sink groups*. A*sink group* consists of one buffering elements and the flip-flops it drives. a buffer and a flip-flop are filled with the same color, to represent they form one sink group. The buffers of each sink group are denoted by e_i, $(i = 0, 1, 2, 3)$, and the flip-flops are denoted by R_1 through R_4, respectively. We added two non-leaf buffers, to observe that how the clock buffers not in leaf node have an impact on current peak. Except the root clock buffer, all the elements are supplied from the center of power/ground mesh. The root clock buffer receives power from ideal voltage source. To compare the current peak when only leaf buffers are considered versus the current peak when leaf buffers plus sink flip-flops are considered together, we conducted SPICE simulation by changing mapping of e_i and R_i $(i = 1, 2, 3, 4)$.

Table 3.1 shows all possible mapping of $R_1 \cdots R_4$. We assumed that if R_i is one of $\{FF^{+}_{2inv}$ of $FF^{+}_{1inv}\}$, e_i is mapped to BUF_X1, if R_i is one of $\{FF^{-}_{2inv}$ of $FF^{-}_{1inv}\}$, e_i is mapped to INV_X2. In the table, the topmost row is the initial mapping, where all the flip-flops is the conventional D flip-flop, FF^{+}_{2inv}, which results in $359.7uA$ of maximum current peak. The objective is to minimize the maximum current peak, our objective is reduce this $359.7uA$ of current peak appearing in I_{SS}. The next below 4 rows represent the situation where only leaf buffers are considered to be optimized, as in the previous work. In this case, the possible mapping of R_i is either FF^{+}_{2inv} or FF^{-}_{2inv}. With these candidates, we can at most only reduce by 7.34% of peak value. Now if we broaden the mapping candidates with considering all 4 types of flip-flops, we can reduce the current peak up to about 19%. The rest rows represents this result. One thing to note is the last row from the Table 3.1. In that case, mapping all of the flip-flops into FF^{-}_{1inv} that only using one-inverter-absent types does not

guarantee to minimize current peak. From this result, we can see that a problem of what type should be mapped to each sink group is not trivial, and requires an algorithmic approach to solve it.

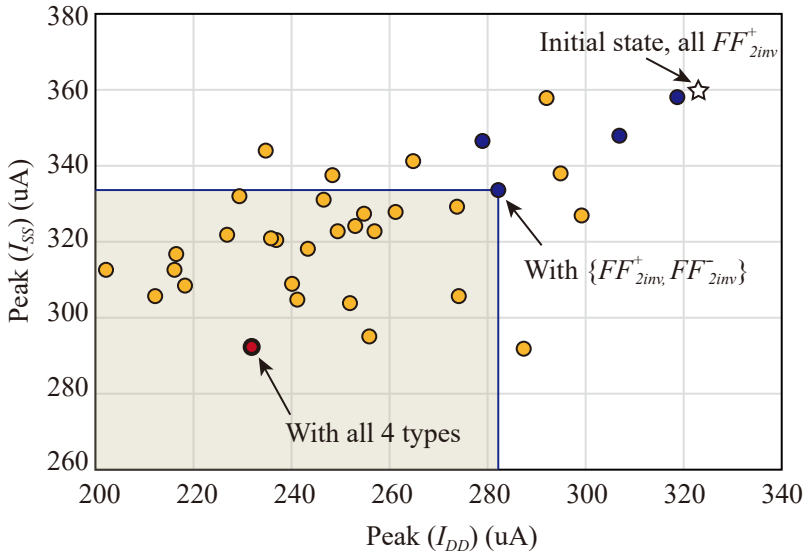
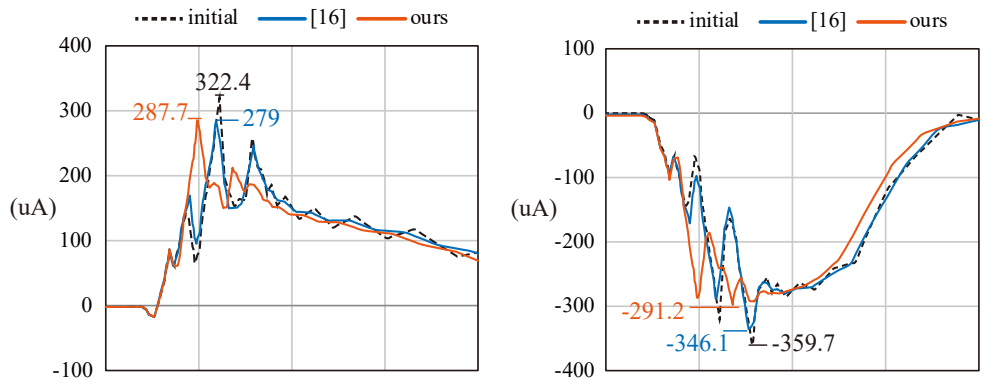


Figure 3.5: Graphical representation of the (I_{DD}, I_{SS}) values of Table 3.1. Lying at more left and more bottom position represents to it is a mapping with lower I_{DD} and I_{SS}. The blue dots are the mapping only FF^{+}_{2inv} and FF^{-}_{ainv} re considered, as in the previous work, while the yellow dots are the mapping all four types of flip-flops are considered. The yellow-filled area represents the region the previous work cannot find or cannot reach. With our extended structure of flip-flops, we can find current-peak more reduced mapping solution, which is marked as red dot in this figure.

Meanwhile, Fig. 3.5 graphically represents the values of Table 3.1. We marked every (I_{DD}, I_{SS})) in each mapping of R_i, where the x and y values are I_{DD} and I_{SS}, respectively. For example, the point marked as a star tells that the initial state where all the flip-flops are FF^{+}_{2inv}, resulting in 359.7 uA as



(a) I_{DD} (b) I_{SS}

Figure 3.6: Comparison of I_{DD} and I_{SS} of the initial state (dotted line), optimal mapping when only FF^{+}_{2inv} and FF^{-}_{2inv} are considered as in the previous works (blue line), and optimal mapping when all types of FF^{+}_{2inv}, FF^{-}_{2inv}, FF^{+}_{1inv}, and FF^{-}_{1inv} are considered. The peak current values of the initial state, (I_{DD}, I_{SS})=(322.4, -359.7) (uA) are reduced to (279, -346.1) (uA) when only FF^{+}_{2inv} and FF^{-}_{2inv} are considered. Finally, the current peaks are further reduced to (287.7, -291.2) (uA). Even the peak of I_{DD} becomes little higher (287.7(uA) > 279(uA)), the peak of I_{SS} becomes a lot lower (-359.7(uA) $\rightarrow$ -346.1(uA) $\rightarrow$ -291.2(uA).

the maximum current peak. Hence, a dot lying more left and more bottom position is better mapping having more minimized peak current. The four dark blue dots represents the cases where only FF^{+}_{2inv} and FF^{-}_{2inv} are considered, while rest of yellow dots represents the cases where all four types of flip-flops are considered. Among them, the mapping with minimum current peak is marked as red, that is, $\{FF^{-}_{2inv}, FF^{-}_{1inv}, FF^{-}_{1inv}, FF^{-}_{1inv}\}$, and (I_{DD}, I_{SS})=(287.7, 291.2) (uA), which is about 19% lower than the initial state.

