M.S. THESIS

Design and Algorithm for Clock Gating and Flip-flop Co-optimization

클럭 게이팅 및 플립 플롭 동시 최적화를 위한 설계 및 알고리즘

BY

Yang Giyoung

FEBURARY 2019

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

M.S. THESIS

Design and Algorithm for Clock Gating and Flip-flop Co-optimization

클럭 게이팅 및 플립 플롭 동시 최적화를 위한 설계 및 알고리즘

BY

Yang Giyoung

FEBURARY 2019

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

Design and Algorithm for Clock Gating and Flip-flop Co-optimization

클럭 게이팅 및 플립 플롭 동시 최적화를 위한 설계 및 알고리즘

지도교수 김 태 환
이 논문을 공학석사 학위논문으로 제출함

2019년 2월

서울대학교 대학원

전기 컴퓨터 공학부

양 기 용

양기용의 공학석사 학위 논문을 인준함

2019년 2월

위 원 장: _____
부위원장: _____
위 원: _____

Abstract

In this paper, we introduce dynamic power optimization techniques applicable for various design stage from standard cell to placement stage. This work firstly investigates the problem of how designing data-driven (i.e., toggling based) clock gating can be closely integrated with the synthesis of flip-flops, which has never been addressed in the prior clock gating works. Our key observation is that some internal part of a flip-flop cell can be reused to generate its clock gating enable signal. Based on this, we *propose a newly optimized flip-flop wiring structure*, called eXOR-FF, in which an internal logic can be reused for every clock cycle to decide if the flip-flop is to be activated or inactivated through clock gating, thereby achieving area saving (thus, leakage as well as dynamic power saving) on every pair of flip-flop and its toggling detection logic. Then, we propose a comprehensive methodology of placement/timing-aware clock gating exploration that provides two unique strengths: best suited for *maximally exploiting the benefit of eXOR-FFs* and precise *analyses on the decomposition of power consumptions and timing impact, and translating them into cost functions* in core engine of clock gating exploration.

Through experiments with benchmark circuits in ISCAS89, ITC89, ITC99 and IWLS 2005, it is shown that our proposed method is able to reduce the total power by 5.6% and total cell area by 5.3% compared with the previous data-driven clock gating method in [1].

keywords: Low Power, Standard cell, Flip-flop, Logic Synthesis, Clock-gating
student number: 2017-27057

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Power Consumption in CMOS Digital Design

There are two major components that constitute the power used by a CMOS integrated circuit [4]: Static power and Dynamic power. Static power essentially consists of the power used when the transistor is not in the process of swtiching and is essentially determined by th formula

$$P_{static} = I_{static} \cdot V_{dd} \tag{1.1}$$

where V_{dd} is the supply voltage and I_{static} is the total current flowing through the device [5]. Typically, CMOS technology has been prasied for its low static power. However, as devices are scaled, gated oxide thickness decrease and there is increased probability of tunnealing, resulting in larger currents. This subthreshold leakage current is goverend by thermodynamics, more specifically the Boltzmann distribution [6].

Dynamic power is the sum of transient power consumption and capacitive load power consumption induced by charging/discharging the node. Transient power consumption represents the amount of power consumed when the device changes logic states. (i.e. "Low" to "High" bit or vice versa), and capcitive load power consumption represents the power used to charge the load capacitance. In Eq. 1.2 and Fig. 1.1, we calculate the disspatied energy(E_{dyn}) in CMOS inverter circuit, where C is the sum of

the load(C_{Load}) and internal capacitance of integrated circuit ($C_{internal}$),V_{dd} is supply voltage, $C \cdot V_{dd} \int_{t_0}^{t_1} dv$ is supplied energy and $C \cdot \int_{t_0}^{t_1} vdv$ is stored energy.

$$E_{dyn} = \int_{t_0}^{t_1} P(t)dt = \int_{t_0}^{t_1} (V_{dd} - v) \cdot i(t)dt = \int_{t_0}^{t_1} (V_{dd} - v) \cdot C(dv/dt)dt$$
$$= C \cdot V_{dd} \int_{t_0}^{t_1} dv - C \cdot \int_{t_0}^{t_1} vdv = C \cdot {V_{dd}}^2 - 1/2C \cdot {V_{dd}}^2 = 1/2C \cdot {V_{dd}}^2 \quad (1.2)$$

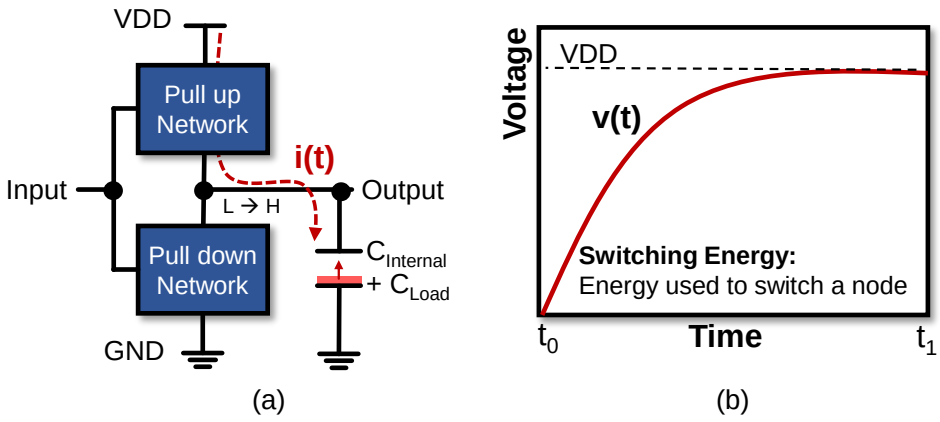


Figure 1.1: (a) Dynamic power consumption occured when charing of the CMOS inverter circuit, (b) voltage(V)-time(t) curve in load capacitance.

We can extract dynamic power from dissipated energy(E_{dyn}) with below Eq. 1.3 , when node voltages are changing.

$$P_{dyn} = E_{dyn}/T_{tran} = 1/2 \cdot C \cdot {V_{dd}}^2 \cdot f \quad (1.3)$$

1.2 Low Power Design Methodologies

With suffering some radical changes in electronic design, such as computation and consumer electronics, power consumption of individual components is reaching the limits resulting in reduced device reliablity. Dealing with power is rapidly becoming

one of the most demanding issues in digital design. By that reason, various power optimization techniques are researched over the several decades. In fact, to be meaninigful, power optimization has been occured at all levels of the design hierachy, including the technology, circuit, layout, logic, architectural and algorithmic levels. [7]

As we looked at the two types of power ahead, there are two main key categories for enabling effective power optimization in chip. Firstly, reducing the static power. It can be implemented in different phase of desgin phase, such as minimizing usage of low threshold voltage(V_{th}) transistors in process stage and back biasing at block circuit level in design stage. In particular, power gating is one of the most commonly used techniques to save the standby leakage by shutting off the current to blocks of the circuit that are not in use. This is accomplished by adding switch cell either to VDD or GND supply [8].

Secondly, decreasing the dynamic power such as Dynamic voltage and frequency scaling(DVFS), Voltage island and clock gating. DVFS is the adjustment of power and speed setting on a device's processor and controller chips to optimize resource allotment for tasks and maximize power saving when those resources are not needed. By that, DVFS allows the devices to perform needed tasks with the minimum amount of required power. Another techniques is multiple-supply voltage(MSV) with considering floorplan , level-shifter placement and power planning. Above techniques are effective and powerfull, if those design methods are implement in early design stage, however from the early stages of chip design, high-level designing with power limitation in early phase is not easy. For that reason, those methodologies such as DVFS and MSV are difficult to implement at the design stage where fine tunning is desired. Therefore, this paper focus on clock gating one of the low power design methodolgies.

In synchronous digital system, most of dynamic power is consumed by the clock signal, reaching up to 70% of total power consumption in enitre systems [9].

This clock induced power dissipation can be classified into two groups: (*Group 1*) the power consumed by the clock delivery network including the storge elements

(i.e., flip-flops/latches) and (*Group 2*) the power consumed by the combinational logic synchronized by the sequential elements. As a means to save the dynamic power consumption in group-2, As a means to save dynamic power, *clock gating* has been known to be one of the most effective techniques. Clock gating saves power by shutting off a subtree of clock network during 'idle' state of the driven logic blocks or by disabling the clock signals to a group of flip-flops during their 'untoggling' states [10].

For the situation where designers have a prior knowledge of or can extract logic conditions for some blocks in circuit that can be safely in an idle state, the idle state based clock gating can be applied to the part of clock network that drives the blocks. (The clock enable/disable conditions are usually extracted in the system or RTL design stage since the mutual dependencies of various functions can be found well in those levels.) However, in case where it is not easy to extrat clock gating logic or there is no subcircuit block of reasonable size for clock gating, the toggling based clocking gating can be an alternative. One common consideration in applying the toggling based gating is that since the toggling based clock gating is relatively fine-grained (i.e., flip-flop level) one, the supplemental logic for generating gating enabling/disabling signal together with detecting toggling/untoggling state of every flip-flop is a big burden on extra power consumtion. Nevertheless, there are designs in which the toggling based clock gating is more effective in reducing power over the idle state based clock gating [11], for example, being reported for some controller design that its idle based gating to RTL blocks reduces the power dissipation by 23$\sim$27% whereas its toggling based gating to flip-flops reduces the power more than double [12].

This work belongs to the area of toggling based clock gating and addresses (1) how this supplemental logic for clock gating can be elegantly reduced by proposing a practically feasible solution through simultaneously designing and optimizing gating enabling logic and internal structure of flip-flops, and (2) how the power saving and timing impact caused by clock gating can be accurately formulated and integrated into the framework of clock gating exploration.

1.3 Contribution of This Thesis

The traditional toggling based clock gating can be implemented at various design levels (e.g., [13, 14, 15]). The main focuses of most of the existing methods of the clock gating are aggressively selecting candidate flip-flops for clock gating with little or no consideration of timing impact and grouping the flip-flops for gating to share gating logic at the expense of degradation of power saving (e.g., [16, 17]). The unique contributions of this work are summarized as:

1. We propose a *new internal structure of flip-flops dedicated to toggling based clock gating* called eXOR-FF, referring to as *XOR embedded flip-flop*. The benefit of using eXOR-FF cells provdes a complete elimination of XOR implementation in the gating enabling logic.

2. We provide a comprehensive data on the *timing and power analyses to validate the practical feasibility of eXOR-FF*. In short, 12$\sim$16% less power consumption is measured for a pair of flip-flop and its gating logic.

3. We propose a *new design methodology that is best suited for the clock gating to maximally utilize the strength of eXOR-FF*.

4. We formulate a set of fine-grained cost functions on *clock gating induced power saving and timing impact*, and devise a framework for exploiting them as core engine in our new *placement/timing-aware clock gating exploration*.

5. Extensive experiments with benchmark circuits are performed to check how much our clock gating methodology combined with eXOR-FFs is effectve in saving power with no timing violation. In summary, ours is able to achieve 5.6% more power saving over the existing aggressive power-saving clock gating sacrificing timing slack constraints.

