Design Flow for Flip-Flop Grouping in Data-Driven Clock Gating

Shmuel Wimer, *Member, IEEE*, and Israel Koren, *Fellow, IEEE*

***Abstract*— Clock gating is a predominant technique used for power saving. It is observed that the commonly used synthesis-based gating still leaves a large amount of redundant clock pulses. Data-driven gating aims to disable these. To reduce the hardware overhead involved, flip-flops (FFs) are grouped so that they share a common clock enabling signal. The question of what is the group size maximizing the power savings is answered in a previous paper. Here we answer the question of which FFs should be placed in a group to maximize the power reduction. We propose a practical solution based on the toggling activity correlations of FFs and their physical position proximity constraints in the layout. Our data-driven clock gating is integrated into an Electronic Design Automation (EDA) commercial backend design flow, achieving total power reduction of 15%–20% for various types of large-scale state-of-the-art industrial and academic designs in 40 and 65 manometer process technologies. These savings are achieved on top of the savings obtained by clock gating synthesis performed by commercial EDA tools, and gating manually inserted into the register transfer level design.**

***Index Terms*— Clock gating, clock networks, dynamic power reduction.**

I. Introduction

ONE of the major dynamic power consumers in computing and consumer electronics products is the system's clock signal, typically responsible for 30%–70% of the total dynamic power consumption [1]. Several techniques to reduce the dynamic power are developed, of which clock gating is predominant. Ordinarily, when a logic unit is clocked, its underlying sequential elements receive the clock signal, regardless of whether or not they will toggle in the next cycle. With clock gating, the clock signals are ANDed with explicitly predefined enabling signals. Clock gating is employed at all levels: system architecture, block design, logic design, and gates [2], [3]. Several methods to take advantage of this technique are described in [4]–[6], with all of them relying on various heuristics in an attempt to increase clock gating opportunities.

With the rapid increase in design complexity, computer-aided design tools supporting system-level hardware description have become commonly used. Although substantially increasing design productivity, such tools require the employment of a long chain of automatic synthesis algorithms, from register transfer level (RTL) down to gate level and net list. Unfortunately, such automation leads to a large number of unnecessary clock togglings, thus increasing the number of wasted clock pulses at flip-flops (FFs) as shown in this paper through several industrial examples. Consequently, development of automatic and effective methods to reduce this inefficiency is desirable. In the sequel, we will use the terms toggling, switching, and activity interchangeably.

This paper studies data-driven clock gating, employed for FFs at the gate level, which is the most aggressive possible. The clock signal driving a FF is disabled (gated) when the FFs state is not subject to change in the next clock cycle [7]. Data-driven gating is causing area and power overheads that must be considered. In an attempt to reduce the overhead, it is proposed to group several FFs to be driven by the same clock signal, generated by oring the enabling signals of the individual FFs. This may however, lower the disabling effectiveness. It is therefore beneficial to group FFs whose switching activities are highly correlated and derive a joint enabling signal. In a recent paper, a model for data-driven gating is developed based on the toggling activity of the constituent FFs [9]. The optimal fan-out of a clock gater yielding maximal power savings is derived based on the average toggling statistics of the individual FFs, process technology, and cell library in use. In general, the state transitions of FFs in digital systems depend on the data they process. Assessing the effectiveness of data-driven clock gating requires, therefore, extensive simulations and statistical analysis of the FFs' activity.

Another grouping of FFs for clock switching power reduction, called multibit FF (MBFF), has recently been proposed in [10] and [11]. MBFF attempts to physically merge FFs into a single cell such that the inverters driving the clock pulse into its master and slave latches are shared among all FFs in a group. MBFF grouping is mainly driven by the physical position proximity of individual FFs, while grouping for data-driven clock gating should combine toggling similarity with physical position considerations.

While [9] answered the question of what is the group size that maximizes power savings, this paper studies the questions of: 1) which FFs should be placed in a group to maximize the power reduction and 2) how to algorithmically derive those groups. We also describe a backend design flow implementation.

In the next section, we briefly overview data-driven clock gating, which motivates this paper. Section III presents the

Manuscript received June 19, 2012; revised January 22, 2013; accepted March 14, 2013. Date of publication May 6, 2013; date of current version March 18, 2014. This work was supported in part by the MAGNET Program of Israel Ministry of Industry.

S. Wimer is with the Electrical Engineering Faculty, Technion–Israel Institute of Technology, Haifa 32000, Israel, and also with the Engineering Faculty, Bar-Ilan University, Ramat-Gan 5900, Israel (e-mail: wimer@ee.technion.ac.il).

I. Koren is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003 USA (e-mail: koren@ecs.umass.edu).

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.2013.2253338

problem of optimal FF grouping and its inherent difficulty. Section IV introduces layout considerations into FF grouping and describes a near-optimal grouping algorithm. Section V discusses the implementation of a practical design flow. Section VI presents experimental results obtained for digital signal processor (DSP) and 3-D graphic designs. Final conclusions are presented in Section VII.

II. Data-Driven Clock Gating

Clock enabling signals are very well understood at the system level and thus can effectively be defined and capture the periods where functional blocks and modules do not need to be clocked. Those are later being automatically synthesized into clock enabling signals at the gate level. In many cases, clock enabling signals are manually added for every FF as a part of a design methodology. Still, when modules at a high and gate level are clocked, the state transitions of their underlying FFs depend on the data being processed. It is important to note that the entire dynamic power consumed by a system stems from the periods where modules' clock signals are enabled. Therefore, regardless of how relatively small this period is, assessing the effectiveness of clock gating requires extensive simulations and statistical analysis of FFs toggling activity, as presented subsequently.

Fig. 1 shows the FFs' toggling activity in an arithmetic block comprising 22K FFs, designed in 40-nm technology, taken from Ceva's X1643 DSP core for multimedia and wireless baseband applications [21]. The statistics is obtained from extensive simulations of typical modes of operation, consisting of 240-K clock cycles. The average time window when the FFs clock signal is enabled is only 10%, which is still responsible for the entire dynamic power consumed by that block. The clock enabling signals are obtained by RTL synthesis and manual insertions. As Fig. 1 shows, a FF toggled its state only 2.9% of the clock enabled time period, on the average, thus more than 97% of the clock pulses driving FFs are useless. Such a low toggling rate (of nonclock signals) is very common [12].