Fig. 3.6 shows the current waves of the initial mapping (all FF^{+}_{2inv}), the optimal mapping when only two types, FF^{+}_{2inv} and FF^{-}_{2inv} are considered (3 FF^{+}_{2inv} s and 1 FF^{-}_{2inv}), and the optimal mapping when all four types of flip-flops are considered, which is (1 FF^{-}_{2inv} and 3 FF^{-}_{1inv} s). Though we get slightly larger I_{DD}, eventually we can further minimize the peak current in I_{DD} from -359.7 (uA) to -291.2(uA), which is about 19% reduction, while the previous work can reduce the peak from -359.7 (uA) to -346.1 (uA), about 7.3%.

3.4 Problem Formulation

We formally describe the boundary optimization for minimizing peak current problem as follows: BoundaryNoiseMin problem: (**Clock tree boundary optimization for peak current minimization**) *Given a buffered clock tree $\mathcal{T}$ with a set E of (already allocated) leaf buffering elements, a library B of buffers, a library I of inverters, a set $\mathcal{R}$ of (already allocated) flip-flops driven by the cells in E, clock skew bound constraint δ, clock slew rate constraint κ, and time sampling slots P, replace the cells in E and $\mathcal{R}$ by finding mapping functions $\phi_1 : E \mapsto B \cup I$ and $\phi_2 : \mathcal{R} \mapsto \{FF^{+}_{2inv}, FF^{+}_{1inv}, FF^{-}_{2inv}, FF^{-}_{1inv}\}$ that*

minimizes the quantity of

$$\max_{p\in P}\left\{\sum_{e_i\in E, f_j\in R} current(\phi_1(e_i),\phi_2(f_j),p)\right\} \tag{3.1}$$

$$s.\ t.\ \max_{f_i\in\mathcal{R}}\{t_i\}-\min_{f_i\in\mathcal{R}}\{t_i\}<\delta,$$

$$\max_{e_j\in E}\{s_j\}<\kappa$$

where t_i *and* s_i *are the clock arrival time at flip-flop* $\phi_2(f_i)$ *and the output slew rate of the driving buffer or inverter* $\phi_1(e_i)$*, respectively. The term* $current(\phi_1(e_i),\phi_2(f_j),p)$ *is the value of peak current at a time sampling slot* k *caused by by the switching of* e_i *and* f_j *when it is assigned with* $\phi_1(e_i)\in\{B\cup I\}$*, and* $\phi_2(f_j)\in\{\ FF^{+}_{2inv}, FF^{+}_{1inv}, FF^{-}_{2inv}, FF^{-}_{1inv}\ \}$

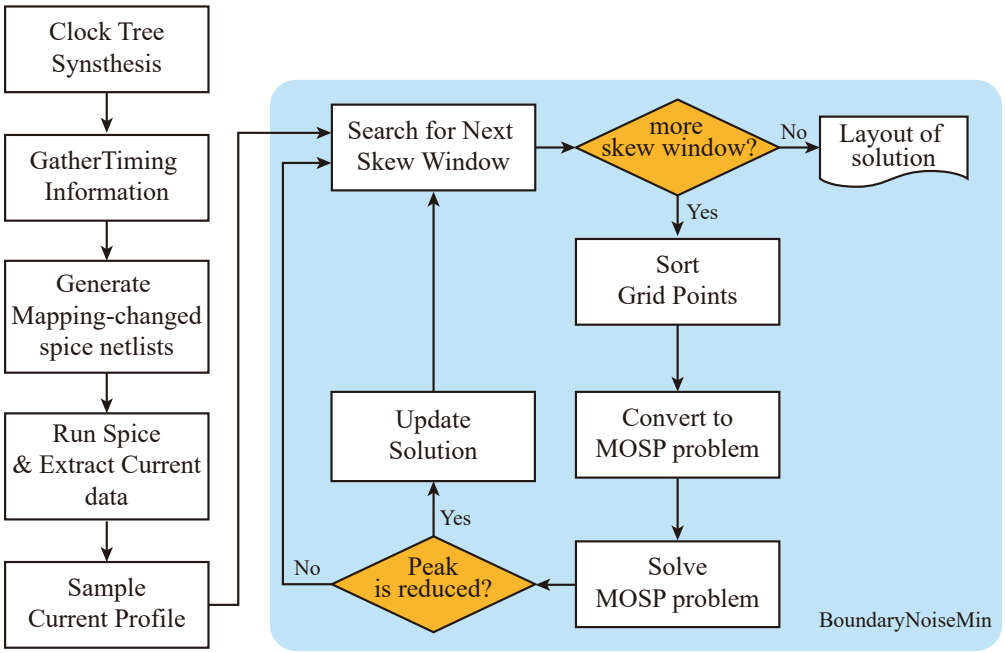


Figure 3.7: Flow of BoundaryNoiseMin algorithm.

3.5 Proposed Algorithm

3.5.1 An Overview

Fig. 3.7 shows the flow of our proposed algorithm of BoundaryNoiseMin. The inputs to our framework are a synthesized buffered clock tree, and clock skew and slew constraints δ and κ, from which the preprocessing of gathering all the clock arrival times to each flip-flop, which are extracted[1] by mapping every leaf buffer and flip-flop to different cell types, is performed, followed by generating current profile and sampling through HSPICE simulation.

We first sorted the grid points on a circuit according to the initial peak current value, and apply our assignment algorithm in a greedy manner, from the grid point with the highest current peak to the point with the lowest peak.

We transform the flip-flop type assignment problem of a circuit into an instance of min-max problem, which is then transformed to an instance of multi-objective shortest path (MOSP) problem, for which we use a polynomial approximation algorithm devised by Warburton in [48] whose time and is bounded by $O(rn^3(n/\epsilon)^{2r})$ and space by $O(rn(n/\epsilon)^r)$ where r is the arc weight dimension and n is the number of vertices in MOSP graph. Our formulation to the MOSP problem is described in Sec. 3.5.3. When solving the MOSP problem, we adopt a concept of superposition of current flows, described in Sec. 3.5.2. In addition, the derivation procedure of an instance of MOSP problem is illustrated in Sec. 3.5.3, and the heuristic of selecting a target grid points for reducing peak current is described in Sec. 3.5.4. Finally, integrating clock power minimization into our framework is described in Sec. 3.5.5.

[1] We used Synopsys IC Compiler for our experiment.

3.5.2 Superposition of Current Flows

Definition 1 ($R(i), R(i,j)$, $I(i,j)$) : $\underline{R(i)}$ is defined to be the set of flip-flops that are directly driven by clock buffer b_i. (We call $R(i)$ the set of sink flip-flops of buffer b_i.) Let p_j be a junction point in power mesh P. Then, $\underline{R(i,j)}$ is defined to be a maximal subset of $R(i)$ such that every flop-flops in the subset pulls current through p_j. (We call $R(i,j)$ the set of sink flip-flops of buffer b_i on power grid p_j.) Finally, $\underline{I(k,j)}$ is defined to be the current profile pulled by a flip-flop f_k through power grid point p_j.[2]

Note that every flip-flop is directly driven by exactly one clock buffer. That is, $R(i_1) \cap R(i_2) = \phi$ if $i_1 \neq i_2$, but it may pull current through more than one junction point in power mesh. Superposition of current flow on a junction p_i in power mesh P states that the total current flow flowing at p_i equals to the sum of the amount of currents that are pulled through p_i by the cells connected to p_i.

