Probability-Driven Multibit Flip-Flop Integration With Clock Gating

Doron Gluzer and Shmuel Wimer

***Abstract*— Data-driven clock gated (DDCG) and multibit flip-flops (MBFFs) are two low-power design techniques that are usually treated separately. Combining these techniques into a single grouping algorithm and design flow enables further power savings. We study MBFF multiplicity and its synergy with FF data-to-clock toggling probabilities. A probabilistic model is implemented to maximize the expected energy savings by grouping FFs in increasing order of their data-to-clock toggling probabilities. We present a front-end design flow, guided by physical layout considerations for a 65-nm 32-bit MIPS and a 28-nm industrial network processor. It is shown to achieve the power savings of 23% and 17%, respectively, compared with designs with ordinary FFs. About half of the savings was due to integrating the DDCG into the MBFFs.**

***Index Terms*— Clock gating (CG), clock network synthesis, low-power design, multibit flip-flop (MBFF).**

I. Introduction

The data of digital systems are usually stored in flip-flops (FFs), each of which has its own internal clock driver. In an attempt to reduce the clock power, several FFs can be grouped into a module called a multibit FF (MBFF) that houses the clock drivers of all the underlying FFs. We denote the grouping of kFFs into an MBFF by a k-MBFF. Kapoor *et al.* [1] reported a 15% reduction of the total dynamic power in a 90-nm processor design. Electronic design automation tools, such as Cadence Liberate, support MBFF characterization.

The benefits of MBFFs do not come for free. By sharing common drivers, the clock slew rate is degraded, thus causing a larger short-circuit current and a longer clock-to-Q propagation delay t_{pCQ}. To remedy this, the MBFF internal drivers can be strengthened at the cost of some extra power. It is therefore recommended to apply the MBFF at the RTL design level to avoid the timing closure hurdles caused by the introduction of the MBFF at the backend design stage. Due to the fact that the average data-to-clock toggling ratio of FFs is very small, which usually ranges from 0.01 to 0.1 [2], the clock power savings always outweigh the short-circuit power penalty of the data toggling.

An MBFF grouping should be driven by logical, structural, and FF activity considerations. While FFs grouping at the layout level have been studied thoroughly, the front-end implications of MBFF group size and how it affects clock gating (CG) has attracted little attention. This brief responds to two questions. The first is what the optimal bit multiplicity k of data-driven clock-gated (DDCG) MBFFs should be. The second is how to maximize the power savings based on data-to-clock toggling ratio (also termed *activity* and *data toggling probability*).

An MBFF usage at the RTL logic synthesis design stage can be found in [3] for a 55-nm 230-MHz design of a system on a chip. Santos *et al.* [3] restricted the MBFF grouping into FFs belonging

Manuscript received April 9, 2016; revised August 5, 2016; accepted September 25, 2016. This work was supported by the Israel Chief Scientist's Magnet Program, HiPer consortium.

The authors are with the Engineering Faculty, Bar-Ilan University, Ramat Gan 52900, Israel (e-mail: bdgluzer@gmail.com; wimers@biu.ac.il).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2016.2614004

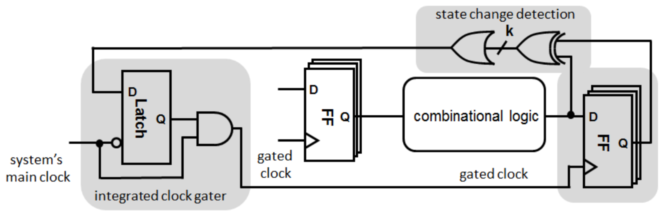

Fig. 1. DDCG integrated into a k-MBFF.

to the same bus. Both 2-MBFFs and 4-MBFFs were used with a 20% increase in t_{pCQ}. A dynamic power reduction of 13% was achieved with some degradation in timing convergence. This was remedied by applying low voltage threshold cells on critical paths, which somewhat increased the leakage power. The total area was increased by 2.3%, because of the timing fixes.

Most published works on MBFF have focused on physical implementation, driven primarily by the postplacement layout [4], [5], [7], [8], [13], [16]. In these works, FF activities tend to be ignored. Each FF is associated with time margins derived from the layout comprising 1-bit FFs. The wires connected to the data input and output of an FF are anchored on their opposite side to the rest of the logic, whereas the position of the FF is allowed to move around without violating timing. This defines the region in the layout where the FF can be displaced and merged into the MBFF. The 2-MBFF merging is formulated as an optimization problem that aims at maximizing the number of merged FFs. Other works [9]–[11] have introduced clock-tree layout considerations as well.