Another example of a 40-nm control block comprising 37-K FFs (part of Mellanox ConnectX network processor [23]) has also been examined. There, the clock signal is enabled 20% of the time and within that window the average FF toggling is only 1.3%, and here too more than 98% of the clock pulses driving FFs are useless. It follows from the above examples that no matter what RTL and gate levels clock enabling signals are followed, there are still many opportunities to gate the clock signal at the FF level.

The data-driven gating proposed in [9] is illustrated in Fig. 2. A FF finds out that its clock can be disabled in the next cycle by XORing its output with the present data input that will appear at its output in the next cycle. The outputs of k XOR gates are ORed to generate a joint gating signal for k FFs, which is then latched to avoid glitches. The combination of a latch with AND gate is commonly used by commercial tools and is called integrated clock gate (ICG) [13]. Such data-driven gating is used for a digital filter in an ultralow-power design [24]. A single ICG is amortized over k FFs. There is

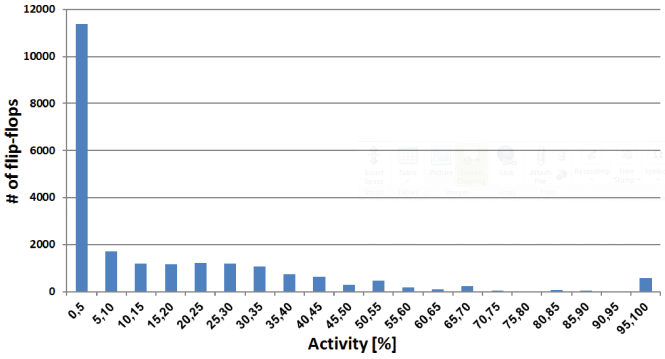


Fig. 1. Toggling statistics of Ceva's X1643 DSP core over 240-K clock cycles.

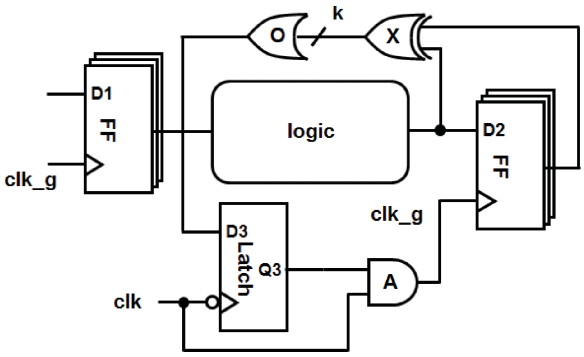


Fig. 2. Practical data-driven clock gating. The latch and gater (AND gate) overheads are amortized over k FFs.

a clear tradeoff between the number of saved (disabled) clock pulses and the hardware overhead. With an increase in k, the hardware overhead decreases but so does the probability of disabling, obtained by ORing the k enable signals.

Let the average toggling probability of a FF (also called activity factor) be denoted by p ($0 < p < 1$). Under the worst-case assumption of independent FF toggling, and assuming a uniform physical clock tree structure, it is shown in [9] that the number k of jointly gated FFs for which the power savings are maximized is the solution of

$$(1-p)^k \ln(1-p)\,(c_{\mathrm{FF}} + c_{\mathrm{W}}) + c_{\mathrm{latch}}/k^2 = 0 \qquad (1)$$

where c_{FF} is the FFs clock input capacitance, c_{W} is the unit-size wire capacitance, and c_{latch} is the latch capacitance including the wire capacitance of its *clk* input. Table I shows how the optimal k depends on p. Such a gating scheme has considerable timing implications, which are discussed in [9]. We will return to those when discussing the implementation of data-driven gating as a part of a complete design flow.

For the scheme proposed in Fig. 2 to be beneficial, the clock enabling signals of the grouped FFs should preferably be highly correlated. Data-driven clock gating is shown to achieve savings of more than 10% of the total dynamic power consumed by the clock tree [15]. Reference [24] reported 20% power savings. It took advantage of the very low dynamic range of the data in a digital filter. The gating logic is tailored to the structure of the filter, whereas the approach discussed in this paper is more general and applies to large scale and a wide range of designs. The experiments described in Section VI

TABLE I
DEPENDENCY OF OPTIMAL FF GROUP SIZE ON TOGGLING PROBABILITY

p	0.01	0.02	0.05	0.1
k	8	6	4	3

show 15%–20% total savings, achieved on top of the savings obtained by synthesis and manual insertions.

III. OPTIMAL FFS GROUPING FOR JOINT CLOCK GATING

Knowing the optimal group size k, the next step is to partition the FFs of a system into k-size sets such that the power savings will be maximized. Grouping FFs for joint clock gating is described in [14] as a part of the physical layout synthesis. Such tools are focusing on skew, power, and area minimization, but they are not aware of the toggling correlations of the underlying FFs, which this paper is focusing on. The optimal value of k is obtained from (1) under toggling independence assumption, but in reality the toggling may be correlated, so in practice one can expect higher saving than the theoretical lower bound obtained under independence assumption. Furthermore, a practical design methodology should preserve the integrity of system clock enabling signals. This means that the FFs of a k-size set must all belong to the same enabled clock (called hereafter pre-enabled). We will first introduce a graph model followed by a formulation of the associated optimization problem. We then show the inherent difficulty of the problem.

