Smart Non-Default Routing for Clock Power Reduction

Andrew B. Kahng[†‡], Seokhyeong Kang[†] and Hyein Lee[†]
[†]ECE and [‡]CSE Departments, University of California at San Diego
abk@ucsd.edu, shkang@vlsicad.ucsd.edu, hyeinlee@ucsd.edu

ABSTRACT

At advanced process nodes, *non-default routing rules* (NDRs) are integral to clock network synthesis methodologies. NDRs apply wider wire widths and spacings to address electromigration constraints, and to reduce parasitic and delay variations. However, wider wires result in larger driven capacitance and dynamic power. In this work, we quantify the potential for capacitance and power reduction through the application of *"smart" NDR* (SNDR) that substitute narrower-width NDRs on selected clock network segments, while maintaining skew, slew, delay and EM reliability criteria. We propose a practical methodology to apply smart NDRs in standard clock tree synthesis flows. Our studies with a 32/28nm library and open-source benchmarks confirm substantial (average of 9.2%) clock wire capacitance reduction and an average of 4.9% clock switching power savings over the current fixed-NDR methodology, without loss of QoR in the clock distribution.

Categories and Subject Descriptors

B.7.2 [**Hardware**]: INTEGRATED CIRCUITS—*Design Aids*; J.6 [**Computer Applications**]: COMPUTER-AIDED ENGINEERING

General Terms

Algorithms, Design, Performance

Keywords

Clock Network Synthesis, Clock Network Optimization, Power Minimization

1. INTRODUCTION

Clock distribution is well-known to have a large impact on integrated-circuit performance, area and power consumption. From the 40nm node onward, *non-default routing rules* (NDRs) have become an integral element of clock tree synthesis (CTS) methodology, as a means of reducing electromigration (EM) violations and delay variations. NDRs specify per-net, per-layer requirements for the router to use wiring geometries that differ from the default single-width, single-spacing (1W1S) configuration. Example NDRs might include (2W2S) (double-width, double-spacing), (1W3S), (4W2S), (3W3S), etc., where W denotes width and S denotes spacing. Figure 1 illustrates sample NDRs, along with their respective routing track costs when wire segments are centered on track gridlines, as in the outputs of modern detailed routers.

At today's leading-edge process nodes, NDRs are employed in CTS for several basic reasons.

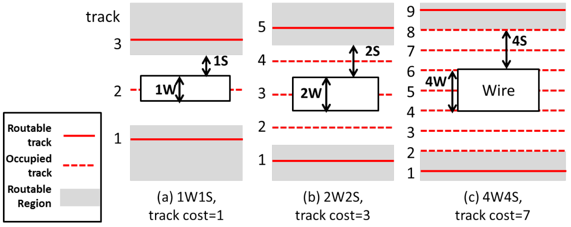

Figure 1: Illustration of three example NDRs, showing the track cost (number of tracks unavailable to (1W1S) wires) given the required spacing to neighbor wires. We assume that the detailed router centers each wire segment on a track gridline, and that track pitch = 2 × minimum width = 2 × minimum spacing; (a) (1W1S) NDR, with track cost 1; (b) (2W2S) NDR, with track cost 3; and (c) (4W4S) NDR, with track cost 7.

- Signal electromigration (EM) limits are violated by minimum-width wires when large buffers (e.g., 32X) are used to drive large fanouts (e.g., anywhere from 16 to 40 loads for each clock buffer instance in a typical buffered clock tree solution). To satisfy EM limits, wider wiring must be used.[1]
- Smaller geometries and more resistive interconnects, in conjunction with multi-patterning in lithography, result in higher parasitic and delay variability. Wider wires are less sensitive to these variations.
- Coupling capacitance, and therefore coupling-induced delay uncertainty, will be reduced with larger spacing [11].
- Though wider wires have higher capacitances, overall delays tend to be less due to reduced resistances.

