A Parameterized Timing-aware Flip-flop Merging Algorithm for Clock Power Reduction

Chaochao Feng*, Daheng Yue*, Zhenyu Zhao*, Zhuofan Liao†
*School of Computer, National University of Defense Technology, Changsha 410073, China
{fengchaochao, yuedaheng, zyzhao}@nudt.edu.cn
†School of Computer and Communication Engineering, Changsha University of Science & Technology, China
csulzf@gmail.com

***Abstract*—In modern integrated circuits, the clock power contributes a dominant part of the chip power. Clock power can be reduced effectively by utilizing multi-bit flip-flops. In this paper, a parameterized timing-aware flip-flop merging algorithm is proposed for clock power reduction. The single-bit flip-flops are merged into multi-bit flip-flops after placement & optimization and before clock network synthesis with consideration of function information, scan chain information, distance and timing constraints. The algorithm can be configured with different parameters, such as the bit-number of MBFF, the setup timing margin and the distance margin. Experimental results under an industrial design show that compared with the basic design without MBFF, the design with 2-bit, 4-bit, 6-bit, and 8-bit MBFFs can save 7.5%, 12%, 11.8% and 11.1% total power consumption respectively. Using MBFF4 to replace 1-bit FFs is the best choice for the design optimization, which achieves minimum area and total power consumption. We also compare the designs with MBFF4 replacement under five different setup timing margins and distance margins. Without violating any timing constraint, it is better to set the setup timing margin as small as possible to achieve best power optimization. The distance margin (100μm, 30μm) is the best choice for this industry design to achieve minimum power consumption.**

I. INTRODUCTION

With the enhancement of the integration degree, power consumption has become a limiting constraint in modern multicore SoC design. Among the overall chip power consumption, the switching power of the clock network consumes more than 40% of the total power [1]. Clock signals switching each cycle is one reason that clock consumes so much power. Another reason is that the clock network needs to drive a large number of flip-flops which have huge load capacitance. Thus many design methodologies have been proposed to reduce clock power, such as buffer sizing [2], clock gating [3], register clustering [4] and DVFS [5]. Recently, multi-bit flip-flops (MBFFs) replacement has become a promising technology for clock power optimization [6][7].

An MBFF consists of two or more 1-bit flip-flops (FFs), which share common clock drivers. Replacing multiple 1-bit FFs with one MBFF will significantly reduce the number of clock inverters and the number of clock sinks, which will reduce the wire length of the clock network and the buffers/inverters required in the clock network. Thus, power consumed by the clock network will be substantially reduced. Fig. 1 shows an example of replacing two 1-bit FFs with one 2-bit MBFF.

Utilizing MBFFs was first proposed in [8], which groups 1-bit FFs into MBFFs before floorplanning and placement to reduce clock latency and skew. A design methodology for applying MBFFs during logic synthesis was proposed in [9] for power reduction. However, it is hard to consider its effect on timing if applying MBFFs in early design stage, e.g., before placement. Recently, a few works [10]–[13] considered merging 1-bit FFs into MBFFs at the post-placement stage for clock power optimization.

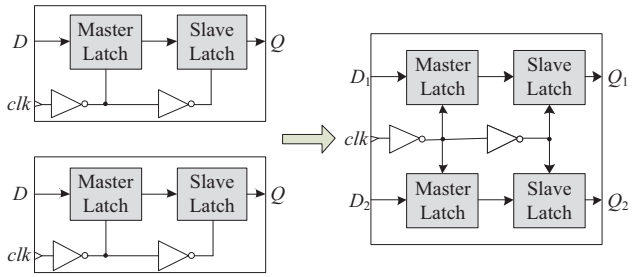

Fig. 1. Example of merging two 1-bit flip-flops into one 2-bit flip-flop.