Let a pre-enabled clock signal have n FFs and be switching during $m+1$ cycles. The first step toward an optimal FFs grouping is to take advantage of the correlations of their toggling. Let the vector $\mathbf{a}=(a_1,\ldots,a_m)$ denote the activity of a FF, where $a_t=0$, $1\le t\le m$, if the FF stays unchanged (no toggling) from $t-1$ to t, and $a_t=1$, otherwise. The norm $\|\mathbf{a}\|$ is the number of 1 s in $\mathbf{a}$, which is proportional to the power consumed by the FFs switching. All the $n(n-1)/2$ pairs $(\mathbf{a}_i,\mathbf{a}_j)$, $1\le i<j\le n$, are bit-wise xored to yield the number $\|\mathbf{a}_i\oplus\mathbf{a}_j\|$ of redundant clock pulses occurring if FF_i and FF_j are clocked by a common gater. The term $r_{ij}=\|\mathbf{a}_i\oplus\mathbf{a}_j\|/m$ measures the fraction of redundant clock pulses that will occur if FF_i and FF_j are clocked by a common gater. This fraction satisfies $0<r_{ij}<1$. A key consideration in selecting FFs to be driven by a common gater is their activity similarity given by $1-r_{ij}$. The closer to 1 it is, the more desirable it is to jointly drive FF_i and FF_j.

Fig. 3 shows the activity similarities of the FFs in the design referred to in Fig. 1, where only FFs in the same pre-enabled clock signal are paired. Notice that different pre-enabled clock signals may have a different duration m of enabled window. As expected, the activity similarity is very high, mostly due to the very low FFs' toggling during their enabled clock window. Still, highly active and uncorrelated FF pairs exist and are encircled. A good FFs grouping algorithm should avoid putting the FFs of an encircled pair into the same group.

To model the switching power consumed when driving FFs pairs with a common gater ($k=2$), an n-vertex complete weighted graph $G(V,E,w)$, called FF pairwise activity graph, is defined. Assume, without loss of generality, that n is even as otherwise we could artificially add a never toggling FF and set

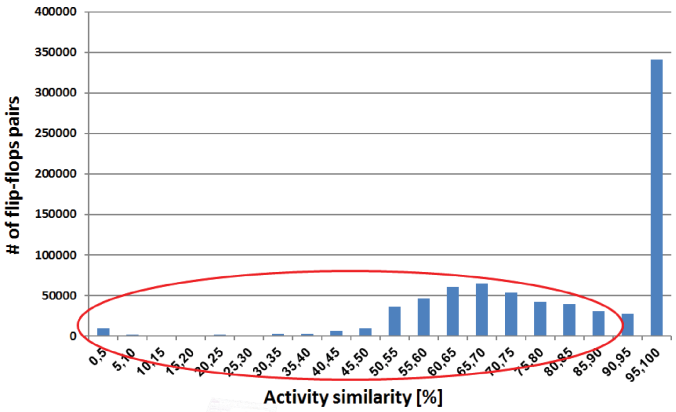


Fig. 3. Activity similarity of FFs in a DSP core (see Fig. 1). FFs are paired only if they share same pre-enabled clock signal. Undesirable pairs are encircled.

the weight of its entire incident edges to zero. A vertex $v_i\in V$ is associated with FF_is activity $\mathbf{a}_i$. An edge $e_{ij}=(v_i,v_j)\in E$ is associated with a joint activity vector $\mathbf{a}_i|\mathbf{a}_j$, where the or is a bit-wise operation. An edge e_{ij} is assigned a weight $w(e_{ij})=\|\mathbf{a}_i\oplus\mathbf{a}_j\|$, which counts the number of redundant clock pulses incurred by clocking FF_i and FF_j with a common gater. Let $E'\subset E$, $|E'|=n/2$, be a vertex matching of $G(V,E,w)$. The total power P consumed by the clock signal depends on the number of clock pulses driving the FFs and is given by

$$P=2\sum_{e_{ij}\in E'}\|\mathbf{a}_i|\mathbf{a}_j\|=\sum_{v_i\in V}\|\mathbf{a}_i\|+\sum_{e_{ij}\in E'}\|\mathbf{a}_i\oplus\mathbf{a}_j\|. \quad (2)$$

The first sum in the right-hand side of (2) is the contribution due to the toggling of the individual FFs and is independent of the pairing. Therefore, to consume minimum dynamic power (or alternatively, achieve maximum dynamic power savings), it is necessary to minimize $\sum_{e_{ij}\in E'}\|\mathbf{a}_i\oplus\mathbf{a}_j\|=\sum_{e_{ij}\in E'}w(e_{ij})$, which turns into the well-known minimal cost perfect graph matching (MCPM) problem, for which polynomial complexity algorithms are known [16].

The extension for $k>2$ is straightforward. Assume without loss of generality that n is divisible by k, as otherwise we could artificially add a few never toggling FFs and set to zero the weight of the edges incident to their corresponding vertices. A complete k-uniform weighted hypergraph $H(V,E,w)$, called FF grouping activity hypergraph, is defined, where for a subset $\boldsymbol{v}\subset V$ and $|\boldsymbol{v}|=k$, $e_{\boldsymbol{v}}=\{v_u\}_{u\in\boldsymbol{v}}\in E$ defines a hyper edge. It follows that

$$|E|=\binom{n}{k}.$$

A hyper edge $e_{\boldsymbol{v}}$ is associated with a joint activity vector $\bigcup_{u\in\boldsymbol{v}}\mathbf{a}_u$, defined by the bit-wise oring of the k toggling vectors. A hyper edge $e_{\boldsymbol{v}}$ is assigned a weight

$$w(e_{\boldsymbol{v}})=\sum_{v\in\boldsymbol{v}}\left\|\mathbf{a}_v\oplus\bigcup_{u\in\boldsymbol{v}}\mathbf{a}_u\right\| \quad (3)$$

which is the total number of redundant clock pulses incurred by clocking the k FFs corresponding to $e_{\boldsymbol{v}}$ with a common gater.

