The Optimal Fan-Out of Clock Network for Power Minimization by Adaptive Gating

Shmuel Wimer and Israel Koren, *Fellow, IEEE*

***Abstract*—Gating of the clock signal in VLSI chips is nowadays a mainstream design methodology for reducing switching power consumption. In this paper we develop a probabilistic model of the clock gating network that allows us to quantify the expected power savings and the implied overhead. Expressions for the power savings in a gated clock tree are presented and the optimal gater fan-out is derived, based on flip-flops toggling probabilities and process technology parameters. The resulting clock gating methodology achieves 10% savings of the total clock tree switching power. The timing implications of the proposed gating scheme are discussed. The grouping of FFs for a joint clocked gating is also discussed. The analysis and the results match the experimental data obtained for a 3-D graphics processor and a 16-bit microcontroller, both designed at 65-nanometer technology.**

***Index Terms*—Clock gating, clock networks, clock tree, dynamic power minimization, optimal fan-out.**

I. Motivation

THE increasing demand for low power mobile computing and consumer electronics products has refocused VLSI design in the last two decades on lowering power and increasing energy efficiency. Power reduction is treated at all design levels of VLSI chips, from architecture through block and logic levels, down to gate-level, circuit and physical implementation. One of the major dynamic power consumers is the system's clock signal, typically responsible for up to 50% of the total dynamic power consumption [1]. Clock network design is a delicate procedure, and is therefore done in a very conservative manner under worst case assumptions. It incorporates many diverse aspects such as selection of sequential elements, controlling the clock skew, and decisions on the topology and physical implementation of the clock distribution network [2].

A. Clock Gating

Several techniques to reduce the dynamic power have been 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. (We will use the terms toggling, switching and activity to mean the same). With clock gating, the clock signals are ANDed with explicitly defined enabling signals. Clock gating is employed at all levels: system architecture, block design, logic design, and gates [3]. In [4] the impact of on-chip variations (OCV) on timing due to clock gating was discussed. Clock enabling signals are usually introduced by designers during the system and block design phases, where the interdependencies of the various functions are well understood. In contrast, it is very difficult to define such signals at the gate level, especially in control logic, since the interdependencies among the states of various flip-flops (FFs) depend on automatically synthesized logic. We claim that a big gap exists between clock disabling that is derived from the HDL definitions and what can be achieved through detailed knowledge regarding the FFs' activities and how they are correlated with each other. The above gap has been studied in [23] for a programmable interrupt controller (PIC), where HDL-based gating has reduced the clock power by 25% while manual insertion of gating logic to every FF increased the power savings by twofold to 50%. This paper proposes a systematic way to bridge this gap and construct a clock network that takes full advantage of FFs' activity statistics.

We present an approach to maximize clock disabling at the gate level, where the clock signal driving a FF is disabled (gated) when the FF state is not subject to a change in the next clock cycle. Few attempts to take advantage of this principle have been made before [5]–[7] for design levels higher than individual FFs; all of them rely on various heuristics in an attempt to increase clock gating opportunities. An activity driven clock tree was presented in [6] for data flow blocks. This is a good example where the designer is aware of the interrelations between the various modules, and therefore can introduce appropriate enable signals.

Clock gating does not come for free. Extra logic and interconnects are required to generate the clock enabling signals and the resulting area and power overheads must be considered. In the extreme case, each clock input of a FF can be disabled individually, yielding maximum clock suppression. This, however, results in a high overhead; thus suggesting the grouping of several FFs to share a common clock disabling circuit in an attempt to reduce the overhead. On the other hand, such grouping may lower the disabling effectiveness since the clock will be disabled only during time periods when the inputs to all the FFs in a group do not change.

In the worst case, when the FFs' inputs are statistically independent, the clock disabling probability equals the product

Manuscript received April 25, 2011; revised July 08, 2011; accepted July 19, 2011. This work was supported in part by MAGNET Program of Israel Ministry of Industry.

S. Wimer is with the School of Engineering, Bar-Ilan University, Ramat-Gan 52900, Israel (e-mail: wimers@macs.biu.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.2011.2162861

of the individual probabilities, which rapidly approaches zero when the number of involved FFs increases. It is therefore beneficial to group FFs whose switching activities are highly correlated and derive a joint enabling signal. The state transitions of FFs in digital systems like microprocessors and controllers depend on the data they process. Assessing the effectiveness of clock gating requires therefore extensive simulations and statistical analysis of FFs activity, as presented in this paper.

Disabling the clock input to a group of FFs (e.g., a register) in data-path circuits is very effective since many bits behave similarly. Registers enabled by the same clock signal yield a high ratio of the saved power to circuit overhead. Furthermore, the design effort to create the disabling signal is low. Unlike data-path, control logic requires far greater design effort for successful clock gating. This stems from the "random" nature of the control logic. The effectiveness of the proposed gating methodology is demonstrated in this paper through the examples of a 3-D graphics accelerator and a 16-bit microcontroller. These units were designed with full awareness of the internal data dependencies and appropriate clock enabling signals were defined within the RTL code. When the RTL code was then compiled and simulated at gate level, considerable "hidden" disabling opportunities have been discovered.

In many cases clock gating is applied only to the first level of gaters directly driving FFs, since the majority of the load occurs at the leaves of the clock tree where the FFs are connected [11]. Even if we could ideally stop the clock from driving all the FFs when it is not required, the rest of the network will continue pumping clock signals and wasting energy. We consider therefore gating higher levels of the clock tree (closer to root). These portions of the tree may also consume considerable power since they are using long and wide wires plus intermediate drivers to avail robust clock signals for far end FFs. The proposed gating will dynamically prune large portions of the clock tree if it becomes clear that none of the driven FFs is subject to a change in the next cycle.

B. Gated Clock Network Modeling

The construction of gated clock trees raises two questions. The first is: what should be the fan-out of a gater, i.e., how many FFs should a leaf gater drive, and similarly for higher levels of the tree, how many children gaters should be driven by one parent? The second question concerns what FFs should be grouped to share a common gater, and similarly for higher levels of the tree, which sibling gaters should be grouped for maximum power savings? To answer the first question we will use a power model similar to that in [9] and [10] which accounts for interconnects of clock signal and the enabling (gating) signals overhead. While all works so far have assumed a binary clock tree model, we derive the optimal fan-out of the clock tree which maximizes the net switching power savings, accounting for the overhead incurred by the extra logic circuitry required to generate the gating signals. This, to the best of our knowledge, has not been addressed before. For the second question, the matching technique heuristically applied in [6] is used here in a more formal way, but the problem was lately shown to be NP-complete [25].

A key aspect of the optimal solution of the above problems is the probabilistic behavior of FFs' toggling which was also addressed in [9] and [10]. However, unlike their register toggling and gating model that was developed based on random simulations, this paper uses a worst case probabilistic model, yielding a result that provides a provably lower bound on the power savings. It is therefore uniformly applicable to any design and the actual power reduction obtained by the methodology proposed here can only be higher than that predicted by our worst case model. It is important to note that the proposed methodology tests a large set of typical applications prior to clock tree construction in an attempt to find the probability and correlation of FF toggling and follow the best-case rather than the worst case lower bound. FF toggling correlation is used for optimally grouping the FFs.

