Integrated Placement and Skew Optimization for Rotary Clocking

Ganesh Venkataraman, *Student Member, IEEE*, Jiang Hu, *Member, IEEE*, and Frank (Ying) Liu, *Member, IEEE*

***Abstract*—The clock distribution network is a key component of any synchronous VLSI design. High power dissipation and pressure volume temperature-induced variations in clock skew have started playing an increasingly important role in limiting the performance of the clock network. Rotary clocking is a novel technique which employs unterminated rings formed by differential transmission lines to save power and reduce skew variability. Despite its appealing advantages, rotary clocking requires flip-flop locations to match predesigned clock skew on rotary clock rings. This requirement poses a difficult chicken-and-egg problem which prevents its wide application. In this paper, we propose an integrated placement and skew scheduling methodology to break this hurdle, making rotary clocking compatible with practical design flows. A network flow based flip-flop assignment algorithm and a cost-driven skew optimization algorithm are developed. We also present an integer linear programming formulation that minimizes maximum capacitance loaded at any of the rotary rings, thereby maximizing the operating frequency. Experimental results on benchmark circuits show that our method can reduce the tapping cost (measured as the total length of the wire segments connecting the rotary rings to the clock sinks) for rotary clocking by 33%–53%.**

***Index Terms*—Clock distribution, clock skew, low power, rotary clocking, variation.**

I. Introduction

POWER and variation are two important factors that limit the performance of deep submicrometer VLSI circuits. The function of the clock distribution network is to deliver the clock signal from the clock source to the sequential circuit elements (which are referred to as flip-flops or clock sinks). These clock sinks are usually located at various locations in the chip. It is well-known that the clock distribution network dissipates a large amount of power, up to 40% of the entire chip power budget [1]. This power is becoming increasingly significant as frequencies scale well beyond the gigahertz limits. As the feature size keeps shrinking, clock skew becomes increasingly sensitive to manufacturing process variations [2], [3]. Liu *et al.* [3] analyze the impact of process variations on a gigahertz microprocessor. It was observed that interconnect variations alone account for 25% deviation of the clock skew from its nominal value. For proper circuit functionality, the clock skew should be within a certain limit [4] (referred to as the **permissible range**). If the uncertainty in the clock skew is high, there might be a need to run the clock at a lower speed (a higher clock period increases the skew permissible range [4]).

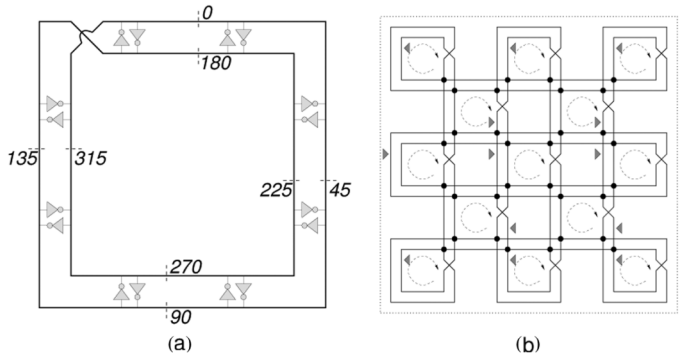

Fig. 1. (a) Rotary clock ring. The numbers indicate relative clock signal phase. (b) Array of 13 rotary clock rings. The small triangles point to the equal-phase points for the 13 rings.

The power dissipation of a clock network is mainly due to frequent charging/discharging of the huge capacitive load and the leakage power through clock buffers. Many efforts have been made to reduce the clock network power by minimizing clock network size [5]–[8]. Another popular technique used for low-power design is clock gating [9], [10]. Gated clocks save power by switching off portions of a circuit that are inactive. Even though these approaches can alleviate the clock network power to some degree, the improvements are intrinsically limited by the fundamental charging/discharging power dissipation characteristics in the conventional clock distribution networks.

Minimizing clock skew variation is another major objective in many works on clock distribution network design [11], [12]. However, many of these methods achieve reduction of skew variation at the expense of increased clock network size and power. In particular, the very effective approach of clock mesh [11] may result in excessive wirelength and power overhead. Conventional clocking structure consists of a clock signal driven by a single clock source (and possibly regenerated by buffers). Accurate control of this clock signal becomes a hard problem since it is propagated over a long distance.

In order to solve the power and the variation problem more effectively, several novel clocking technologies have been developed [13]–[15]. Among them, rotary traveling-wave clock [13] is a promising approach. The basic component of a rotary clock is a pair of cross connected differential transmission line circles, namely a rotary clock ring [as shown in Fig. 1(a)]. A clock signal propagates along the ring without termination so that the energy can be recirculated and the charging/discharging

Manuscript received January 26, 2006; revised September 7, 2006. This work was supported in part by Semiconductor Research Corporation under Contract 2004-TJ-1205.

G. Venkataraman and J. Hu are with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77840 USA (e-mail: ganesh@ece.tamu.edu; jianghu@ece.tamu.edu).

F. Liu is with IBM Austin Research Laboratory, Austin, TX 78758 USA (e-mail: frankliu@us.ibm.com).

Digital Object Identifier 10.1109/TVLSI.2007.893577

power dissipation is greatly reduced. Reference [16] shows that rotary clocks can reduce power dissipation by 70% compared to conventional clock networks. The measurement results from a test chip also showed that a low skew variation of 5.5 ps at 950 MHz can be achieved [13]. Rotary clock can provide uniform clock signal amplitude in contrast to the nonuniform amplitude in standing wave clock [14].

There is one technical hurdle that prevents wide applications of the rotary clock: the clock signal has different phases at different locations on the rotary clock ring. If zero skew design is insisted, the usage of rotary clocks would be very restrictive. Hence, non-zero intentional skew design is a better approach to fully utilize the rotary clock. Unlike the intentional skew design in the conventional clocking technology where no restrictions are imposed on the flip-flop locations, the skew at each flip-flop has to be matched with a specific location at the rotary clock ring. This requirement forms a difficult chicken-and-egg problem: the flip-flop placement depends on skew optimization while it is well known that skew optimization depends on flip-flop locations. This is quite different from traditional intentional skew designs [8] where the placement does not depend on skew optimization.

The goal of this paper is to break this technical hurdle so that rotary clocks can easily be deployed in practice to alleviate the power and variation issues. We make the following contributions in this paper.

- A relaxation technique based on flexible tapping is suggested to break the loop in the chicken-and-egg problem.
- An integrated placement and skew optimization methodology is proposed. This methodology has the advantage that it fits well with the current computer-aided design (CAD) flow and the placers can be used without any change.
- A min-cost network flow algorithm is found to assign flip-flops to the rotary clock rings so that the movement of flip-flops has the least disturbance to traditional placement.
- A pseudo-net technique is introduced to guide flip-flops toward their preferred locations without intrusive disturbance to traditional placement.
- A cost driven skew optimization formulation is developed to reduce the connection cost between flip-flops and their corresponding rotary clock rings.
- We develop an integer linear programming (ILP) formulation to minimize the maximum capacitance loaded at any rotary ring. Since the operating frequency is inversely proportional to the load capacitance, this formulation could be used in delay critical designs. We also detail a fast and effective heuristic used to solve the formulation.