To further save power, [6] introduced CG, but the relationship among the CG strategy, the FF activities, and their grouping was not conclusive. Wang *et al.* [12] described another postplacement algorithm that accounted implicitly for switching data to estimate the expected power. Although [6] and [12] used switching data as a secondary criterion in postplacement FF grouping, our strategy is to use it as a primary clustering criterion, and do so at the preplacement RTL level.

The requirement in [4]–[13] and [16] of having a timing-converged postplacement layout as the starting point for the MBFF design flow is a burden. Timing constraints may be very tight, thus limiting the potential FF merging. Moreover, it considerably reduced the solution space by confining the merging to layout proximity, whereas RTL merging is free of such constraints. No less important, having a timing-converged layout as a prerequisite reduces the impetus to modify the design to MBFFs. Our rationale by contrast is that the clock-gated MBFF should be introduced at the RTL design level, based on architectural, structural, and, most importantly, FF activity considerations.

The main contributions of this brief are as follows:

1) a design methodology that fuses MBFF and DDCG, yielding considerable power savings;
2) a probability-driven algorithm that minimizes the expected DDCG MBFF power consumption.

The remainder of this brief is organized as follows. Section II discusses the effect of data-to-clock toggling probabilities on energy savings and how it affects the integration of DDCG with MBFF.

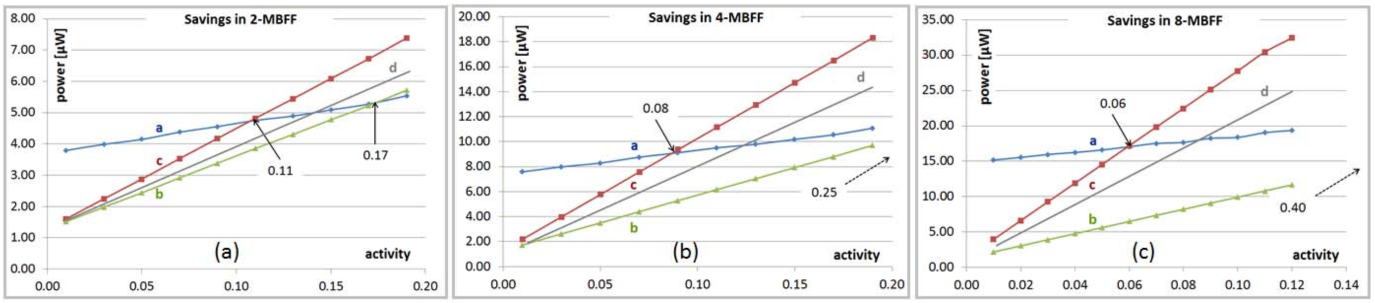

Fig. 2. Power consumption of k 1-bit FFs compared to k-MBFF: 2-MBFF (a), 4-MBFF (b) and 8-MBFF (c). Line (a) is the power consumed by k 1-bit FFs driven independently of each other. Line (b) is the ideal case of simultaneous (identical) toggling. Line (c) is the worst case of exclusive (disjoint) toggling. Line (d) is an example of realistic toggling.

TABLE I
DEPENDENCE OF THE OPTIMAL MBFF MULTIPLICITY ON DATA-TO-CLOCK TOGGLING ACTIVITY

p	0.01	0.02	0.05	0.1
k	8	6	4	3

Section III analyzes what FFs should be grouped into the DDCG MBFF used in Section IV in a grouping algorithm and a design flow. Section V presents the experimental results, and Section VI draws conclusions.

II. INTEGRATING CLOCK GATING INTO MBFF

Let p be the data-to-clock toggling probability. The expected energy E_1 consumed by a 1-bit FF is

$$E_1(p) = \lambda_1 + \mu_1 p \tag{1}$$

where λ_1 is the energy of the FF's internal clock driver and μ_1 is the energy of data toggling. In the general case of k-MBFF, let λ_k be the energy of the MBFF's internal clock driver and μ_k its per-bit data toggling energy. Assume that the FFs toggle with probability p independently of each other. It has been shown in [14] that the expected energy is

$$E_k(p) = \sum_{j=0}^{k} (\lambda_k + j\mu_k)\binom{k}{j} p^j (1-p)^{k-j} = \lambda_k + k\mu_k p. \tag{2}$$

It is important to note that toggling independence is a pessimistic assumption. In reality, the correlation between FF toggling yields higher energy savings than the model in [2].