In this paper, a parameterized timing-aware flip-flop merging algorithm is proposed for clock power reduction. The single-bit flip-flops are merged into multi-bit flip-flops after placement & optimization and before clock network synthesis with consideration of function information, scan chain information, distance and timing constraints. The flip-flops merging algorithm is a heuristic algorithm which can be configured with different parameters, such as the bit-number of MBFF, the setup timing margin and the distance margin. Experimental results under an industrial design show that compared with the basic design without MBFF, the design with 2-bit, 4-bit, 6-bit, and 8-bit MBFFs can save 7.5%, 12%, 11.8% and 11.1% total power consumption respectively. Using MBFF4 to replace 1-bit FFs is the best choice for the design optimization, which achieves minimum area and total power consumption. We also compare the designs with MBFF4 replacement under five different setup timing margins and distance margins. Without violating any timing constraint, it is better to set the setup timing margin as small as possible to achieve best power optimization. The distance margin (100μm, 30μm) is the best choice for this industry design to achieve minimum power consumption.

II. PROBLEM FORMULATION

The post-placement flip-flops merging problem can be defined as follow: Given a placed design D, a library L with MBFFs, replace 1-bit flip-flops by MBFFs as much as possible such that the power consumption and clock wire-length are minimized and the timing, scan chain and distance constraints

are satisfied. In this section, we discuss the flip-flops merging problem from the following four aspects: function, scan chain, distance and timing.

A. Function Consideration

Replacing a few 1-bit flip-flops (i.e., $f_1,\ldots f_n$, n>1) with an MBFF f_m without changing the original function must satisfy the following conditions:

(1) The n 1-bit flip-flops are the same FF type in the library;
(2) The bit width of MBFF f_m must be equal to the sum of bit widths of the n 1-bit FFs and the MBFF f_m has the same control pins with the n 1-bit flip-flops;
(3) The n 1-bit flip-flops can be merged into an n-bit MBFF if and only if their control pins, such as clock, set, reset and scan-enable, are driven by the same nets.

B. Scan Chain Consideration

Merging flip-flops into an MBFF must also consider the scan chain information such that the scan chain is not changed due to the merging operation. In this paper, the flip-flops to be merged must obey the following rules:

(1) The flip-flops to be merged must be on the same scan chain;
(2) The scan-in (*SI*) pins of the flip-flops to be merged must have only one fan-in;
(3) The clock (*clk*) and scan-enable (*SE*) pins of the flip-flops to be merged must connect the same nets;
(4) The scan chain order must be preserved after merging.

C. Distance Consideration

Merging FFs into MBFF arbitrarily without considering the location information may lead to timing violation and routing problem. So besides the function and scan chain information, we also consider the location information of the flip-flops to be merged. We define the distance constraints of the flip-flops to be merged as follow:

Given n flip-flops ($f_1,\ldots f_n$, n>1), the n flip-flops can be replaced by an n-bit MBFF f_m if and only if the maximum distances along X and Y axis among these flip-flops do not exceed two threshold values respectively.

The maximum distance along X axis, called ΔX, is defined as: $\max\{|x_i-x_j|\}$ ($1\le i, j\le n$ && $i\ne j$), where x_i/x_j is the x coordinate value of the location of flip-flop f_i/f_j.

The maximum distance along Y axis, called ΔY, is defined as: $\max\{|y_i-y_j|\}$ ($1\le i, j\le n$ && $i\ne j$), where y_i/y_j is the y coordinate value of the location of flip-flop f_i/f_j.

In our algorithm, the two threshold values for ΔX and ΔY are set to be configured parameters to achieve better power optimization effect. Fig. 2 shows the example of distance constraints for four flip-flops to be merged.

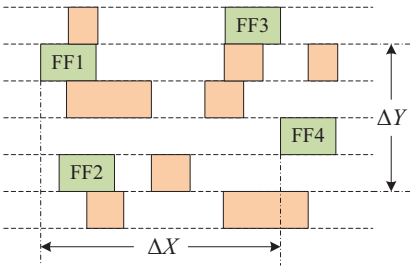

Fig. 2. Example of distance constraints for FFs to be merged.