The work presented in [9] describes a complete clock gating solution at module granularity, where the collective activity of all its underlying FFs represents the module's activity. The approach considers both the logical and physical aspects of the gated clock network implementation. On the logic side there is an attempt to discover "hidden" clock gating beyond the straight forward derivation obtained from enabling signals described in the RTL, thus increasing its clock disabling periods.

The elementary gated objects considered in [5] and [6] are modules, and the activity periods are defined by execution tasks, which may comprise many clock cycles each. Unlike [5] and [6], the resolution of gating proposed in this paper is of individual FFs at individual clock cycles. Gating at that resolution has been proposed for regularly structured circuits such as Linear feedback shift register (LFSR) [13] and counters [19], where the amount of power savings can be predicted from the circuit's structure. An attempt to discover an explicit clock disabling condition was made in [19]. It requires detailed knowledge of the state transitions and state coding, based on which clock signal requirements were derived and used for gating. The method is useful for simple and well-structured circuits such as counters, but may be very difficult to apply to general control logic whose state coding assignment is usually determined by automatic synthesis tools.

Regarding the clock tree structure, both [5] and [6] propose a tree that allows gaters at each internal node, depending on the activity of the node. The latter is defined by ORing the activities of the node's children (and hence leaves of the rooted sub tree). The authors targeted zero skew but have ignored the delay aspects of the generated enabling signals. For a large and high performance chip this may result a clock latency of a few clock cycles, making their solution undesirable. To demonstrate the importance of activity correlation, the authors have obtained the correlation between leaf nodes by generating a few dozens of random activity patterns over a small number of clock periods. Such decisions, however, should rely on extensive simulations of real data traces rather than random ones. Finally, the authors of [5] have set the relation between the capacitive load of the clock tree and that of the enabling control logic by using relative weights without specifying how these weights are determined. In this paper we accurately derive the load incurred by clock enabling, taking into account both the logic gates and the interconnects involved.

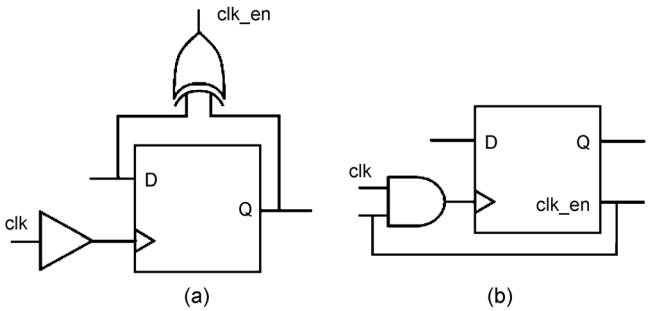


Fig. 1. Enabling of the clock signal.

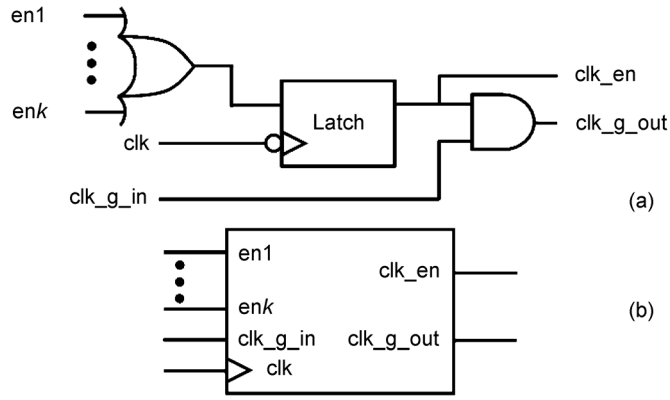


Fig. 2. Joining k enabling signals generated by distinct flip-flops into one gating signal.

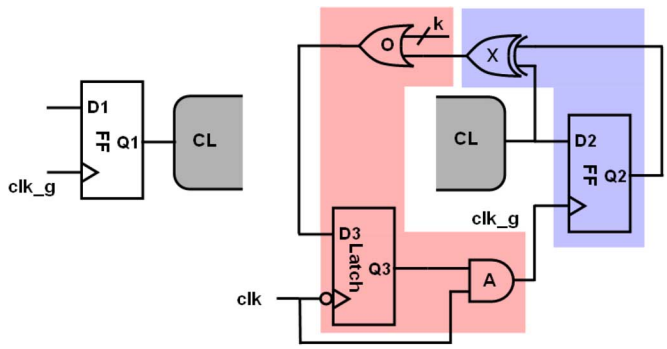


Fig. 3. Application of clock gating with feedback enabling signals. The components highlighted in red centered shaded area comprise the clock gater, while those in blue reside in FF.

The rest of this paper will establish the structure of the adaptive disabling circuits and show how they are combined in the traditional clock tree. A power savings model is developed in Section III and expressions yielding an optimal fan-out are derived. Section IV discusses the problem of what FFs to put together to be jointly driven by a common clock gater. Section V concludes this paper.

II. Adaptive Clock Gating Implementation

Fig. 1 shows how a FF can find out that its clock can be disabled in the next cycle. A XOR gate compares the FF's current output with the present data input that will appear at the output in the next cycle [12]. The XOR's clk_en output indicates whether a clock signal will be required in the next cycle. The clock driver in Fig. 1(a) is then replaced by a 2-way AND gate called *clock gater*. We will use the symbol in Fig. 1(b) to represent FFs that incorporate generation of clk_en. In practice the XOR is connected to the output of FF's internal master rather than D, as it is guaranteed to be stable when the FF's slave is transparent.

Controlling the clock in each FF by a dedicated gater was studied in [12]. An implementation for a linear feedback shift register (LFSR) has been presented in [13], where a 10% net power reduction was reported. Additional power reduction can be achieved by lowering the number of clock gaters. We could drive several FFs with a common gater if we knew that they are toggling simultaneously most of the time, thus achieving almost the same power reduction, but with fewer gaters. The grouping may place up to several dozens of FFs in a single group, and is usually done by synthesizers during the physical design phase [14]. Such tools are focusing on skew, power, and area minimization, and are not aware of the toggling correlations of the underlying FFs.

Fig. 2 shows how to join k clk_en signals generated by distinct FFs into one gating signal. It saves the individual clock gaters at the expense of an OR gate and a negative edge triggered latch that is required to avoid glitches of the enable signal. Due to the power consumed by the latch such joining is justified only for $k \geq 3$ [23]. The combination of a latch with an AND gate is commonly used by commercial tools and is called integrated clock gate (ICG) [15]. Clearly, the hardware savings increases with k, but the number of disabled clock pulses is decreasing. Thus, for the scheme proposed in Fig. 2 to be beneficial, the clock enabling signals of the grouped FFs must be highly correlated. Such correlation and its implication on the clock savings is thoroughly studied below. Since the enabling signals ORed in Fig. 2 are the outputs of XOR gates, which may have glitches, the question of the power penalty is in order. Fortunately, the probability of signal toggling is well known to be very low [18] so the average amount of glitches is expected to stay small as well.