For all practical purposes, modern use of NDRs means that IC designs intentionally spend extra wire capacitance nearly everywhere in the clock distribution network. Within a typical clock tree solution, the clock *subnet* driven by a given clock buffer[2] will be routed entirely with an NDR, except for the few microns of (1W1S) wiring needed to connect to input pins of the buffer's fanouts (loads). This increases capacitance and clock dynamic power. Given the need to reduce overall IC power consumption without sacrificing performance – particularly in mobile applications – we revisit the classic idea of optimizing wire width to reduce wire capacitance and dynamic power.

Motivating Studies

Consider a clock subnet driven by a large clock buffer. It is reasonable for the wiring of the subnet incident to the source (driver) to be wide: a large amount of downstream capacitance is being switched,

DAC'13, May 29 - June 07 2013, Austin, TX, USA.
Copyright 2013 ACM 978-1-4503-2071-9/13/05 ...$15.00.

[1] In light of random variation models and on-chip variation-aware (OCV) signoff, clock distribution methodologies usually seek to minimize source-to-sink insertion delays. This implies a trend toward fewer levels and larger (stronger) drivers in the clock topology.

[2] We say that a buffered *clock tree* consists of a number of *clock subnets*. Each subnet is driven by a buffer and has fanouts that are either other clock buffers or clock sinks (flip-flops). A routed subnet consists of a number of *edges*.

and this takes more time (with more current flow) with more downstream load. However, as we follow the subnet's wiring topology away from the source, the number of downstream loads continues to decrease with every branching of the topology, until each leaf segment in the subnet's routing tree has only one downstream load. Figure 2 summarizes statistics for 64-sink clock subnets routed by *Cadence Encounter DIS V10.1* [24]. The number of downstream loads (reflecting the average current) is highest near the source (driver) buffer, but rapidly decreases for the overwhelming majority of the clock subnet wirelength. Thus, there is no (electromigration-reliability) reason for a 2W or 3W NDR to persist for the entire routing topology of a given clock subnet.

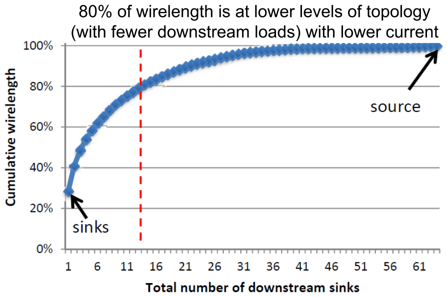

Figure 2: Study of 64-sink subnets routed by *Cadence Encounter DIS V10.1* [24]. Approximately 80% of clock subnet wirelength is in the lower levels of the tree, with few downstream loads and lower currents. The wire segment incident to the source has highest current but on average is only a small fraction of a subnet's total wirelength.

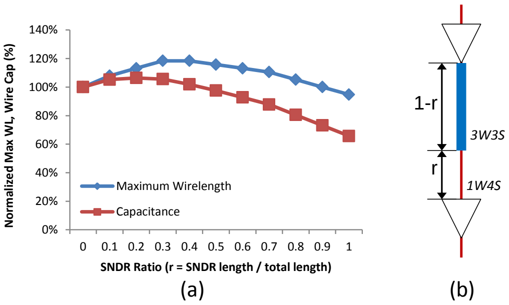

Figure 3: (a) Electrical performance when an equivalent-pitch SNDR (1W4S) is applied to a fraction r of a wire with a given original NDR (3W3S). Shown are maximum wirelength possible while satisfying a prescribed slew constraint, and total wire capacitance (both values normalized, and plotted against the y-axis). The ratio $r = 0.9$ achieves 27% capacitance reduction without incurring any reduction of maximum wirelength. (b) Illustration of tapering of wire from the original NDR (blue color) to the SNDR (red color). SPICE simulation is used to determine delay and slew with different SNDR ratios r.