An extended abstract of this paper appeared in [18]. However, [18] does not detail the ILP-based formulation to minimize the load capacitance. The network flow approach is suitable if the design objective is to minimize wirelength and the ILP-based approach works well for designs where operating frequency is more critical.

The remaining portion of this paper is organized as follows. Section II discusses the basic structure of rotary clocks and how it fits with the traditional design flow. Section III details the relaxation technique used to find the tapping point. We provide the basic outline of the proposed flow in Section IV. Sections V and VI discuss the problem formulations used for reducing the total tapping cost (measured as the total length of the wire segments connecting the rotary rings to the clock sinks) and maximum load capacitance, respectively. In Section VII, we discuss the skew optimization technique. The experimental results are presented in Section VIII followed by Section VIIII on conclusions and future work.

II. Rotary Traveling-Wave Clock and Traditional Design Flow

In this section, we discuss the basics of rotary clocking, its advantages, whether a rotary clock can fit into a traditional design flow, and if not, what are the difficulties. The basic component of a rotary clock is a pair of cross connected transmission line circles as shown in Fig. 1(a). In the rotary clock ring, an oscillation can start spontaneously upon any noise event [13]. When the oscillation is established, the square wave signal can travel along the ring without termination. An arbitrary point on the ring can be designated as the reference point with clock signal delay $t = 0$ and clock phase $\phi = 0$. Starting from this reference point, the clock signal travels along the ring and reaches back to the reference point with delay equal to clock period T and phase $\phi = 360$. The numbers in Fig. 1(a) indicate clock signal phases. Clock signal delay t and clock signal phase ϕ can be converted to each other by $(\phi/360) = (t/T) - \lfloor t/T \rfloor$.

The energy loss due to the wire resistance is compensated by the anti-parallel inverters as shown in Fig. 1(a). In addition, these inverter pairs help to achieve phase locking as the phases of the two circles at the same location are always opposite. In order to maintain uniform capacitance distribution along the ring, dummy capacitive load needs to be inserted at places where no flip-flops exist. In chip level designs, multiple rotary clock rings can be connected together to form an array as shown in Fig. 1(b), where the dashed arrows indicate the signal propagation directions and the small triangles indicate equal phase locations for all rings.

A rotary clock has the advantage of both low power dissipation and low skew variation. It consumes less power as the energy is recirculated along the ring as opposed to energy loss during the charging/discharging through transistors in conventional clocking. In the rotary clock ring array [Fig. 1(b)], the phase averaging at the junction points can reduce skew variation remarkably.

In spite of the appealing advantages, the rotary clocking scheme is not compatible with existing design flows. Consider the cross-coupled rings shown in Fig. 1(b). Since at each spot on a rotary clock ring, the clock signal has a distinct phase, a zero clock skew design implies that only one spot on each ring can be utilized. In Fig. 1(b), there are 13 rings and there are only 13 useful spots for a zero clock skew design. Obviously, such usage of rotary clock is very inefficient. In order to fully utilize rotary clock, intentional skew design is a much better choice. Even with intentional clock skews, rotary clocking poses an important problem. A typical intentional skew design flow, which is employed in IBM high performance ASIC designs [8] proceeds in the order of the following stages.

1) Placement. In placement [19], [20], cells are placed in a nonoverlapping manner so that an objective function, such as the total signal net wirelength, congestion, critical path timing, or a combination of them, is minimized.
2) Clock skew optimization. For intentional skew designs, clock signal delay target to each flip-flop is found so as to minimize clock period or maximize timing slack, subject to long path and short path constraints. Since the long path and short path constraints depend heavily on cell and flip-flop locations, placement information is essential in order to perform skew optimization. We will discuss skew optimization in Section VII.
3) Clock distribution network synthesis. In this stage, a clock distribution network layout is generated to approximately deliver intentional skews [8], which correspond to the clock delay targets obtained in skew optimization. Of course, the clock distribution network layout depends on flip-flop locations.

The feasibility of this flow is based on its one-way dependency that each stage relies on the result of the previous stages, but not *vice versa*. More specifically, the placement (with the inclusion of flip-flops), does not depend on skew optimization or clock distribution network synthesis. Likewise, skew optimization does not depend on clock distribution network synthesis.

Rotary clock is usually designed independently. In placement, each flip-flop needs to be placed at a rotary clock ring and the clock phase at its location has to match the clock signal delay target for the flip-flop. This requirement causes a cyclic dependency. The flip-flop placement depends on its clock signal delay target, which is generated by skew optimization. But skew optimization is always dependent on placement. This chicken-and-egg problem has to be solved to enable the application of the rotary clocking technique.

III. Relaxation Via Flexible Tapping

For a complex constrained optimization problem, relaxation is an effective technique to handle troublesome constraints. The difficulty of applying rotary clock can be alleviated if we relax the constraint which requires each flip-flop to be attached exactly on a rotary clock ring. For a flip-flop at an arbitrary location (x_f, y_f), we can always find a tapping point p on a rotary clock ring and deploy a buffer at p to drive the flip-flop through a wire such that clock signal delay t_f at (x_f, y_f) satisfies a prespecified clock delay target $\hat{t}_f$ for the flip-flop. Of course, we do not want the flip-flop to be too far away from the ring. Otherwise, the long wire between the tapping point and the flip-flop may cause significant power and variability degradation that it becomes meaningless to use rotary clock. On the other hand, if a flip-flop is very close to its tapping point on the ring, the buffer can be omitted. The wirelength between the tapping point p and the flip-flop can be counted as a cost to be minimized in placement and skew optimization. The approach of transforming troublesome constraints to cost is very similar to Lagrangian relaxation.

Now, we will show how to find the location of the tapping point so that the delay target $\hat{t}_f$ for the flip-flop at (x_f, y_f) can be satisfied. We illustrate this procedure through an example in Fig. 2. A rotary clock ring is implemented in square shape in

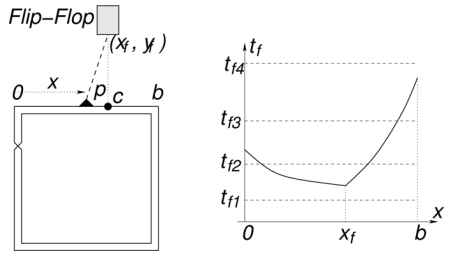

Fig. 2. Tapping point p for a flip-flop can be found by solving the delay satisfying clock signal delay target.

layout and is composed by four inside segments and four outside segments, as shown in Fig. 2. Without loss of generality, we only consider the case when the tapping point is on the top segment. Other cases can be derived in an almost identical manner. We assume that the left end of the segment is set at coordinate $(0, 0)$, the location for the right end, the flip-flop and the tapping point are represented as $(b, 0)$, (x_f, y_f) and $(x, 0)$, respectively. If the delay of the clock signal at $(0, 0)$ is t_0, then the delay at p can be expressed as $t_0 + \rho x$ [13] with ρ being a positive constant. We denote the distance between the tapping point and the flip-flop as l, assuming wire resistance and capacitance per unit length are r and c, respectively. Let $C_{\text{flip-flop}}$ denote the input capacitance of the flip-flop. In order to satisfy the clock signal delay target at the flip-flop, the clock signal delay t_f has to satisfy

$$t_f(x) = t_0 + \rho x + \frac{1}{2} rcl^2 + rlC_{\text{flip-flop}} = \hat{t}_f. \tag{1}$$

Since $l = |x - x_f| + y_f$, l is a function of x. The location of the tapping point can be obtained by solving the previous equation.