By following the current superposition theorem, the total amount of current flow pulled by flip-flops on a power grid p_j can be expressed as:

$$\sum_{f_k \in \mathcal{F}} I(k,j) = \sum_{b_i \in \mathcal{B}} \sum_{f_k \in R(i,j)} I(k,j) \tag{3.2}$$

where $\mathcal{F}$ and $\mathcal{B}$ are the set of flip-flops and the set of sink clock buffers, respectively.

Based on *Eq.*(3.2), we extract a current profile of each $R(i,j)$ and perform the superposition of current profiles. Fig. 3.8 illustrates our procedure of deriving the current profiles on power junctions.

[2]We assume that the currents are pulled only through power grid points.

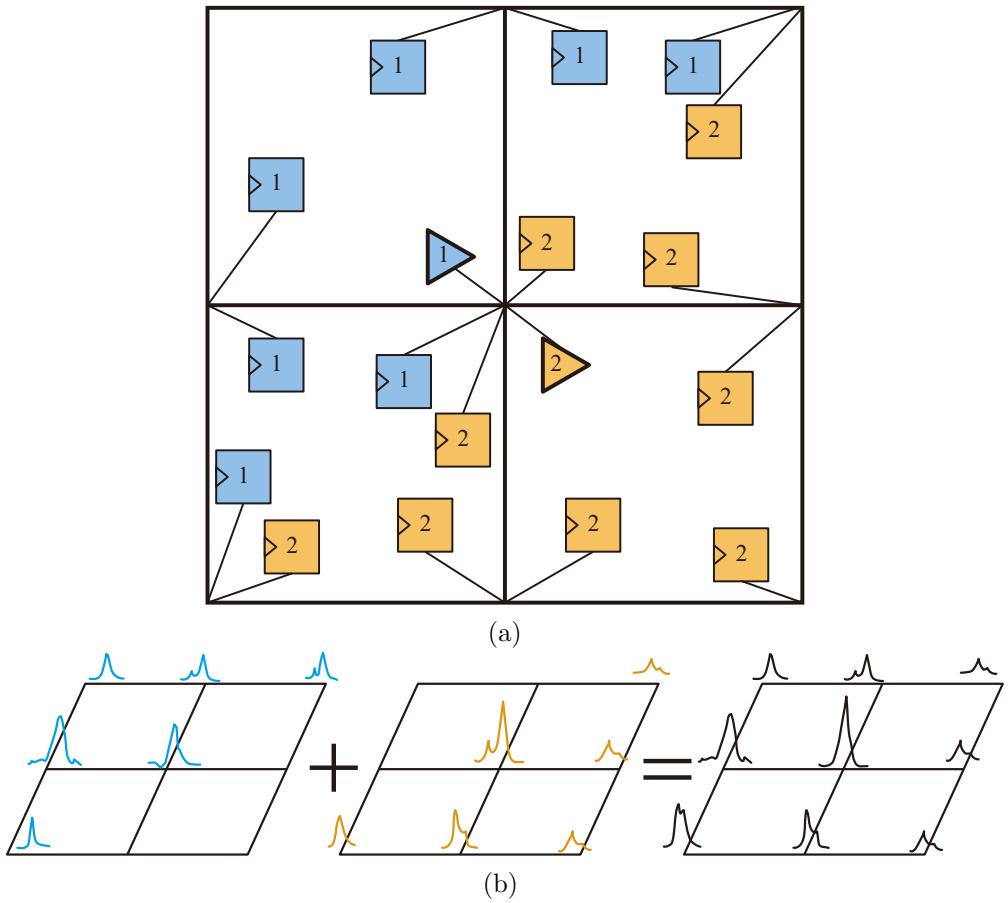


Figure 3.8: An illustration for superposition of currents. (a) A example circuit with a power mesh having two sink groups. A ground mesh is omitted in this illustration for simplicity. Each sink group is denoted as 1 and blue color, 2 and yellow color, respectively. Each cell is connected to its nearest junction of power mesh. (b) Superposition of currents. The current shown at a junction equals to the summation of currents each cell connected to that junction pulls.

3.5.3 Formulation to Instance of MOSP Problem

For a target power grid point p_i and the sets of $R(i)$ in which their flip-flop types in $\mathcal{R}$ have not been determined, we want to find a mapping of $R(i)$ to flip-flop types that minimizes the peak current on p_i. For example, suppose we have three flip-flop groups $R(1)$, $R(2)$, and $R(3)$ on p_i and their flip-flop types are not determined yet. Further, we suppose that the mapping candidates without violating the clock skew bound constraint have been extracted, as shown in Table 3.2. We call such mapping candidates *feasible mappings*. For example, mapping candidate 1 for $R(1)$ indicates that the initial driving buffer is converted to an inverter with size of X8 and its driven flip-flops to FF^{-}_{1inv}. Note that a driving buffer may be converted to an inverter according to the assignment of flip-flop type to its driven set of flip-flops.

Table 3.2: An illustration of feasible mappings for three sets of flip-flops, each of which is driven by buffers b_1, b_2, and b_3 in the initial clock tree.

Sink Group	T_1		T_2		T_3	
	FF type	B/I	FF type	B/I	FF type	B/I
R_1	FF^{-}_{1inv}	INV_X8	FF^{-}_{2inv}	INV_X4	FF^{+}_{2inv}	BUF_X4
R_2	FF^{-}_{1inv}	INV_X8	FF^{+}_{1inv}	BUF_X4	-	-
R_3	FF^{+}_{1inv}	BUF_X4	FF^{+}_{2inv}	BUF_X4	-	-

Then, we construct an L-layered network $G(V, A)$ for the sets, say $R(1)$, $\cdots$, $R(L)$, of flip-flops that are directly driven by L clock buffers. Each vertex in V indicates a distinct mapping candidate and all vertices corresponding to set $R(i)$, $i = 1, 2, \cdots, L$ are arranged in the i-th layer in G. Then, for every pair of vertices between two consecutive layers in G, we create an arc $(v_i, v_j) \in A$. Finally, we add two dummy vertices called *src* and *dest*, placing at the top and the other at the bottom of the L-layered network G. Fig. 3.9 show the graph $G(V, A)$ corresponding to the mapping candidates in Table 3.2. The weight of arc (v_i, v_j) is a vector of size s whose elements indicate the current values at the