The ratio $(kE_1(p) - E_k(p))/kE_1(p)$ expresses the energy saving potential of k-MBFF. The coefficients λ and μ of the 65-nm 1-MBFF, 2-MBFF, and 4-MBFF were derived with SPICE simulations. Zero activity ($p = 0$) yields 35% savings for the 2-MBFF and 55% savings for the 4-MBFF, whereas full activity ($p = 1.0$) yields 15% savings for the 2-MBFF and 23% savings for the 4-MBFF. In typical VLSI systems, the average p does not exceed 0.1, so high savings are achievable. Section III generalizes the energy consumption model in (2) to the case of distinct data toggling probabilities.

Fig. 1 illustrates a DDCG integrated into a k-MBFF. The shaded circuits reside within a library cell. Given an activity p, the group size k that maximizes the energy savings solves the equation

$$(1-p)^k \ln(1-p) C_{\text{FF}} + \frac{C_{\text{latch}}}{k^2} = 0 \tag{3}$$

where C_{FF} and C_{latch} are the clock input loads of an FF and a latch, respectively [2]. The solution to (3) for various activities is shown in Table I for typical C_{FF} and C_{latch}. The above optimization does not take into account the clock driver sharing, which also affects the optimal grouping as shown below.

To grasp the power savings of a k-MBFF achievable by DDCG, Fig. 1 was simulated with SPICE for various activities p and $k = 2, 4, 8$. Fig. 2(a) shows the power consumption of a 2-MBFF. Line (a) is the power consumed by two 1-bit FFs driven independently of each other. The 3.8-μW power at zero activity is due to the toggling of the clock driver at each FF, and it is always consumed regardless of the activity.

Line (b) corresponds to the ideal case where the two FFs toggle simultaneously (identically). In this case, the clock driver shared by the two FFs either toggles for the sake of the two or is disabled by the internal gate shown in Fig. 1. As expected, the power consumed for zero activity is smaller than two 1-bit FFs. As the activity increases, the power of line (b) rises faster than that of line (a) since the gating circuit overhead consumes power proportionally to the activity. There is no point in using a 2-MBFF beyond the 0.17 activity crossing point, the case where power starts to be lost.

Line (c) shows the case where the FFs toggle exclusively (disjoint). This is obviously the worst case: although the clock driver works for the two FFs, only one needs it. Similar to line (b), in exclusive toggling, there is no point in using the 2-MBFF if the FF activities are higher than 0.11.

For a given activity and toggling scenarios of line (b) or (c), the power saving of the 2-MBFF is their distance to line (a). Note that for zero activity, the per-bit power saving is $(3.8 - 1.8)/2 = 1.0$ μW. The interim line, line (d), shown between the extreme cases of lines (b) and (c), represents a more realistic operation where FFs within an MBFF toggle neither identically nor exclusively.

Fig. 2(b) shows the power consumed by the 4-MBFF, where line (a) corresponds to four 1-bit FFs driven independently of each other, line (b) represents the best case of simultaneous toggling of the four FFs, and line (c) represents the worst case of exclusive toggling. For zero activity, the per-bit power saving is $(7.4 - 2.2)/4 = 1.3$ μW, which is larger than the 1.0 μW in the 2-MBFF. Note, however, that for the worst case of exclusive toggling, the 4-MBFF stops saving at 0.08 activity, compared with 0.11 in the 2-MBFF. In the best case of simultaneous toggling, the 4-MBFF is always favored over the 2-MBFF. Similar conclusions hold for the 8-MBFF shown in Fig. 2(c). Its per-bit power saving for zero activity is $(15.3 - 2.5)/8 = 1.6$ μW. The saving of the 8-MBFF stops at 0.06 activity in the worst case and at 0.40 in the best case.

III. WHAT FFS SHOULD BE GROUPED IN AN MBFF?

Clearly, the best grouping of FFs that minimizes the energy consumption can be achieved for FFs whose toggling is highly correlated. Using toggling correlations for MBFF grouping has the drawback of requiring early knowledge of the value change dump vectors of a typical workload. Such data may not exist in the early design stage. More commonly available information is the average toggling bulk probability of each FF in the design, which can be estimated from earlier designs or the functional knowledge of modules. FFs' toggling probabilities are usually different from each other. An important question is therefore how they affect their grouping. We show below that data-to-clock toggling probabilities matter and should be considered for energy minimization.