Fig. 2 shows an example of curve $t_f(x)$ which is composed by two parabolas joining at $x = x_f$. The function $t_f(x)$ can be decomposed into a quadratic function plus the term of $|x - x_f|$. The quadratic function leads to a single parabola. However, the term of $|x - x_f|$ consists of two pieces of linear parts with a nondifferentiable joint point at $x = x_f$. Therefore, the overall shape of the $t_f(x)$ curve becomes two pieces of parabolas joining at $x = x_f$. Depending on the value of $\hat{t}_f$, there are four cases for solving (1) shown as follows.

- *Case 1:* $\hat{t}_f$ is very small like t_{f1} in Fig. 2. There is no direct solution for this case. This case can be circumvented by reducing t_0 by integer number of clock period time T. Note that such reduction does not affect clock phase. This is equivalent to lowering the curve $t_f(x)$ by multiple T and eventually results in one of the following three cases. Obviously, the number of T for the reduction needs to be minimized.
- *Case 2:* $\hat{t}_f$ is moderately small like t_{f2} in Fig. 2. There are two solutions in the case and the solution with smaller wirelength is selected.
- *Case 3:* $\hat{t}_f$ is at middle level like t_{f3} in Fig. 2. There is a unique solution.
- *Case 4:* $\hat{t}_f$ is large like t_{f4} in Fig. 2. There is no direct solution for this case. However, we can choose $(b, 0)$ as the tapping point and intentionally introduce wire detour between p and the flip-flop so that delay target $\hat{t}_f$ is satisfied. This is almost the same as the wire snaking in clock tree routing [6].

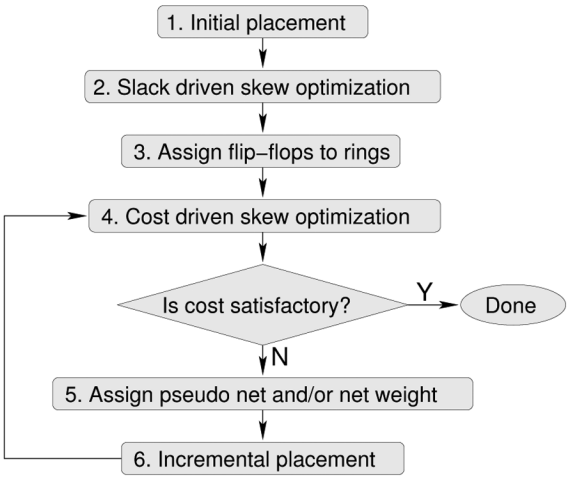

Fig. 3. Proposed methodology flow.

After the tapping points on each of the eight segments are calculated, the one leading to the minimum wirelength is selected as the tapping point for the ring. The actual wirelength for achieving the clock signal delay target is defined as the tapping cost. Note that we could also use a buffer to drive the signal from point p. If needed, (1) can be easily modified to take care of the buffer delay.

It may be noted that in a rotary ring, a phase, and its complementary phase (two phases are complementary to each other if they are 180° apart, for example, 90 and 270 are complementary phases) are available at points that are physically close to each other. Let ϕ_1 and ϕ_2 denote the two complementary phases available at points p_1 and p_2 in the rotary ring. If two flip-flops F_1 and F_2 are both assigned to p_1, then one can connect F_1 to point p_1 (with phase ϕ_1), F_2 to point p_2 (with phase ϕ_2) and assign opposite polarities to F_1 and F_2 (that is we can make one positive edge triggered and another one as negative edge triggered).

IV. Proposed Methodology

The flexible tapping technique breaks the cyclic dependency between the placement and skew optimization. It also facilitates a new methodology flow as shown in Fig. 3, in which skew awareness for the placement is achieved indirectly through a pseudo-net technique. By doing so, the traditional placement methods can be used without any change. This is an appealing feature because placement is a much more complicated problem than skew optimization.

The first two stages are almost the same as the traditional flow. The initial placement can be implemented with any existing placement method [19], [20] without special considerations of flip-flop locations or their skews. In stage 2, skew optimization [4] is performed based on the placement of stage 1 to maximize the timing slack. The details of this part will be described in Section VII.

In stage 3, each flip-flop is assigned to a ring of the rotary clock ring array. This assignment only establishes an association between a particular flip-flop to a ring and does not change the flip-flop location. When a flip-flop is assigned to a specific ring, the corresponding tapping cost can be computed according to Section III. Depending upon the design objective, we propose the following two different techniques for flip-flop assignment.

1) A network flow-based assignment algorithm which aims at minimizing the total tapping length subject to capacity constraints. This technique is detailed in Section V.
2) An ILP-based approach that aims at minimizing the maximum capacitance loaded at any of the rotary rings. This technique (detailed in Section VI) can be used in high-speed designs.

After each flip-flop is assigned to a ring, another skew optimization can be performed to reduce the tapping cost. This is somewhat different from traditional skew optimization and will be discussed in details in Section VII. After stage 4, the overall cost is evaluated as a weighted sum of total tapping cost and traditional placement cost, which is usually total signal wirelength. If the overall cost is sufficiently small, this flow is completed. Otherwise, the flow proceeds to stage 5.

Since the initial placement is based on the traditional objectives such as signal wirelength and ignores tapping cost, it is quite likely that the tapping cost is very high when we enter stage 5 for the first time. In order to reduce tapping cost, we insert a pseudo net between each flip-flop and its ring. The flip-flops can be pulled toward their associated rings in the placement stage 6. Since stage 6 is an incremental placement, it normally runs considerably faster than the initial placement. Of course, the incremental placement is preferred to be a stable one [19], i.e., small changes on the netlist should not cause dramatic change on the placement result.