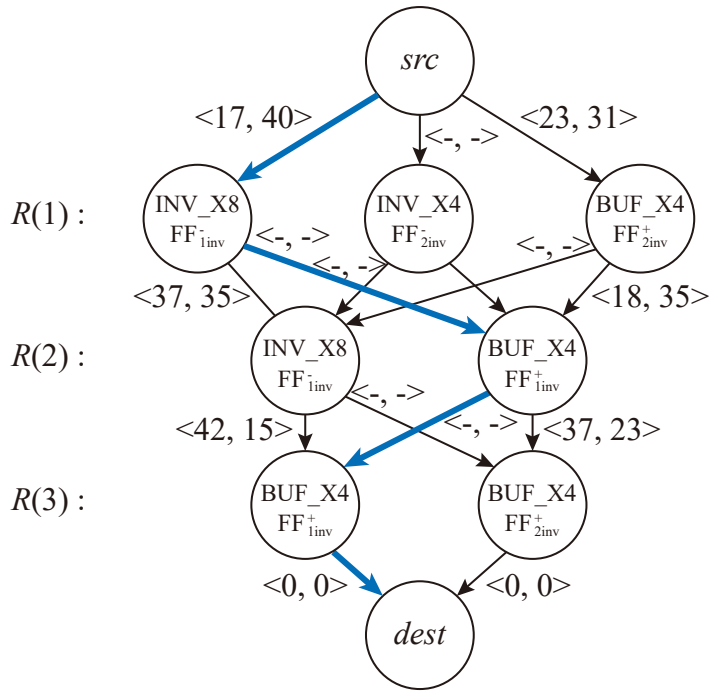


Figure 3.9: Conversion to a network graph $G(V, A)$ for the mapping candidates in Table 3.2. The peak current minimization problem is then translated to find a solution for an instance of multi-objective shortest path problem.

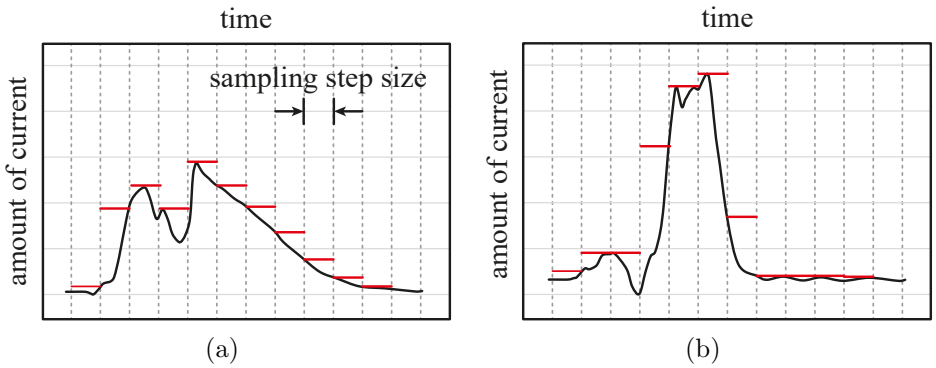


Figure 3.10: An illustration of sampling on current waves in (a) power line and (b) ground line. The maximum value of each range will be chosen. The sampling step size determines the number of sampling slots, i.e., the value of s.

s sampling slots on the current profile caused by the mapping corresponding to v_j. An illustrative example of sampling on current waves is shown Fig. 3.10. The number of sampling slots is determined by designers. Using more sampling slots would measure more value of peak current at the expense of computation time. Then, we want to find, among all possible paths from *src* to *dest* in G, a path that minimizes the largest value among the elements in the element-wise vector sum on the path. This problem is referred to as the multi-objective shortest path (MOSP) problem. (If $s = 1$, it becomes the ordinary shortest path problem.) The lines with blue color indicate the minimum peak current path. Its vector sum, for example $< 30, 12 >$, means that peak current of $30uA$ is the least height that can be achieved by the feasible mappings. The resulting mapping are INV_X8 + FF^{-}_{1inv}, BUF_X4 + FF^{+}_{1inv}, and BUF_X4 + FF^{+}_{1inv} for $R(1)$, $R(2)$, and $R(3)$, respectively. In practice, for large values of s and L, finding an exact or bounded solution considering all power grid points is a time consuming process.

3.5.4 Selecting Target Power Grid Points

To speedup the mapping process of flip-flops, we employ a greedy approach for the selection of a power grid points on which we want to minimize the peak current.

Definition 2 ($I_{peak}(i)$, $N_{tot}(i)$, $N_{self}(i)$) : Let p_i be a power grid point. $\underline{I_{peak}(p_i)}$ is defined to be the value of peak (i.e., max.) current on p_i extracted on an initial clock tree with flip-flops, $\underline{N_{tot}(p_i)}$ is defined to be the total number of buffers such that *some* of their driven flip-flops pull current through p_i, and $\underline{N_{self}(p_i)}$ ($< N_{tot}(p_i)$) be the number of buffers such that *all* the driven flip-flops pull current *entirely* through p_i.

For example, Table 3.3 shows the values of $I_{peak}(\cdot)$ extracted by HSPICE simula-

Table 3.3: Current peak data of all the grid points on circuit s1423 with 3x3 power/ground lines when VDD=0.95V is applied. The last column indicates the sets of flip-flops whose current source come from the corresponding grid points.

location	power (uA)	ground (uA)	max (uA)
(0, 0)	156.8	214.4	214.4
(1, 0)	450.9	497.1	497.1
(2, 0)	140.5	213.7	213.7
(3, 0)	212.2	321.3	321.3
(0, 1)	513.2	575.0	575
(1, 1)	187.0	305.8	305.8
(2, 1)	53.5	71.4	71.4
(3, 1)	293.0	273.0	293
(0, 2)	465.4	709.9	709.9
(1, 2)	48.1	70.9	70.9
(2, 2)	463.8	656.4	656.4
(3, 2)	242.8	391.6	391.6
(0, 3)	144.2	213.1	213.1
(1, 3)	592.1	667.9	667.9
(2, 3)	105.6	165.4	165.4
(3, 3)	547.5	757.4	757.4

tion for circuit s1423 in ISCAS'89. In addition, from the last column in Table 3.3 the values of $N_{tot}(p_i)$ and $N_{self}(p_i)$ for each grid point p_i are computed.

Given the values of $I_{peak}(\cdot)$, $N_{tot}(\cdot)$, and $N_{self}(\cdot)$ for all power grid points, we select the grid point (p_i) that minimizes the quantity of:

$$C(p_i) = w \cdot I_{peak}(p_i) + (1 - w) \cdot \frac{N_{self}(pi)}{N_{tot}(p_i)} \tag{3.3}$$

where w $(0 \leq w \leq 1)$ is a weighting factor.

The first term in $Eq.(3.3)$ prefers, as next target, the power grid point that is very likely to expose highest peak current while the second term in $Eq.(3.3)$ ensures that our mapping assignment is effective in lowering down the peak current.

Once the mapping assignment for a target grid point is done, we update the current profiles on all the grid points that have not been processed. Then, the selection process repeats until there is no grid point with unmapped flip-flops.