The proposed adaptive clock gating has considerable timing implications. Fig. 3 illustrates a FF to FF logic stage with its driving clock signals. In the physical implementation, the XOR gate is integrated into the FF, while the OR gate, AND gates and the latch are integrated into the clock gater. Fig. 4 depicts the timing sequence and its implied constraints. There are two distinct clock signals: clk_g is the ordinary gated signal driving the registers, while clk is driving the latches of the clock gaters. Both have the same period denoted by T_C.

Using the notations $t_{\mathrm{pA}}, t_{\mathrm{pO}}, t_{\mathrm{pX}}$ for the propagation delay of AND, OR, and XOR gates, respectively, $t_{\mathrm{pd\_logic}}$ for the propagation delay of a logic stage between two FFs, and $t_{\mathrm{pcq}}, t_{\mathrm{setup}}$ for clock to Q propagation delay and setup time, respectively, the following constraint (derived from Fig. 4) must be satisfied for proper operation

$$t_{\mathrm{pcq\_FF}} + t_{\mathrm{pd\_logic}} + t_{\mathrm{setup\_FF}} \leq T_{\mathrm{C}}. \tag{1}$$

This is the ordinary constraint used in VLSI design practice, without adaptive gating, that is imposed by clk_g. The introduction of gating results in the following constraint (derived from Fig. 4):

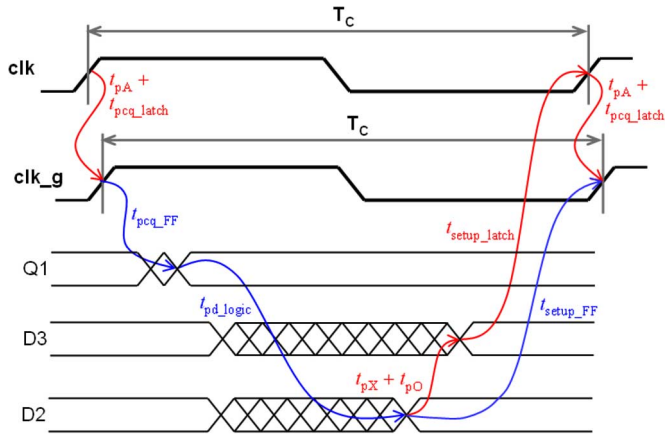


Fig. 4. Timing sequencing and overhead of adaptive clock gating.

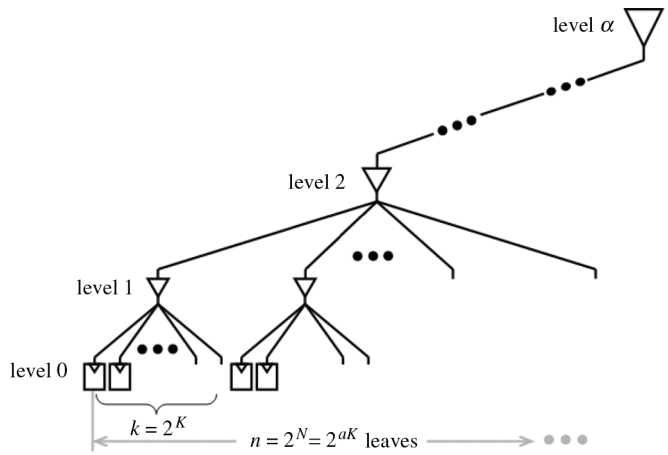


Fig. 5. Sized clock tree distribution network.

$$t_{\mathrm{pA}} + t_{\mathrm{pcq\_latch}} + t_{\mathrm{pcq\_FF}} + t_{\mathrm{pd\_logic}} + t_{\mathrm{pX}} + t_{\mathrm{pO}} + t_{\mathrm{setup\_latch}} \le T_{\mathrm{C}} \quad (2)$$

that is required for proper latching of the enabling signal as imposed by clk. It follows from (1) and (2) that

$$t_{\mathrm{pcq\_FF}} + t_{\mathrm{pd\_logic}} + T' \le T_{\mathrm{C}}, \quad \text{where}$$
$$T' = \max\{t_{\mathrm{setup\_FF}}, t_{\mathrm{pA}} + t_{\mathrm{pcq\_latch}} + t_{\mathrm{pX}} + t_{\mathrm{pO}} + t_{\mathrm{setup\_latch}}\}. \quad (3)$$

Equation (3) imposes tight constraints on the setup times of the latch and FF and the delay of the gating logic. Furthermore, it may happen that (2) will not be satisfied unless the clock period is relaxed or the logic propagation delay stays small enough.

Joining enabling signals of individual FFs suits well the commonly used clock tree distribution networks [16]. A typical topological structure is shown in Fig. 5. The clock signal enters the block at a pin called root, and is then being driven to the far-end FFs through chains of drivers connected in a tree topology. It is possible to replace the drivers of the tree in Fig. 5 by k-way gaters shown in Fig. 6. A gater receives the enabling signals of its k children and delivers the clock signal downstream accordingly.

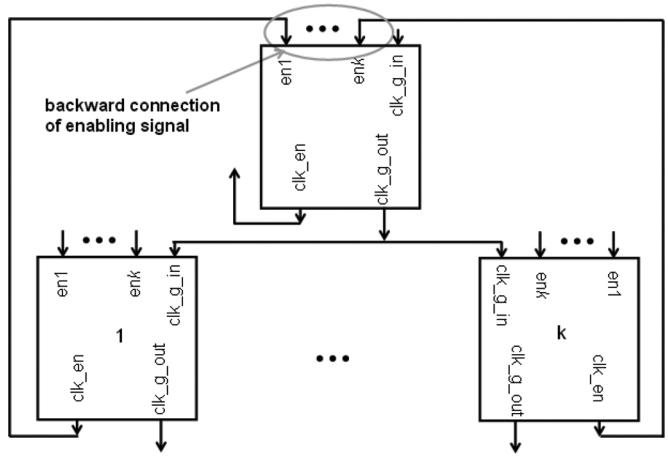


Fig. 6. Replacement of clock drivers by gaters in a k fan-out clock tree. k children gaters feed their enabling signals back to their parent where they are ORed and latched.