Chapter 2

Preliminary and Motivations

2.1 Background

Data toggling based clock gating shuts off the clock signal to a flip-flop when the state of the flip-flop is not subject to change at the next clock cycle. A block-level circuit structure supporting the clock gating for a group of k flip-flops ($R = \{f_1, f_2, \cdots, f_k\}$) is shown in Fig. 2.1(a), in which the newly added logic blocks (i.e., *Clock Disable* (CD) and *Integrated Clock Gating* (ICG) [18]) to the original circuit are marked with yellow color.

The logic operations of CD and ICG are:

CD: Boolean equation for CD can be expressed as:

$$g = (D_1 \oplus Q_1) + (D_2 \oplus Q_2) + \cdots + (D_k \oplus Q_k) \tag{2.1}$$

Thus, the implementation requires k 2-input XOR gates, one for each flip-flop, and an ORtree consisting of $k-1$ 2-input OR gates. The equation indicates that if g is 1, there is at least one flip-flop in R that will change its state at the next clock cycle. Thus, the condition of disabling clock signal to *all* k flip-flops in R at the next clock cycle is $\bar{g}$.

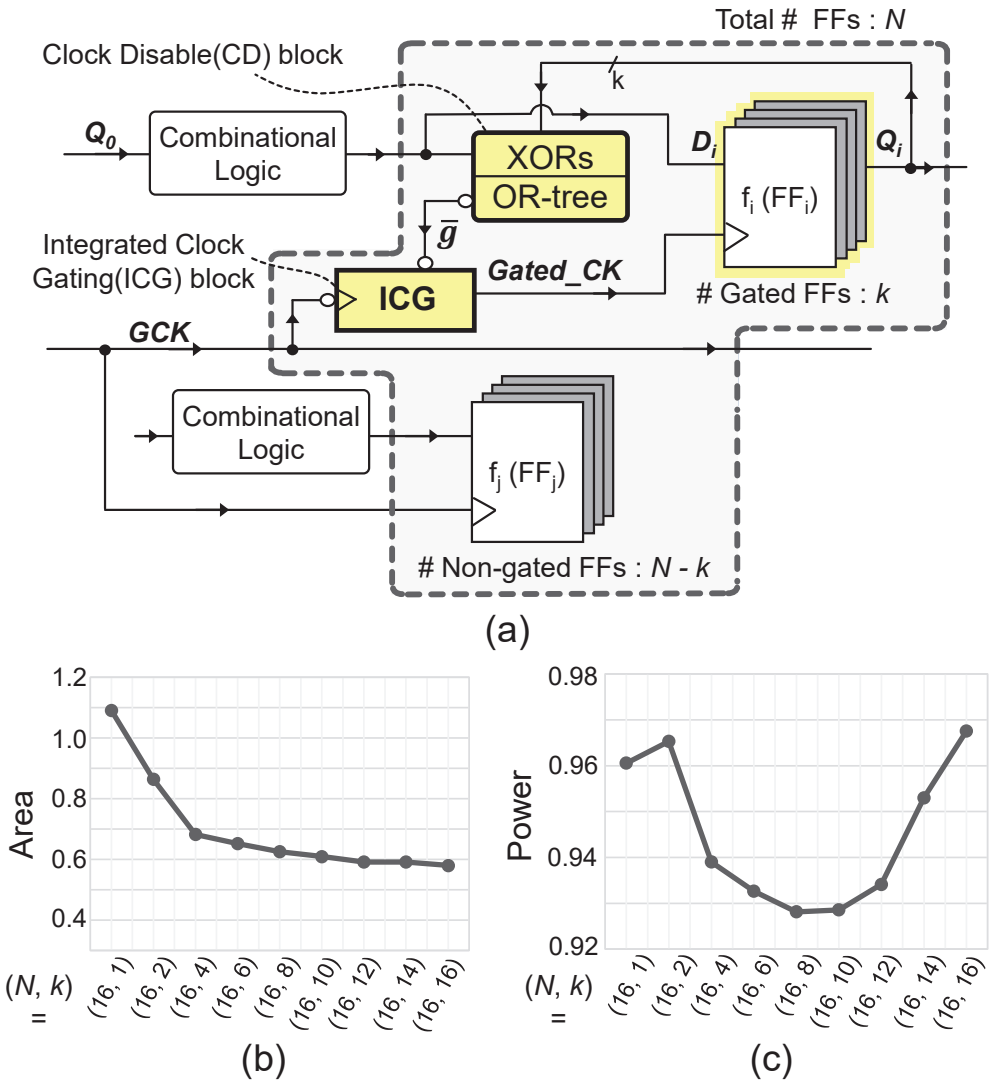


Figure 2.1: (a) Block-level structure of data toggling based clock gating, in which the logic blocks added for clock gating are marked with yellow color. (b) The changes of area in the dotted circle as the clock gating size changes. (c) The changes of power consumption in the dotted circle as the clock gating size changes.

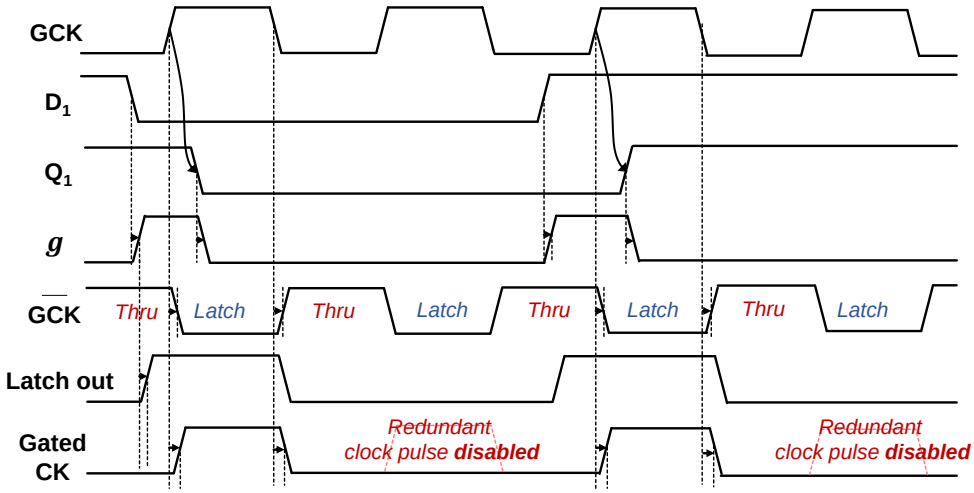


Figure 2.2: Timing diagram of conventional data-driven (i.e toggling based) clock gating

ICG: It receives the logic value of $\bar{g}$ from CD and decides if the clock signal is to be disabled or not while synchronizing it to the rising edge of *GCK*.

The effectiveness of toggling based clock gating on reducing power closely relies on the number of occurances of the clock cycles at which $\bar{g}$ is true (i.e., all flip-flops simultaneously untoggling). One strong evidence of the usefulness of the toggling based clock gating can be found in [11], which measured statistical toggling data of more than 22 000 flip-flops in a DSP core and found that on average 95% of clock signals to the flip-flops in a clock cycle was untoggled.

However, there are two important factors that have not been carefully addressed by the existing works on the toggling based clock gating. Precisely, the resulting clock gating logic blocks (CD and ICG) incur area overhead, which causes nontrivial impact on power and timing. The following two subsections describe in-depth analyses of the two factors.

2.2 Observation on Area and Power Saving

We performed HSPICE circuit simulation[1] to the blocks in the dotted circle in Fig. 2.1(a), varying the flip-flop group size k = 1, 2, 4, 6, 8, 10, 12, 14, 16 for clock gating, assuming that the circuit has a total of 16 (= N) filp-flops, and each flip-flop toggles indepedently with toggling probability of 0.02. The curves in Fig. 2.1(b) and 2.1(c) show the changes of area and power consumption of the blocks in the dotted circle in Fig. 2.1 as the gating size k varies.

Observation 1. The numbers on the area curve are obtained by computing the $\frac{k \cdot Area(XOR)+(k-1) \cdot Area(OR)+Area(ICG)}{k \cdot Area(FF)}$, which is the amount of the average area overhead per flip-flop required for gating k flip-flops. Even though the values of $k \cdot Area(XOR)$ and $(k-1) \cdot Area(OR)$ linearly increase as k increases, since $k \cdot Area(FF)$ increases faster (i.e., steeper slope), the overall clock gating area cost per flip-flop goes slow down as k increase, eventually becomming a *constant*, as shown in Fig. 2.1(b). Thus, it is important to find a design technique which is able to *reduce a part of clock gating area* that determines the constant factor.

Observation 2. The numbers on the power curve in Fig. 2.1(c) are obtained by summing the power consumptions of the clock gating logic (i.e., XOR, ORtree and ICG), the k flip-flops selected for clock gating, and the remaining 16-k flip-flops. (The details of the calculation of their power consumption will be described in Sec. 4.2 and Sec. 4.3.) The power curve indicates that the power consumption gradually decreases as k increases, and then grows back. It means that the total *power saving heavily depends on the value of group size k*, which in turn closely relies on the joint toggling probabilities among the grouped flip-flops. Thus, it is very important to accuratly reflect the joint toggling probabilities in the power cost formulation.

Based on the observations, our strategy targets two directions: (i) *investigating co-optimizing possibility in designing flip-flops and clock gating logic* to save area

[1]We used 28nm cell library and PDK.

(Sec. 3), and (ii) *exploring flip-flop grouping for clock gating, based on accurate power cost formulation* on the constituents of clock gating to minimize the total power consumption (Sec. 4).

2.3 Observation on Timing Impact

Fig. 2.3 shows the timing diagram of the signals asserted to a clock gated flip-flop and gating logic blocks in Fig. 2.1(a). We can see that the flip-flop will be activated when signal *Gated_CK* is triggled from 0 to 1, which occurs $T_{clk2Q}(ICG)$ time later after triggering of *GCK*, as indicated by the red arrow in Fig. 2.3, where $T_{clk2Q}(ICG)$ represents the clock-to-Q delay of the latch in ICG.

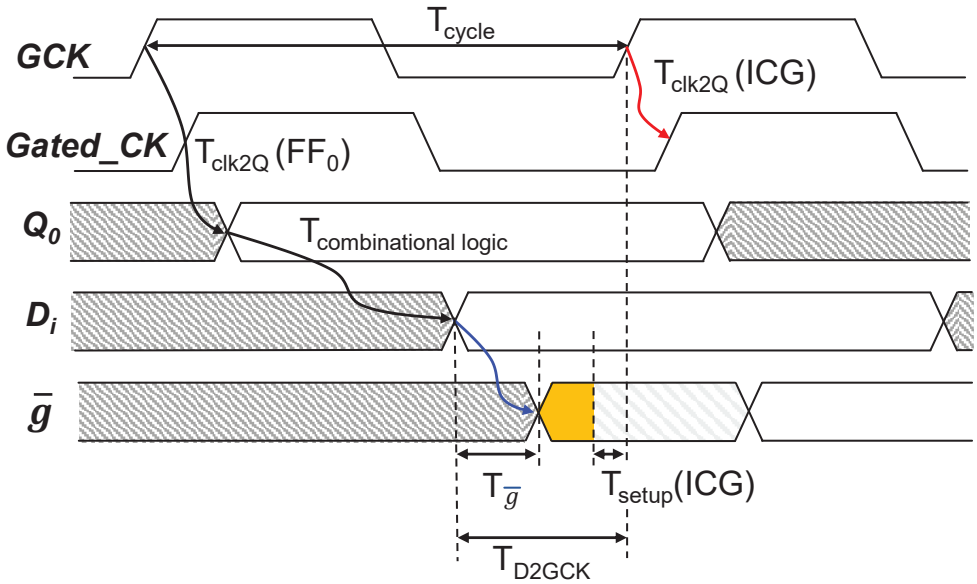


Figure 2.3: Timing diagram for the gating signals in $Fig.2.1$ to a flip-flop, in which the intervals marked by red and blue arrows are the delay caused by clock gating. The yellow interval indicates the setup slack of flip-flop with clock gating.