Let $E'\subset E$ be an exact cover of the vertices of $H(V,E,w)$ by n/k hyper edges (a vertex belongs to one and only one

hyper edge). The total power P consumed by the clock signal linearly depends on the total number of pulses driving the FFs and is given by

$$P = \sum_{e_{\boldsymbol{v}} \in E'} k \left\| \bigcup_{u \in \boldsymbol{v}} \mathbf{a}_u \right\| = \sum_{v_i \in V} \|\mathbf{a}_i\| + \sum_{e_{\boldsymbol{v}} \in E'} \sum_{v \in \boldsymbol{v}} \left\| \mathbf{a}_v \oplus \bigcup_{u \in \boldsymbol{v}} \mathbf{a}_u \right\|. \tag{4}$$

The first sum in the right-hand side of (4) is the contribution due to the toggling of the individual FFs and is independent of the grouping. Therefore, to consume minimum dynamic power (or alternatively, achieve maximum dynamic power savings), it is necessary to minimize the term $\sum_{e_{\boldsymbol{v}} \in E'} \sum_{v \in \boldsymbol{v}} \left\| \mathbf{a}_v \oplus \bigcup_{u \in \boldsymbol{v}} \mathbf{a}_u \right\| = \sum_{e_{\boldsymbol{v}} \in E'} w(e_{\boldsymbol{v}})$. We refer to it in this paper as the MIN_CLK_GATE problem.

In the above representation, a solution to the problem of finding n/k hyper edges exactly covering the n vertices and yielding minimum redundant clock pulses can be derived from a solution to the well-known NP-hard weighted set partitioning problem (SPP) [17], where hyper edges are the variables covering the vertex constraints. Although the reduction to SPP does not imply that MIN_CLK_GATE by itself is a difficult problem, it may provide a hint. Indeed, it can be shown that at MIN_CLK_GATE is also NP-hard, but the proof is beyond the scope of this paper.

Several papers have addressed the grouping problem based on FFs pairing ($k = 2$) and applied the solution iteratively to larger sets where $k = 2^K$. The authors of [18] propose heuristics where FFs are sorted by their activity and then paired in that order. It is possible, however, to construct a counterexample where this heuristic would increase the number of redundant clock pulses rather than reduce them. In [6] and [19], FFs are paired based on intuitive arguments without formal proof, and it may sometimes yield inferior gating.

A bottom-up process for a coarse, block-level gating is proposed in [5] by repeating the MCPM algorithm. We have adapted this idea to FF-level gating. Starting with n individual FFs and constructing the associated n-vertex FF pairwise activity graph, an MCPM algorithm then finds the best FFs pairing. A new $n/2$-vertex pairwise activity graph is then defined where its vertices correspond to the matching ($n/2$ edges) found in the former step. The process repeats K times until groups of size $k = 2^K$ are determined. For $k = 2$ ($K = 1$), MCPM indeed solves the problem of minimizing the number of redundant clock pulses, but its repetitive application for $k > 2$ ($K > 1$) may not find the minimum, as otherwise this would contradict the NP-hardness. Still, the iterative MCPM algorithm is practical and has acceptable run time. Experiments showed that it is about 5% off SPPs optimal solution. This is observed by solving accurately a variety of relatively small SPP problems with CPLEX solver [20].

The 3-D graphics accelerator in [9] containing 4.9×10^3 FFs is used to demonstrate the advantage of the correlation-based algorithm over a random grouping. A test bench of 10^5 clock cycles with an average FF toggling of $p = 0.05$ is simulated and the toggling vectors of the FFs are recorded. The expression $\sum_{e_{\boldsymbol{v}} \in E'} \sum_{v \in \boldsymbol{v}} \left\| \mathbf{a}_v \oplus \bigcup_{u \in \boldsymbol{v}} \mathbf{a}_u \right\|$ for the total number of redundant clock pulses is calculated, and the results are shown in Table II. The advantage of the correlation-based grouping over a random grouping is considerable and is rapidly increasing with k up to a certain maximum. The increase is explained by the quadratic growth of $k(k-1)/2$ in the redundancy expression, governing the count of redundant pulses in random grouping. For large random groups, the clock is enabled most of the time and the redundancy saturates. Notice that when the toggling probability p is high, random grouping with large k may nullify the dynamic power savings.

TABLE II
COMPARISON OF THE NUMBER OF REDUNDANT CLOCK PULSES BETWEEN CORRELATION-BASED AND RANDOM-BASED FF GROUPINGS

group size k	2	4	8	16	32	64	128
correlation-based ($\times 10^6$)	1.79	3.28	4.26	5.39	6.91	8.95	11.6
random-based ($\times 10^6$)	3.11	10.1	19.8	41.3	50.7	54.7	56.2
random-based/correlation-based	1.73	3.08	4.65	7.66	7.34	6.11	4.84

IV. PHYSICAL LAYOUT CONSIDERATIONS IN FFS GROUPING

Finding sets of FFs that minimize the number of redundant clock pulses is not enough to maximize power savings. Grouping must account for the on-die locations of FFs and gaters, which affect the power consumption due to the capacitive loads resulting from their connections. The physical locations of FFs affect also the delay and clock skew, and it is therefore desirable for FFs driven jointly by the same clock gater, to be placed in proximity of each other.

A scheme for constructing clock trees when the positions of the FFs in leaves are known is described in [7]. A cost function weighting the sum of clock activities and clock pin distances is minimized. Such a cost function is problematic since the physical meaning of a weighted sum of activities and distances is not well defined and requires delicate tuning of the weights. An alternative of summing the products of activity by the diameter of smallest circles enclosing FF sets looks more appropriate and is used in this paper. This sum of products has the physical units of effective capacitance, thus explicitly measuring power consumption, and no weights are needed.