III. Joint Gating and Gater's Optimal Fan-Out

Assume that a circuit contains $n = 2^N$ FFs whose clock signals are driven by the tree shown in Fig. 5. Its leaves are connected to the FFs and the gaters' fan-out is $k = 2^K$. We assume that $N = \alpha K$, where α is the number of levels of the clock tree. A leaf gater has unit size (driving strength). The gater at the first level is connected to the leaf by a wire of unit length and unit width. We now introduce the following notations to quantify and analyze the power savings achieved by joint clock enabling: c_{FF}—FF's clock input capacitance; c_{latch}—latch capacitance; including the wire capacitance of its clk input; c_{w}—unit wire capacitance; c_{gater}—unit drive gater capacitance; c_{OR}—OR gate capacitance; β—level to level gater's sizing factor; γ—level to level wire width sizing factor; δ—level to level wire length sizing factor.

In this notation the size of a gater in level j is β^{j-1} and the size of a wire connecting level j to $j-1$ is $(\gamma\delta)^{j-1}, 1 \le j \le \alpha$, as commonly happens in tree networks such as the H-tree [17]. The total capacitive load of the resulting clock tree is

$$C_{\mathrm{tree}} = nc_{\mathrm{FF}} + c_{\mathrm{gater}} \sum_{j=1}^{\alpha} \left(\frac{n}{k^j}\right) \beta^{j-1} + c_{\mathrm{w}} \sum_{j=1}^{\alpha} \left(\frac{n}{k^{j-1}}\right) (\gamma\delta)^{j-1}$$
$$= n\left[c_{\mathrm{FF}} + \frac{c_{\mathrm{gater}}}{\beta} \frac{1-(\beta/k)^{\alpha}}{1-\beta/k} + c_{\mathrm{w}} \frac{k}{\gamma\delta} \frac{1-(\gamma\delta/k)^{\alpha}}{1-\gamma\delta/k}\right]. \quad (4)$$

Consider for example the well-known clock H-tree [17], for which $k = 4 (K = 2)$. To illustrate (4) and examine the relative contribution of the various capacitances to power consumption let $n = 1024$ and then $N = 10$ and hence $\alpha = 5$. Setting $\beta = 2$, $\gamma = 2$, and $\delta = 4$, yields $C_{\mathrm{tree}} = 1024(c_{\mathrm{FF}} + c_{\mathrm{gater}} 32/31 + c_{\mathrm{w}} 31/2)$.

To asses the clock gating impact on power we consider the toggling of FF as an independent random variable. A FF has probability p to change state and $q = 1 - p$ to stay unchanged. The probability of a group of k FFs to stay unchanged (as a group) is therefore q^k. The probability p is sometimes called *activity factor*. It is well known that the average activity factor of non clock signals is very low, since a typical signal toggles very infrequently [18].

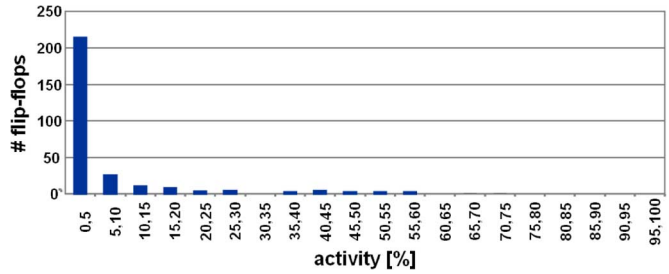


Fig. 7. FF activity of 16-bit microcontroller (register file excluded).

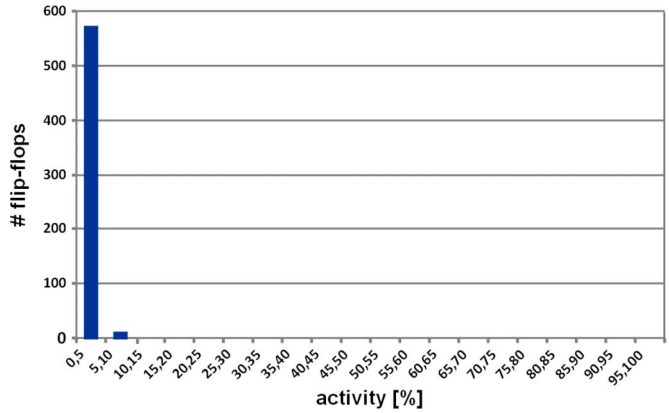


Fig. 8. FF activity of a triangle's rasterization unit in a 3-D graphics accelerator.

The toggling probabilities of individual FFs are obtained by running gate-level verilog simulation with a representative test bench of the application in hand. This is demonstrated in Fig. 7 that shows the activity factors measured for a 16-bit microcontroller. A test bench of its instruction set has been simulated and the toggling of every FF in its ALU and control circuits (register file was excluded) was recorded. As shown, the majority of FFs are toggling a very small fraction of time, less than 5%. Similar statistics are shown in Fig. 8 for a triangle's rasterization unit used in a 3-D graphics accelerator.

A gater at level j of the tree is driving k child gaters of size β^{j-2} and k wires of size $(\gamma\delta)^{j-1}$. Since the number of FFs spanned by that gater is k^j (the number of leaves in the sub-tree rooted at that gater), the probability of a disabling clock signal is q^{k^j}. The dynamic power saved by the gater is the product of its disabling probability and the capacitive load it is driving. This load is given by $kq^k(c_{\rm FF}+c_{\rm w})$ for first level gater and by $kq^{k^j}\left[c_{\rm gater}\beta^{j-2}+c_{\rm w}(\gamma\delta)^{j-1}\right]$ for the second level and above. There are n/k^j nodes at level j of the tree. Let $\alpha' \le \alpha$ be the highest gated level. Consider the total power savings $C_{\rm saving}^{1-\alpha'}$ resulting from replacing the ordinary drivers by clock gaters, without accounting for gating logic and interconnects overhead. It is obtained by summation of the savings over all nodes of the gated levels, given by

$$C_{\rm saving}^{1-\alpha'} = n(c_{\rm FF}+c_{\rm w})q^k + \sum_{j=2}^{\alpha'}(n/k^{j-1})q^{k^j}\left[c_{\rm gater}\beta^{j-2}+c_{\rm w}(\gamma\delta)^{j-1}\right]. \quad (5)$$

Clock gating incurs certain power and area costs. As shown in Figs. 1–3, FFs need additional XOR gates and every gater requires a k-way OR gate and a latch. Moreover, there is a wiring penalty resulting from the separation of clk_g and clk. The interconnections realizing clk_g are switching only when the clock is required for FF toggling. These are the real functional clock wires with the full sizing required to deliver high quality clock signal. The interconnections propagating clk are needed for the latches residing at the gaters and are used at each cycle. Notice that clk exists only at gaters in the first level of the tree and above, but does not exist at the leaves (FFs). There are also the clk_en signals, feeding back the activity of k children gaters (or FFs at leaves) to the OR gate at their parent. The wires of clk and clk_en, shown in Fig. 6, generate a "shadow" of the clock tree in Fig. 5. These wires can be of a minimum width, subject to delay constraints shown in Fig. 4. A reasonable assumption for the subsequent analysis is that their length is similar to that of clk_g since they connect the same elements as clk_g does.