D. Timing Consideration

Finally, the most important constraint is the timing constraint for the flip-flops to be merged. Given n flip-flops to be merged into a functionally-equivalent MBFF, if these flip-flops meet the scan chain and distance constraints, they can be merged if and only if the input D-pin timing slack and the output Q-pin timing slack for each flip-flop should be positive. The input D-pin timing slack is defined as the minimum setup timing slack among all timing paths to the D-pin of a flip-flop. The output Q-pin timing slack is defined as the minimum setup timing slack among all timing paths from the Q-pin of a flip-flop. In our algorithm, the D-pin and Q-pin timing slack is also set to be a configured parameter which can be 0 or any positive number to achieve better power optimization effect.

In addition, another rule which is related to the timing should be obeyed in our algorithm if the useful skew [14] option is opened in the placement stage. The place-and-route tool will preset a clock latency on the clock pin of the flip-flop when the useful skew option is opened during placement. So the rule is that before flip-flops merging, we also check that the clock latency on the clock pins of the flip-flops to be merged must be 0, which means the useful skew should not apply on the flip-flops to be merged.

III. Parameterized Timing-Aware Flip-Flop Merging Algorithm

The pseudo code of the parameterized timing-aware flip-flop merging algorithm is shown in Fig. 3. The inputs of the algorithm are a placed design database D without MBFF, a library L with MBFFs and four configurable parameters. In order to simplify the complexity, we only use the MBFF with the same number of bits each time, so the bit-number n of MBFF should be set before. The other three parameters are the D-pin and Q-pin setup timing margin *slack* and distance margin (ΔX, ΔY) which have been discussed in Section II. The output of the algorithm is a new placed design database D_{new} with MBFFs.

Firstly, all flip-flops with scan pins are gathered in a queue *scan_FFs_queue* in line 1. From line 2 to 33, the while loop performs until the length of the *scan_FFs_queue* reaches 0. In line 3, the first element of the *scan_FFs_queue* is fetched into a variable *f*. In line 5, n sequential flip-flops including *f* are searched and checked starting from *f* on the scan chain by a procedure get_merge_group. A variable *bad_chain* which is set to be 0 initially in line 4 is used for marking whether the n sequential flip-flops are met the scan chain rule or not. The function of the procedure get_merge_group is to search n FFs on the scan chain starting from the flip-flop *f* and to check if the n FFs meet the scan chain constraint or not. The procedure will return the *merge_group* containing n FFs and the *bad_chain*. In line 6-9, if the *bad_chain* equals to 1, the flip-flop *f* will be removed from *scan_FFs_queue* and the while loop continues. From line 10 to 14, the distance constraints are checked. The maximum distances of FFs in the *merge_group* along X and Y axis are calculated by the procedure get_distance. If the maximum distances are larger than the preset margin ΔX or ΔY, *f* will be removed from *scan_FFs_queue* and the while loop continues. In line 15-22, the type of FFs in *merge_group* is checked by the procedure check_FF_type and the clock and scan-enable pins are checked if driven by the same nets

through the procedure check_CLK_SE_pin_net. If the type of FFs is not the same one or the *clk*-pin and *SE*-pin are not connected by the same nets, *f* will be removed from *scan_FFs_queue* and the while loop continues. The timing margin and clock latency are checked by procedures check_timing_margin and check_clock_latency from line 23 to 30. If the timing margin and clock latency checkings are not satisfied, *f* will be removed from *scan_FFs_queue* and the while loop continues. If all above checkings pass, the *n* FFs in *merge_group* will be replaced by an MBFF with *n* bits in line 31. Then the MBFF will be placed in a suitable location in line 32. To decide the location, we get the minimum *x* and *y* coordinates (*min_x*, *min_y*) of the *n* FFs in *merge_group* by the procedure get_distance. Thus the MBFF will be placed at the coordinate ($min\_x+\Delta X/2$, $min\_y+\Delta Y/2$). When the while loop finishes, a refine-place operation (line 34) will be executed on the whole design to solve the overlap problems due to the replacement operation.