Given n FFs $\{\mathrm{FF}_i\}_{i=1}^{n}$, let us consider their grouping in 2-MBFFs. We denote by $\mathrm{FF}_{(i,j)}$ a 2-MBFF, comprising FF_i and FF_j, toggling independently of each other with probabilities p_i and p_j, respectively. When neither is toggling, the clock of $\mathrm{FF}_{(i,j)}$ is disabled by the gate and the internal clock driver does not consume dynamic energy. When both FF_i and FF_j are toggling, the clock of $\mathrm{FF}_{(i,j)}$ is enabled and the clock driver energy is useful for both FFs so there is no waste. Waste occurs when one FF is toggling, but its counterpart is not, a case where the enabled clock signal drives both FFs, but only one needs it. Waste $W_{(i,j)}$ of half the internal clock driver energy λ_2 thus occurs [see (2) for $k = 2$]

$$\begin{aligned} W_{(i,j)} &= \frac{\lambda_2}{2}[p_j(1-p_i) + p_i(1-p_j)] \\ &= \frac{\lambda_2}{2}(p_i + p_j - 2p_i p_j). \qquad (4) \end{aligned}$$

Given four FFs FF_i, FF_j, FF_k, and FF_l, their pairing in two 2-MBFFs yields the following energy waste:

$$W_{(i,j)} + W_{(k,l)} = \frac{\lambda_2}{2}[(p_i + p_j + p_k + p_l) - 2(p_i p_j + p_k p_l)]. \qquad (5)$$

Whereas $(p_i + p_j + p_k + p_l)$ in (6) is independent of the pairing, $(p_i p_j + p_k p_l)$ is dependent on the pairing. $W_{(i,j)} + W_{(k,l)}$ is minimized when $(p_i p_j + p_k p_l)$ is maximized. If $p_i \le p_j \le p_k \le p_l$, the pairing $\{\mathrm{FF}_{(i,j)}, \mathrm{FF}_{(k,l)}\}$ is favored over $\{\mathrm{FF}_{(i,k)}, \mathrm{FF}_{(j,l)}\}$ and $\{\mathrm{FF}_{(i,l)}, \mathrm{FF}_{(j,k)}\}$ [14]. The generalization for pairing of n FFs is straightforward. Let n be even and $\mathbf{P}: \{\mathrm{FF}_{(s_i,t_i)}\}_{i=1}^{n/2}$ be a pairing of $\mathrm{FF}_1, \mathrm{FF}_2, \ldots, \mathrm{FF}_n$ in $n/2$ 2-MBFFs. The energy waste is

$$\sum_{i=1}^{n/2} W_{(s_i,t_i)} = \frac{\lambda_2}{2}\left[\sum_{j=1}^{n} p_j - 2\sum_{i=1}^{n/2} p_{s_i} p_{t_i}\right]. \qquad (6)$$

Since $\sum_{j=1}^{n} p_j$ is independent of the pairing, (6) is minimized when $\sum_{i=1}^{n/2} p_{s_i} p_{t_i}$ is maximized. Let the FFs be ordered such that $p_1 \le p_2 \le \cdots \le p_n$. The pairing $\{\mathrm{FF}_{(2i-1,2i)}\}_{i=1}^{n/2}$ of the FFs in successive order was shown in [14] to minimize (6). A generalization for n/k k-MBFFs proved that $\{\mathrm{FF}_{(k(i-1)+1,\ldots,ki)}\}_{i=1}^{n/k}$, which groups k successive FFs, minimizes the energy waste. The case where n is not divisible by k has also been addressed.

IV. Capturing Everything in a Design Flow

In the following paragraphs, we combine the activity p and the MBFF multiplicity k in a design flow aimed at minimizing the expected wasted energy. Fig. 2(a)–(c) illustrates that the power savings of the 2-MBFF, 4-MBFF, and 8-MBFF, respectively, are used. Knowing the activity p of an FF, the decision as to which MBFF size k it best fits follows the interim lines, lines (d). To obtain the per-bit power consumption, lines (d) in Fig. 2(a)–(c), representing an MBFF realistic operation, were divided by their respective multiplicity. The result is shown in Fig. 3.