The calculation of the power consumed by the shadow tree with its logic overhead is based on toggling probabilities. An enabling signal informs the gater at level j whether its child gater at level $j-1$ needs the clock pulse in the next cycle. The toggling independence is a worst case assumption since toggling correlation increases power savings as it reduces the probability of a gater to send a clock signal to a FF when it does not need it. We calculate the net power savings, denoted by $c_{\rm net\_saving}^j$, $1 \le j \le \alpha'$, for a single branch of the tree and then sum over all branches. At the leaves where FFs are connected ($j=1$), the net power savings per branch satisfies

$$c_{\rm net\_saving}^1 \ge q^k(c_{\rm FF}+c_{\rm w}) - \left[c_{\rm latch}/k + (1-q)(c_{\rm w}+c_{\rm OR})\right]. \quad (6)$$

The term $q^k(c_{\rm FF}+c_{\rm w})$ in (6) is the savings due to the disabling of clk_g. The term $c_{\rm latch}/k$ is the overhead due to the latch at the parent gater being always clocked by the clk signal. The division by k stems from the fact that the latch overhead is amortized among the k branches connected to the gater. The overhead $(1-q)(c_{\rm w}+c_{\rm OR})$ is due to the switching of clk_en. Notice that if the probability of a FF to toggle is $p=1-q$, then $\Pr(clk\_en=1)=1-q$ and hence its switching probability cannot exceed $1-q$.

For the internal nodes of the tree ($j \ge 2$) we follow a similar analysis as done for $j=1$. It is shown in (5) that the savings for a forward branch of clk_g due to its disabling probability q^{k^j} is given by

$$c_{\rm saving}^j = q^{k^j}\left[c_{\rm gater}\beta^{j-2}+c_{\rm w}(\gamma\delta)^{j-1}\right] \quad (7)$$

where $c_{\rm gater}$ and $c_{\rm w}$ are multiplied by their appropriate sizing factors.

In parallel to the forward clock signal clk_g, there is a "shadow" feedback enabling signal clk_en, issued from the latch output of the $(j-1)$-level gater (see Fig. 2), driving one input of the k-input OR gate of the j-level gater, whose output is latched at level j. The latch at level j is always clocked by clk, but it is amortized among the k forward branches of the gater. clk_en is 1 when its corresponding $(j-1)$-level gater needs the clock signal in the next cycle and 0 if it does not. Since the toggling probability of the $(j-1)$-level gater is $1-q^{k^{j-1}}$ it

follows that $\Pr(clk\_en = 1) = 1 - q^{k^{j-1}}$ and hence its relative switching count cannot exceed $1 - q^{k^{j-1}}$.

In summary, the power overhead per branch to generate the enabling signal is given by

$$c^j_{\text{ovehaed}} = c_{\text{latch}}/k + \left(1 - q^{k^{j-1}}\right)\left[c_{\text{w}}(\gamma\delta)^{j-1} + c_{\text{OR}}\right],\ 2 \le j \le \alpha'. \tag{8}$$

Notice that we made a worst case assumption by using the same sizing factor $(\gamma\delta)^{j-1}$ for clk_en wire as for clk_g. Subtracting of (8) from (7) yields the net power savings per branch as follows:

$$\begin{aligned} c^j_{\text{net\_saving}} \ge\ & q^{k^j}\left[c_{\text{gater}}\beta^{j-2} + c_{\text{w}}(\gamma\delta)^{j-1}\right] \\ & - \left\{c_{\text{latch}}/k + \left(1 - q^{k^{j-1}}\right)\left[c_{\text{w}}(\gamma\delta)^{j-1} + c_{\text{OR}}\right]\right\}, \\ & 2 \le j \le \alpha'. \end{aligned} \tag{9}$$

Notice that (6) can be obtained from (9) by substituting $j = 1$ and replacing $c_{\text{gater}}\beta^{j-2}$ with c_{FF}.

The total net power savings $C^{1-\alpha'}_{\text{net\_saving}}$ in a clock tree gated up to level α' is obtained by summation of the net savings over all branches of the gated levels. There are n wires connected to FFs whose savings is given in (6), and n/k^{j-1} wires connected from level j to level $j-1$ for $2 \le j \le \alpha'$, whose savings is given in (9), thus yielding

$$C^{1-\alpha'}_{\text{net\_saving}} = nc^1_{\text{net\_saving}} + \sum_{j=2}^{\alpha'}(n/k^{j-1})c^j_{\text{net\_saving}}. \tag{10}$$

The importance of (10) stems from the fact that it describes the relationship between the clock signal disabling probabilities and the circuit's capacitance factors on one hand, and the clock tree structural parameters (gater's fan-out k) on the other hand. This enables the construction of a clock tree that yields maximum power savings. Solving the equation $(d/dk)C^{1-\alpha'}_{\text{net\_saving}} = 0$ yields the optimal k. This equation is complex and not analytically solvable but can be solved numerically.

Consider the common case in logic-gate design-level where clock gating usually takes place only at the first level of the tree. Such gating is what is currently supported by several CAD tools, leaving to the user the decision regarding the value of k, usually by relying on past experience. Equating to zero the derivative of (6) with respect to k yields the following implicit equation for the optimal k:

$$q^k \ln q(c_{\text{FF}} + c_{\text{W}}) + c_{\text{latch}}/k^2 = 0. \tag{11}$$

Notice that the gating overhead term $(1-q)(c_{\text{w}} + c_{\text{OR}})$ appearing in (6) does not affect the optimal k since it is being paid by each of the n FFs, regardless of the value of k.

In an attempt to find the optimal value of k, Fig. 9 is showing the normalized power savings per FF derived from (6). The savings is compared to the nongated situation. Various values of $q = 1 - p$ have been examined to explore the behavior of the optimal k. The relative capacitance of FFs, latches, OR gate and unit wires connecting the first level gater to the FFs depend on the specific technology and cell library in hand. We assumed all

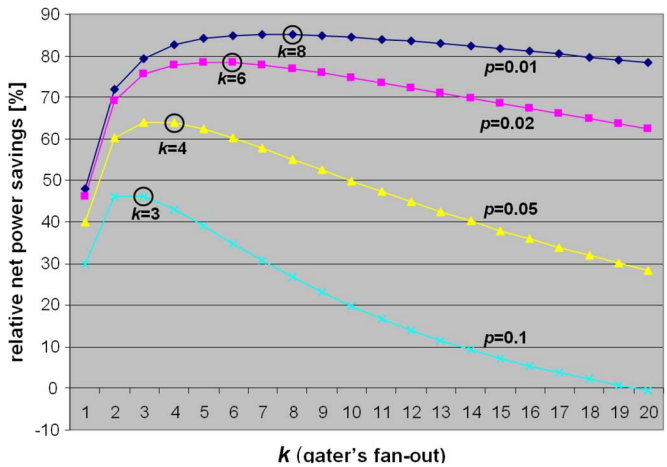


Fig. 9. Normalized power net savings per FF obtained by adaptive gating at 1st level of clock tree in (6). The savings is compared to the non gated situation. The optimal fan-out is marked for each toggling probability.

to be equal in Fig. 9. As expected, the lower the toggling probability of FF is, the higher the optimal k is. The optimal k values obtained in the plots agree with the common practice of EDA tools. It is shown that significant savings can be achieved. Recall however that there is delay and area overhead and though high fan-out values results less gaters, the OR fan-in is increasing accordingly, which will further increase area and delay overheads.