Observation 3. Since the arrival times of input data (e.g., D_i in the timing diagram) to a flip-flop can be extracted in a pre-processing step, and the value of $T_{clk2Q}(ICG)$ is constant regardless of the group size of flip-flops for clock gating, the remaining

time allowed for the CD delay ($T_{\bar{g}}$) and the (minimally required) setup time of ICG (T_{setup}(ICG)) should not be greater than T_{D2GCK} in Fig. 2.3. That is, the following inequality should be satisfied:

$$T_{\bar{g}} \leq T_{D2GCK} - T_{setup}(ICG) \tag{2.2}$$

Since T_{setup}(ICG) is constant and T_{D2GCK} is fixed for each flip-flop, $T_{\bar{g}}$ is the only variable, the value of which we can control by clock gating. Thus, by letting T_{slack} = $T_{D2GCK} - T_{setup}(ICG)$, a meaningful guideline for the selection of flip-flops for clock gating is to *prefer the flip-flops with large values of* T_{slack}.

Based on the observation, our strategy to mitigate the timing impact is (iii) *to develop, utilizing* T_{slack} *term, a timing penalty cost function* in the exploration of placement and timing-aware clock gating (Sec. 4).

Chapter 3

Redesign of Flip-flops Specialized for Clock Gating

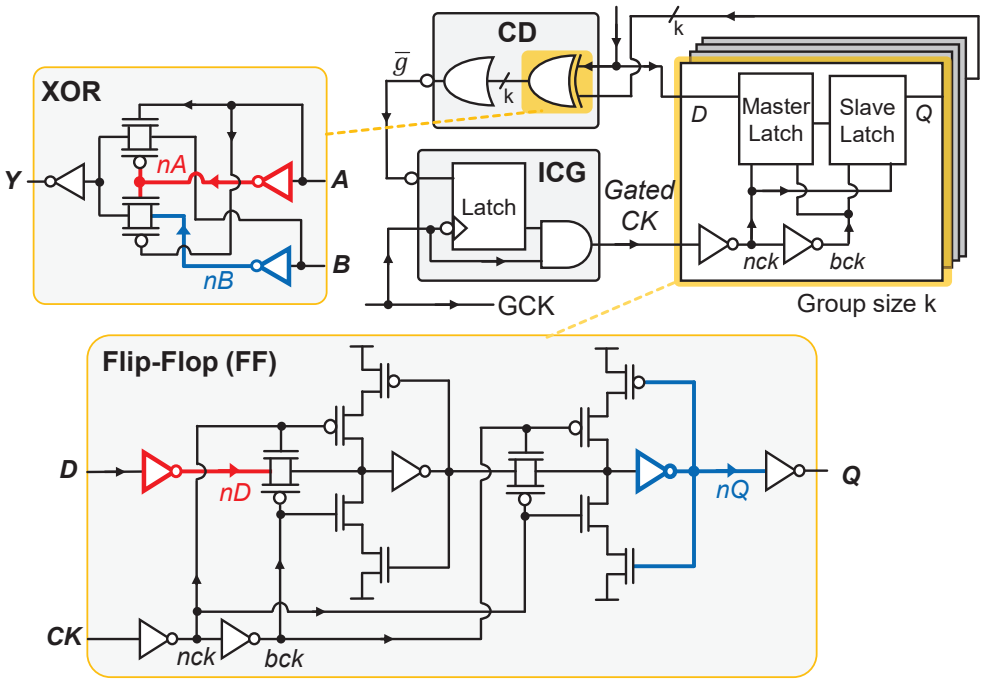


Figure 3.1: Internal circuit netlist of clock gating logic block CD and flip-flop (FF), in which XOR [2] in CD and FF [3] are further elaborated on the left and bottom sides, respectively.

Fig. 3.1 shows the internal circuit structure of CD (*Clock Disable*) and a flip-flop (FF) in which the 2-input XOR [2] in CD and the FF [3] are further elaborated in the

transistor level, as shown on the left and bottom sides, respectively. The role of XOR gate is to compare the current state (i.e., Q) of FF and the coming data (i.e., D) to FF to check if output toggling will occur at the next clock cycle or not. Since the signals D and Q directly connected from/to FF, as specified with red and blue thick lines, the two red and blue inverters in XOR can be shared with the inverters connected to D and Q in FF. In other words, signal *nD* in FF can be reused as *nA* in XOR and signal *nQ* in FF be reused as *nB* in XOR. Based on this observation, we combine FF and XOR together. Fig. 3.2 shows our co-optimized design for a pair of FF and XOR, in which each of the red and blue inverters plays a double role. We call the optimized design eXOR-FF, meaning *XOR embedded flip-flop*.

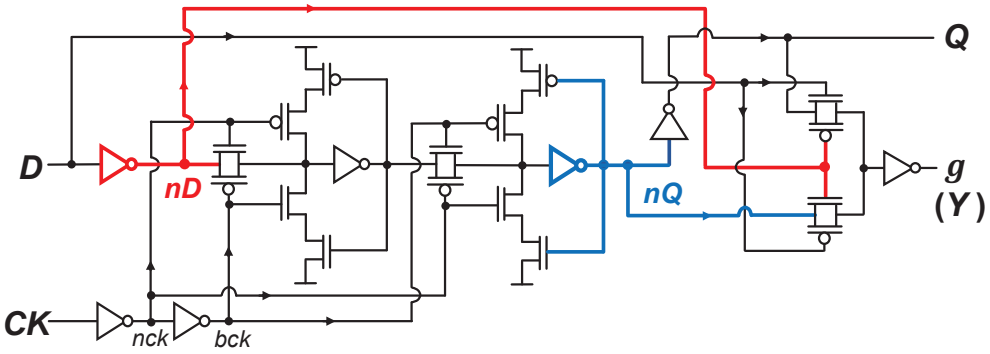


Figure 3.2: Newly designed and optimized flip-flop (eXOR-FF) that includes 2-input XOR logic function. The red and blue components participate both of flip-flop and XOR functions.

We applied HSPICE simulation to extract the power and delay numbers on the blocks (i.e., flip-flops, XOR and OR-tree in CD, and ICG) in the dotted circle in Fig. 2.1(a).

• **Power analysis**: We measured the power consumption on the five types of components in the highlighted box in Fig. 2.1 for three design options: (1) not applying clock gating to all N flip-flops, (2) applying clock gating to all (i.e., $k = N$) flip-flop without using eXOR-FFs, and (3) applying clock gating to all (i.e., $k = N$) flip-flops using eXOR-FFs, assuming the toggling probability of 0.02 on each flip-flop. The nor-

malized comparison between the power consumptions of the three options is obtained for k $(= N)$ = 2, 4, 8, and 16, In summary, the power saving by eXOR-FFs is 12.0%, 14.5%, 15.0% and 15.7% for k = 2, 4, 8 and 16, respectively.

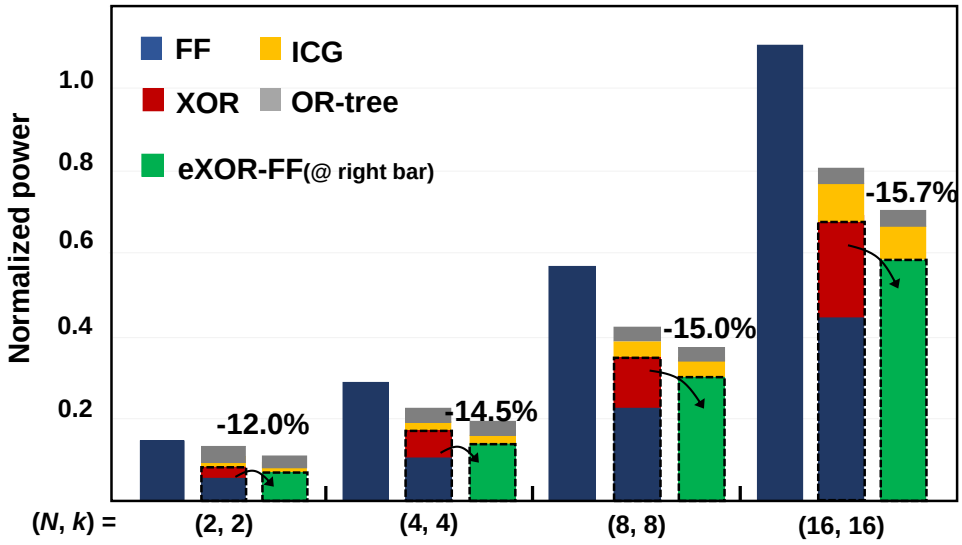


Figure 3.3: Normalized comparison of power consumed by the clock gating logic blocks and flip-flops in $Fig.2.1$ for non-gated initial designs (left bar), clock gated designs for all flip-flops without using eXOR-FFs (middle bar), and clock gated designs for all flip-flops using eXOR-FFs (right bar).

• **Timing analysis**: We measured two delay parameters.

Clock-to-Q delay: Fig. 3.4(a) compares the timing waveforms for clock-to-Q (yellow one) on a regular flip-flop and clock-to-Q (red one) on an eXOR-FF. We measured 3.49ps delay increase. The amounts of delay increase for various output loads on Q are listed in the upper part in Table 3.1. The delay increase is 3.35ps on average, which can be translated to 2.53% increase over the clock-to-Q delay on a regular flip-flop. Even though the delay increase is very little,[1] designers, if they wish, can exclude the flip-flops with very tight time margin from the consideration of using eXOR-FF cells

[1]For example, for a design of 1 GHz clock frequency, the timing penalty by eXOR-FFs is merely 0.35%.