To support activity-diameter products, the FF grouping activity hypergraph $H(V, E, w)$ defined before is modified in order to account for the FFs' layout proximity. It is assumed that some knowledge of the preferred FF locations in the layout is available. This can, for instance, be obtained by running first a placement of the nominal design without the data-driven clock gating circuits. It is supposed to place FFs close to the logic where they are being used and also place closely FFs belonging to the same pre-enabled clock signal (existing in system prior to data-driven gating insertion). Based on this data, the weight $w(e_{\boldsymbol{v}})$ of a hyper edge in (3), which considered only the number of redundant clock pulses, is modified as follows:

$$w(e_{\boldsymbol{v}}) = d(\boldsymbol{v}) \sum_{v \in \boldsymbol{v}} \left\| \mathbf{a}_v \oplus \bigcup_{u \in \boldsymbol{v}} \mathbf{a}_u \right\| \tag{5}$$

where $d(\boldsymbol{v})$ is the diameter of the smallest circle enclosing the $\boldsymbol{v}$s FFs. Substituting (5) in (4), the problem of maximizing

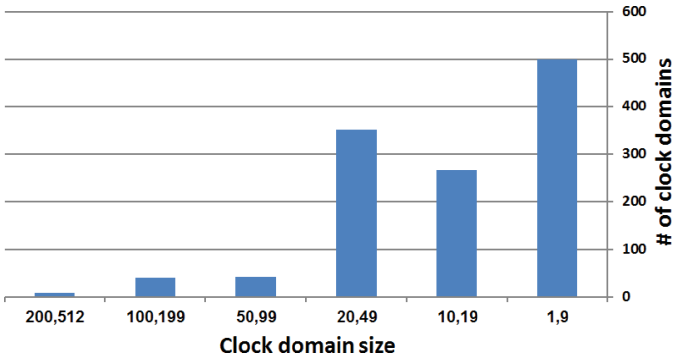


Fig. 4. Distribution of the number of FFs sharing a common pre-enabled clock signal of a DSP core (see Fig. 1).

the power savings turns into finding a subset $E' \subset E$ of n/k hyper edges exactly covering the vertices of $H(V, E, w)$ such that the expression

$$\sum_{e_v \in E'} w(e_v) = \sum_{e_v \in E'} d(\boldsymbol{v}) \sum_{v \in \boldsymbol{v}} \left\| \mathbf{a}_v \oplus \bigcup_{u \in \boldsymbol{v}} \mathbf{a}_u \right\| \tag{6}$$

is minimized. As mentioned before, any algorithm for solving SPP (e.g., [20]) is adequate to accurately solve the MIN_CLK_GATE problem, hence yielding maximum savings. Although SPP is NP-hard, and hence its corresponding algorithms may have limited capability, the number n of FFs (vertices of H) sharing the same pre-enabled clock is limited in practice. Our experience shows that the introduction of FFs proximity constraints increases by up to 10% the amount $\sum_{e_v \in E'} \sum_{v \in \boldsymbol{v}} \left\| \mathbf{a}_v \oplus \bigcup_{u \in \boldsymbol{v}} \mathbf{a}_u \right\|$ of redundant clock pulses in (4). This, however, pays off afterward by avoiding excessive interconnections.

Fig. 4 illustrates the distribution of the number of FFs sharing a common pre-enabled clock signal of the DSP block of Fig. 1. As shown in Table I, the typical size of k falls between 3 and 8, so solving SPP with

$$\binom{n}{k}$$

variables (hyper edges of H, FF sets) and n constraints (vertices of H, FFs) is feasible. Moreover, imposing a constraint $d(\boldsymbol{v}) \leq D$ on the diameter of the smallest circle enclosing the FFs (vertices) in a FF set (hyper edge), where D bounds the allowable diameter, further contracts $H(V, E, w)$. The resulting SPPs can then be optimally solved for each pre-enabled clock signal by the CPLEX solver [20], rather than getting suboptimal solution with iterative MCPM algorithm. We have observed about 10% degradation in power savings resulting by restricting FFs grouping to common pre-enabled clock signals and physical proximity. For large blocks comprising tens to hundred thousands of FFs, those restriction are unavoidable as otherwise the grouping problem is practically unsolvable.

V. Implementation and Integration in a Design Flow

In the following, we describe the implementation of data-driven clock gating as a part of a standard backend design flow. It consists of the following steps.

1) Estimating the FFs toggling probabilities involves running an extensive test bench representing typical operation modes of the system to determine the size k of a gated FF group by solving (1).
2) Running the placement tool in hand to get preliminary preferred locations of FFs in the layout.
3) Employing a FFs grouping tool to implement the model and algorithms presented in Sections III and IV, using the toggling correlation data obtained in Step 1 and FF locations' data obtained in Step 2. The outcome of this step is k-size FF sets (with manual overrides if required), where the FFs in each set will be jointly clocked by a common gater.
4) Introducing the data-driven clock gating logic into the hardware description (we use Verilog HDL). This is done automatically by a software tool, adding appropriate Verilog code to implement the logic described in Fig. 2. The FFs are connected according to the grouping obtained in Step 3. A delicate practical question is whether to introduce the gating logic into RTL or gate-level description. This depends on design methodology in use and its discussion is beyond the scope of this paper. We have introduced the gating logic into the RTL description.
5) Re-running the test bench of Step 1 to verify the full identity of FFs' outputs before and after the introduction of gating logic. Although data-driven gating, by its very definition, should not change the logic of signals, and hence FFs toggling should stay identical, a robust design flow must implement this step.
6) Ordinary backend flow completion. From this point, the backend design flow proceeds by applying ordinary place and route tools. This is followed by running clock-tree synthesis.

Few timing-related comments are in order. The extra gating delay introduced by the feedback loop in Fig. 2 should not exceed the delay margins of paths from the clock input *clk_g* of FF_1 to the data input D_2 of FF_2. In ordinary designs, notably in automatically synthesized blocks, most of the delay margins are large enough to absorb the introduction of the gating logic. If at a later stage timing violations due to the gating are found, one can simply drop the data-driven gating from the troublesome FFs. We found very few of those in our designs, less than 5% of the FFs. Relaxation of the clock cycle can also overcome this problem, but it must be considered in a wider context of power-delay tradeoff and product specifications, which is beyond the scope of this paper.