An implementation of adaptive gating has been reported in [13], where, after taking into account the power consumed by the extra circuitry, a 10% net power savings was reported. We observed similar amounts of savings based on gate-level verilog simulations of designs, where adaptive gating was added to the first level of clock gater. This translates to 5% of total dynamic power savings of the entire chip. The net savings were obtained on top of savings obtained by clock enabling signals which have already been introduced by the designer at the RTL verilog.

Additional savings can be obtained by gating at higher levels of the tree. The normalized net power savings per FF for gating at three levels is illustrated in Fig. 10 as a percentage of the non gated situation. There, gater's drive, wire width and wire length sizing factors of $\beta = \sqrt{2}$, $\gamma = \sqrt{2}$, and $\delta = 2$, respectively, have been used. As can be seen, higher power savings per FF are achieved by gating at the second and third levels. For low toggling probabilities more power savings is obtained. Though the percentage is a bit lower than in Fig. 9, the total is higher since it is from a larger capacitance. On the other hand, once FFs toggling probabilities increases, the savings goes rapidly down, and for $p > 0.2$ there is only power loss. The area implications of the proposed scheme for acceptable values of the fan-out need to be further investigated by incorporating it into a backend layout flow. This, however, is beyond the scope of this study.

A comment on the gating depth α' is in order. The term q^{k^j} in (9) is rapidly approaching zero with increasing j, turning $c^j_{\text{net\_saving}}$ into a negative value. This in turn results in power waste rather than savings as can be seen in Fig. 10. Adaptive gating should therefore be restricted to the lower levels of the clock tree.

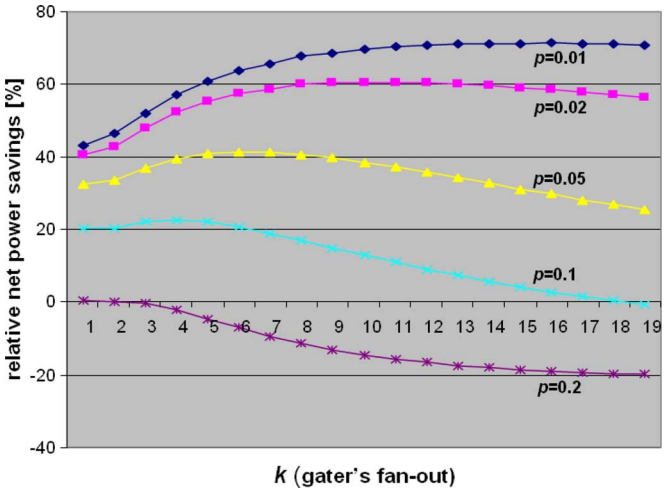


Fig. 10. Normalized power savings per FF obtained by adaptive gating at first lower three levels of clock tree. The savings is compared to the nongated situation.

Another comment concerns latency. Timing constraints applicable to FFs at the leaves of the clock tree have been derived in (1)–(3). In the proposed gating scheme, the next cycle enabling signals are bottom-up propagated in the "shadow" tree towards its root. Each node in a path from leaf to root determines whether it needs the clock signal clk_g for the next cycle and then transmits its decision to its parent. clk_g is then delivered through the main clock tree from the root down to the FFs. The delay of this round trip must fall within a single clock cycle, which is unlikely to happen for a high clock speed and a clock tree comprising many levels. This is another reason for restricting adaptive gating to the lower levels of the clock tree.

IV. Grouping of FFs and Gaters for Clock Tree Construction

Section III developed the probabilistic model of adaptive gating and derived expressions for the optimal gater's fan-out. We made a worst-case assumption that the FFs are toggling independently of each other. In reality, toggling of FFs are correlated to some degree, which can only increase the power savings in (10). This follows from the disabling probabilities appearing in the positive terms of (6) and (9) that can only become greater than q^{k^j}, while the feedback toggling probabilities appearing in the negative terms will get smaller than $1-q^{k^{j-1}}$. The next step is to decide on the groups of k FFs to be driven by a common clock signal, and similarly determine the grouping of internal tree gaters when constructing the entire clock tree shown in Fig. 5.

FFs and gaters groupings have logic and physical aspects. The logic aspect attempts to minimize the number of clock pulses delivered to FFs and gaters when they are not needed; these are called redundant clock pulses. The physical aspect has to do with the on-die locations of FFs and gaters which directly affect the amount of routing required for their connection, and hence their capacitive load, delay, and clock skew.

Solving the logic aspect has been shown to be an NP-complete problem [25] and hence a heuristic solution is in order. In this section we present an approach towards a practical solution. Few papers have addressed the grouping problem when constructing a binary clock tree ($k=2$). A heuristic for sorting FFs according to their activity and pairing them in that order was presented in [22]. It is possible, however, to construct an example where this heuristic would increase the number of redundant clock pulses rather than minimize them. In [10] and [20], FFs and gaters were paired based on intuitive arguments without a formal proof, and it may sometimes yield inferior gating. It has been correctly pointed out in [6] that for a binary tree the FF pairing at leaves can be optimally solved using a minimum weight perfect matching algorithm [21].

A scheme for constructing clock trees when the positions of the leaves are known was described in [9]. The leaves can be FFs or modules' input clock pins for higher design levels. Clock activities and clock pin distances are weighted and summed, but this is problematic since the physical meaning of a weighted sum is not well defined and requires delicate setting of the weights. It is also possible to generate an example where the weighted pairing heuristic yields the worst solution. We believe that summing of products of activity by distance is more appropriate since it explicitly measures power consumption and no weights are needed.

Considering the logic aspect, let a circuit run for $T+1$ clock cycles. Let the vector $\mathbf{a}=(a_1,\ldots,a_T)$ denote the activity of a FF, where$a_t=0$, $1\le t\le T$ 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 1s in $\mathbf{a}$, which is proportional to the power consumed by FF switching. Each of the $n(n-1)/2$ FF's activity pairs $(\mathbf{a}_i,\mathbf{a}_j)$, $1\le i<j\le n$, are bit-wise XORed and $\|\mathbf{a}_i\oplus\mathbf{a}_j\|$ is therefore the number of redundant clock pulses occurring if F_i and FF_j are jointly clocked by the same gater. Two correlations are defined. The first equals $1-\|\mathbf{a}_i\oplus\mathbf{a}_j\|/T$, measuring FFs pair activity correlation during the entire period T. For FFs whose toggling rate is very low this value is nearly 1, regardless of their toggling overlap. The second correlation equals $1-\|\mathbf{a}_i\oplus\mathbf{a}_j\|/\|\mathbf{a}_i|\mathbf{a}_j\|$ (where the OR is a bit-wise operation), measuring their joint toggling. Large values of those indicate a high potential of joining FFs for a common drive such that the number of redundant clock pulses is reduced, thus yielding higher power savings.