FFs in the *merge_group* can be merged without violating any timing constraint corresponds to the *D*-pin and *Q*-pin setup timing slack. The procedure check_timing_margin uses the following two commands in the place-and-route tool, like Cadence Innovus, to get the *D*-pin and *Q*-pin setup timing slack of all *n* FFs in the *merge_group*. If all *D*-pin and *Q*-pin setup timing slacks are larger than or equal to the preset timing margin *slack*, the *n* FFs in the *merge_group* can be replaced by an *n*-bit MBFF.

```
get_property [get_pins $f/D] slack_max
get_property [get_pins $f/Q] slack_max
```

As discussed in Section II, the FFs which use useful skew cannot be merged. In the procedure check_clock_latency, the following command is used to get clock latency for all clock pins of *n* FFs in the *merge_group*. If any clock latency of clock pins is not equal to 0, the *n* FFs in the *merge_group* cannot be merged.

```
get_property [get_pins $f/CP] latency_rise_max
```

Parameterized timing-aware flip-flop merging algorithm

Input: a placed design database *D*, a library *L* with MBFFs
bit-number of MBFF *n*
D-pin and *Q*-pin setup timing margin *slack*
distance margin (ΔX, ΔY)

Output: a new placed design database D_{new} with MBFFs

```
1:  scan_FFs_queue ← get_all_scan_FFs()
2:  while{length_of_queue(scan_FFs_queue)>0}
3:   f ← get_first_element(scan_FFs_queue)
4:   bad_chain ← 0
5:   {merge_group, bad_chain} ← get_merge_group(f, n)
6:   if bad_chain == 1 then
7:     scan_FFs_queue ← scan_FFs_queue - {f}
8:     continue
9:   end if
10:  {distance_x, distance_y} ← get_distance(merge_group, n)
11:  if distance_x > ΔX || distance_y > ΔY then
12:    scan_FFs_queue ← scan_FFs_queue - {f}
13:    continue
14:  end if
15:  if check_FF_type(merge_group) == 0 then
16:    scan_FFs_queue ← scan_FFs_queue - {f}
17:    continue
18:  end if
19:  if check_CLK_SE_pin_net(merge_group) == 0 then
20:    scan_FFs_queue ← scan_FFs_queue - {f}
21:    continue
22:  end if
23:  if check_timing_margin(merge_group, slack) == 0 then
24:    scan_FFs_queue ← scan_FFs_queue - {f}
25:    continue
26:  end if
27:  if check_clock_latency(merge_group) == 0 then
28:    scan_FFs_queue ← scan_FFs_queue - {f}
29:    continue
30:  end if
31:  Replace n FFs in merge_group by an MBFFn
32:  Place_MBFF(MBFFn)
33: end while
34: RefinePlace()
```

Fig. 3. Parameterized timing-aware flip-flop merging algorithm.

IV. Experimental Results

We implement the flip-flops merging algorithm in TCL language and integrate it in Cadence Innovus flow. The timing-aware FF merging is performed after placement. After that merged MBFF will be relocated in reasonable location and other related standard cells will be made a legalization. Then, clock tree synthesize will be performed and followed by routing and optimization. We also use Cadence Conformal to check the functional equivalence of the two netlists before and after merging, which demonstrates the correctness of the algorithm.

The parameterized timing-aware flip-flop merging algorithm has been tested on a 16*nm* industrial design which has about 300 thousands instances and 100 thousands flip-flops. The flip-flops merging algorithm can be configured with different parameters. We set five different setup timing margins *slack*, i.e., 0*ps*, 5*ps*, 10*ps*, 15*ps* and 20*ps* and set five distance margins (ΔX, ΔY) with unit μm, i.e., (100, 30), (50, 15), (30, 10), (20, 5) and (10, 3) in the experiments. In the library, we choose MBFF2, MBFF4, MBFF6 and MBFF8 respectively in the experiments.