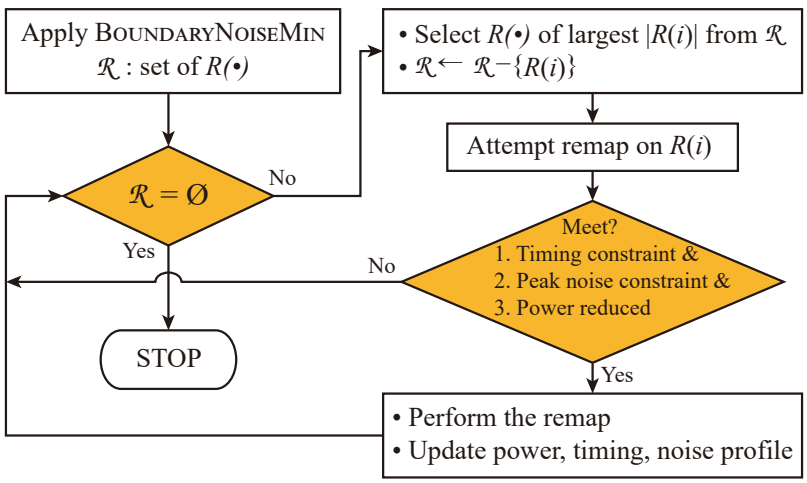


Figure 3.11: Flow of post power minimization.

3.5.5 Consideration of Reducing Power Consumption

Our clock boundary optimization methodology can be extended to support minimizing power consumption under current noise constraint, described in Fig. 3.11. Initially we start from the result of peak noise minimal-mapping produced by our BoundaryNoiseMin. Then, at each iteration, we attempt to remap a set of flip-flops as long as it results in reducing power consumption while still meeting timing and current noise constraints. Note that depending on the relative importance of noise, power, and timing, the trade-off between them can be explored by restructuring the flow diagram.

3.6 Experimental Results

Our proposed algorithm called BoundaryNoiseMin was implemented in C++ and Python language on a Linux machine with i5-4670 CPU and 8GB RAM. ISCAS'89 benchmark circuits were synthesized with Synopsys Design compiler and Synopsys IC compiler using Nangate 45nm Open Cell Library[39]. Initially, clock trees were synthesized by IC compiler with the slew and skew constraints of $100ps$. After extracting RC information of routed clock tree, HSPICE simulation is conducted with a power/ground mesh. We used power delivery network (PDN) models modelled as an RL network to clearly observe power/ground noise by simultaneous switching. Each RL segment of the on-chip power/ground mesh has parameters of $R = 0.21\Omega/um$ and $L = 0.5fH/um$. Each cell on a clock tree is connected to the closest grid point of power/ground mesh.

The off-chip PDN is also modeled with the model and parameters in [49], for which the parameters are summarized in Table 3.4. The off-chip supply voltage is transferred to the on-chip power/ground mesh through an RLC circuit. Each of four power/ground bumps of off-chip PDN is connected to a corner on the

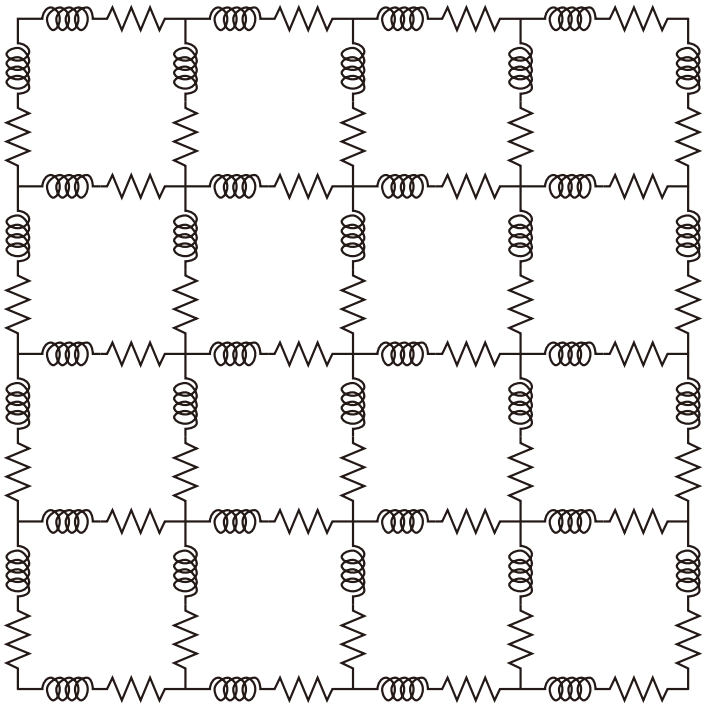


Figure 3.12: Model of the on-chip power delivery network.

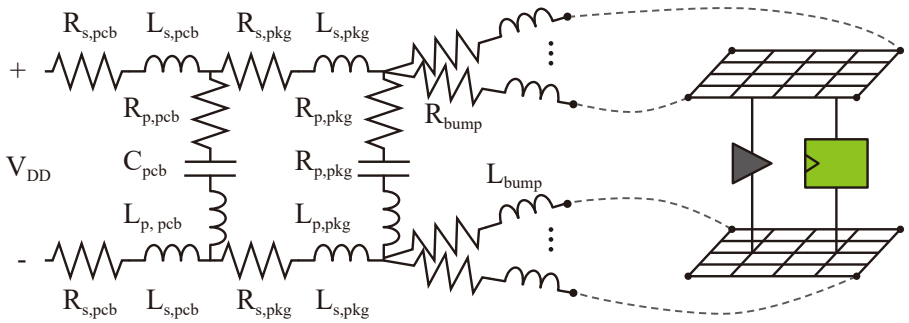


Figure 3.13: Model of the off-chip power delivery network (PDN). The off-chip and on-chip PDNs are connected through four bumps around the corners.

Table 3.4: Off-chip PDN parameters for HSPICE simulation

$R_{s,pcb}$	0.094 $m\Omega$	$R_{s,pcb}$	0.166 $m\Omega$
$L_{s,pcb}$	21 pH	$L_{s,pcb}$	0 pH
$R_{s,pkg}$	1 $m\Omega$	$C_{s,pcb}$	240 $m\Omega$
$L_{s,pkg}$	120 pH	$R_{p,pkg}$	0.54 $m\Omega$
R_{bump}	20 $m\Omega$	R_{bump}	5.61 pH
L_{bump}	30 pH	C_{pkg}	26 μF

on-chip power/ground mesh, shown by dotted lines in Fig. 3.13.

The input clock signal has 30ps of slew and frequency of 500MHz. Clock skew and slew constraints are both 100ps. To measure the current peak in the PDN, HSPICE simulation was executed on the ISCAS'89 benchmark circuits. After we get the new cell type mapping by BoundaryNoiseMin so that the current peak is minimized, we replace the buffers and flip-flops in the initial circuit to the new cell types. Finally RC extraction and HSPICE simulation are performed again on the new circuit.

The simulation results are summarized in Table 3.5. We attached 3x3 power and ground mesh to each benchmark circuit, where every cell is connected to its nearest junction of power/ground mesh. We measure peak current and power/ground noise on every junction in the mesh, and take maximum value of them. Each column of *Base*, *polarity assignment only* [1], *Ours* represents the initial clock tree, the results produced by applying the polarity assignment in [1], and the results by our BoundaryNoiseMin, respectively. The *Peak* and *Noise* columns indicate the maximum peak current and maximum peak-to-peak voltage fluctuation appearing at the junctions in power/ground mesh, respectively.