V. Flip-Flop Assignment to Minimize Tapping Cost

In stage 3 of the previous flow, each flip-flop needs to be assigned to a ring in the rotary clock ring array. It is required that the assignment minimizes the tapping cost defined in Section III. We denote the tapping cost as $c_{i,j}$, when a flip-flop i is assigned to ring j. Each ring j has limited space and can accommodate no more than U_j flip-flops. We introduce a decision variable $x_{i,j}$: if flip-flop i is assigned to ring j, $x_{i,j} = 1$, otherwise $x_{i,j} = 0$. The flip-flop assignment problem can then be formulated as the following 0–1 programming problem:

$$\begin{aligned} \text{Minimize} \quad & \sum_{i,j} c_{i,j} x_{i,j} \\ \text{Subject to} \quad & \sum_{j} x_{i,j} = 1 \quad \forall\, i \\ & \sum_{i} x_{i,j} \le U_j \quad \forall\, j \\ & x_{i,j} \in \{0,1\} \quad \forall\, i,j. \end{aligned}$$

This assignment problem can be solved using the min-cost network flow model [22] as shown in Fig. 4. The vertices in the network include a column of flip-flop vertices, a column of ring vertices, a source vertex and a target vertex. There is an arc from a flip-flop vertex to a particular ring vertex only if the corresponding flip-flop is considered to be a potential candidate of the ring. If a flip-flop and a ring are too far away from each other, it is not necessary to insert an arc between them. Each arc between a flip-flop vertex i and a ring vertex j has a cost of $c_{i,j}$. The rest arcs have zero cost. Each arc from a ring vertex j to the

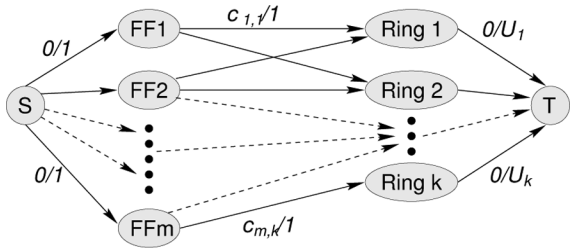

Fig. 4. Min-cost network flow model for flip-flop assignment. Each arc is associated with cost/capacity.

target vertex has a capacity of U_j. The other arcs have a capacity of 1. It is well known that this min-cost network flow problem can be solved optimally in polynomial time [22].

VI. Flip-Flop Assignment to Minimize Load Capacitance

In this section, we present an alternative formulation that can be used in place of the one discussed in Section V. The frequency of operation of the rotary clock is given by [11]

$$f_{\text{osc}} = \frac{1}{2\sqrt{L_{\text{total}} C_{\text{total}}}} \tag{2}$$

where C_{total} consists of two components: the ring capacitance and the capacitance of the load that the ring drives. The latter shall hence forth be referred to as load capacitance. Since the ring capacitance for a given ring dimension is a constant, we can minimize C_{total} by minimizing the load capacitance. If the design objective is to run the maximum possible frequency, then the load capacitance at the rings is a more relevant optimization objective. The network flow formulation discussed in Section V attempts to distribute the flip-flops uniformly across the rotary rings while optimizing the tapping length. While this is ideal if the design objective is to minimize the tapping length, it is not a direct measure of the load capacitance.

We present an alternative formulation which takes into account the load capacitance at the rotary rings. Let C_p^{ij} denote the load capacitance of a flip-flop i when it is assigned to a ring j. This capacitance includes both the interconnect capacitance as well as the flip-flop input capacitance (or buffer capacitance). Let x_{ij} denote the indicator variable that is set to one if flip-flop i is assigned to ring j and zero otherwise. Then the formulation to minimize the maximum load capacitance can be stated as follows:

$$\begin{aligned} \min \max \quad & \sum C_p^{ij} x_{ij} \\ & \sum_{\forall\, j} x_{ij} = 1 \quad \forall\, i \\ & x_{ij} \in \{0,1\}. \end{aligned} \tag{3}$$

The following key observations can be made about formulation (3).

- The cost $\left(C_p^{ij}\right)$ is a direct measure of the load capacitance in contrast to the cost used in formulation (2)
- The formulation is an integer programming (ILP) problem. ILP in general is NP-hard and hence there is a need to employ fast and effective heuristics.

Procedure: Greedy Rounding
Input: $X^{lp} = \{x_{ij}^{lp}\}$: Solution for the relaxed LP I - set of flip-flops, J - set of rings
Output: x_{ij}^{ilp}: Solution to the ILP
1. For each $i \in I$ do 1.1 If $x_{ij_1}^{lp} = 1$ for some $j_1 \in J$, $x_{ij_1}^{ilp} = 1$ and $x_{ij}^{ilp} = 0\ \forall j \neq j_1$ 1.2 Else Find $j_{max} \in J$ such that $x_{ij_{max}}^{lp} \geq x_{ij}^{lp} \forall j \in J$ $x_{ij_{max}}^{ilp} = 1$ and $x_{ij}^{ilp} = 0\ \forall j \neq j_1$ 2. Return $X^{ilp} = \{x_{ij}^{ilp}\}$

Fig. 5. Greedy rounding algorithm.

A. Solution by LP-Relaxation

LP-relaxation has been used to solve a certain class of combinatorial optimization problems like the set cover [25]. The basic steps involved in LP-relaxation are as follows.

1) Relax the integer constraints (usually 0–1 constraints on decision variables) to continuous constraints. For example, an integer constraint $x_{ij} \in \{0,1\}$ would be replaced by $0 \leq x_{ij} \leq 1$.
2) Solve the resulting LP. For a minimization problem, the optimal solution to the LP gives a lower bound on the optimal solution to the ILP. This is due to the fact that any feasible solution to the ILP is also a feasible solution to the LP.
3) Now, round the LP solution to the get an integer solution.

The key to any LP-relaxation procedure is the rounding process. The rounding process should be such that 1) feasibility of the ILP is maintained and 2) the deviation of the objective function is minimized. With the two previous restrictions in mind, we employ a rounding procedure which shall henceforth be referred to as **greedy rounding**.

Let I denote the set of flip-flops, J the set of rotary rings, $X^{\text{lp}} = \left\{x_{ij}^{\text{lp}}\right\}$ the solution obtained from the LP-relaxation and $X^{\text{ilp}} = \left\{x_{ij}^{\text{ilp}}\right\}$ the ILP solution obtained on LP-rounding. For feasibility, we need to make sure that each flip-flop is assigned to exactly one ring. This is done by ensuring upon rounding x_{ij}^{ilp} is set to exactly one $j \in J$.

The greedy rounding procedure is described in Fig. 5. If the LP solution also happens to be integral, we retain it. If not, each flip-flop i is assigned to the ring j whose corresponding assignment variable x_{ij}^{lp} is maximum. Hence, our procedure is referred to as greedy rounding. The complexity of the rounding procedure is linear in the number of flip-flops and the number of rings. Since the number of rings is much smaller than the number of flip-flops and the solution to LP generally runs in linear time (in practice), the overall procedure of greedy rounding is linear in the problem size.