A. Results of MBFFs with different number of bits

TABLE I shows the design characteristics of basic design and design with 2-bit, 4-bit, 6-bit and 8-bit MBFFs replacement. The setup timing margin *slack* is set to be 20*ps* and the distance margin (ΔX, ΔY) is set to be (100μm, 30μm). The industrial design should run at the maximum frequency 1.4GHz. All designs with MBFFs replacement have met the performance requirement, which means FFs merging does not violate any timing constraint. Compared with the basic design without MBFF, the design with 2-bit, 4-bit, 6-bit, and 8-bit MBFFs can save 7.5%, 12%, 11.8% and 11.1% total power consumption respectively. It can be concluded that using MBFF4 to replace 1-bit FFs is the best choice for the design optimization, which achieves minimum area and total power consumption. In comparison to the basic design, the design with MBFF4 replacement can save 2.2% area and 12% total power consumption. The design with MBFF4 has 63.7% shorter clock wire length and achieves 13.4% less clock network power than the basic design.

TABLE I. DESIGN CHARACTERISTICS OF MBFFS WITH DIFFERENT NUMBER OF BITS

	Total cells	Total area (μm^2)	Total regs	MBFF regs	Total wire length (mm)	Clock wire length (mm)	Clock buffers	Total clock cap (pF)	Freq (GHz)	Total power (mW)	Clock network power (mW)	Reg power (mW)
Base	310305	212850	102346	0	2955	575	12113	191.72	1.468	171.4	94.9	41.5
MBFF2	280897	213764	74751	27595	2316	209	9673	62	1.418	158.6	86.3	37.2
MBFF4	254044	208272	52036	16770	2356	209	7020	63	1.427	150.9	82.2	34.4
MBFF6	253033	208312	50841	10301	2446	199	6865	62	1.420	151.1	82.6	34.4
MBFF8	255617	208546	53514	6976	2516	206	6915	64	1.473	152.3	83.3	35.0

B. Results of different setup timing margins

TABLE II shows the results of designs with MBFF4 replacement under five different setup timing margins from 0ps to 20ps. The distance margin is set to be (100μm, 30μm). The maximum frequency requirement is met under all five setup timing margins. It can be observed that as the setup timing margin decreases, more flip-flops can be replaced by MBFFs, which is consistent with our expectations. The design with setup timing margin of 0ps achieves the minimum total power consumption, which is 2.8% less than the design with setup margin of 20ps. It can be concluded that without violating any timing constraint, it is better to set the setup timing margin as small as possible to achieve best power optimization.

TABLE II. RESULTS UNDER DIFFERENT SETUP TIMING MARGINS

Setup timing margin (ps)	Total cells	Total area (μm^2)	Total regs	MBFF regs	Total power (mW)	Clock network power (mW)
0	247309	207595	45892	18818	146.7	79.5
5	248193	207846	46846	18500	148.7	80.6
10	250152	207841	48742	17868	148.5	80.8
15	252140	207832	50428	17306	148.5	80.5
20	254044	208272	52036	16770	150.9	82.2

C. Results of different distance margins

TABLE III shows the results of the design with MBFF4 under five different distance margins. The setup timing margin is set to be 20ps. It can be seen that the distance margin (100μm, 30μm) is the best choice for this industry design to achieve minimum power consumption. If the distance margin is set to be too small, like (10μm, 3μm), fewer FFs will be replaced by MBFFs. To achieve better replacement effect, the distance margin should set to be a suitable value, which is according to the specific design case by case.

TABLE III. RESULTS UNDER DIFFERENT DISTANCE MARGINS

Distance margin (μm)	Total cells	Total area (μm^2)	Total regs	MBFF regs	Total power (mW)	Clock network power (mW)
(10, 3)	289909	210952	83827	6173	161.2	88.5
(20, 5)	277130	209798	72451	9965	156.9	85.9
(30, 10)	263238	209638	58687	14553	155.2	85.8
(50, 15)	255827	208182	53866	16160	152.3	84.1
(100, 30)	254044	208272	52036	16770	150.9	82.2

V. CONCLUSIONS

In this paper, we propose a parameterized timing-aware flip-flop merging algorithm for clock power reduction. The algorithm is a heuristic algorithm which can be configured with different parameters, such as the bit-number of MBFF, the setup timing margin and the distance margin. In order to ensure the correctness, single-bit flip-flops are merged into multi-bit flip-flops with consideration of function information, scan chain information, distance and timing constraints. This algorithm has been integrated into Cadence's Innovus place-and-route tool flow, achieving good results in an industrial design. In future work, we will consider placement and routing congestion information in the flip-flop merging algorithm.