The highest current peak occurs at the ground line in every case, hence voltage noise on V_{DD} is larger than that on V_{SS}. As shown in Table 3.5, our proposed

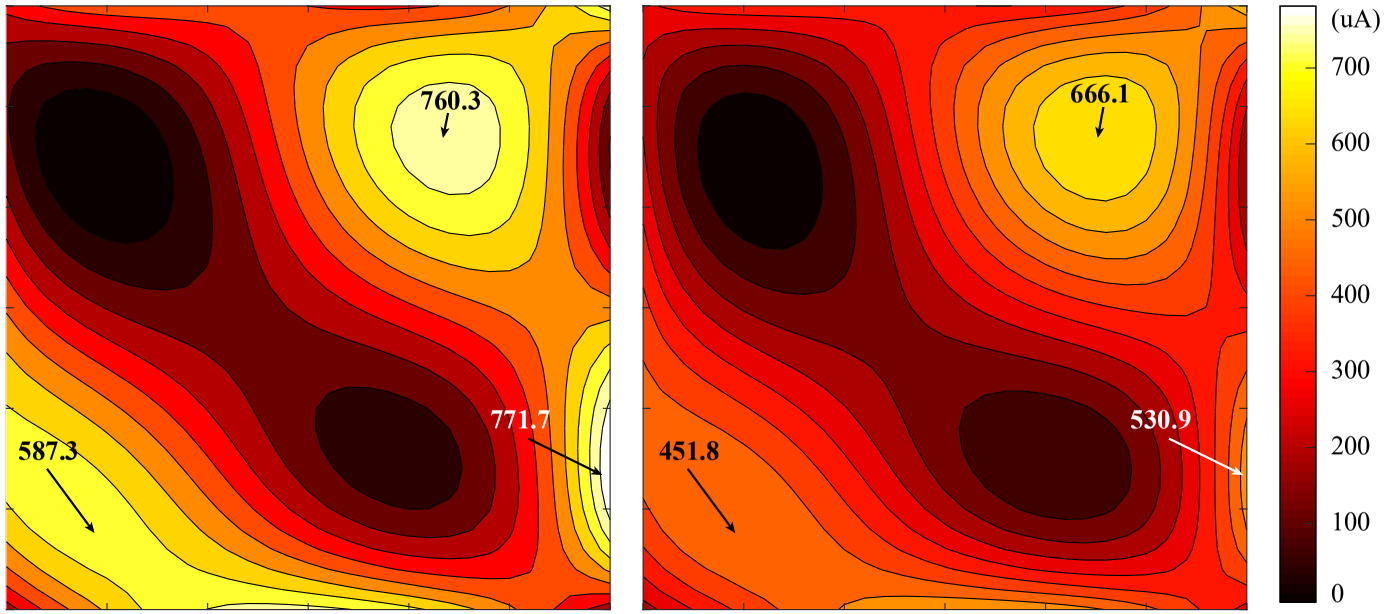
algorithm BoundaryNoiseMin outperforms the polarity assignment method in [1] where BoundaryNoiseMin reduces peak current and power/ground noise by 9.36%∼19.54% and 27.69%∼30.94%, respectively, over the initial clock tree, and by 2.92%∼10.80% and 12.67%∼17.55%, respectively over the work in [1]. Overall, our proposed mapping solution consistently reduces the peak current and power/ground noise under the clock skew and slew constraints over the design optimized by [1] as well as the initial designs. Fig. 3.14 shows the current maps before and after the application of BoundaryNoiseMin to circuit s1423.

3.7 Summary

This chapter proposed an algorithm for clock tree boundary optimization with the objective of minimizing peak current. The key enabler was exploiting the four types of flip-flop mapping at the clock tree boundary. We formulated the mapping problem of minimizing peak current into a multi-objective shortest path problem and solved it efficiently using an approximation algorithm. Through testing benchmark circuits, it was shown that our algorithm was able to reduce the peak current by 27.7%∼30.9%. In addition we suggested an extended design flow of integrating the peak current noise minimization with the clock power minimization.

Table 3.5: Comparison of peak current and power/ground noise for initial circuits and ones produced by [1] and our BoundaryNoiseMin.

Circuits	Base				[1] (Polarity assignment only)				Ours (Buffer and FF co-optimization)					Improv. over Base (%)				Improv. over [1] (%)			
	Peak (uA)		Noise (mV)		Peak (uA)		Noise (mV)		Peak (uA)		Noise (mV)		Time	Peak (%)		Noise (%)		Peak (%)		Noise (%)	
	vdd	vss	vdd	vss	vdd	vss	vdd	vss	vdd	vss	vdd	vss	(sec)	vdd	vss	vdd	vss	vdd	vss	vdd	vss
s1423	672	790	30.28	31.21	561	717	12.41	12.99	445	701	7.49	9.35	13.3	+33.78	+11.27	+75.25	+70.03	+20.68	+2.23	+39.61	+27.99
s15850	899	1168	40.71	42.53	820	942	31.82	34.82	820	942	31.82	34.82	20.0	+8.79	+19.35	+21.84	+18.13	0.00	0.00	0.00	0.00
s5378	1238	1907	127.10	137.30	1229	1827	138.00	149.70	1212	1685	95.25	101.60	47.5	+2.10	+11.64	+25.06	+26.00	+1.38	+7.77	+30.98	+32.13
s13207	2043	2837	77.54	81.45	1749	2610	42.74	45.48	1433	2405	31.16	44.07	189.0	+29.86	+15.23	+59.81	+45.89	+18.07	+7.85	+27.09	+3.10
s38584	3902	6545	64.04	113.20	3619	6406	69.13	117.20	3633	6395	60.32	107.00	2597	+6.89	+2.29	+5.81	+5.48	-0.39	+0.17	+12.74	+8.70
s38417	5456	7521	90.12	167.50	4744	7597	83.16	157.40	3863	7421	72.11	139.70	4877	+29.20	+1.33	+19.98	+16.60	+18.57	+2.32	+13.29	+11.25
s35932	5288	7236	84.35	151.60	4719	6927	76.28	141.70	3904	6919	76.93	133.90	3798	+26.17	+4.38	+8.80	+11.68	+17.27	+0.12	-0.85	+5.50
Average														**+19.54**	**+9.36**	**+30.94**	**+27.69**	**+10.80**	**+2.92**	**+17.55**	**+12.67**



(a) Initial State (b) After applying BoundaryNoiseMin algorithm

Figure 3.14: Peak current distribution of ISCAS'89 s1423 benchmark circuit. More brighter color means larger peak current. The initially bright yellowish region changed into more dark yellow and orange color, which means peak current value is minimized.

Chapter 4

Conclusion

The contributions of this dissertation is summarized as follows.

4.1 Clock Buffer and Flip-flop Co-optimization for Reducing Power Consumption

In this chapter, we solve a new problem of optimizing the boundary of buffered clock trees, which has not been addressed in the design automation as yet. Precisely, we want to show that the clock cells that directly drive flip-flops should not necessarily be buffers. By taking into account the internal structure of flip-flops, we can have a freedom of choosing either buffers or inverters for the cell implementation from library. This in fact leads to cancel out the two inverters, one in the driving buffer and another in each flip-flop, thereby reducing the power consumption on the clock tree, including flip-flops. We generalize this idea to look into the possibility of co-optimizing the driving buffers and flipflops together to reduce the clock power at the boundary of clock trees, and propose an effective four-step synthesis algorithm of clock tree boundary for

low power. By applying our proposed technique to benchmark circuits, it is observed that the clock power is able to be reduced by 7.9%∼10.2% further on average without timing violation.