The toggling correlations of the FFs in a 16-bit micro-controller whose activities are shown in Fig. 7, have been measured. Fig. 11 shows the $1-\|\mathbf{a}_i\oplus\mathbf{a}_j\|/T$ activity correlation metric. Unsurprisingly, for the majority of pairs this value is nearly 1. This happens since their toggling probability is very low and hence $\|\mathbf{a}_i\oplus\mathbf{a}_j\|\ll T$. Fig. 12 shows the joint toggling correlation. Indeed, there are many FFs pairs that can be driven by a common gater with low redundant clock pulses. The correlations measured for the triangle's rasterization unit of a 3-D graphics accelerator shown in Fig. 8 are illustrated in Figs. 13 and 14, with similar activity and toggling correlations.

In order to group FFs at the leaves, and similarly gaters at the tree's internal nodes, we first address the case of $k=2$, which is the binary tree model used in most prior works. We define a weighted complete graph $G(V,E,w)$ as follows. A vertex $v_i\in V$ corresponds to FF_i and an edge $e_{ij}\in E$ connecting two vertices $v_i,v_j\in V$, $1\le i<j\le n$, is associated with a weight

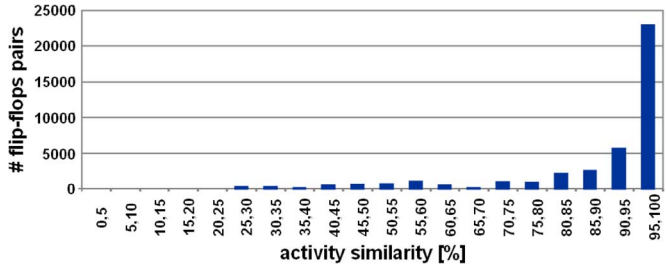


Fig. 11. FF pair-wise activity correlation for 16-bit microcontroller.

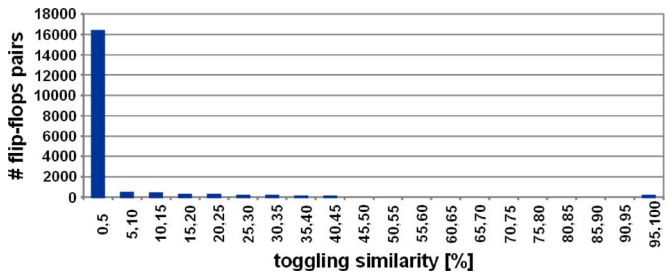


Fig. 12. FF pair-wise joint correlation for 16-bit microcontroller.

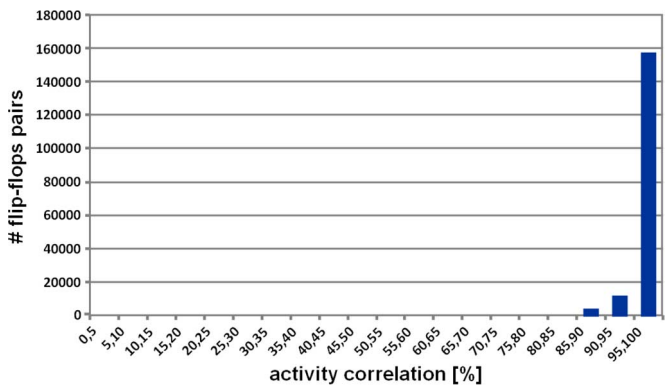


Fig. 13. FF pair-wise activity correlation for a triangle's rasterization unit in 3-D graphics accelerator.

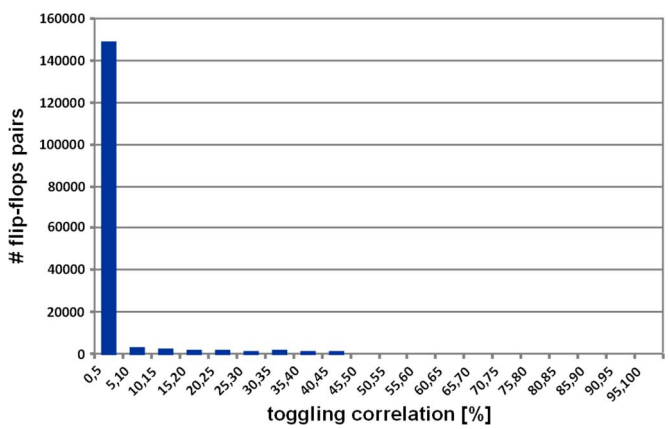


Fig. 14. FF pair-wise joint correlation for a triangle's rasterization unit in 3-D graphics accelerator.

$w(e_{ij}) = \|\mathbf{a}_i \oplus \mathbf{a}_j\|$. The weight represents the number of redundant clock pulses driving FF_i and FF_j, resulting from being clocked by a common gater. The optimal FF pairing is therefore equivalent to covering V by $n/2$ edges of minimum weight sum [6]. This is the well-known minimal perfect matching problem [21].

Figs. 7 and 8 which show a very small average toggling probability, and the gater's optimal fan-out obtained from (11) and (10), and depicted in Figs. 9 and 10, respectively, indicate that k should be usually greater than 2 and the minimal perfect graph matching model must therefore be modified. Several papers have proposed repeated application of perfect matching for a bottom-up construction of a binary clock tree. At each level of the hierarchy, a complete graph with half the number of vertices of in its lower level, is defined. A vertex is associated with a toggling vector defined by the union (bit-wise ORing) of its two children, while an edge is weighted by the number of redundant clock pulses incurred by driving the two graph's vertices through a joint gater. Though intuitive, it does not yield the optimal grouping as was shown in [25].

To consider the matching of $k > 2$ vertices in an attempt to minimize the number of redundant clock pulses, we can use a complete k-uniform hyper graph $H(V, E, w)$, modeling the "toggling proximity" of FFs groups as follows. A hyper edge $e(V') \in E$, $V' \subset V$, satisfies $|V'| = k$. Denote by $\mathbf{a}_v$ the toggling vector of FF_v, $v \in V$. The weight of a hyper edge represents the number of redundant clock pulses driving V''s FFs, and is given by

$$w\left(e(V')\right) = \sum_{v \in V'} \left\| \mathbf{a}_v \oplus \bigcup_{u \in V'} \mathbf{a}_u \right\|. \tag{12}$$

The union in (12) is the bit-wise ORing of the k toggling vectors, while XORing the union with an individual toggling vector $\mathbf{a}_v$ yields the number of redundant clock pulses drivingFF_v. It follows that $|E| = \binom{n}{k}$ and the problem of finding the n/k hyper edges covering the n vertices and yielding minimum redundant clock pulses turns into the well-known NP-complete minimal weight exact covering problem [24] and any approximation of the latter will apply.