Further motivation is obtained from SPICE experiments that evaluate the potential for SNDR-based capacitance reduction *without loss of electrical performance*. We study a symmetric H-tree [1] with a large buffer as driver, and 16 identical buffers as sinks. We sweep an *SNDR ratio*, r (i.e., wirelength with a given SNDR, divided by total wirelength – see Figure 3 (b)), to observe the impacts of SNDR wire tapering from different fixed original NDR widths and spacings. We assume that the signal transition at the input pin of the driver cell has 50ps slew time, and we find the maximum wirelength which maintains the same slew time at the end of the wire (i.e., at sink input pins), as well as the corresponding total wire capacitance.[3] For example, Figure 3 shows maximum wirelength under the 50ps maximum slew time constraint, and wire capacitance (both values normalized and plotted against the y-axis), when a (1W4S) SNDR is applied to save capacitance from an equivalent-pitch (3W3S) original NDR. The library and interconnect technology used to generate the figure are from a public-domain 32/28nm PDK [26]; 32X buffers are used for driver and sinks. In this case, application of the SNDR can reduce wire capacitance by 27% "for free" - that is, with zero decrease in maximum driven wirelength, at $r = 0.9$. Alternatively, wire capacitance can be reduced by 34% at the cost of 5% decrease in maximum driven wirelength, at $r = 1.0$.

With the above studies as our starting point, we explore the dichotomy between today's *fixed NDR* methodology and a possible *smart NDR* methodology. The cartoon of Figure 4 shows that *fixed NDRs* on clock subnets result in larger driven capacitance, potentially leading to larger currents, larger drivers, and increased dynamic power. (Indeed, if the larger drivers violate signal EM limits, even the larger (wider) NDRs may be required.) By contrast, *smart NDRs* taper wire widths as the number of downstream loads decreases, reducing driven capacitance, dynamic power, and the number and size of buffers.

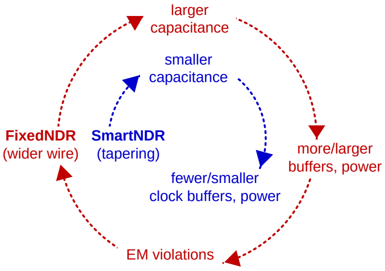

Figure 4: Intuitive dichotomy between today's "fixed NDR" methodology and a potential "smart NDR" methodology.

This Work

In this work, we assess the potential for capacitance and power reductions through use of *"smart" NDRs* (SNDRs) that substitute narrower-width NDRs for selected clock net segments while maintaining skew, slew, delay and EM reliability criteria. We formulate the optimal application of SNDRs for a given clock subnet as a quadratically constrained program. We also demonstrate the practicality of a flow that applies SNDRs at the end of the CTS phase in a commercial place-and-route tool.

The main contributions of our work are summarized as follows.

- We perform studies of practical clock routing instances and NDRs to establish the potential for substantial dynamic power reductions using SNDRs.
- We formulate optimal application of SNDRs as a quadratically constrained program to minimize wire capacitance under skew, slew, delay and EM constraints, and we efficiently solve this problem for each subnet of a given clock tree.
- We extend our SNDR solution approach to the entire clock tree by propagating skew constraints from downstream subnets to upstream subnets.
- We also propose a practical flow to apply SNDRs transparently post-CTS within a standard commercial place-and-route tool.
- We empirically confirm an average of 16% clock wire capacitance reduction, and 5% total clock power reduction, achieved by our proposed technique, as compared to traditional CTS approaches with fixed NDRs.

The remainder of this paper is organized as follows. In Section 2, we give an overview of related literature. Section 3 formulates the SNDR problem, and Section 4 describes our wire-tapering approach. Section 5 provides experimental results and analysis. We give conclusions and ongoing research directions in Section 6.

[3] Because the H-tree is symmetric, greater wirelength means that the sinks of the H-tree are spaced farther apart.

2. RELATED WORK

Wire width optimization (tapering) for clock and signal distribution has been extensively studied since the early 1990s. In this section, we survey related literature according to three main categories: (1) works on wire width optimization in clock trees; (2) electromigration-constrained wire sizing methods; and (3) coupling noise-driven wire sizing methods.