4.2 Clock Buffer and Flip-flop Co-optimization for Reducing Power/Ground Noise

In this chapter, we discussed and previous polarity assignment technique to mitigate simultaneous switching noise on power/ground mesh, by reducing peak current, which is an extended concept of previous work as including our new proposed flip-flop structures as the possible candidates for allocating at the clock boundary. Consequently, we have a flexibility of selecting (i.e., allocating) clock boundary components in a way to reduce peak current under timing constraint. We formulate the component allocation problem of minimizing peak current into a multi-objective shortest path problem and solve it efficiently using an approximation algorithm. We have implemented our proposed approach and tested it with ISCAS benchmark circuits. The experimental results confirm that our approach is able to reduce the peak current by 27.7%∼30.9% on average.

Bibliography

[1] D. Joo and T. Kim, "A fine-grained clock buffer polarity assignment for high-speed and low-power digital systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 3, pp. 423–436, Mar. 2014.

[2] P. E. Gronowski, W. J. Bowhill, R. P. Preston, M. K. Gowan, and R. L. Allmon, "High-performance microprocessor design," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 5, pp. 676–686, May 1998.

[3] V. Tiwari, D. Singh, S. Rajgopal, G. Mehta, R. Patel, and F. Baez, "Reducing power in high-performance microprocessors," in *Proceedings 1998 Design and Automation Conference. 35th DAC. (Cat. No.98CH36175)*, June 1998, pp. 732–737.

[4] P. Pillai and K. G. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems," in *SOSP '01: Proceedings of the eighteenth ACM symposium on Operating systems principles*. New York, NY, USA: ACM, 2001, pp. 89–102. [Online]. Available: http://doi.acm.org/10.1145/502034.502044

[5] C. M. Krishna and Y. H. Lee, "Voltage-clock-scaling adaptive scheduling techniques for low power in hard real-time systems," *IEEE Transactions on Computers*, vol. 52, no. 12, pp. 1586–1593, Dec 2003.

[6] M. Donno, A. Ivaldi, L. Benini, and E. Macii, "Clock-tree power optimization based on rtl clock-gating," in *Proceedings 2003. Design Automation Conference (IEEE Cat. No.03CH37451)*, June 2003, pp. 622–627.

[7] R. Bhutada and Y. Manoli, "Complex clock gating with integrated clock gating logic cell," in *2007 International Conference on Design Technology of Integrated Systems in Nanoscale Era*, Sept 2007, pp. 164–169.

[8] W. M. D. J. G. Xi, "Buffer insertion and sizing under process variations for low power clock distribution," in *Proceedings of IEEE/ACM Design Automation Conference*, 1995, pp. 491–496.

[9] J. Lillis, C.-K. Cheng, and T. T. Y. Lin, "Optimal wire sizing and buffer insertion for low power and a generalized delay model," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 3, pp. 437–447, March 1996.

[10] K. Banerjee and A. Mehrotra, "A power-optimal repeater insertion methodology for global interconnects in nanometer designs," *IEEE Transactions on Electron Devices*, vol. 49, no. 11, pp. 2001–2007, Nov 2002.

[11] S. Pullela, N. Menezes, and L. T. Pillage, "Low power ic clock tree design," in *Proceedings of IEEE Custom Integrated Circuits Conference*, 1995, pp. 263–266.

[12] S. C. Chan, P. J. Restle, T. J. Bucelot, J. S. Liberty, S. Weitzel, J. M. Keaty, B. Flachs, R. Volant, P. Kapusta, and J. S. Zimmerman, "A resonant global

clock distribution for the cell broadband engine processor," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 1, pp. 64–72, Jan 2009.

[13] V. S. Sathe, S. Arekapudi, A. Ishii, C. Ouyang, M. C. Papaefthymiou, and S. Naffziger, "Resonant-clock design for a power-efficient, high-volume x86-64 microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 1, pp. 140–149, Jan 2013.

[14] S. Ahn, M. Kang, M. C. Papaefthymiou, and T. Kim, "Design methodology for synthesizing resonant clock networks in the presence of dynamic voltage/frequency scaling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 12, pp. 2068–2081, 2016.

[15] M. Edahiro, "A clustering-based optimization algorithm in zero-skew routings," in *30th ACM/IEEE Design Automation Conference*, June 1993, pp. 612–616.

[16] T.-H. Chao, Y.-C. Hsu, J.-M. Ho, and A. B. Kahng, "Zero skew clock routing with minimum wirelength," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 39, no. 11, pp. 799–814, Nov 1992.

[17] R. S. Tsay, "An exact zero-skew clock routing algorithm," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 2, pp. 242–249, Feb 1993.

[18] J. Cong and C.-K. Koh, "Minimum-cost bounded-skew clock routing," in *Circuits and Systems, 1995. ISCAS '95., 1995 IEEE International Symposium on*, vol. 1, Apr 1995, pp. 215–218 vol.1.

[19] A. B. Kahng and C. W. A. Tsao, "More practical bounded-skew clock routing," in *Proceedings of the 34th Design Automation Conference*, June 1997, pp. 594–599.

[20] J. Lu, V. Honkote, X. Chen, and B. Taskin, "Steiner tree based rotary clock routing with bounded skew and capacitive load balancing," in *2011 Design, Automation Test in Europe*, March 2011, pp. 1–6.

[21] A. B. K. D. J.-H. Huang, "On the bounded-skew clock and steiner routing problems," in *32nd Design Automation Conference*, 1995, pp. 508–513.

[22] C. W. A. Tsao and C. K. Koh, "Ust/dme: a clock tree router for general skew constraints," in *IEEE/ACM International Conference on Computer Aided Design. ICCAD - 2000. IEEE/ACM Digest of Technical Papers (Cat. No.00CH37140)*, Nov 2000, pp. 400–405.

[23] J. G. Xi and W. W. M. Dai, "Useful-skew clock routing with gate sizing for low power design," in *33rd Design Automation Conference Proceedings, 1996*, Jun 1996, pp. 383–388.

[24] A. Rajaram and D. Z. Pan, "Meshworks: An efficient framework for planning, synthesis and optimization of clock mesh networks," in *2008 Asia and South Pacific Design Automation Conference*, March 2008, pp. 250–257.

[25] H. Seo, J. Kim, M. Kang, and T. Kim, "Synthesis for power-aware clock spines," in *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2015, pp. 126–131.

[26] Y. Kim and T. Kim, "Algorithm for synthesis and exploration of clock spines," in *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 2017, pp. 263–268.

[27] A. Rajaram and D. Z. Pan, "Meshworks: An efficient framework for planning, synthesis and optimization of clock mesh networks," in *2008 Asia and South Pacific Design Automation Conference*, March 2008, pp. 250–257.

[28] G. Venkataraman, Z. Feng, J. Hu, and P. Li, "Combinatorial algorithms for fast clock mesh optimization," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 1, pp. 131–141, Jan 2010.