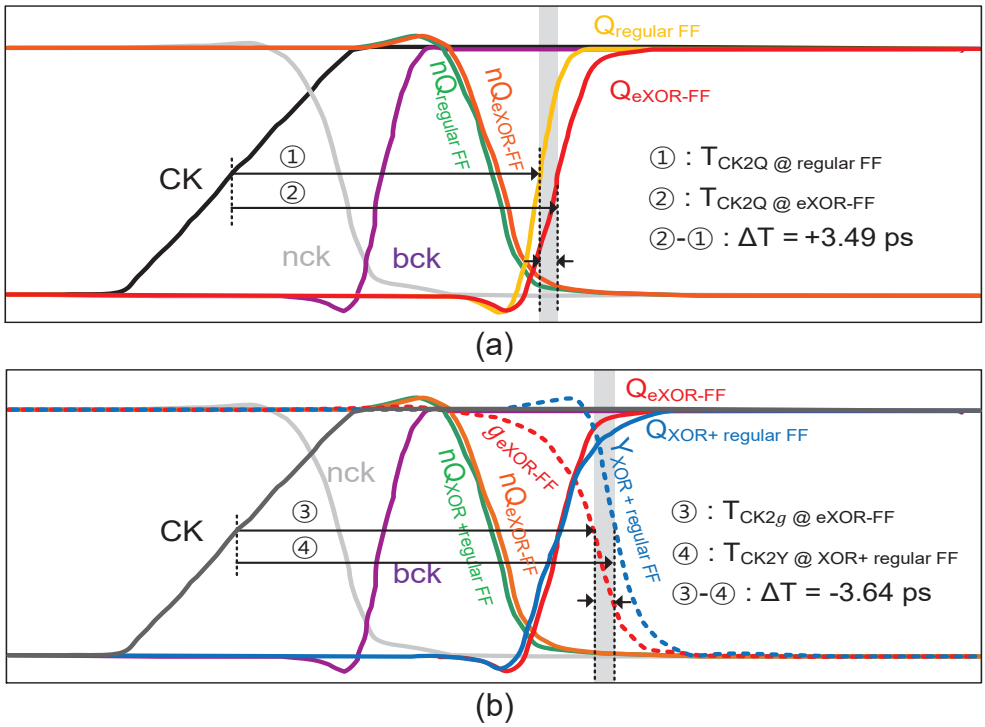


Figure 3.4: Timing waveforms for (a) clock-to-Q (measuring 3.49ps delay increase) and (b) clock-to-Y (measuring 3.64 delay decrease) in eXOR-FF in $Fig.3.2$.

Table 3.1: **Upper table**: The values of the *increase* of clock-to-Q delay on eXOR-FF for various combinations of Q loads, arranged light to heavy and clock's input transition times based on the cell chracterization look-up-table (LUT). The increase is 3.35ps on average; **Lower table**: The values of the *decrease* of clock-t-Y delay on eXOR-FF for various combination of Y loads, arranged light to heavy and clock's input transition times based on the cell charcterization LUT. The decrease is 20.18ps on average.

The increase (+ps) of clock-to-Q delay for eXOR-FF								
		Input trans. time						
		slew1	slew2	slew3	slew4	slew5	slew6	slew7
Load cap. @ Q	load1	3.51	3.49	3.49	3.47	3.50	3.50	3.50
	load2	3.39	3.36	3.35	3.36	3.40	3.30	3.30
	load3	3.32	3.34	3.31	3.30	3.40	3.30	3.30
	load4	3.40	3.40	3.30	3.30	3.40	3.30	3.30
	load5	3.40	3.40	3.40	3.50	3.40	3.30	3.20
	load6	3.40	3.40	3.20	3.20	3.20	3.30	3.20
	load7	3.80	3.20	3.20	3.20	3.40	3.20	3.30
Avg.		**3.35ps**						

The decrease (-ps) of clock-to-Y delay for eXOR-FF								
		Input trans. time						
		slew1	slew2	slew3	slew4	slew5	slew6	slew7
Load cap. @ Y	load1	-3.63	-3.62	-3.64	-3.65	-3.60	-3.60	-3.60
	load2	-4.70	-4.71	-4.64	-4.68	-4.70	-4.70	-4.80
	load3	-8.52	-8.47	-8.60	-8.70	-8.60	-8.50	-8.60
	load4	-15.20	-15.40	-15.50	-15.50	-15.30	-15.30	-15.50
	load5	-24.50	-24.60	-24.60	-25.70	-24.50	-24.20	-24.30
	load6	-36.20	-36.90	-35.50	-35.90	-35.60	-35.80	-35.70
	load7	-48.60	-48.40	-48.50	-48.40	-49.20	-48.00	-48.00
Avg.		**-20.18ps**						

in clock gating.

Clock-to-Y delay: Fig. 3.4(b) compares the timing waveforms for clock-to-Y (blue dashed one) on an XOR connected to a regular flip-flop and clock-to-Y (red dashed one) on an XOR in an eXOR-FF where Y indicates the output signal of XOR. We observed 3.64ps delay improvement. The amounts of delay decrease for various output loads on Y are listed in the lower part in Table 3.1. On average, 20.18ps reduction of delay on Y is possible if eXOR-FF is used.

3.1 Observation on Area Impact

In this section, we cover the area effect of the eXOR-FF. To calculate the cell area, we multiply the cell height and number of poly (number of criticla poly pitch). Table 3.2 shows the reference standard cell area of basic gates. In Fig. 3.5 (a), there are three basic gates of the cell library, inverter, 2-input nand and 2-input nor gate with small drivability. Fig. 3.5 (b) shows the cell size comparison. Individual 2-input XOR gate and 1 bit FF can be replaced into the single combined cell, called eXOR-FF. We can reduce the cell size more than 9.7% by combining two different cell into eXOR-FF, such as the cell side boundries and inverter pair which plays a double role

Table 3.2: Relative physical cell area in the standard cell library. Standard cell area is calculated by multiplying cell height and cell width. Cell width indicate number of CPP(critical poly pitch).

Cell name	# of CPP	Relative Cell area
Inverter	3	1.00
2-input NAND	4	1.33
2-input XOR	9	3.00
D Flip-Flop w/ Reset	22	7.33
Integrated Clock Gating (ICG)	15	5.00

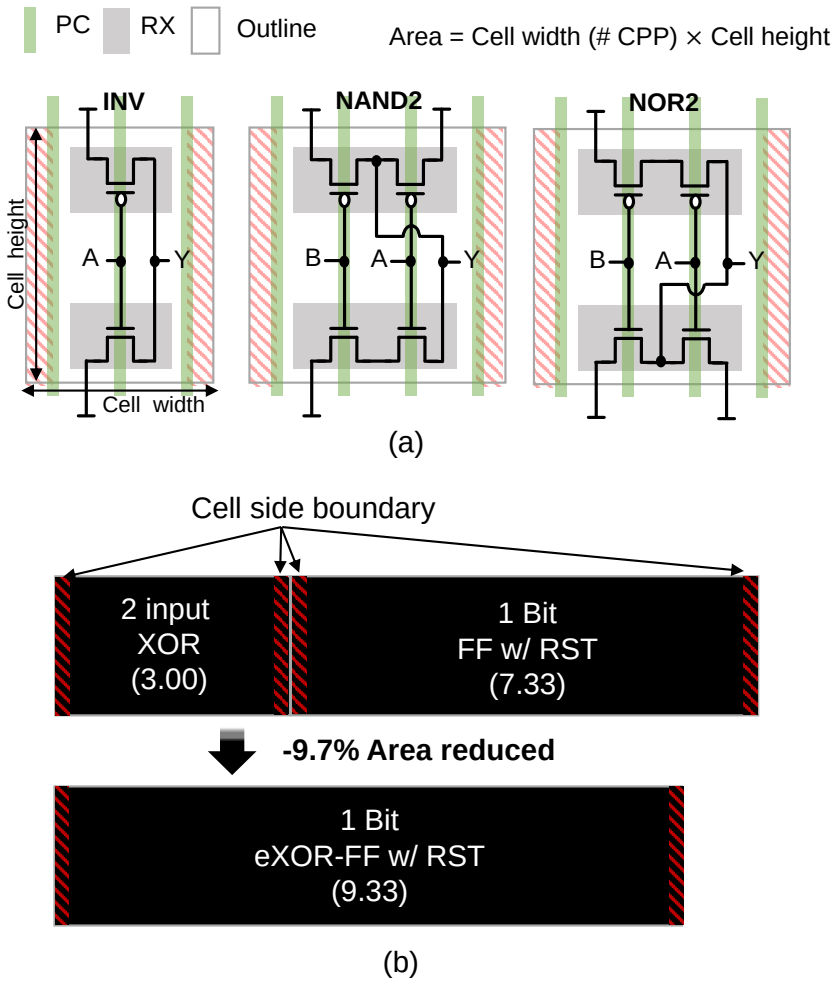


Figure 3.5: (a) Reference standard cell size of basic gate: inverter, nand2 and nor2 (b) Cell size comparison with the eXOR-FF and the conventional FF with XOR cell We can reduce the cell size more than 9.7% by combining two different cell into eXOR-FF, such as the cell side boundries and inverter pair which plays a double role.

Chapter 4

Placement-aware Clock Gating Methodology Utilizing eXOR-FF Cells

4.1 Overall Design Flow

Besides the compact eXOR-FF cells, another strength of our clock gating methodology over the existing ones is that our methodology explores clock gating solution space based on a set of accurate cost formulations for the power saving and timing impact. Since an accurate timing information is available only after the completion of cell placement, we devise two mechanisms to cope with the timing vulnerability caused by clock gating: (*Mechanism 1*) One is to iteratively explore clock gating solution space that leads to a minimal power saving while meeting timing constraint; (*Mechanism 2*) The other is to inlay our cost formulations of clock gating algorithhm with the flip-flops' slack data discussed in Observation 3 in Sec. 2.3.

In our clock gating exploration algorithm, a placement control parameter ρ is employed to balance the amounts of power saving and timing penalty during iterations (i.e., for $Mechanism1$). Thus, the objective function to be maximized at each iteration for searching clock gating is:

$$Cost_{CG} = PS^{new} - \rho \cdot T_{penalty} \tag{4.1}$$

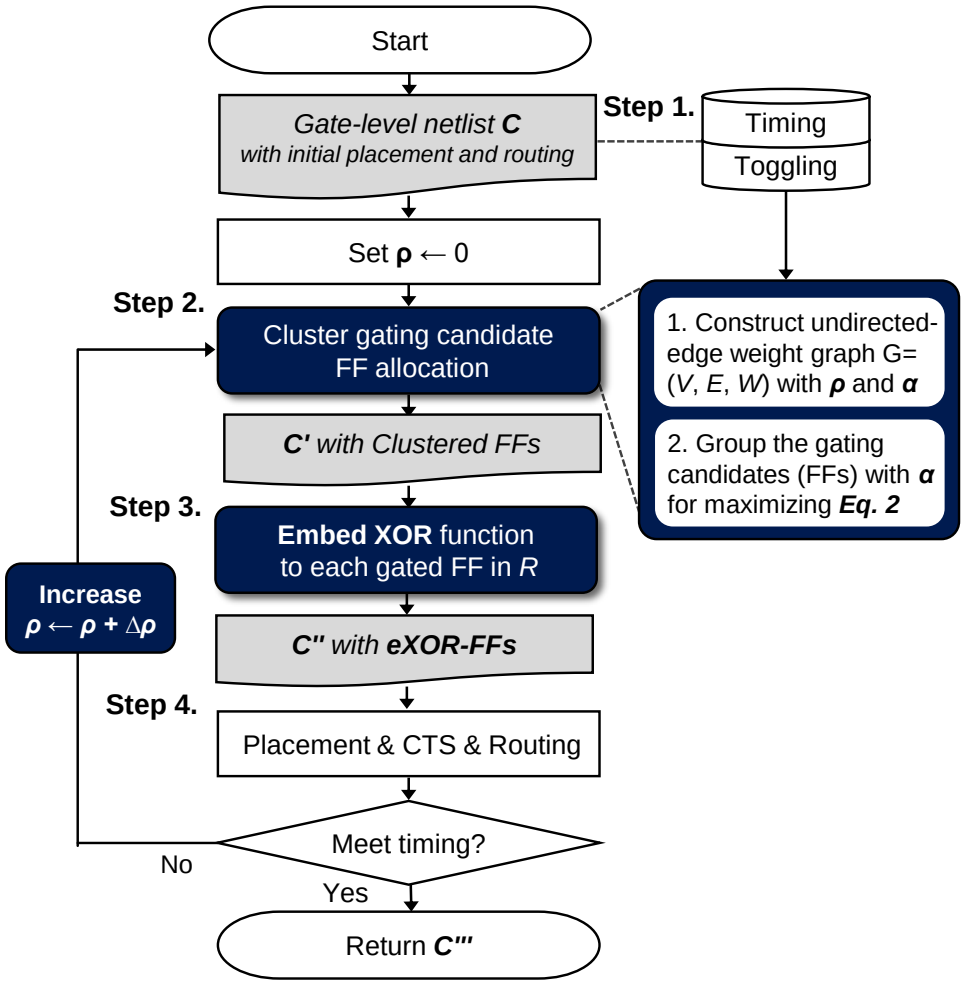


Figure 4.1: The flow of our proposed 4-step placement/timing-aware clock gating exploration algorithm.

where PS is the amount of power saving by clock gating and $T_{penalty}$ reflects the timing penalty in terms of timing slacks of the flip-flops grouped for clock gating (i.e., for $Mechanism2$). (The details on the cost formulations are provided in Sec. 4.2 and Sec. 4.3.)

As shown in Fig. 4.1, our placement/timing-aware clock gating exploration algorithm consists of four steps.

- **Step 1. (Pre-processing step)** *Extracting information about the slack times, togglings of flip-flops, and power statistics*.

  We gather the slack timing information (i.e. $T_{slack}(V_i)$) for all flip-flops in the block at initial physical design. It is possible to get the timing information in logic gate level, however we obtained slack information at the physical design stage for more detailed timing information calculated by parasitic RC and distance information. Each flip-flop could be distinguished by whether it is the starting point or the end point, but we set the worst timing margin of each flop as the unique timing information of the flops. And then, the unique inherent timing values of the flops are normalized by the worst timing value of the flops in the block. (i.e. Slack(V_i))

  To extract the toggling information of flip-flops, we apply N number of patterns to the input of the circuit for 100K clock cycles. By that, through a logic gate verilog simulation, it checks whether there is a transition of output of the flip-flops every clock cycle, and becomes a digital bit vector of 100K size with a value of 1 if there was a change, and a vlaue of 0 otherwise.

  We use the spice simulation to divide the power of the data-driven clock gating structure according to the group size for each function (XOR, flip-flop, ICG and OR-tree) to create the statistical power model. This power model is the linear function with X-axis is the toggling activity of the grouped flip-flop with same gated clock, and Y-axis is the power consumption in Fig. 4.2 (c) .

Those informations are used in computing the timing cost $T_{penalty}$ in *Eq.4.1* and *Eq.4.7*, joint toggling probablity $\alpha(R_j)$, and power saving costs in Sec. 4.2 and Sec. 4.3.

- **Step 2**. *Applying any conventional clock gating algorithm* that is based on graph based clustering for extracting flip-flop groups of clock gating (e.g., [11, 16]).

  The difference is that we use the following function, $w(i, j)$, as the edge weight of two flip-flops f_i and f_j by assuming a single group of two flip-flops, i.e., $\mathcal{R}$ = $\{R_1\}$ = $\{\{f_{1,i}, f_{1,j}\}\}$: (The detailed formulation on PS^{new} and $T_{penalty}$ are given in Sec. 4.3.)

$$w(i, j) = f_{CG} = PS^{new} - \rho \cdot T_{penalty} \tag{4.2}$$

  where ρ is the weight of the $T_{penalty}$. As this value increases, it plays the role of shifting the weight from the cost to the power to the timing effect.

  Note that as the grouping of flip-flops progresses, we update the flip-flop grouping set $\mathcal{R}$ accordingly, thus gradually approximating the cost function in *Eq. 4.1*, producing circuit $\mathcal{C}'$ from an initial circuit $\mathcal{C}$.

- **Step 3**. *Replacing all flip-flops selected for clock gating with eXOR-FF cells*, producing circuit $\mathcal{C}''$ from $\mathcal{C}'$. The clock gating logic corresponding to each group of filp-flops is also synthesized in this step Fig 4.4 (a) shows the case where clock gating is implementes withoout using eXOR-FF. Since XOR must compare input and output of the FF, it is connected as shown in Fig 4.4 (a). In the PnR stage, scoording to the routing priority, the inputs of the XOR may come in through the detoruing path. This detoured routing path causes an unintended loading penalty. In case of conventional data-driven clock gating algorithm, XOR and FF is placed seperately. It make the detoured routing path and it causes an intended loading penalty to the timing of each cells. On the other

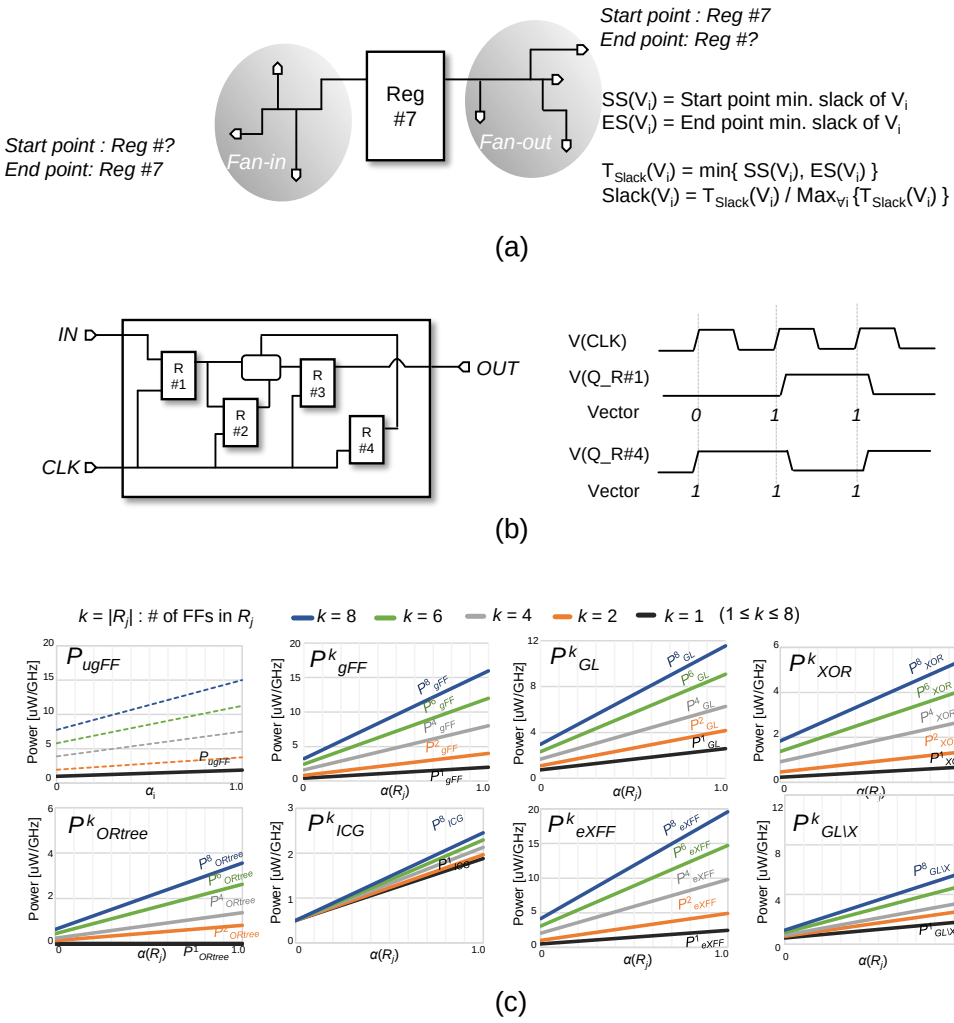


Figure 4.2: (a) shows how to get detailed timing information of flip-flop and extract Slack(V_i values. (b) shows how to extract toggling vector on individual flip-flop (c) indicates the linear power model with X-axis is the toggling activity and Y-axis is the power consumption

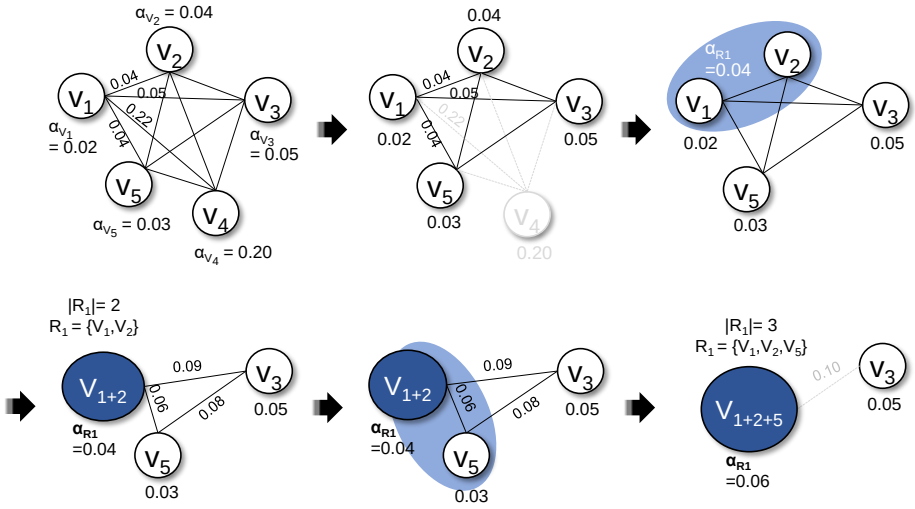


Figure 4.3: Example of the graph based clustering algorithm. G = $\{V,E,W\}$ is the undirected edge weighted graph and V is the individual 1-bit flip-flop and edge weight is the joint toggling probability of grouped flip-flops, where α_{V_1} is the toggling probability of the flip-flop V_1 and α_{R_1} is the joint toggling probability of the Group1 R_1 (i.e. V_1 and V_2)

hand, eXOR-FF does not generate the detouring path while using up the Metal 2 in the cell. To prove the effectiveness of eXOR-FF, we calculate the physical distance in Fig 4.4 (c) between XOR and FF in case of conventional clock gating by using manhattan distance in Fig 4.4 (b).