To maximize the power savings, Fig. 3 divides the range of FF activity into regions. The black line follows the power consumed by a 1-bit ungated FF. The triangular areas bounded by the black line and each of the green, blue, and red per-bit lines show the amount

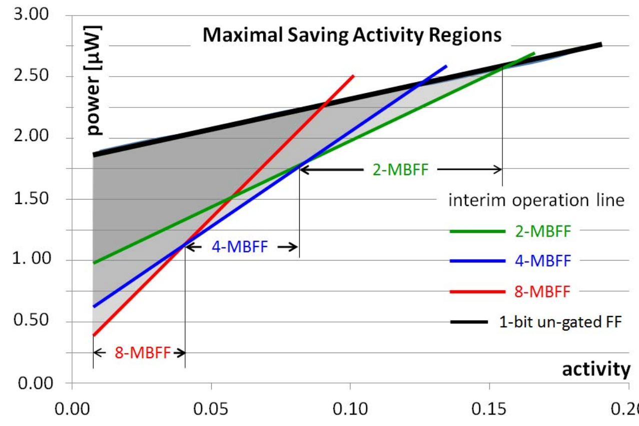

Fig. 3. Division of the activity into ranges of maximal savings.

of power savings per activity obtained by grouping an FF in the 2-MBFF, 4-MBFF, and 8-MBFF, respectively. It shows that for a very low activity, it pays to group FFs into an 8-MBFF. As activity increases, there will be some point where the 4-MBFF overtakes and pays off more than the 8-MBFF. At some higher activity, the 2-MBFF overtakes and pays off more than the 4-MBFF, up to an activity where the power savings stops. The remaining FFs can be grouped into ungated MBFFs, simply to reduce the number of internal clock drivers. We take advantage of this behavior and the optimal grouping by the monotonic activity ordering shown in Section III. The following MBFF grouping algorithm is proposed.

1) Sort the n FFs such that $p_1 \le p_2 \le \cdots \le p_n$.
2) $i \leftarrow 1$.
3) Decide on optimal k by p_i, based on Fig. 3.
4) If $i > n$ or $k < 2$ stop.
5) Group $\mathrm{FF}_i, \mathrm{FF}_{i+1}, \ldots, \mathrm{FF}_{i+k-1}$ in a k-MBFF.
6) $i \leftarrow i + k$.
7) Go to 3.

A few practical comments are in order. The grouping should not cross clock domains. The clock enable signals introduced by the RTL synthesis and manually by designers are untouched. Groupings should also consider logical relations and practical layout concerns. One example is the pipeline registers of a microprocessor, which are natural candidates for MBFF implementation (see Section V). It is expected that the place and route tool will locate bits belonging to the same register close to each other, whereas FF clusters of registers belonging to distinct pipeline stages will be placed away from each other. FFs belonging to different pipeline registers should therefore not be mixed in an MBFF. Similar arguments hold for other system buses and registers such as those storing data, addresses, counters, statuses, and the like. Another example is the FFs of finite-state machines, whose MBFF grouping should not cross control logic borders.

Finally, the aforementioned postplacement MBFF clustering must consider the timing constraints, which are built into their algorithms. By contrast, the MBFF grouping algorithm does not require explicit timing constraints since it works at the RTL design level. In order to bridge the gap between the RTL grouping and the grouping driven by backend timing-closure considerations, we suggested appropriate DDCG design flow. The main idea involves providing "natural" physical layout directives for FF grouping by employing a prior placement. The main steps are described below. More details can be found in [18]:

1) estimation of the FFs toggling probabilities;
2) running the placement to get preliminary preferred locations of FFs in the layout (dry run);

TABLE II
POWER SAVINGS IN THE PIPELINE REGISTER OF A 32-BIT 65-nm MIPS

		IF/ID	ID/EXE	EXE/MEM	MEM/WB	total
data-to-clock activity	sort	0.105	0.0856	0.0711	0.0473	
	matrix mult	0.127	0.118	0.0799	0.0582	
power [μW]		980	1056	952	916	3904
savings [μW]	MBFF only	171.0	240.5	160.7	129.3	701.5
	with DDCG	284.4	344.0	332.0	388.4	1348
	post-layout	199.5	273.0	253.3	232.0	957.8
savings [%]	MBFF only	17.4	22.8	16.9	14.1	18.0
	with DDCG	29.1	32.6	34.8	42.4	34.6
	post-layout	20.4	25.9	26.6	25.3	24.5