[29] H. Chen, C. Yeh, G. Wilke, S. Reddy, H. Nguyen, W. Walker, and R. Murgai, "A sliding window scheme for accurate clock mesh analysis," in *ICCAD-2005. IEEE/ACM International Conference on Computer-Aided Design, 2005.*, Nov 2005, pp. 939–946.

[30] L. H. Chen, M. Marek-Sadowska, and F. Brewer, "Buffer delay change in the presence of power and ground noise," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 3, pp. 461–473, June 2003.

[31] J.-T. Yan and Z.-W. Chen, "Construction of constrained multi-bit flip-flops for clock power reduction," *Proceedings of IEEE International Conference on Green Circuits and Systems*, pp. 675–678, 2010.

[32] Y.-T. Chang, C.-C. Hsu, M. P.-H. Lin, Y.-W. Tsai, and S.-F. Chen, "Post-placement power optimization with multi-bit flip-flops," in *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, 2010, pp. 218–223.

[33] I. Jiang, C. Chang, and Y. Yang, "INTEGRA: Fast multibit flip-flop clustering for clock power saving," *IEEE Transactions of Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 2, pp. 192–204, 2012.

[34] S. H. Wang, Y. Y. Liang, T. Y. Kuo, and W. K. Mak, "Power-driven flip-flop merging and relocation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 2, pp. 180–191, 2012.

[35] C.-C. Hsu, Y.-C. Chen, and M. P.-H. Lin, "In-placement clock-tree aware multi-bit flip-flop generation for power optimization," in *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, vol. 1, Nov. 2013, pp. 592–598.

[36] Z.-W. Chen and J.-T. Yan, "Routability-constrained multi-bit flip-flop construction for clock power reduction," *Integration, the VLSI Journal*, vol. 46, no. 3, pp. 290–300, Jun. 2013.

[37] Y.-T. Shyu, J.-M. Lin, C.-P. Huang, C.-W. Lin, Y.-Z. Lin, and S.-J. Chang, "Effective and efficient approach for power reduction by using multi-bit flip-flops," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 4, pp. 624–635, 2013.

[38] T. K. Hyoungseok Moon, "Design and allocation of loosely coupled multi-bit flip-flops for power reduction in post-placement optimization," in *Proceedings of IEEE Asia and South Pacific Design Automation Conference*, 2016, pp. 268–273.

[39] "NanGate FreePDF45 Open Cell Library," 2011. [Online]. Available: http://www.nangate.com/?page_id=2325

[40] P. Vuillod, L. Benini, A. Bogliolo, and G. De Micheli, "Clock skew optimization for peak current reduction," in *Proceedings of the 1996 international symposium on Low power electronics and design*. IEEE Press, 1996, pp. 265–270.

[41] A. Vittal, H. Ha, F. Brewer, and M. Marek-Sadowska, "Clock skew optimization for ground bounce control," in *Proceedings of International Conference on Computer Aided Design*, Nov 1996, pp. 395–399.

[42] W.-C. Lam, C.-K. Koh, and C.-W. Tsao, "Power supply noise suppression via clock skew scheduling," in *Quality Electronic Design, 2002. Proceedings. International Symposium on.* IEEE, 2002, pp. 355–360.

[43] S.-H. Huang, C.-M. Chang, and Y.-T. Nieh, "Fast multi-domain clock skew scheduling for peak current reduction," in *Asia and South Pacific Conference on Design Automation, 2006.*, Jan 2006, pp. 6 pp.–.

[44] A. Vijayakumar, V. C. Patil, and S. Kundu, "An efficient method for clock skew scheduling to reduce peak current," in *2016 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems (VLSID)*, Jan 2016, pp. 505–510.

[45] Y.-T. Nieh, S.-H. Huang, and S.-Y. Hsu, "Minimizing peak current via opposite-phase clock tree," in *Proceedings. 42nd Design Automation Conference, 2005.*, June 2005, pp. 182–185.

[46] R. Samanta, G. Venkataraman, and J. Hu, "Clock buffer polarity assignment for power noise reduction," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 6, pp. 770–780, June 2009.

[47] H. Jang, D. Joo, and T. Kim, "Buffer sizing and polarity assignment in clock tree synthesis for power/ground noise minimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 1, pp. 96–109, Jan. 2011.

[48] A. Warburton, "Approximation of pareto optima in multiple-objective, shortest-path problems," *Oper. Res.*, vol. 35, pp. 70–79, Feb. 1987.

[49] M. S. Gupta, J. L. Oatley, R. Joseph, G. Y. Wei, and D. M. Brooks, "Understanding voltage variations in chip multiprocessors using a distributed power-delivery network," in *2007 Design, Automation Test in Europe Conference Exhibition*, April 2007, pp. 1–6.

초록

초고밀도 집적(VLSI) 회로에서는 데이터를 저장하는 모든 플립플롭들의 동작은 클럭 네트워크를 통해 전달되는 클럭 신호에 의해 동기화된다. 클럭 신호의 매우 높은 스위칭 주파수로 인해서, 클럭 네트워크에서 소모하는 동적 전력(dynamic power)은 회로의 전체 전력 소모 중에서 상당히 큰 부분을 차지한다. 또한, 클럭 네트워크에서 가장 많은 양의 전력 소모는 바로 클럭 네트워크의 경계에 있는 플립플롭과, 그 플립플롭들을 드라이브하는버퍼에서 일어난다. 게다가, 동기 회로에서 모든 플립플롭의 동시적인 활성화는 클럭 네트워크의 경계에서 큰 피크 전류(즉, 전압 하강)를 야기시킨다. 이 점과 관련하여 본 논문에서는 다음의 두 가지 새로운 문제, 클럭 네트워크 경계에서의 클럭 전력 감소 문제와 클럭 네트워크 경계에서의 전류 노이즈 감소 문제를 언급한다. 플립플롭의 최적화와 클럭 버퍼의 최적화를 분리해서 고려했던 이전 연구들과는 달리, 플립플롭과 클럭 버퍼의 동시최적화를 고려한다. 더 엄밀히 말하면, 플립플롭과 그것을 드라이브하는 버퍼 한 쌍을 하나의 싱글 유닛으로 구현할 수 있는 네 종류의 하드웨어요소를 제안한다. 이러한 네 종류의 하드웨어 요소를 이끌어내는 가장 중요한 점은 바로, 플립플롭의 기능을 바꾸지 않고도, 플립플롭을 드라이브하는 버퍼 내의 인버터 하나와 각 플립플롭 내의 인버터 하나를 결합해서 제거한다는 것이다. 그 결과 주어진 타이밍 조건 하에서 전력 소모와 피크 전류를 더 감소시킬 수 있도록 클럭 경계 요소들을 선택하는, 즉 할당하는 더 많은 자유도를 얻는다. 클럭 시차 상한 조건 하에서 ISCAS'89 벤치마크 회로에 대해 본 논문에서 제안된 알고리즘을 구현하였다. 실험 결과는 본 논문에서 제안된 알고리즘이 클럭 전력 소모와 파워 노이즈를 평균적으로 각각 7.9∼10.2%, 27.7%∼30.9% 줄일 수 있음을 보여준다.

주요어: 클락 트리합성, 저전력, 최적화, 전류 피크, 클럭 시차, MOSP
학번: 2012-30200