VI. Experimental Results

The design flow described in Section V is experimented on a DSP core comprising 22 k FFs [21] (Figs. 1, 3, and 4), another large vectored DSP core comprising 100 k FFs [22], a 3-D graphics accelerator [9], and a network processor control block [23]. The DSP cores include both arithmetic and control circuits. Table I shows the theoretical optimal group size for various toggling probabilities. For FF toggling probabilities from $p = 0.01$ to $p = 0.05$, the group size maximizing the

TABLE III
TOTAL POWER REDUCTION FOR A DSP CORE [21]

Group size	none	k=2	k=4	k=8	k=16	k=32	k=64	k=128
Combinational power (mW)	31.6	33.5	33.1	33.2	33.5	33.4	33.3	33.4
Sequential power (mW)	8.87	6.91	5.63	5.04	5.66	5.91	6.13	6.38
Clock power (mW)	18.2	20.4	14.2	11.5	12.4	12.5	12.7	13.0
Leakage power (mW)	0.5	0.59	0.57	0.55	0.53	0.52	0.51	0.51
Total power (mW)	59.0	61.4	53.4	50.2	52.0	52.3	52.6	53.3
Net savings (%)		**-4.1**	**9.5**	**15.0**	**11.9**	**11.4**	**10.9**	**9.7**

TABLE IV
TOTAL POWER REDUCTION FOR A DSP CORE COMPRISING 100 k FFS [22]

Group size	none	k=4	k=8	k=16	k=32	k=64	k=128
Combinational power (mW)	28.7	32.9	33.0	33.1	33.1	33.1	33.1
Sequential power (mW)	27.1	20.1	19.7	19.6	20.3	20.4	20.4
Clock power (mW)	28.9	24.2	20.7	19.0	20.7	20.5	20.5
Leakage power (mW)	0.58	0.67	0.65	0.64	0.66	0.66	0.66
Total power (mW)	84.8	77.3	73.5	71.8	74.1	74.0	74.0
Net saving (%)		**8.8**	**13.3**	**15.3**	**12.6**	**12.7**	**12.7**

TABLE V
TOTAL POWER REDUCTION FOR A 3-D GRAPHICS ACCELERATOR [9], [25]

Group size	none	k=2	k=4	k=8	k=16	k=32	k=64	k=128
Combinational power (mW)	3.14	3.37	3.37	3.36	3.36	3.36	3.36	3.36
Sequential power (mW)	1.75	0.87	0.88	0.92	1.02	1.15	1.32	1.50
Clock power (mW)	1.92	2.23	1.48	1.16	1.19	1.20	1.34	1.45
Leakage power (mW)	0.1	0.12	0.11	0.11	0.1	0.1	0.1	0.1
Total power (mW)	6.91	6.59	5.84	5.54	5.67	5.81	6.12	6.41
Net savings (%)		**4.63**	**15.5**	**19.8**	**18.0**	**15.9**	**11.5**	**7.24**

net power savings is between $k = 8$ and $k = 4$. We measured the power savings compared with the nominal designs using the SpyGlass EDA power simulator [8]. The measurements accounted for the logic and latch overhead required by the gating (see Fig. 2), thus they reflect the net power savings. Tables III–V show the resulting power for a wide range of group sizes, where the maximum power savings are achieved for $k = 8$ in Tables III and V and $k = 16$ in Table IV. The latter is due to the very low toggling rate of that processor.

As shown in the tables, the net dynamic power savings are 15% for the DSP cores and 20% for the 3-D graphics accelerator. Experiments for the network processor control block also yield nearly 20% power savings where the optimal group size is $k = 8$. Notice that the above results include not only the clock network and sequential power but also the power consumed in the combinational logic, which is about half of the total dynamic power. Hence, considering clocking power savings alone, 30%–40% is achieved.

Since the toggling probability is averaged across the entire FFs, it may happen that different sub-blocks will have different probabilities. It is therefore possible to further reduce the power by using various k values in different sub-blocks. By mixing $k = 8$ with $k = 16$, the power savings in Table IV reached 16.3%.

Another interesting observation is the slight growth in the combinational logic power, due to the extra XOR connected at every FF and the other logic involved in data-driven gating (Fig. 2). The growth is almost independent of k. Since SpyGlass is charging the latch overhead to the sequential power, this can be explained by the fact that most of the increase is due to the XOR gates, which is independent of k.

TABLE VI
COMPARISON OF GATING METHODS. THE TOTAL POWER COMPARED WITH THE NATIVE DESIGN WITHOUT ANY GATING IN SHOWN. IN ALL THREE DESIGNS, APPLYING ONLY THE DATA-DRIVEN GATING YIELDS THE BEST RESULT

		Synthesis-based		
		no	yes	
Data driven	no	(*i*) **8.2**mw (100%)	(*ii*) **6.91**mW (84.3%)	3D graphics accelerator ([9], [25])
	yes	(*iii*) **5.31**mW (64.8%)	(*iv*) **5.54**mW (67.6%)	
Data driven	no	(*i*) **173**mw (100%)	(*ii*) **59.0**mW (34.1%)	DSP core ([21])
	yes	(*iii*) **47.3**mW (27.3%)	(*iv*) **50.2**mW (29.0%)	
Data driven	no	(*i*) **144**mw (100%)	(*ii*) **84.8**mW (58.9%)	DSP core ([22])
	yes	(*iii*) **60.5**mW (42.0%)	(*iv*) **71.8**mW (49.9%)	

It is interesting to compare the relative power savings achieved by applying synthesis-based gating only or data-driven gating only or both. To this end, we ran RTL power simulation of the processors in Tables III–V in four gating modes: 1) no gating; 2) synthesis-based available from an EDA vendor [26]; 3) data-driven; and 4) both combined. Table VI summarizes the total power consumption of each case and compares it with the native design without any gating [shown in (1)]. The synthesis-based gating alone 2) reduced the total power to 84%, 34%, and 59%, respectively. Applying data-driven gating on top of the synthesis-based one 4) further reduced the power to 68%, 29%, and 50%, respectively.

However, for all three designs, the application of data-driven gating alone yielded higher power savings than the combination of data-driven and synthesis-based, reduction the power to 65%, 27%, and 42%, respectively. This is due to the fact that data-driven gating stops any unnecessary clock pulse, and the inclusion of the synthesis-based gating only adds logic circuits that becomes redundant once the data-based gating is applied. Although synthesis-based clock gating is a well-established design methodology, the above experiments encourage its replacement by data-driven gating.