(1) Wire sizing in clock trees. Many works ([19], [9], [23], [15], [14], [17]) have applied wire sizing to minimize skew in clock trees. Tsai et al. [19] propose a dynamic programming method for simultaneous buffer insertion and wire sizing to optimize delay and power of a given zero-skew or useful-skew clock tree. They calculate a feasible delay-capacitance region for all nodes in a bottom-up phase, then determine buffer locations/widths and wire widths. Guthaus et al. [9] propose sequential linear programming as well as quadratic programming based clock buffer/wire sizing to minimize skew. The former technique uses first-order sensitivities for a small region of buffer/wire size solutions to represent a nonlinear objective using a set of linear functions. Zhu et al. [23] perform wire sizing to minimize skew using Gauss-Marquardt least-squares minimization. Pullela et al. [15] use wire width widening to reduce clock skew, delay and process variability impact. Their method starts with a minimum-delay tree and then optimizes skew by widening wires based on a delay sensitivity to wire width derived from Elmore delay [8]. In subsequent work [17], they suggest an analytical moment-sensitivity based methodology, using the moments of the transfer function of an RC tree circuit, to simultaneously reduce skew and slew. Liu et al. [14] also propose simultaneous clock routing, wire sizing and buffer insertion based on the Deferred-Merge Embedding algorithm [2] [3].

To our understanding, previous literature on clock tree wire sizing emphasizes timing optimization; exceptions are [19] and [9], the latter of which incorporates a power constraint.[4] In contrast to previous works, use of NDRs in clock routing is now largely driven by *signal EM reliability* limits. Moreover, previous works have various limitations such as continuous sizing that limit application within conventional physical design methodologies.

(2) Electromigration-constrained wire sizing. The literature on EM-constrained wire sizing has centered on power/ground networks. Tan et al. [20] suggest sequential linear programming-based wire sizing that considers IR drop, EM and other design constraints while minimizing area. A nonlinear function of constraints is relaxed to a sequence of linear programs which always enables convergence to an optimum solution. Wu et al. [22] propose a power/ground wire sizing algorithm with IR drop, EM and minimum-width constraints. A penalty method is used to solve the nonlinear problem so as to achieve area minimization of the power/ground network. For clock trees, Pullela et al. [16] consider an EM constraint in their low-power clock tree design methodology. They optimize a clock tree with buffer insertion by decreasing wire width while satisfying bounds on process variation-dependent skew and current density. Among all previous works of which we are aware, the work of [16] is the closest to our present target; however, buffer insertion is not our objective as it potentially draws more power and causes more EM violations in a vicious cycle (recall Figure 4).

(3) Noise-driven wire sizing. Last, wire sizing has been analyzed in conjunction with spacing to address coupling noise sensitivity. Cong et al. [5] propose symmetric and asymmetric wire sizing and spacing to minimize the weighted sum of coupling capacitance-induced delay at all sinks. Lagrangian relaxation is used to solve the same problem as in [4], but it is shown that proper wire sizing can further reduce delay when coupling capacitance is considered. In a subsequent work, Cong et al. [6] study a simultaneous wire spacing problem for multiple nets, to deal with crosstalk noise. In our methodology, although we do not address the noise problem directly, by increasing wire spacing, we reduce the coupling capacitance (and coupling-induced delay variation) as well.

3. PROBLEM FORMULATION

We seek to apply different NDRs (i.e., *smart NDRs*) to each wire segment of a given routed clock net, to reduce total wire capacitance and dynamic power subject to reliability and electrical requirements. More specifically, we want to minimize tree capacitance without violating slew, skew, clock source-to-sink insertion delay, and EM reliability constraints. Our focus is on an "iso-area" use model that is *transparent* to existing CTS flows: we start with a routed CTS solution, then identify a set of wire segment width reductions that save power without affecting any (skew, slew, etc.) solution metric.[5] Below, we describe such a transparent flow, which imports implemented SNDRs back into the P&R tool as DRC-clean modified DEF (Design Exchange Format, [30]) for extraction and performance analysis, with no ECO routing needed for any other nets.