- **Step 4**. *Performing placement* for $\mathcal{C}''$, producing $\mathcal{C}'''$, from which the timing analysis will be carried out to check if there is a timing violation. If violated, ρ increases by $\Delta\rho$ and continue the iteration. Otherwise, return $\mathcal{C}'''$ with power saving numbers.

The details on the formulation and computation of the power and timing costs are given in the following two subsections. Let $\mathcal{R} = \{R_1, R_2, \cdots, R_M\}$ be the set of groups of flip-flops selected for clock gating. In addition, let $\alpha(R_j)$, $j = 1, \cdots, M$ be the joint toggling probability for the flip-flops in R_j.[1] We assume a circuit has N total number of flip-flops.

4.2 Cost Formulation for Conventional Clock Gating

The power saving by the conventional clock gating is:

$$PS^{old} = \sum_{i=1}^{N} P_{ugFF}(\alpha_i) - \sum_{j=1}^{M} \{P^k_{gFF}(\alpha(R_j)) + P^k_{GL}(\alpha(R_j))\} \tag{4.3}$$

$$P^k_{GL}(\alpha(R_j)) = P^k_{XOR}(\alpha(R_j)) + P^k_{ORtree}(\alpha(R_j)) + P^k_{ICG}(\alpha(R_j)) \tag{4.4}$$

[1]The joint probability is measured by intersecting the toggling vectors of the flip-flops in R_j that has been obtained by simulation in the pre-processing step. We simply use notation α_i to denote the toggling probability of a single flip-flip f_i.

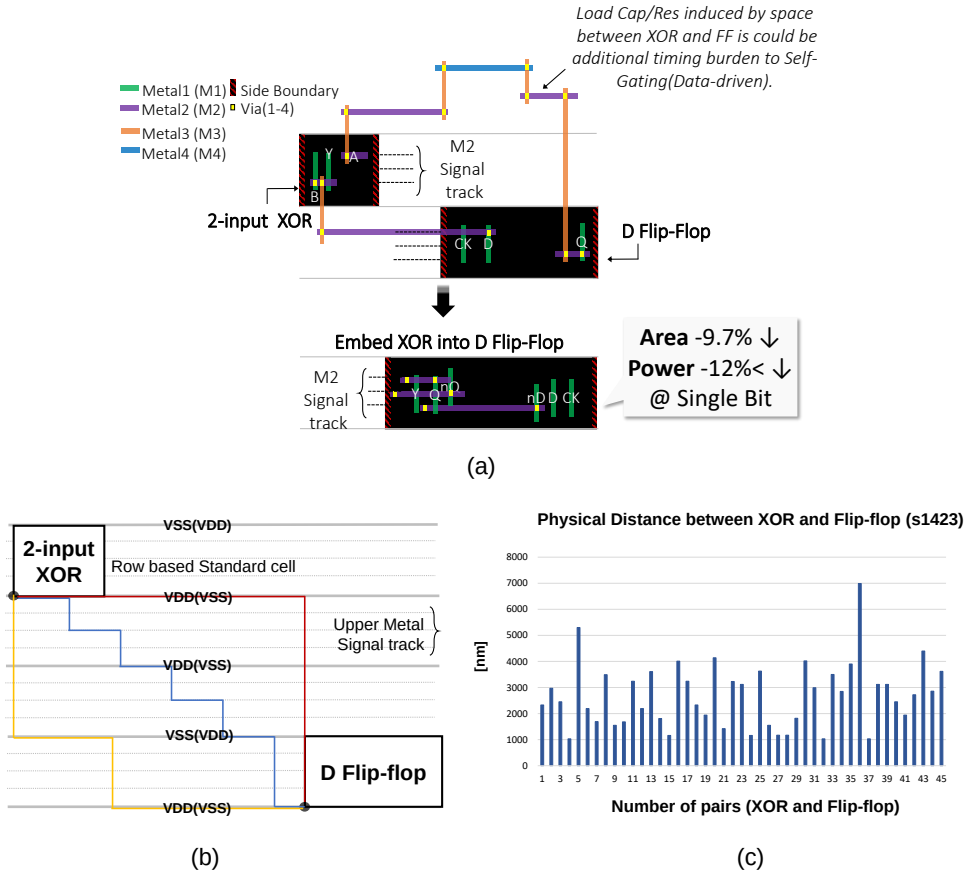


Figure 4.4: (a) This is the case where clock gating is implemented without using eXOR-FF. Since XOR must compare input and output of the FF, it is connected as shown in figure. In the PnR stage, scoording to the routing priority, the inputs of the XOR may come in through the detoruing path. This detoured routing path causes an unintended loading penalty. In case of conventional data-driven clock gating algorithm, XOR and FF is placed seperately. It make the detoured routing path and it causes an intended loading penalty to the timing of each cells. On the other hand, eXOR-FF does not generate the detouring path while using up the Metal 2 in the cell. To prove the effectiveness of eXOR-FF, we calculate the physical distance in (c) between XOR and FF in case of conventional clock gating by using manhattan distance (b).

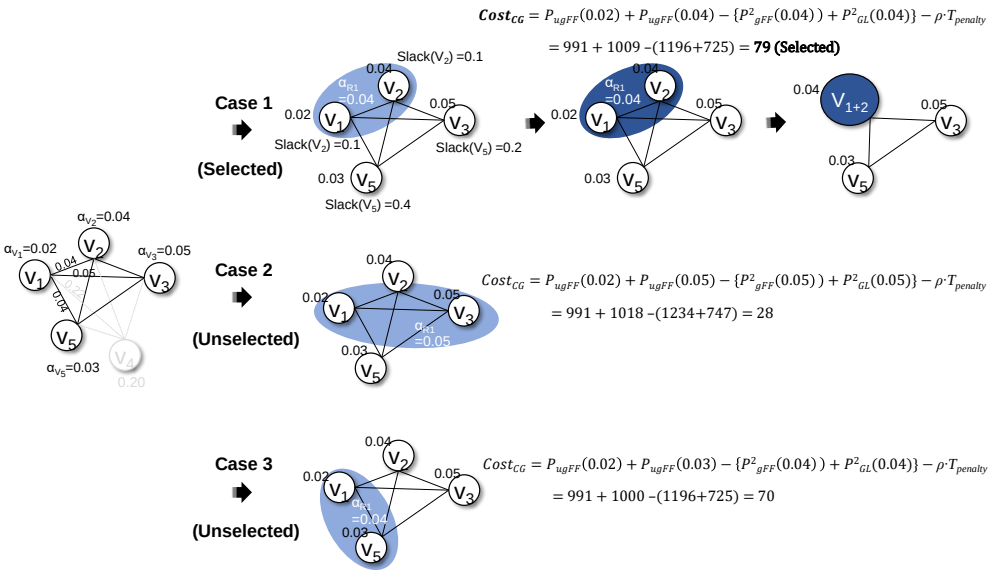


Figure 4.5: This figure shows the example of stepwise procedure of our algorithm for grouping FFs. Given the initial graph of FFs, we compute the cost function for grouping of FFs. At a first, we set the trade-off parameter between timing and power saving to zero. We can determine the grouping candidates for the flip-flop as V1 and V2. We compare all the grouping set R cost by evaluating the grouping candidates among all FFs.

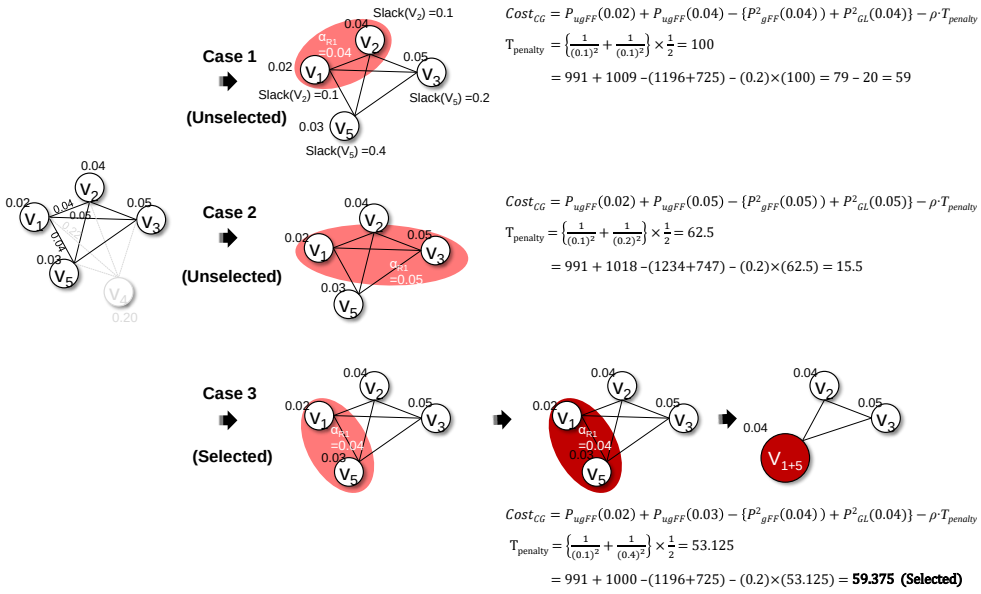


Figure 4.6: This figure shows another example of stepwise procedure of our algorithm for grouping FFs. If the timing is not met, we control the trade-off parameter between timing and power saving and repeat the process, until the timing is met. Suppose that the timing violation is occured in the final circuit, and then the ρ value is increased to 0.2. Due to the timing is not met, the trade-off parameter is changed, As the result, the grouping set has been changed.

· $P_{ugFF}(\alpha_i)$: the power consumed by an ungated individual flip-flop with toggling probability of α_i.

· $P^k_{gFF}(\alpha(R_j)$: the total power consumed by the gated flip-flops in group R_j. $\alpha(R_j)$ is the joint toggling probability of the flip-flops in R_j.

· $P^k_{GL}(\alpha(R_j))$: the total power consumed by the gating logic i.e., XORs, ORtree, and ICG for gating the flip-flops in R_j, whose powers are respectively represented by $P_{XOR}(\alpha(R_j))$, $P_{ORtree}(\alpha(R_j))$ and $P_{ICG}(\alpha(R_j))$.

4.3 Cost Formulation for Our Clock Gating using eXOR-FFs

The power saving by our clock gating using eXOR-FFs (eXFF for short). It combines a flip-flop and an *XOR* into a single unit, The new power cost is then expressed as:

$$PS^{new} = \sum_{i=1}^{N} P_{ugFF}(\alpha_i) - \sum_{j=1}^{M}\{P^k_{eXFF}(\alpha(R_j)) + P^k_{GL\setminus X}(\alpha(R_j))\} \tag{4.5}$$

$$P^k_{GL\setminus X}(\alpha(R_j)) = P^k_{ORtree}(\alpha(R_j)) + P^k_{ICG}(\alpha(R_j)) \tag{4.6}$$

· $P^k_{eXFF}(\alpha(R_j))$: the total power consumed by the eXOR-FFs that are replaced with the flip-flops in group R_j.