To further characterize the benefits of the two gating modes, we analyzed several different circuit types, for example, control units, arithmetic units, and register files. We selected such units from the large designs that we analyzed above [21], [22], [25], where the type of the circuit is specified in Table VII. The resulting power consumptions (in percentages), shown in Table VII, are relative to the no clock gating case, and the lowest power is indicated in red (circled). It can be seen that for control circuits, data-driven gating 3) is outperforming synthesis-based gating 2). This is explained by their very low toggling rate, where data-driven is most useful. Similar

TABLE VII
IMPACT OF THE VARIOUS GATING MODES FOR DIFFERENT CIRCUIT TYPES ON THE TOTAL REMAINING POWER

Circuit Type	Gating Modes		
	Synthesis (*ii*)	Data-driven (*iii*)	Combined (*iv*)
Control I (3D graphics pipeline)	81%	65%	71%
Control II (DSP video processing)	83%	56%	64%
Adder I (24 bit Carry-select)	92%	59%	61%
Adder II (32 bit Carry-skip)	89%	58%	62%
Multiplier (24 bit Booth Multiplier)	91%	71%	74%
Reg-file I (3D 32 Regs of 24-bit)	42%	57%	61%
Reg-file II (DSP 64 Regs of 32-bit)	47%	60%	69%

TABLE VIII
AMOUNT OF REDUNDANT CLOCK PULSES WITHOUT AND WITH RADIUS LIMIT FOR THE 3-D GRAPHICS ACCELERATOR OF [9], [25]

	group size k	2	4	8	16	32
redundant clock pulses (X 10^6)	no limit	1.786	3.277	4.258	5.391	6.912
	radius limit	1.915	3.463	4.643	5.927	7.583
	increase	7%	5.6%	9%	9.9%	9.7%

behavior is observed for arithmetic circuits (their IO registers are included). Expectedly, synthesis-based gating is still favored for register files. This follows since only one register is changing its data at a time, a condition that can easily be caught and defined in the RTL code. Therefore, applying data-driven gating on top of synthesis-based mostly adds circuit overhead. The results of the combined synthesis-based and data-based gating scheme are worse than the data-driven only gating for all seven circuits. Thus, unless register files can undergo only synthesis-based gating and data-based gating will not be applied to them, synthesis-based gating should be completely replaced by data-based gating.

As mentioned earlier, the gating scheme may have considerable timing implications. Potential timing violations are avoided using Step 3 of the flow proposed in Section IV, which groups FFs based on the preliminary preferred locations of FFs in the layout found by Step 2. Such restriction increases the amount of redundant clock pulses in (4), but this is compensated by shortening the wires connecting the clock gaters to their corresponding FFs. Table VIII compares grouping with and without FF location constraints, for a design comprising 4.9 k FFs and a test of 10^5 clock cycles, showing a 5%–10% increase in the redundant pulses. This has only a little impact on the saving measured in simulation (less than 1%).

To quantify the timing impact of data-driven clock gating, we ran static timing analysis on the native 3-D graphics design without gating and then compared it with the design with the gating. Fig. 5 illustrates the margin (slack) distribution for 200-MHz clock cycle. It can be seen that the margin distribution has slightly worsened as more paths now have a negative slack. The violations need to be taken care of for timing closure and a variety of actions and techniques are possible, but their discussion is beyond the scope of this paper.

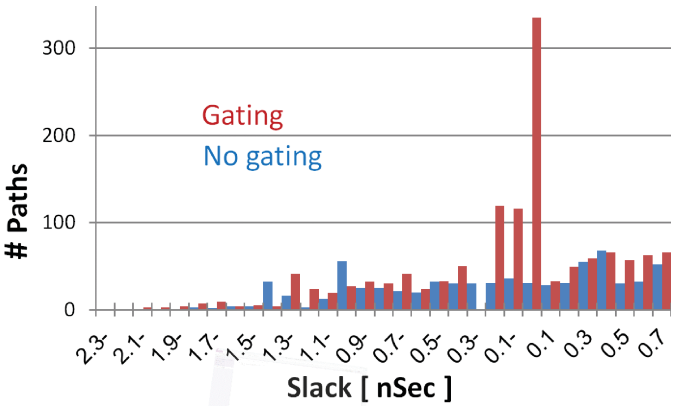


Fig. 5. Negative slack distribution in the 3-D graphics accelerator [9], [25], for 200-MHz clock cycle. The introduction of clock gating resulted in a few additional delay paths with negative slacks.

VII. CONCLUSION

This paper studied the problem of grouping FFs for joint clocking by a common gater to yield maximal dynamic power savings. Although the problem was NP-hard, we discussed several practical algorithms to solve it and found several of them to be useful in a real design automation implementation. The solution was integrated in a practical design flow. Experimental results of DSP cores, a network processor control block, and a 3-D graphics accelerator were presented, achieving 15%–20% total power reduction.

The FF grouping problem also arised in MBFF [11], where distinct FFs were combined in one physical cell to share their internal clock drivers. It is interesting to consider the combination of data-driven gating with MBFF in an attempt to yield further power savings.

ACKNOWLEDGMENT

The authors would like to thank the Ceva and Mellanox Corporations for availing their design data, R. Mioni for implementing the design flow, and the Technion VLSI Lab for the development of the FF grouping software. They are also grateful for the useful comments made by the anonymous reviewers.

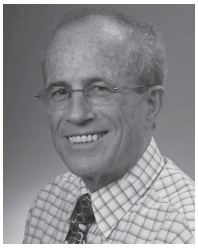
REFERENCES

[1] V. G. Oklobdzija, *Digital System Clocking—High-Performance and Low-Power Aspects*. New York, NY, USA: Wiley, 2003.
[2] L. Benini, A. Bogliolo, and G. De Micheli, "A survey on design techniques for system-level dynamic power management," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 8, no. 3, pp. 299–316, Jun. 2000.
[3] M. S. Hosny and W. Yuejian, "Low power clocking strategies in deep submicron technologies," in *Proc. IEEE Intll. Conf. Integr. Circuit Design Technol.*, Jun. 2008, pp. 143–146.
[4] C. Chunhong, K. Changjun, and S. Majid, "Activity-sensitive clock tree construction for low power," in *Proc. Int. Symp. Low Power Electron. Design*, 2002, pp. 279–282.
[5] A. Farrahi, C. Chen, A. Srivastava, G. Tellez, and M. Sarrafzadeh, "Activity-driven clock design," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 6, pp. 705–714, Jun. 2001.
[6] W. Shen, Y. Cai, X. Hong, and J. Hu, "Activity and register placement aware gated clock network design," in *Proc. Int. Symp. Phys. Design*, 2008, pp. 182–189.