ACKNOWLEDGMENT

The research is partially supported by the National Natural Science Foundation of China under Grant No. 61471375, No. 61402056 and No. 61303066.

REFERENCES

[1] D. Papa, C. Alpert, C. Sze, Z. Li, N. Viswanathan, G.-J. Nam, and I. L. Markov, "Physical synthesis with clock-network optimization for large systems on chips," *IEEE Micro*, vol. 31, no. 4, pp. 51–62, 2011.

[2] K. Wang and M. Marek-Sadowska, "Buffer sizing for clock power minimization subject to general skew constraints," in *Design Automation Conf. (DAC)*, pp. 159–164, 2004.

[3] M. Donno, A. Ivaldi, L. Benini, and E. Macii, "Clock-tree power optimization based on RTL clock-gating," in *Design Automation Conf. (DAC)*, pp. 622–627, 2003.

[4] Y. Cheon, P.-H. Ho, A. B. Kahng, S. Reda, and Q. Wang, "Power-aware placement," in *Design Automation Conf. (DAC)*, pp. 795–800, 2005.

[5] S. M. Martin, K. Flautner, T. Mudge, and D. Blaauw, "Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads," in *Int. Conf. on Computer-Aided Design (ICCAD)*, pp. 721-725, 2002.

[6] G. Wu, Y. Xu, D. Wu, M. Ragupathy, Y.-y. Mo, and C. Chu, "Flip-flop clustering by weighted k-means algorithm," in *Design Automation Conf. (DAC)*, 2016.

[7] I. Seitanidis, G. Dimitrakopoulos, P. Mattheakis, L. M. Navette and D. Chinnery, "Timing driven incremental multi-bit register composition using a placement-aware ILP formulation", in *Design Automation Conf. (DAC)*, 2017.

[8] R. Pokala, R. Feretich, and R. McGuffin, "Physical synthesis for performance optimization," in *IEEE Int. ASIC Conf. and Exhibit*, pp. 34–37, 1992.

[9] Y. Kretchmer, "Using multi-bit register inference to save area and power: The good, the bad, and the ugly," in *EE Times Asia*, 2001.

[10] Y. T. Chang, C. C. Hsu, P. H. Lin, Y. W. Tsai, and S. F. Chen, "Post-placement power optimization with multi-bit flip-flops," in *Int. Conf. on Computer-Aided Design (ICCAD)*, pp. 218–223, 2010.

[11] I. H. R. Jiang, C. L. Chang, Y. M. Yang, E. Y. W. Tsai, and L. S. F. Chen, "INTEGRA: Fast multi-bit flip-flop clustering for clock power saving based on interval graphs," *IEEE Trans. on Computer-Aided Design (TCAD)*, vol. 31, no. 2, pp. 192-204, 2011.

[12] J.-T. Yan and Z.-W. Chen, "Construction of constrained multi-bit flipflops for clock power reduction," in *IEEE Int. Conf. on Green Circuits and Systems*, pp. 675–678, 2010.

[13] Z.-W. Chen and J.-T. Yan, "Routability-driven flip-flop merging process for clock power reduction," in *IEEE Int. Conf. on Computer Design (ICCD)*, pp. 203–208, 2010.

[14] W. Shen, Y. Cai and W. Chen, "Useful clock skew optimization under a multi-corner multi-mode design framework", in *Int. Symp. on Quality Electronic Design (ISQED)*, pp.598-601, 2010.