Let OPT(LP) denote the optimum solution of the LP-relaxation and SOLN(ILP) denote the solution of obtained after solving the ILP (by any procedure). Then, we define *integrality gap* IG as

$$\text{IG} = \frac{\text{SOLN(ILP)}}{\text{OPT(LP)}}. \tag{4}$$

TABLE I
IG OF GREEDY ROUNDING AND ILP SOLVER

Circuit	Greedy Rounding		ILP-Solver	
	IG	CPU(s)	IG	CPU
s9234	1.32	0.25	1.83	$>$ 10hrs
s5378	1.57	0.31	22.05	$>$ 10hrs
s15850	1.32	2.0	–	$>$ 10hrs
s38417	1.23	6.10	–	$>$ 10hrs
s35932	1.63	13.10	–	$>$ 10hrs

The following observations can be made about IG.

1) IG ≥ 1, since the LP solution is always lower than the ILP solution.
2) ILP, in general, is NP-complete. But, whether the ILP described in (3) can be solved optimally in polynomial time remains open. Since we currently do not know the optimal solution, IG acts as a good base case for comparison.
3) The value of IG should not be interpreted in an absolute sense. For example, IG $= 1.40$ does NOT mean that the solution of the heuristic is 40% away from optimal value.

In order to compare the efficacy of the proposed heuristic, we ran the same formulation using a public domain ILP solver [26]. Table I presents the comparison between greedy rounding and the ILP solver. The comparisons are made in terms of both runtime and quality of solution (measured by the IG). While greedy rounding produced a solution within a few seconds, the ILP solver took several hours. We bounded the simulation time for the ILP solver to 10 hr and report the best solution that it produced within this time. For three of the test cases, the ILP solver did not produce a feasible solution. For the remaining two, the greedy rounding outperformed the ILP solver by a significant margin. This is not unexpected since the ILP solver is generic in nature, whereas our rounding procedure exploits the problem structure.

It was observed that the ILP formulation reduces the maximum capacitance significantly. But this reduction comes at the expense of increased signal and clock wire length. The details are discussed in Section VIII.

VII. SKEW OPTIMIZATION

Skew optimization is an important part of the proposed flow. The skew optimization in stage 2 is the same as the max-slack version of the traditional method [4] and we summarize it here for completeness. If two flip-flops i and j are said to be *sequentially adjacent* if they have only combinational logic between them. Sequential adjacency is denoted by $i \longmapsto j$. Let $D^{ij}_{\max}\left(D^{ij}_{\min}\right)$ denote the maximum (minimum) combinational logic delay between i and j. Let $\hat{t}_i$ denote the clock signal arrival time at i, $t_{\text{setup}}(t_{\text{hold}})$ the setup (hold) time, and T the clock period. Then the max-slack version of skew optimization can be formulated as [4]

$$\begin{aligned} \text{Maximize} \quad & M && (5)\\ \text{Subject to} \quad & \hat{t}_i - \hat{t}_j + M \leq T - D^{ij}_{\max} - t_{\text{setup}} \quad i \longmapsto j && (6)\\ & \hat{t}_i - \hat{t}_j \geq M + t_{\text{hold}} - D^{ij}_{\min} \quad i \longmapsto j && (7) \end{aligned}$$

where M is the slack. Inequalities (6) and (7) are referred to as the long path constraint and short path constraints, respectively. It has been shown that this problem can be solved using linear programming [4] or graph-based algorithms [23], [24].

We propose a cost driven method so that skew optimization can be leveraged to assist placement on reducing the total tapping cost. The computation of the tapping cost (Section III) is based on a known clock signal delay target while the delay target is a decision variable in skew optimization. Therefore, we try to minimize the tapping cost indirectly by finding clock signal delay targets such that the tapping point can be moved to the flip-flop as close as possible. For example, in Fig. 2, if the delay target of the flip-flop can result in tapping point at c, then the tapping cost is minimized. In other words, if the actual delay to a flip-flop i is t_i through tapping point at c, we try to make the delay target $\hat{t}_i$ to be as close to t_i as possible.

For flip-flop i, we first find the closest point c on its ring and the distance between i and c, which is the shortest distance between i and the ring, is denoted as l_i. The delay t_i at i through tapping point at c depends on the reference clock signal delay on the rotary clock rings. We can arbitrarily choose a set of equal phase points for all the rotary clock rings as reference points like the small triangular spots in Fig. 1(b). Let the clock signal delay at the reference points be t_{ref}. If the delay from the reference point to c is $t_{\text{ref},c}$, then the clock signal delay at c is $t_c = t_{\text{ref}} + t_{\text{ref},c}$. The delay from c to i is $t_{c,i} = (1/2)rcl_i^2 + rl_iC_{\text{flip-flop}}$, which is very similar to (1). Hence, the clock signal delay at i is $t_i = t_c + t_{c,i}$, If the difference $|t_i - \hat{t}_i|$ is minimized, there is a good chance that the tapping point is the closest to c and the flip-flop. When a flip-flop i is far from its ring, i.e., $t_{c,i}$ is large, it is especially desired that the tapping point is closest to c. Therefore, $|t_i - \hat{t}_i| + t_{c,i}$ is minimized in the cost driven skew optimization

$$\begin{aligned} \text{Minimize} \quad & \Delta \\ \text{Subject to} \quad & \hat{t}_i - \hat{t}_j + M \leq T - D^{ij}_{\max} - t_{\text{setup}} \quad i \longmapsto j \\ & \hat{t}_i - \hat{t}_j \geq M + t_{\text{hold}} - D^{ij}_{\min} \quad i \longmapsto j \\ & t_{\text{ref}} + t_{\text{ref},c} + 2t_{c,i} - \hat{t}_i \leq \Delta \quad \forall\, i \\ & \hat{t}_i - t_{\text{ref}} - t_{\text{ref},c} \leq \Delta \quad \forall\, i \end{aligned}$$

where Δ is the maximum difference and M is a prespecified slack. The constraint from the two inequalities in this formulation is equivalent to $|t_i - \hat{t}_i| + t_{c,i} \leq \Delta$. Obviously, this problem can be solved through linear programming [4].

Alternatively, the skew optimization problem can be formulated to minimize a weighted sum of the differences as follows:

$$\begin{aligned} \text{Minimize} \quad & \sum_{\forall\, i} w_i\delta_i \\ \text{Subject to} \quad & \hat{t}_i - \hat{t}_j + M \leq T - D^{ij}_{\max} - t_{\text{setup}} \quad i \longmapsto j \\ & \hat{t}_i - \hat{t}_j \geq M + t_{\text{hold}} - D^{ij}_{\min} \quad i \longmapsto j \\ & t_i - \hat{t}_i \leq \delta_i \quad \forall\, i \\ & \hat{t}_i - t_i \leq \delta_i \quad \forall\, i \end{aligned}$$

where δ_i is the difference for flip-flop i and w_i is its weighting factor. A natural choice of the weighting factors is to let $w_i = l_i$, as we wish to focus our effort on those flip-flops far away from their rings. Again, this problem can be solved directly through linear programming [4].