· $P^k_{GL\setminus X}(\alpha(R_j))$: the total power consumed by the gating logic i.e., ORtree and ICG for gating the eXOR-FFs corresponding to the flip-flops in R_j.

Finally, the term $T_{penalty}$ in *Eq.4.1* is formulated to:

$$T_{penalty} = \sum_{j=1}^{M}\sum_{i=1}^{|R_j|} \frac{1}{(Slack(f_{j,i}))^2} \times \frac{1}{|R_j|} \tag{4.7}$$

where $T_{slack}(f_{j,i})$ is $T_{slack} = T_{D2GCK} - T_{setup}(ICG)$ for the i^{th} flip-flop in group R_j, defined in *Eq.2.2* and Fig. 2.3. And then, the unique inherent timing values of the

flops are normalized by the worst timing value of the flops in the block (i.e. Slack(V_i)) The more flip-flops have large values of $Slack$, the less the value of $T_{penalty}$ is, offering the subsequent placement step being more likely to meet the timing constraint.

Note that a fast calculation of the power and timing costs is enabled by simply referring the pre-computed data in a lookup table (LUT) forms.

Chapter 5

Experiments

5.1 Experimental Setup

We used 28nm cell library and physical design kit (PDK) for evaluating the effectiveness of our proposed cell design and clock gating algorithm. We tested three sets of benchmark circuit netlists taken from ISCAS89, ITC99, and IWLS 2005. To build a worse timing environment, we used slow PVT corners. In addition, we generated various types of $eXOR$ library and characterized them to cover the cell driverability to the implementation of eXOR-FFs. We performed logic synthesis using Synopsys Design Compiler (DC) and physical design running IC Compiler (ICC), applying default tool options to all design process stages of logic synthesis, cell placement, and clock tree synthesis. The experiments were repeated several times, varying operating clock frequency in range of 0.325GHz to 1GHz, to genetate tough conditions that can cause timing violation.

5.2 Experimental Results

Table 5.1 summarizes the results (distribution of the size of clock gating groups of flip-flops, power saving, timing impact, and area overhead) for the designs produced

Table 5.1: Comparison of the distribution of clock gating groups of flip-flops, power saving, timing impact, and area overhead for the designs produced by the conventional data toggeling based clock gating in [1] and our clock gating methodology using eXOR-FF cells, new power/timing cost functions and placement-aware gating exploration flow. P_{total} columns represent the sum of cells' internal power, net switching power, and leakage power of the corresponding circuit; *Worst slack* columns indicate the timing margin on the longest (i.e., critical) combinational logic path; $Area$ columns represent the total cell area of the corresponding circuit.

Circuit	# of gates	# of FFs	Conventional (graph-based) clock gating ([1])				[1] + Ours (eXOR-FF cells, new power/timing costs, design flow)			
			Size dist. of gating FF groups (R)	Power [uW] P_{total} [uW] (Red.)	Timing [ps] *Worst slack*	Area [um^2] $Cell_{total}$ (Inc.)	Size dist. of gating FF groups (R)	Power [uW] P_{total} [uW] (Red.)	Timing [ps] *Worst slack*	Area [um^2] $Cell_{total}$ (Inc.)
s1423	386	74	2≤|R|≤3: 4 4≤|R|≤5: 9 6≤|R|≤8: 32	118.1 (-43.7%)	-16.8	493.7 (+29.4%)	2≤|R|≤3: 4 4≤|R|≤5: 5 6≤|R|≤8: 46	114.2 (-45.5%)	14.6	463.4 (+21.4%)
b11	325	30	2≤|R|≤3: 4 4≤|R|≤5: 4 6≤|R|≤8: 15	52.2 (-30.0%)	-2.3	264.9 (+26.1%)	2≤|R|≤3: 4 4≤|R|≤5: 8 6≤|R|≤8: 8	45.6 (-38.7%)	42.6	243.6 (+16.0%)
s9234	520	132	2≤|R|≤3: 0 4≤|R|≤5: 5 6≤|R|≤8: 87	282.2 (-29.1%)	-10.2	762.6 (+26.6%)	2≤|R|≤3: 0 4≤|R|≤5: 5 6≤|R|≤8: 88	252.8 (-36.7%)	2.2	724.2 (+20.4%)
s13207	839	213	2≤|R|≤3: 6 4≤|R|≤5: 5 6≤|R|≤8: 109	551.3 (-36.5%)	-2.2	1022.2 (+24.3%)	2≤|R|≤3: 3 4≤|R|≤5: 14 6≤|R|≤8: 90	573.5 (-34.0%)	12.8	959.4 (+16.7%)
b12	631	119	2≤|R|≤3: 0 4≤|R|≤5: 0 6≤|R|≤8: 108	102.1 (-64.5%)	-17.6	828.2 (+33.7%)	2≤|R|≤3: 2 4≤|R|≤5: 4 6≤|R|≤8: 88	95.0 (-66.9%)	3.4	784.3 (+26.6%)
s15850	335	128	2≤|R|≤3: 0 4≤|R|≤5: 0 6≤|R|≤8: 111	168.8 (-63.2%)	-9.5	675.6 (+41.8%)	2≤|R|≤3: 0 4≤|R|≤5: 0 6≤|R|≤8: 110	167.2 (-63.6%)	2.8	634.6 (+33.2%)
s38417	5007	1462	2≤|R|≤3: 11 4≤|R|≤5: 4 6≤|R|≤8: 1150	1346.5 (-44.9%)	-7.4	8456.4 (+32.7%)	2≤|R|≤3: 0 4≤|R|≤5: 5 6≤|R|≤8: 1165	1315.7 (-46.1%)	4.5	8034.2 (+25.6%)
ss_pcm	232	87	2≤|R|≤3: 26 4≤|R|≤5: 4 6≤|R|≤8: 23	252.3 (-30.8%)	-19.5	435.4 (+44.3%)	2≤|R|≤3: 37 4≤|R|≤5: 4 6≤|R|≤8: 22	217.1 (-40.8%)	3.4	416.4 (+38.3%)
b14	3665	215	2≤|R|≤3: 0 4≤|R|≤5: 5 6≤|R|≤8: 166	402.7 (-28.8%)	-4.9	2930.4 (+11.8%)	2≤|R|≤3: 17 4≤|R|≤5: 59 6≤|R|≤8: 102	387.7 (-31.5%)	21.6	2875.6 (+4.3%)
Avg. wrt initial circuits			68.7% coverage	-41.3%	*all violated*	+30.1%	68.3% coverage	**-44.9%**	***all met***	+22.5%
Avg. wrt clock gating [1]							0.1% less	**-5.6**%		**-5.3**%

by the conventional toggeling driven (graph-based) clock gating in [1] and our clock gating methodology using eXOR-FF cells (Fig. 3.2), new power/timing cost functions (Sec. 4.3), and placement-aware gating exploration flow. P_{total} columns indicate the total sum of cells' internal power, net switching power, and leakage power of the corresponding circuit; *Worst slack* cloumns indicate the timing margin on the longest combinational logic path; $Area$ columns represent the total cell area of the corresponding circuit.

In summary, in comparion with the conventional clock gating ([1]), which exhibits timing violation for all designs to aggressively save power, ours is able to synthesize clock gated designs which use 5.6% less power as well as 5.3% smaller area while meeting all timing constraints, even though the number of flip-flops seleced for clock gating by ours is a little less than that by [1].

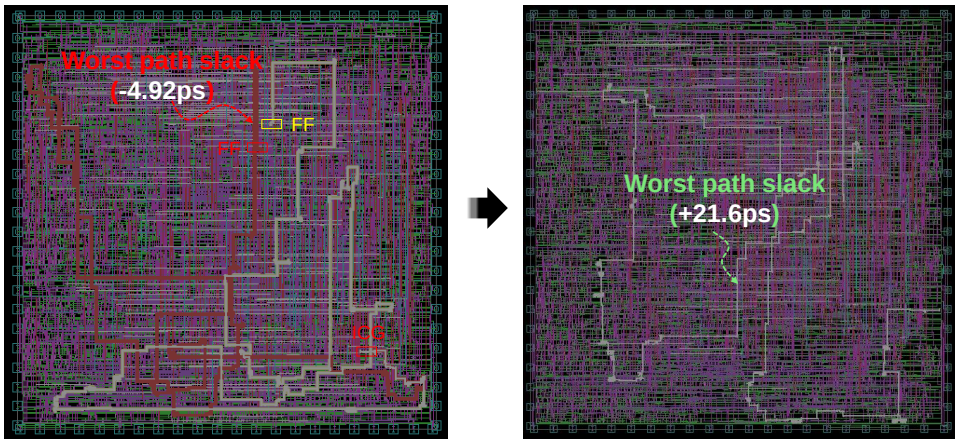
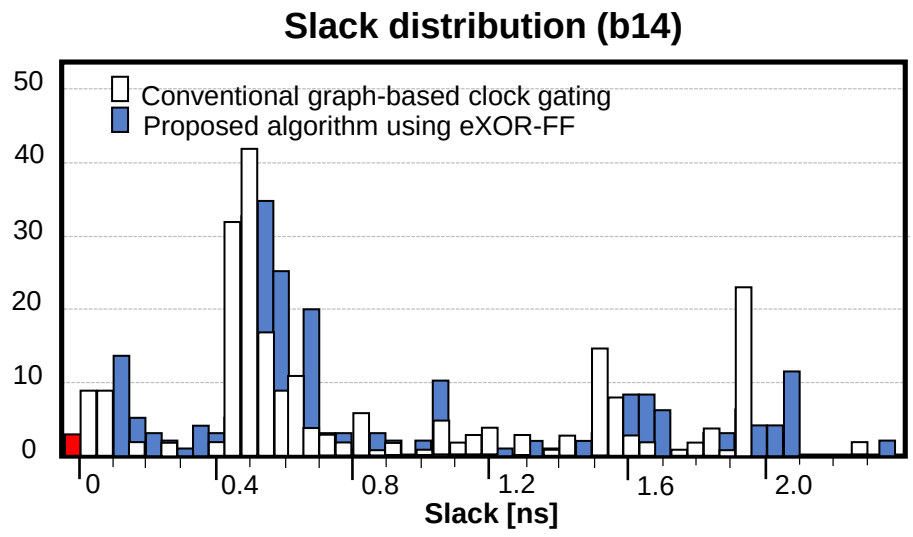


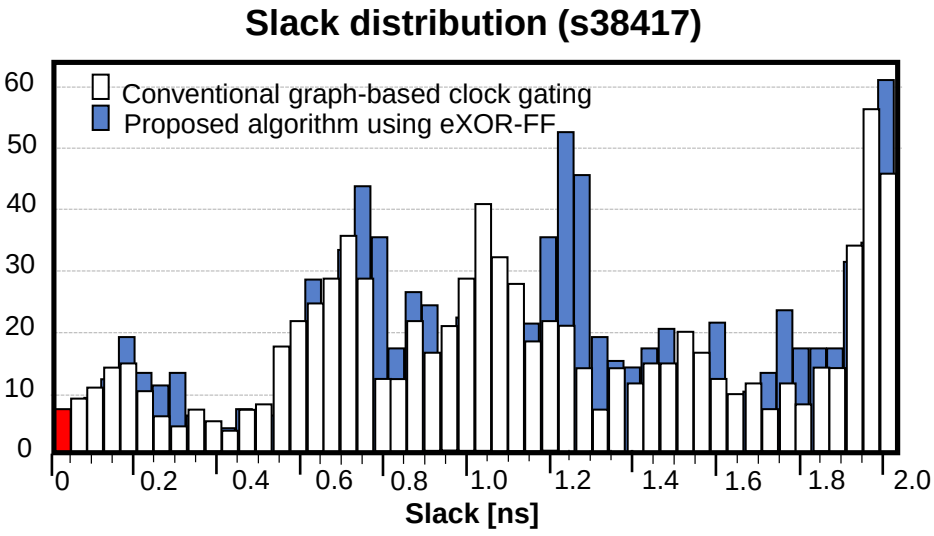
Figure 5.1: This figure shows the chip layout using by conventional algorithm (a) and ours (b) for benchmark circuit b14. Those highlighted lines in (a) indicate the timing violation paths. Red one is the worst timing path and its WNS(Worst Negative Slack) is -4.92ps. On the other hand, the green line in (b) indicates the worst path and its slack time is 21.6ps. And all timing paths meet the constraints.