Table 1 presents the notations that we use to describe our problem formulation and algorithm. Upper bounds on slew (transition) time, skew, and insertion delay are respectively denoted by U_S, U_K, and U_L. We use e to indicate an edge (between clock buffers, clock sinks, or Steiner points) in the clock tree. Given a clock tree T with edges $e \in T$, a set N of allowed SNDRs (indexed as n) with maximum current limit $E_{e,n}$ for edge e with SNDR $n \in N$, and electrical performance bounds U_S, U_K and U_L, our SNDR optimization seeks to map each edge width w_e to an SNDR $n \in N$ while minimizing total wire capacitance, subject to the constraints $E_{e,n}$, U_S, U_K and U_L.

$$\textbf{Minimize:} \sum_{e \in T} C_e$$

$$\textbf{Subject to:}$$

$$S_v \le U_S,\ \ I_e \le E_{e,w_e},\ \ L_v \le U_L,\ \ K_{u,v} \le U_K,\qquad w_e \in N,\ \ w_e > w_{desc(e)},\qquad (\forall v, e \in T) \tag{1}$$

Table 1: Notation

Notation	Meaning
C_e	capacitance of edge e
S_v	slew at the node v
I_e	average current of edge e
w_e	NDR of edge e, $w_e \in N$
x_e	the alternative NDR variable, $x_e = ln(w_e)$
$D_{u,v}$	delay from node u to node v
$K_{u,v}$	skew between node u and node v
L_v	clock latency at sink v
T	given clock tree
N	set of NDRs
E_{e,w_e}	maximum current limit for edge e with NDR $w_e \in N$
U_S	maximum slew constraint
U_K	maximum skew constraint
U_L	maximum clock latency constraint
$desc(e)$	set of all downstream sinks of e

We conclude this section with two comments on the extensibility of our SNDR problem formulation. First, the benefits of SNDR optimization can encompass area, routing congestion, and/or coupling between routes, since the use of SNDRs decreases the track consumption of the clock routing. Realizing these reductions requires ECO routing, and possibly ECO placement as well; we do not implement such a flow in our present work. However, we do report results in Section 5.2 suggesting that a "reduced-area" SNDR

[4] Of separate interest is the literature on wire sizing in general signal routing trees. These works are exemplified by Chen et al. [4], which targets signal routing and does not consider skew. The authors of [4] apply Lagrangian relaxation for gate and wire sizing to minimize area subject to a maximum delay bound.

[5] Section 5.2 also considers a use model where track usage is allowed to decrease as a result of SNDR application.

optimization, where wire widths are reduced and spacings can take on any value that does not result in wire capacitance exceeding that of the original fixed NDRs, can also significantly reduce track consumption. Second, analysis and enforcement of electrical constraints readily extend to include coupling noise, noise-induced delay variation, process variation, and a number of other concerns. However, the basic optimization will remain the same as what we study, and we leave such extensions to future work.

4. SNDR WIRE SIZING

4.1 Wire RC Delay Model for SNDR

RC modeling of wire is given by Equation (2), where l_e, w_e and s_e are the length, width and spacing of edge e, respectively.

$$R_e = \rho \cdot \frac{l_e}{w_e}, \;\; C_e = \varepsilon \cdot \frac{l_e w_e}{s_e} \tag{2}$$

We assume $w_e + s_e$ is a constant Y equal to a fixed track pitch. Then, $s_e = Y - w_e$. To obtain linear formulations in w_e, we approximate R per μm and C per μm as functions of $x_e = ln(w_e)$[6]. Then, R_e and C_e can be approximated as

$$R_e = (\alpha_R \cdot x_e + \beta_R) \cdot l_e, \;\; C_e = (\alpha_C \cdot x_e + \beta_C) \cdot l_e \tag{3}$$

where α_R, β_R, α_C and β_C are fitting coefficients which we obtain by linear regression. Figure 5 shows that our suggested model is fairly accurate with measurements[7]. If the range of wire width increases, our model may not be applicable. However, these models are reasonable for the general wire width range limited by recent process technologies.

We use the Elmore delay model [8] to calculate the delay of clock tree. The delay between node u and v is

$$D_{u,v} = \sum_{e \in P_{u \rightarrow v}} (R_e \cdot \sum_{i \in desc(e)} C_i) \tag{4}$$

where $P_{u \rightarrow v}$ is the path from node u to v.