As mentioned before the "logic proximity" must be accounted together with some knowledge of the proximity of FFs. Weighing $H(V, E, w)$ hyper edges by product of a distance measure (e.g., the diameter of the circle enclosing FFs) and the number of redundant clock pulses in (12) is suggested. It directly measures the wasted switching power. This is presently studied separately and its discussion is beyond the scope of this paper.

V. Conclusion and Further Research

This paper has presented a probabilistic model of the clock gating network that allows quantifying the expected power savings and the implied overhead. It was shown that under reasonable and realistic assumptions, supported by simulations of real VLSI designs, the optimal fan-out of a gater which maximizes the power saving can be derived. The derivation is based on the toggling probability of the FFs comprising the circuit, the relative capacitance factors of the process technology and cell library in hand, and the sizing factors used in the clock tree construction. A backend (layout) implementation in 32-nm process

technology is presently being developed and implemented. It will provide a full picture of area, timing, and OCV implications of the proposed gating method.

The togglings of FFs were assumed to be independent of each other, which is the worst-case assumption, and in case of high FFs activity, the gater's fan-out may be very small. It is therefore interesting to develop a model for the optimal fan-out under the assumption that a certain correlation exists. This will then allow increasing the fan-out and achieving higher power savings.

The question of how to combine FFs into groups whose optimal size is known has also been formalized. Though the logic aspect of FFs toggling correlations is clear, the difficult question of how to account for layout considerations in such grouping needs further study.

REFERENCES

[1] V. G. Oklobdzija, *Digital System Clocking—High-Performance and Low-Power Aspects*. New York: Wiley, 2003.

[2] N. H. E. Weste and D. Harris, *CMOS VLSI Design—A Circuit and System Perspective*. Boston, MA: Addison-Wesley, 2005, ch. 7, 12.

[3] 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.

[4] M. S. Hosny and W. Yuejian, "Low power clocking strategies in deep submicron technologies," in *Proc. IEEE Int. Conf. Integr. Circuit Design Technol. (ICICDT)*, pp. 143–146.

[5] C. Chunhong, K. Changjun, and S. Majid, "Activity-sensitive clock tree construction for low power," in *Proc. ISLPED*, 2002, pp. 279–282.

[6] 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.

[7] W. Shen, Y. Cai, X. Hong, and J. Hu, "Activity and register placement aware gated clock network design," in *Proc. ISPD*, 2008, pp. 182–189.

[8] M. Donno, A. Ivadldi, L. Benini, and E. Macii, "Clock-tree power optimization based on RTL clock gating," in *Proc. Design Autom. Conf.*, 2003, pp. 622–627.

[9] M. Donno, E. Macii, and L. Mazzoni, "Power-aware clock tree planning," in *Proc. ISPD*, 2004, pp. 138–147.

[10] W. Shen, Y. Cai, X. Hong, and J. Hu, "Activity and register placement aware gated clock network design," in *Proc. ISPD*, 2008, pp. 182–189.

[11] Y. Cheon, P.-H. Ho, and A. B. Kahng, "Power-aware placement," in *Proc. Design Autom. Conf.*, 2005, pp. 795–800.

[12] T. Lang, E. Musoll, and J. Cortadella, "Individual flip-flops with gated clocks for low power datapaths," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 44, no. 6, pp. 507–516, Jun. 1997.

[13] W. Aloisi and R. Mita, "Gated-clock design of linear-feedback shift registers," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 5, pp. 546–550, Jun. 2008.

[14] Cadence, Berkshire, U.K., "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] M. Muller, S. Simon, H. Gryska, A. Wortmann, and S. Buch, "Low power synthesizable register files for processor and IP cores," *Integr., VLSI J.*, vol. 39, pp. 131–155, 2006.

[16] A. Raghunatan, S. Dey, and N. K. Jha, "Register transfer level power optimization with emphasis on glitch analysis and reduction," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 18, no. 8, pp. 1114–1131, Aug. 1999.

[17] H. B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI*. Boston, MA: Addison-Wesley, 1990.

[18] N. Magen, A. Kolodny, U. Weiser, and N. Shamir, "Interconnect-power dissipation in a microprocessor," in *Proc. Int. Workshop Syst. Level Interconnect Prediction*, 2004, pp. 7–13.

[19] Q. Wu, M. Pedram, and X. Wu, "Clock-gating and its application to low power design of sequential circuits," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 47, no. 3, pp. 415–420, Mar. 2000.

[20] W. Shen, Y. Cai, X. Hong, and J. Hu, "Activity-aware registers placement for low power gated clock tree construction," in *Proc. ISVLSI*, 2007, pp. 383–388.

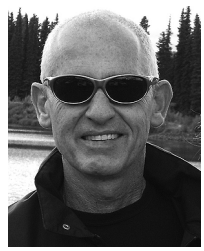
[21] V. Kolmogorov, "Blossom V: A new implementation of a minimum cost perfect matching algorithm," *Math. Prog. Comp.*, vol. 1, no. 1, pp. 43–67, 2009.

[22] C. Chunhong, K. Changjun, and M. Sarrafzadeh, "Activity-sensitive clock tree construction for low power," in *Proc. ISLPED*, 2002, pp. 279–282.

[23] G. Palumbo, F. Pappalardo, and S. Sannella, "Evaluation on power reduction applying gated clock approaches," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2002, pp. IV-85–V-88.

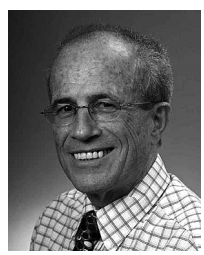
[24] M. R. Garey and D. S. Johnson, *Computers and Intractability*. New York: Freeman, 1979.

[25] S. Wimer and I. Koren, "Optimal flip-flop grouping in VLSI clock gating for maximal power reduction," 2011.



Shmuel Wimer 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 (1999–2009), Sagantec (1997–1999), microCAD (1994–1997), IBM (1985–1994), National Semiconductor (1981–1985), and Israeli Aircraft Industry (IAI) (1978–1981). He is presently 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 areas of interest include VLSI circuits and systems design optimization and combinatorial optimization.



Israel Koren (M'76–SM'87–F'91) is currently a Professor with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst. He has been a consultant to numerous companies including IBM, Analog Devices, Intel, AMD, and National Semiconductors. His research interests include fault-tolerant systems, computer architecture, vlsi yield and reliability, secure cryptographic systems, and computer arithmetic. He publishes extensively and has over 250 publications in refereed journals and conferences. He is the author of the textbook *Computer Arithmetic Algorithms* (2nd Ed., A.K. Peters, 2002) and a co-author of *Fault Tolerant Systems* (Morgan-Kaufman, 2007).

Prof. Koren is an Associate Editor of the *VLSI Design Journal*, and *Sustainable Computing*. He served as General Chair, Program Chair, and Program Committee member for numerous conferences.