5.3 Comparing with Industry Algorithm

Data-driven (i.e toggling-based) clock gating can be performed during the logic synthesis step in Design Compiler(DC), or during the clock tree synthesis and clock optimization step in IC Compiler(ICC). We compare our clock gating algorithm with the prior methodology by using commands, compile_ultra -gate _clock -self_gating using DC. DC recognizes the cell placement in a topological mode and inserts the XOR self-gating using RTL or gate-level netlist. Gate-level netlists produced our clock gat-



(a)



(b)

Figure 5.2: Those graphs are the timing path slack distributions of corresponding circuits. (a) shows the slack distribution for b14 circuit and (b) is for s38417. White graph bars indicate the conventional graph-based clock gating case, on the other hand blue graph bars indicate proposed algorithm using eXOR-FF. Path slack distribution are right-shifted by using ours.

ing algorithm with eXOR-FF do not use topographical mode.

To see the power reduction due to tool embedded optimization algorithm, we measure the power consumption on several designs was measured post-placement. Timing violation has occured in some designs when performaing clock gating using the tool. However, since ours algorithm does not generate timing violations, we do not include timing information for an equivalent comparisons.

We conducted experiments with six different circuits to see the amount of power reduction. Table 5.2 summarizes the results in terms of the numbers of resulting flip-flops, the power $P_{Int.}$, $P_{Sw.}$, P_{Static} and P_{Total} of cell's internal, net's switching, static and total power consumption of the circuits.

Through the analysis of the Table 5.2, we were able to figure out the adavantages and disadavantagbes of out algorithm. 1) Ours gating algorithm saves the internal power of the cell by 10.2% (up to 22.4% in b11)on average and 8.6% (up to 25.0% in b11) on average in the view of the total power. In other words, we can see that this effect has the effect of increasing gating covereage of the flip-flops in the whole chip through our gating algorithm. 2) DC based clock gating algorithm implemented efficient distribution and placement of cell and inter-connect net by using topographical mode considering the physical information from the synthesis stage. As a result, the DC based clock gating algorithm shows a maximum 36.1% increase in net switching power in the same design as s38417.

Table 5.2: Comparison of power saving for the designs produced by original circuit without clock gating, the conventional data toggeling based clock gating in [1] and our clock gating methodology using eXOR-FF cells, new power/timing cost functions and placement-aware gating exploration flow. $P_{Int.}$ columns represent the sum of cells' internal power, $P_{Sw.}$ columns represent the net switching power, and P_{Static} columns indicate the leakage power of the corresponding circuit

Circuit	# of gates	# of FFs	Power $[uW]$											
			Original Circuit (w/o CG)				DC based CG				Ours CG			
			$P_{Int.}$	$P_{Sw.}$	P_{Static}	P_{Total} (Red.)	$P_{Int.}$	$P_{Sw.}$	P_{Static}	P_{Total} (Red.)	$P_{Int.}$	$P_{Sw.}$	P_{Static}	P_{Total} (Red.)
B11	325	30	67.4	6.14	0.95	74.5	46.5	13.5	0.87	60.8 (-18.3%)	36.1	8.19	1.34	45.6 (-38.7%)
S9234	520	132	337.5	59.5	2.26	399.3	219.8	56.5	2.70	279.1 (-30.1%)	193.4	56.3	3.09	252.8 (-36.7%)
S13207	839	213	737.5	127.8	3.21	868.5	480.9	119.4	3.68	603.9 (-30.4%)	458.2	111.2	4.08	573.5 (-34.0%)
B12	631	119	275.8	9.5	2.14	287.4	68.6	29.3	2.50	100.4 (-65.1%)	69.4	22.0	3.60	95.0 (-66.9%)
S15850	335	128	397.3	59.7	1.84	458.9	155.1	35.5	2.64	193.2 (-57.9%)	127.1	37.0	3.08	167.2 (-63.6%)
S38417	5007	1462	2011.2	409.3	23.37	2442.9	861.6	340.3	28.96	1230.8 (-49.0%)	815.5	464.1	37.14	1315.7 (-46.1%)

Chapter 6

Conclusion

This work proposed a set of comprehensive solutions to the problem of toggeling based clock gating [19]. Precisely, our contributions were three folds: (1) reducing control logic area overhead through co-optimzation of control logic and flip-flop cells, (2) accurate decomposition of power consumptions and their formulation, and (3) placement/timing aware clock gating exploration methodology through a careful analysis on timing impact and integrating 1 and 2. Through experiments with benchmark circuits, it was confirmed that our solutions were very effective, reducing power by 5.6% on average further while meeting all timing constraints over the (all timing violated) circuits produced by the conventional clock gating method.

Bibliography

[1] S. Wimmer and I. Koren, "The optimal fan-out of clock network for power minimization bt adaptive gating," *IEEE TVLSI*, vol. 20, no. 10, 2012.

[2] A. DEVICE, "Activity: Cmos logic circuits, transmission gate xor," in *https://wiki.analog.com/university/courses/ electronics/electronics-lab-30*, 2017.

[3] D. Markovic, B. Nikolic, and R. Brodersen, "Analysis and design of low-energy flip-flops," in *Low Power Electronic and Design*, 2002.

[4] S. Iyer, "Stanford university coursework: Cmos power comsumption," in *http://large.stanford.edu/courses/2010/ph240/iyer2/*, 2010.

[5] C.-T. Sah, "Fundamentals of solid state electronics," in *World Scientific*, 1996.

[6] D. J. Frank, "The limits of cmos scaling from a power constrained technology optimization persepective," in *nanoHUB*, 2006.

[7] R. Jan M and P. Massoud, "Low power design methodologies," in *The Springer international series in engineering and computer sceience*, 1996.

[8] D. Yi and T. Kim, "Switch cell optimization of power-gated modern system-on-chips," in *IEEE/ACM ICCAD*, 2017.

[9] V. G. Oklobdzija, V. M. Stojanovic, D. M. Markovic, and N. M. Nedovic, "Digital system clocking: High-performance and low-power aspects," in *Wiley-IEEE Press*, 2003.

[10] S. Jairam, M. Rao, J. Srinivas, and P. Vishwanath, "Clock gating for power optimization in asic design cycle theory & practice," in *ACM/IEEE ISLPED*, 2008.

[11] S. Wimer and I. Koren, "Design flow for flip-flop grouping in data-driven clock gating," *IEEE TVLSI*, vol. 22, no. 4, 2014.

[12] G. Palumbo, F. Pappalardo, and S. Sannella, "Evaluation on power reduction applying gated clock approaches," in *IEEE ISCAS*, 2002.

[13] L. Benini, A. Bogliolo, and G. De Micheli, "A survey of design techniques for system-level dynamic power management," *IEEE TVLSI*, vol. 8, no. 3, 2000.

[14] Q. Wang and S. Roy, "Power minimization by clock root gating," in *ACM/IEEE ISLPED*, 2008.

[15] L. Benini, P. Siegel, and G. De Micheli, "Saving power by synthesizing gated clocks for sequential circuits," *IEEE Design & Test of Computors*, vol. 11, no. 4, 1994.

[16] I. Han, J. Kim, J. Yi, and Y. Shin, "Register grouping for synthesis of clock gating logic," in *IC Design and Technology*, 2016.

[17] R. Fraer, G. Kamhi, and M. K. Mhameed, "A new paradigm for synthesis and propagation of clock gating conditions," in *ACM/IEEE DAC*, 2008.

[18] M. Muller, S. Simon, H. Gryska, A. Wortmann, and S. Buch, "Low power synthesizable register file for processor and ip cores," *Integration, the VLSI Journal*, vol. 39, no. 2, pp. 131–155, 2006.

[19] G. Yang and T. Kim, "Design and algorithm for clock gating and flip-flop co-optimization," in *IEEE/ACM ICCAD*, 2018.

초 록

본 논문에서는 표준 셀에서부터 배치 단계에 이르는 다양한 설계단에에서 칩의 동적 전력을 최적화 기법을 소개한다. 이 연구는 우선 데이터 구동형 (즉, 토글링 기반) 클럭 게이팅이 종래 클럭 게이팅 기법들에서 결코 다루어지지 않았던 플립 플롭의 합성과 밀접하게 통합될 수 있는 방법을 연구한다. 우리의 관측의 핵심은 플립 플롭 셀의 일부 내부 부품이 클럭 게이팅 인에이블 신호를 생성 하기 위해 재사용 될 수 있다는 것이다. 이를 바탕으로 eXOR-FF 라고 불리는 새롭게 최적화된 플립 플롭 배선 구조를 제안합니다. 이 구조에서는 매 클럭 주기마다 내부 로직을 재사용하여 클럭 게이팅을 통해 플립 플롭을 활성화할지 또는 비활성화할지 결정합니다. 모든 쌍의 플립 플롭 및 토글릴 감지 로직에서의 영역을 절약함에 따라서 누설 및 동적 전력의 절전 효과를 달성합니다. 그런 다음, 두 가지고유한 장점을 제공하는 배치/타이밍 인식 클럭 게이팅 탐색에 대한 포괄적인 방법론을 제안합니다. 해당 방법론은 eXOR-FF 의 이점을 극대화하고, 전력 소비 및 타이밍 영향의 분해에 대한 정밀 분석을 수행하고 틀럭 게이팅 참색의 핵심 엔진을 비용기능으로 변환하는데 가장 적합합니다. ISCAS89, ITC89, ITC99 및 IWLS 2005의 벤치 마크 회로를 사용한 실험을 통해 제안 된 방법이 이전의 데이터 구동 클록 게이팅 방식과 비교하여 총 전력을 5.6 % 및 면적으로 5.3 % 줄일 수 있음을 보여 주었다.

주요어: 저전력 설계, 표준 셀, 플립플롭, 논리합성, 클럭 게이팅

학번: 2017-27057