TABLE II
TEST CASES. PL IS THE AVERAGE SOURCE-SINK PATH LENGTH IN CONVENTIONAL CLOCK TREES [5], [7]

Circuit	#Cells	#Flip-flops	#Nets	PL(μm)	# Rings
s9234	1510	135	1471	2471	16
s5378	1112	164	1063	2718	25
s15850	3549	566	3462	5175	36
s38417	11651	1463	11545	8261	49
s35932	17005	1728	16685	8290	49

VIII. EXPERIMENTAL RESULTS

The proposed methodology and algorithms are tested on the ISCAS89 benchmark suite. The benchmark characteristics are summarized in Table II. The first four columns indicate the circuit name, number of standard cells, number of flip-flops, and the number of nets, respectively. In the fifth column of Table II, we report average source-sink path length in conventional clock trees [5], [7] for reference. The number of rotary rings for each test case is indicated in the final column. The rotary clock ring arrays are generated as in [13]. The operating frequency was set at 1 GHz. The circuits are synthesized using SIS [28]. The main algorithms are implemented in $C++$. The initial placement as well as the incremental placement are obtained from an academic placement tool mPL [20], [29]. Soplex was used to solve the LP-relaxation problem [27]. All experiments are performed on a Pentium-4 workstation running Linux operating system with 1-GB RAM. The interconnect parameters are obtained from *bptm*.[1] The skew permissible ranges can be computed using any static timing analysis tool as detailed in [4]. We used the Elmore delay model [21] in our static timing analyzer. However, our techniques are generic and can be extended to a more accurate timing analysis tool without any changes in the underlying skew optimization algorithms or the overall flow.

To the best of our knowledge, there is no published work on placement and skew optimization for rotary clocking. There are the following three issues that we care about.

1) Each flip-flop needs to be close to the ring it associated with so that the off-ring variation effect is negligible
2) Moving flip-flops toward their associated rings should not degrade signal wirelength significantly and the total wirelength including tapping wirelength needs to be minimized.
3) If the design objective is speed, then the maximum load capacitance at any of the rotary ring should be minimized.

Table III gives the results for the base case. These results (base case) are obtained by running the flip-flop assignment algorithm using network flow at stage 3 of our overall flow (indicated in Fig. 3). In all the tables shown, AFD denotes the average flip-flop distance and WL is the wire length. All wire lengths are reported in micrometers, capacitances are reported in picofarads, and power in milliwatts. Table III presents the AFD, tapping wirelength, signal wirelength, total wirelength, power dissipation in the clock net, power dissipation in the signal net, and total power dissipation for the base case. It can be seen that the average flip-flop distance is significantly smaller than the average source-sink path length in conventional clock trees [5], [7] (as shown in the right-most column of Table II). The power dissipation is obtained by the technique(s) detailed in the following. The power dissipation in the clock net includes the dynamic power dissipated in the tapping wires from the rotary ring as well as the power dissipated in the flip-flops. The power dissipated in the signal length includes the power dissipated in the interconnect, logic gates as well the buffers. The power dissipation is measured using the following formula:

$$P_{\text{dynamic}} = \frac{1}{2}\alpha V_{\text{dd}}^2 f_{\text{clk}} C_{\text{load}}. \tag{8}$$

In the previous equation, α denotes the switching activity, f_{clk} the clock frequency, C_{load} the total capacitive load, and V_{dd} the supply voltage. For the clock net, α is set to 1. Estimating α for signal net is a hard problem and setting it to 0.15 usually gives a reasonable approximation [30]. For clock nets, we know the actual interconnect capacitance and the flip-flop capacitance accurately and hence estimating the power is straightforward. The capacitance in the signal net consists of three components: 1) the interconnect capacitance; 2) the input capacitance of logic gates; and 3) the input capacitance of the buffers used in the signal net. The first two capacitances are easy to estimate since we have a placed/mapped design. To estimate the number of buffers in the signal net, we use the technique detailed in [31]. The previous technique estimates the buffer delay (while estimating the number of buffers inserted) at an early stage (floorplan) with a reasonable accuracy. Thus, we estimate the dynamic power dissipated at the clock and signal net and sum them up to get the total power. The total leakage power in a circuit can be approximated to [30]

$$P_{\text{leakage}} = V_{\text{dd}} I_{\text{off}} (S + N_F S_F). \tag{9}$$

In the previous equation I_{off} denotes the unit leakage current, N_F the total number of flip-flops, S the total inverter size, and S_F the gate size of one flip-flop. Since our methodology does not change the gate sizes in the mapped design, the leakage power remains largely unaffected and so we concentrate on the dynamic power dissipation alone.

Table IV gives the results on running stages 4–6 in our overall flow along with the improvements from stage 3 (as reported in Table III). The data shows that the iterations of stage 4–6 can reduce tapping wirelength by 37%–52% with only a 1.3%–4.06% penalty on signal wirelength increase. In fact, the total wirelength is reduced by 2.6%–6.0%. After the iterations of stage 4–6, the average distance has reduced to the range of 100–200μm, which is significantly smaller than the stub length limit indicated in [13]. The CPU time is reported at the right-most column of Table IV. As one can see, most of runtime is dominated by the placer. Our method converges within five iterations for all these circuits.

Table V presents the results for maximum load capacitance for the network flow and ILP formulations. We compare the AFD, maximum cap, and total signal wirelength of the two techniques. We observe that ILP-based formulation reduces the maximum load capacitance by 25.7%–48.33%. However, it increases the AFD (by 11.32%–30.82%) and total wire length (by 0.15%–7.09%). This is expected since the formulations target a different objective. Hence, the designer can choose between

[1][Online]. Available: http://www.eas.asu.edu/~ptm/

TABLE III
EXPERIMENTAL RESULTS FOR THE BASE CASE, WIRELENGTH IN MICROMETERS, POWER IN MILLIWATTS

	AFD	Tap. WL	Signal WL	Tot. WL	Clock Power	Signal Power	Tot. Power	CPU(s)
s9234	285.6	38550	244485	283035	5.06	6.72	11.78	70
s5378	194.1	31839	260931	292770	4.67	6.84	11.51	158.1
s15850	266.6	150907	643336	794243	20.16	18.07	38.23	399
s38417	383.5	559586	1634920	2194506	68.8	48.35	117.15	410
s35932	340.8	588823	1735820	2324643	74.17	57.88	132.05	423

TABLE IV
EXPERIMENTAL RESULTS FOR NETWORK FLOW BASED OPTIMIZATION, WIRELENGTH IN MICROMETERS

	AFD	Tap. WL		Signal WL		Tot. WL		CPU(s)	
Circuit		Final	Imp	Final	Imp	Final	Imp	Stg 2-5	mPL
s9234	136.3	18395	52.28%	247797	-1.35%	266192	5.95%	59	283
s5378	124.51	20419	35.87%	263878	-1.13%	284297	2.89%	25	439
s15850	168	95136	36.96%	664534	-3.3%	759670	4.35%	186	995
s38417	222.9	326136	41.72%	1701352	-4.0%	202744	7.61%	192	930
s35932	223.12	385555	34.52%	1799431	-3.67%	2184986	6.0%	195	1153