TABLE III
POWER SAVINGS OF A 28-nm NETWORK PROCESSOR

unit	FF CLK power [mW]	total CLK power [mW]	total unit power [mW]	FF CLK power save [mW]	FF CLK power save [%]	total CLK power save [%]	total unit power save [%]	area penalty [%]
A	80	1112	1,802	44	57.6	4.09	2.52	1.7
B	304	316	1,638	104	33. 4	32.5	6.22	2.8
C	184	268	760	76	41.9	28.6	10.1	2.7
D	72	172	294	32	45.2	19.2	11.2	2.3
E	162	368	884	88	53.4	23.8	9.90	4.3
F	112	204	252	80	69.7	38.2	31.0	1.3
G	124	368	556	72	57.4	19.7	13.0	1.9
total	1,040	2,804	6,186	496	47.5	17.7	8.00	2.3

3) using the proximity data of FFs' physical locations to constraint probability-driven grouping;
4) adding the DDCG logic to the Verilog HDL code (done automatically by the software tool);
5) ordinary backend flow execution.

V. EXPERIMENTAL RESULTS

The proposed DDCG MBFF design flow was used for two designs: a 32-bit pipelined MIPS processor, implemented in a TSMC 65-nm technology, and an industrial network processor, implemented in a TSMC 28-nm technology. For the MIPS, a workload of sort and matrix multiplication programs was tested, as shown in Table II. The data-to-clock toggling probability for each FF was derived by simulating the workload on the RTL design. For each test, the average data-to-clock switching activity of an FF in the pipelined register is shown under the stage name. Observe the decrease in activity with the progress of the pipeline stage from instruction fetch to write-back.

The MBFF bits of the pipeline registers were grouped by monotonic activity. Table II shows the power savings obtained for the combined benchmark. Each pipeline stage shows the savings for implementation with an ungated MBFF and for DDCG integrated into MBFFs, as proposed here. The results were measured with SpyGlass [15] simulations where the MIPS processor was operated at 1.1 V and 200 MHz. Whereas the ungated MBFF saved 18% of the total power, the integration of DDCG with MBFF yielded almost a double saving of 34.6%. The pipelined registers consumed 65% of the entire MIPS power (memory and IO excluded), so the total power reduction in the entire core was 23%, including the gating overheads.

To examine the advantages of front-end grouping compared with postlayout grouping, we employed ad hoc FF clustering based on their location obtained by the Cadence Virtuoso tool. Both DDCG 2-MBFF and DDCG 4-MBFF were used depending on the FF layout proximity. The results are shown in the postlayout rows of Table II. The front-end RTL grouping outperformed the postlayout, yielding nearly 41% more savings.

The second experiment was a complete industrial network processor designed in the TSMC 28-nm technology. The processor operates in 800 MHz. It is divided into seven units, labeled A–G in Table III. The original design already included extensive clock enabling logic signals and ungated MBFFs, inserted both by the RTL compiler and manually. Each unit contains many clock domains derived by logic conditions. The DDCG MBFF design flow worked on each clock domain separately. The network processor consumed a total of 6.2 W, of which 45% was charged to the clock network, including its underlying FFs. The original design comprised ungated MBFFs, so Table III shows the net power savings obtained solely by the DDCG addition shown in Fig. 1.

The data-to-clock toggling ratios of the FFs were profiled first with the aid of extensive power simulations that are already used by the corporation for power convergence validation. The FFs were then sorted in ascending order by their data-to-clock toggling probability and grouped by the FF activity order, subject to the logical and physical proximity constraints described in Section IV.

Table III shows an additional 8% net power saving on top of the ungated MBFFs in reference. The power measurements included both dynamic and static components and all the gating overheads. The 8% power savings comes on top of the 9% savings achieved using MBFFs in the original design, thus yielding 17% combined savings. Similar to the MIPS, this is about double the power savings compared with ungated MBFFs alone. Such savings are highly appreciated by the industry. The area penalty due to the introduction of CG circuitry was 2.3%.

VI. CONCLUSION

This brief suggests combining MBFFs and probability-driven CG to increase their power savings. A model utilizing the relationship between the optimal MBFF multiplicities to FF data-to-clock toggling probabilities is used in a practical design flow, achieving 17% and 23% power savings, compared with designs with ordinary FFs. About half of these savings can be attributed to the integration of DDCG into MBFFs.