[7] M. Donno, E. Macii, and L. Mazzoni, "Power-aware clock tree planning," in *Proc. Int. Symp. Phys. Design*, 2004, pp. 138–147.
[8] *SpyGlass Power* [Online]. Available: http://www.atrenta.com/solutions/spyglass-family/spyglass-power.htm
[9] S. Wimer and I. Koren, "The Optimal fan-out of clock network for power minimization by adaptive gating," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 10, pp. 1772–1780, Oct. 2012.
[10] 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/ACM Int. Conf. Comput., Aided Design*, Nov. 2010, pp. 218–223.
[11] I. H.-R. Jiang, C.-L. Chang, Y.-M. Yang, E. Y.-W. Tsai, and L. S.-F. Cheng, "INTEGRA: Fast multi-bit flip-flop clustering for clock power saving based on interval graphs," in *Proc. Int. Symp. Phys. Design*, 2011, pp. 115–121.
[12] N. Magen, A. Kolodny, U. Weiser, and N. Shamir, "Interconnect-power dissipation in a microprocessor," in *Proc. Int. Workshop Syst. Level Int. Predict.*, 2004, pp. 7–13.
[13] M. Muller, S. Simon, H. Gryska, A. Wortmann, and S. Buch, "Low power synthesizable register files for processor and IP cores," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst., Low-Power Design Tech.*, vol. 39, no. 2, pp. 131–155, Mar. 2006.
[14] *Low Skew—Low Power CTS Methodology in SOC Encounter for ARM Processor Cores*. (2009) [Online]. Available: http://www.cadence.com/cdnlive/library/Documents/2009/EMEA/DI10_Dave Kinjal_ARM_FINAL.pdf
[15] W. Aloisi and R. Mita, "Gated-clock design of linear-feedback shift registers," *IEEE Trans. Circuits Syst., II, Brief Papers*, vol. 55, no. 5, pp. 546–550, Jun. 2008.
[16] V. Kolmogorov, "Blossom V: A new implementation of a minimum cost perfect matching algorithm," *Math. Programm. Comput.* vol. 1, no. 1, pp. 43–67, Jan. 2009.
[17] E. Balas and M. W. Padberg, "Set partitioning: A survey," *SIAM Rev.*, vol. 18, no. 4, pp. 710–760, Oct. 1976.
[18] C. Chunhong, K. Changjun, and M. Sarrafzadeh, "Activity-sensitive clock tree construction for low power," in *Proc. Int. Symp. Low Power Electron. Design*, 2002, pp. 279–282.
[19] W. Shen, Y. Cai, X. Hong, and J. Hu, "Activity-aware registers placement for low power gated clock tree construction," in *Proc. IEEE Comput. Soc. Ann. Symp. VLSI*, Mar. 2007, pp. 383–388.
[20] M. Lewis, G. Kochenberger, and B. Alidaee, "A new modeling and solution approach for the set-partitioning problem," *Comput. Operat. Res.*, vol. 35, no. 3, pp. 807–813, Mar. 2008.
[21] *CEVA-X1643* [Online]. Available: http://www.ceva-dsp.com/CEVA-X1643.html
[22] *CEVA-XC4000* [Online]. Available: http://www.ceva-dsp.com/CEVA-XC4000-Low-Power-DSP-Architecture-Framework-for-the-LTE-Advanced-Wi-Fi-802.11ac-DVB-T2-and-more.html
[23] [Online]. Available: http://www.mellanox.com/content/pages.php?pg=products_dyn&product_family=3&menu_section=32
[24] A. Bonanno, A. Bocca, A. Macii, E. Macii, and M. Poncino, "Data-driven clock gating for digital filters," in *Proc. 19th Int. Workshop*, 2010, pp. 96–105.
[25] *AGE: A 3D graphics acceleration engine*. (2012) [Online]. Available: http://engineering.biu.ac.il/en/node/2571#Device1
[26] *Synopsys Design Compiler*, IEEE Standard E-2010.12-SP2, 2010.

Shmuel Wimer (M'10) received the B.Sc. and M.Sc. degrees in mathematics from Tel-Aviv University, Tel-Aviv, Israel, and the D.Sc. degree in electrical engineering from the Technion-Israel Institute of Technology, Haifa, Israel, in 1978, 1981, and 1988, respectively.

He worked for 32 years at industry in R&D, engineering and managerial positions, for Intel from 1999 to 2009, Sagantec from 1997 to 1999, micro-CAD from 1994 to 1997, IBM from 1985 to 1994, National Semiconductor from 1981 to 1985, and Israeli Aircraft Industry from 1978 to 1981. He is currently an Associate Professor with the Engineering Faculty, Bar-Ilan University, Ramat-Gan, Israel, and an Associate Visiting Professor with the Electrical Engineering Faculty, Technion. His current interests include VLSI circuits and systems design optimization and combinatorial optimization.



Israel Koren (M'76–SM'87–F'91) is currently a Professor of electrical and computer engineering with the University of Massachusetts, Amherst, MI, USA. He has been a Consultant to numerous companies including IBM, Analog Devices, Intel, AMD and National Semiconductors. He is the author of the textbook "Computer Arithmetic Algorithms," 2nd Edition, A. K. Peters, 2002, and a co-author of "Fault Tolerant Systems," Morgan–Kaufman, 2007. He has authored extensively and has over 250 publications in refereed journals and conferences. His current research interests include Fault-Tolerant systems, computer architecture, VLSI yield and reliability, secure cryptographic systems, and computer arithmetic.

He is an Associate Editor of the VLSI Design Journal, and Sustainable-Computing: Informatics and Systems. He served as General Chair, Program Chair and Program Committee Member for numerous conferences.