TABLE V
COMPARISON OF NETWORK FLOW AND ILP FORMULATIONS, CAP IN PICOFARADS

	Network Flow	ILP Formation						
Circuit	Cap	AFD	Imp	Cap	Imp	Tot. WL	Imp	CPU(s)
s9234	0.49	178.25	-30.82%	0.33	32.65%	273633	-2.8%	0.25
s5378	0.39	138.6	-11.32%	0.29	25.64%	284297	-0.15%	0.31
s15850	1.23	205.8	-22.5%	0.70	43.1%	759670	-1.94%	2.0
s38417	2.52	274.2	-23.0%	1.34	46.83%	2027488	-6.59%	6.10
s35932	1.8	276.54	23.9%	0.93	48.33%	1927039	-7.09%	13.1

TABLE VI
POWER DISSIPATION RESULTS (MILLIWATTS) FOR NETWORK FLOW AND ILP FORMULATIONS

	Network Flow Formulation						ILP Formation					
Circuit	Clock	Imp	Signal	Imp	Total	Imp	Clock	Imp	Signal	Imp	Total	Imp
s9234	3.08	39.13%	6.78	-0.89%	9.86	16.3%	3.63	28.26%	6.81	-1.34%	10.44	11.38%
s5378	3.55	23.98%	6.89	-0.73%	10.44	9.3%	3.79	18.84%	6.84	0%	10.63	7.65%
s15850	14.67	27.23%	18.45	-2.1%	33.12	13.37%	16.77	16.82%	18.06	0.06%	34.83	8.89%
s38417	45.82	33.4%	49.56	-2.5%	95.38	18.58%	53.21	22.66%	48.19	0.33%	101.4	13.44%
s35932	54.15	26.99%	59.05	-2.02%	113.2	14.27%	63.24	14.74%	52.64	9.05%	115.88	12.25%
Ave	29.55	30.15%	28.15	-1.65%	52.4	14.36%	28.13	20.26%	26.51	1.62%	54.64	10.72%

the two formulations depending upon the optimization objective. The runtime for the ILP is low as explained in Section VI (Table I) and repeated here for the sake of completeness.

Table VI gives the power dissipation results (in millwatts) for network flow and the ILP formulations. The results (clock, signal, and total power) are then compared with those obtained for the base case (shown in Table III) and the percentage improvement is also reported. The network flow algorithm gives an average power improvement of 14.36%, whereas the corresponding improvement for the ILP formulation is 10.72%. This is expected since the network flow algorithm tries to minimize the total tapping length which in-turn minimizes the clock net length and the power dissipation. It may be noted that the trend in power dissipation follows that of wirelength reduction for both clock and signal nets (as expected).

It is often difficult to compare two techniques that have different optimization objectives. In such cases, it is better to come up with one objective that can be used for comparison. A classic example of such a case is the power-delay product (PDP). In circuit optimization (especially gate sizing), one could tradeoff power for delay improvement and *vice versa*. Hence, PDP is used a metric to compare different techniques. In our methodology, we tradeoff wirelength and maximum load capacitance and, hence, we introduce the metric wirelength-capacitance product (WCP). This has obvious parallels with the PDP since wirelength is directly related to power dissipation and maximum load capacitance is related to delay. Table VII presents the results for both the techniques in terms of WCP. WCP is reported in micrometer picofarads. It may be observed that the ILP formulation results in much better WCP.

TABLE VII
WIRELENGTH CAPACITANCE PRODUCT COMPARISON

Circuit	Network Flow, WCP	ILP Formation, WCP	Imp
s9234	130434	90298.9	30.77%
s5378	110717.6	82446.1	25.53%
s15850	934394.1	542083.5	41.99%
s38417	5109269.8	2895290.9	43.32%
s35932	3238975.8	1792146.3	44.67%

IX. CONCLUSION

In this paper, we propose an integrated placement and skew optimization method for a novel clocking technique: rotary traveling-wave clock, which is superior to the conventional clocking on both power dissipation and tolerance to variations. By utilizing intentional skew management and flip-flop clustering, both the phase and physical location constraints of the

rotary clocking can be satisfied. We also propose techniques to minimize the maximum load capacitance seen by the rotary rings. To the best of our knowledge, this is the first placement and skew optimization work for rotary clocking. Experimental results indicate that our techniques can reduce the tapping cost by **37%–52%** with **2.6%–6%** reduction in total wirelength. Since our method enables a concurrent clock network and placement design, it is potentially useful for other clocking methodologies like [17] as well.

In our current methodology, we connect the ring directly to the flip-flops assigned to them. However, this could be improved by creating local trees that connect the ring location to a set of flip-flops. In such a construction, care should be taken to take care of the skew permissible ranges of the flip-flop pairs. Such a scheme could lead to potential benefits in wirelength and power dissipation. Further, our formulations take the number of rotary rings as part of the input. A better approach would be to integrate the number of rings as a variable in our methodology. This could lead to better optimization as it increases the solution space. We wish to investigate both these aspects in our future work.

REFERENCES

[1] D. E. Duarte, N. Vijaykrishnan, and M. J. Irwin, "A clock power model to evaluate impact of architectural and technology optimizations," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 10, no. 6, pp. 844–855, Dec. 2002.

[2] S. Zanella, A. Nardi, A. Neviani, M. Quarantelli, S. Saxena, and C. Guardiani, "Analysis of the impact of process variations on clock skew," *IEEE Trans. Semicond. Manuf.*, vol. 13, no. 4, pp. 401–407, Nov. 2000.

[3] Y. Liu, S. R. Nassif, L. T. Pileggi, and A. J. Strojwas, "Impact of interconnect variations on the clock skew of a gigahertz microprocessor," in *Proc. Des. Autom. Conf.*, 2000, pp. 168–171.

[4] J. P. Fishburn, "Clock skew optimization," *IEEE Trans. Comput.*, vol. 39, pp. 945–951, Jul. 1990.

[5] T.-H. Chao, Y.-C. Hsu, J.-M. Ho, K. D. Boese, and A. B. Kahng, "Zero skew clock routing with minimum wirelength," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 39, no. 11, pp. 799–814, Nov. 1992.

[6] R.-S. Tsay, "An exact zero-skew clock routing algorithm," *IEEE Trans. Comput.-Aided Des.*, vol. 12, no. 2, pp. 242–249, Feb. 1993.

[7] M. Edahiro, "A clustering-based optimization algorithm in zero-skew routings," in *Proc. ACM/IEEE Design Autom. Conf.*, 1993, pp. 612–616.

[8] S. Held, B. Korte, J. Maßerg, M. Ringe, and J. Vygen, "Clock scheduling and clock tree construction for high performance ASICs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2003, pp. 232–239.

[9] A. H. Farrahi, C. Chen, A. Srivastava, G. Tellez, and M. Sarrafzadeh, "Activity-driven clock design," *IEEE Trans. Comput.-Aided Des.*, vol. 20, no. 6, pp. 705–714, Jun. 2001.