ACKNOWLEDGMENT

The authors would like to thank N. Geuli, T. Shmueli, O. Itizikovich, and Dr. E. Mentovich of the Mellanox Corporation for their support for this work. They would also like to thank the anonymous reviewers for their helpful comments.

REFERENCES

[1] A. Kapoor *et al.*, "Digital systems power management for high performance mixed signal platforms," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 4, pp. 961–975, Apr. 2014.
[2] S. Wimer and I. Koren, "The optimal fan-out of clock network for power minimization by adaptive gating," *IEEE Trans. VLSI Syst.*, vol. 20, no. 10, pp. 1772–1780, Oct. 2012.
[3] C. Santos, R. Reis, G. Godoi, M. Barros, and F. Duarte, "Multi-bit flip-flop usage impact on physical synthesis," in *Proc. 25th IEEE Symp. Integr. Circuits Syst. Design (SBCCI)*, Sep. 2012, pp. 1–6.
[4] J.-T. Yan and Z.-W. Chen, "Construction of constrained multi-bit flip-flops for clock power reduction," in *Proc. IEEE Int. Conf. Green Circuits Syst. (ICGCS)*, 2010, pp. 675–678.
[5] IH-R. Jiang, C-L. Chang, and Y-M. Yang, "INTEGRA: Fast multibit flip-flop clustering for clock power saving," *IEEE Trans. CAD Integr. Circuits Syst.*, vol. 31, no. 2, pp. 192–204, Feb. 2012.
[6] C. L. Chang and I. H. R. Jiang, "Pulsed-latch replacement using concurrent time borrowing and clock gating," *IEEE Trans. Comput.-Aided Design Integr.*, vol. 32, no. 2, pp. 242–246, Feb. 2013.

[7] M. P.-H. Lin, C-C. Hsu, and Y-T. Chang, "Post-placement power optimization with multi-bit flip-flops," *IEEE Trans. CAD Integr. Circuits Syst.*, vol. 30, no. 12, pp. 1870–1882, Dec. 2011.

[8] 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 Trans. VLSI Syst.*, vol. 21, no. 4, pp. 624–635, Apr. 2013.

[9] S. Liu, W.-T. Lo, C.-J. Lee, and H.-M. Chen, "Agglomerative-based flip-flop merging and relocation for signal wirelength and clock tree optimization," *ACM Trans. Design Autom. Electron. Syst.*, vol. 18, no. 3, article no. 40, Jul. 2013.

[10] M. P. H. Lin, C. C. Hsu, and Y. C. Chen, "Clock-tree aware multibit flip-flop generation during placement for power optimization," *IEEE Trans. Comput.-Aided Design Integr.*, vol. 34, no. 2, pp. 280–292, Feb. 2015.

[11] C. Xu, P. Li, G. Luo, Y. Shi, and IH-R. Jiang, "Analytical clustering score with application to post-placement multi-bit flip-flop merging," in *Proc. ACM Int. Symp. Phys. Design*, 2015, pp. 93–100.

[12] S.-H. Wang, Y.-Y. Liang, T.-Y. Kuo, and W.-K. Mak, "Power-driven flip-flop merging and relocation," *IEEE Trans. CAD Integr. Circuits Syst.*, vol. 31, no. 2, pp. 180–191, Feb. 2012.

[13] S-C. Lo, C-C. Hsu, and MP-H. Lin, "Power optimization for clock network with clock gate cloning and flip-flop merging," in *Proc. ACM Int. Symp. Phys. Design*, 2014, pp. 77–84.

[14] S. Wimer, D. Gluzer, and U. Wimer, "Using well-solvable minimum cost exact covering for VLSI clock energy minimization," *Operations Res. Lett.*, vol. 42, no. 5, pp. 332–336, Jul. 2014.

[15] *SpyGlass Power*, accessed on 2016. [Online]. Available: http://www.atrenta.com/solutions/spyglass-family/spyglass-power.htm

[16] 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 *Proc. IEEE Int. Conf. (CAD)*, Nov. 2010, pp. 218–223.

[17] Hsu, Chih-Cheng, Mark Po-Hung Lin, and Yao-Tsung Chang, "Crosstalk-aware multi-bit flip-flop generation for power optimization," *Integr. VLSI J.* vol. 48, pp. 146–157, 2015.

[18] S. Wimer and I. Koren, "Design flow for flip-flop grouping in data-driven clock gating," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 4, pp. 771–778, Apr. 2014.