[10] J. Oh and M. Pedram, "Gated clock routing for low-power microprocessor design," *IEEE Trans. Comput.-Aided Des.*, vol. 20, no. 6, pp. 715–722, Jun. 2001.

[11] P. J. Restle, T. G. McNamara, D. A. Webber, P. J. Camporese, K. F. Eng, K. A. Jenkins, D. H. Allen, M. J. Rohn, M. P. Quaranta, D. W. Boerstler, C. J. Alpert, C. A. Carter, R. N. Bailey, J. G. Petrovick, B. L. Krauter, and B. D. McCredie, "A clock distribution network for microprocessors," *IEEE J. Solid-State Circuits*, vol. 36, no. 5, pp. 792–799, May 2001.

[12] N. Bindal, T. Kelly, N. Velastegui, and K. L. Wong, "Scalable sub-10 ps skew global clock distribution for a 90 nm multi-GHz IA microprocessor," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2003, pp. 346–355.

[13] J. Wood, T. C. Edwards, and S. Lipa, "Rotary traveling-wave oscillator arrays: A new clock technology," *IEEE J. Solid-State Circuits*, vol. 36, no. 11, pp. 1654–1665, Nov. 2001.

[14] R. O'Mahony, C. P. Yue, M. A. Horowitz, and S. S. Wong, "A 10-GHz global clock distribution using coupled standing-wave oscillators," *IEEE J. Solid-State Circuits*, vol. 38, no. 11, pp. 1813–1820, Nov. 2003.

[15] S. C. Chan, K. L. Shepard, and P. J. Restle, "Uniform-phase uniform-amplitude resonant-load global clock distributions," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 102–109, Jan. 2005.

[16] Z. Yu and X. Liu, "Power analysis of rotary clock," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, 2005, pp. 150–155.

[17] Y. Lu, C. N. Sze, X. Hong, Q. Zhou, Y. Cai, L. Huang, and J. Hu, "Navigating registers in placement for clock network minimization," in *Proc. ACM/IEEE Des. Autom. Conf.*, 2005, pp. 176–181.

[18] G. Venkataraman, J. Hu, F. Liu, and C.-N. Sze, "Integrated placement and skew optimization for rotary clocking," in *Proc. Des. Autom. Test Eur.*, 2006, pp. 756–761.

[19] H. Eisenmann and F. M. Johannes, "Generic global placement and floorplanning," in *Proc. ACM/IEEE Des. Autom. Conf.*, 1998, pp. 269–274.

[20] T. F. Chan, J. Cong, J. Shinnerl, and K. Sze, "An enhanced multilevel algorithm for circuit placement," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2003, pp. 299–306.

[21] J. Rubinstein, P. Penfield, and M. Horowitz, "Signal delay in RC tree networks," *IEEE Trans. Comput.-Aided Des.*, vol. CAD-2, no. 3, pp. 202–211, Jul. 1983.

[22] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Upper Saddle River, NJ: Prentice-Hall, 1993.

[23] R. B. Deokar and S. S. Sapatnekar, "A graph-theoretic approach to clock skew optimization," in *Proc. IEEE Int. Symp. Circuits Syst.*, 1994, pp. 1.407–1.410.

[24] C. Albrecht, B. Korte, J. Schietke, and J. Vygen, "Cycle time and slack optimization for VLSI-chips," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 1999, pp. 232–238.

[25] V. Vazirani, *Approximation Algorithms*. Berlin, Germany: Springer-Verlag, 2001.

[26] Free Software Foundation, Boston, MA, "GNU linear programming kit," GLPK [Online]. Available: http://www.gnu.org/software/glpk/glpk.html

[27] R. Wunderling, "Paralleler und objektorientierter simplex-algorithmus," ZIB, Berlin, Germany, Tech. Rep. TR 96–09, 1996.

[28] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, *SIS: A System for Sequential Circuit Synthesis*. Berkeley: Univ. California, 1992.

[29] "CPMO-constrained placement by multilevel optimization," Comput. Sci. Dept., State California Univ., Los Angeles, (2005). [Online]. Available: http://ballade.cs.ucla.edu/cpmo/

[30] W. Liao and L. He, "Full-chip interconnect power estimation and simulation considering concurrent repeater and flip-flop insertion," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2003, pp. 574–580.

[31] C. J. Alpert, J. Hu, S. S. Sapatnekar, and C. N. Sze, "Accurate estimation of global buffer delay within a floorplan," *IEEE Trans. Comput.-Aided Des.*, vol. 25, no. 6, pp. 1140–1145, Jun. 2006.

Ganesh Venkataraman (S'05) received the B.E. degree (honors) in electrical and electronics engineering and the M.Sc. degree (honors) in physics from the Birla Institute of Technology and Science, Pilani, India, both in 2000, the M.S. in electrical and computer engineering, from University of Iowa, Iowa City, in 2003. He is currently pursuing the Ph.D. degree in computer engineering at Texas A&M University, College Station.

Between 2000 and 2001, he worked as a Design Engineer at Cypress Semiconductor, Bangalore, India. His research interests include VLSI physical design, variation tolerant clock distribution, low power, graph theory, and combinatorial optimization.

Mr. Venkataraman was a recipient of the Graduate Merit Fellowship from Texas A&M University.

Jiang Hu (S'98–M'01) received the B.S. degree in optical engineering from Zhejiang University, Zhejiang, China, in 1990, and the M.S. degree in physics and the Ph.D. degree in electrical engineering from the University of Minnesota, Minneapolis, in 1997 and 2001, respectively.

Currently, he is an Assistant Professor in the Department of Electrical and Computer Engineering, Texas A&M University, College Station. From 2000 to 2002, he was with IBM Microelectronics Division, Austin, TX, where he received six U.S. patents. His research interests include computer-aided design for VLSI circuits, especially on physical design for robustness and manufacturability.

Dr. Hu is a recipient of an IBM First Plateau Invention Achievement Award from IBM Microelectronics Division and a Best Paper Award from the ACM/IEEE Design Automation Conference in 2001. He is an Associate Editor of IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS. He has served as technical program committee member of DAC, ICCAD, ISPD, DATE, ISQED, ICCD, and ISCAS.

Frank (Ying) Liu (S'96–M'00) received the M.S. degree in applied mathematics from the University of Minnesota, Minneapolis, in 1996 and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 1999.

Since 1999, he has been with IBM Austin Research Laboratory, Austin, TX, where he is currently a Research Staff Member. His research interests include circuit analysis, model order reduction, numerical analysis, as well as variability characterization and modeling. He has authored and coauthored over 30 conference and journal papers.

Dr. Liu's work has been nominated for a Best Paper Award at the Design Automation Conference (DAC) and the International Conference on Computer-Aided Design (ICCAD). He has served on the Technical Program Committees of ICCAD, the Asia and South Pacific Design Automation Conference (ASP-DAC), and the International Symposium on Circuits and Systems (ISCAS).