By substituting clock source s to u in Equations (3) and (4), we get the clock latency at node v as

$$L_v = \sum_{i \in P_{s \rightarrow v}} ((\alpha_R \cdot x_i + \beta_R) \cdot l_i \sum_{j \in desc(i)} (\alpha_C \cdot x_j + \beta_C) \cdot l_j) \tag{5}$$

For wire slew calculation, we apply the PERI model [10]. The slew at node v, where s is the clock source, is

$$S_v = \sqrt{{S_s}^2 + \ln 9 \cdot {D_{s,v}}^2} \tag{6}$$

where S_s is the output slew of clock buffer at the clock source. The output slew of clock buffers depends on the input slew and the output capacitances. Since the input slew of clock buffer is the output slew of the upstream net which is constrained by U_S, we can assume U_S to be the worst case value. With a constant input slew U_S, we use a piecewise linear model to approximate the output slew as a function of the total output capacitance.

$$S_s = \alpha_S \cdot \sum_{e \in desc(s)} C_e + \beta_S \tag{7}$$

where α_S and β_S are fitting coefficients. We assume the total output capacitance before optimization as the initial capacitance. This assumption is pessimistic since the total output capacitance will be minimized after applying SNDR. However, for large sizes of buffers, which are typically used in clock trees, the output slew is not very sensitive to the output capacitance. Thus, with reasonable pessimism, we obtain a constant S_s with given U_S and the initial total capacitance.[8]

$$K_{u,v} = |D_{s,u} - D_{s,v}| \tag{8}$$

The skew constraint should be checked for all pairs of source-to-sink timing paths with the upper bound U_K.

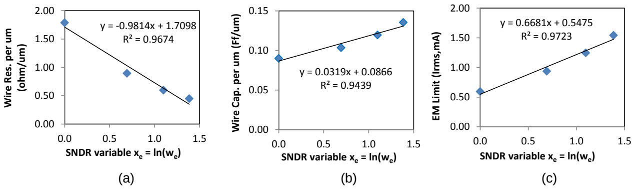

Figure 5: (a) Wire resistance per unit length (b) wire capacitance per unit length, and (c) EM limit (I_{RMS}); each is fitted as a linear function of $x_e = ln(w_e)$, where w_e is the wire width in the limited range that we are using.

4.2 EM Current Model and EM Rule

For EM constraints, we use a simplified I_{RMS} model derived from Black's Equation [12]. If V_{dd} is the operating voltage, F is the operating frequency and sw is the switching activity on the net,

$$I_{RMS}(e) = \sum_{i \in desc(e)} C_i \cdot V_{dd} \cdot F \cdot \sqrt{sw} \tag{9}$$

where $I_{RMS}(e)$ is the root mean square current of the edge e. We use Synopsys 32/28nm PDK [26], where the I_{RMS} limit is defined as a polynomial function of wire width and the metal layer,

$$EMLimit(w) = \alpha_E w^2 + \beta_E w + \gamma_E \tag{10}$$

where α_E, β_E and γ_E are fitting coefficients. We obtain a linear function of x_e as shown in Equation (11), where α'_E and β'_E are fitting coefficients. For the ranges of SNDRs considered, this model accurately captures characterized values as shown in Figure 5.

$$E_{e,w_e} = \alpha'_E x_e + \beta'_E \tag{11}$$

4.3 Iterative Linear Programming (LP)

To avoid quadratic constraints arising from the Elmore delay model, we separate the sizing problem into two (alternating) linear programs by fixing the R values and the C values in alternation. Our method recalls the rescaled simple iteration of [21], and has the following steps.

Delay constraints are formulated based on Equation (5). First, we fix x_i with a constant $x_{e,0}$ (= initial NDR value) and formulate a linear function of x_j ($Constr_C$). In a similar way, we fix x_j with a constant x_0 and formulate a linear function of x_i ($Constr_R$). And then, we solve the problem with both the constraints ($Constr_R$ and $Constr_C$) simultaneously using the objective function (Equation (1)). Second, we formulate the constraints in the same way as the first step, but with different initial values $x_{e,1}$, which are the solutions derived from the previous step. We iteratively solve the problem until all x_e for each edge e are determined. In our experiments, our approach obtained results that are essentially identical to those of the QCP-based method, but with runtime reductions of 6× to 30×[9].

4.4 Applying SNDR to an Entire Clock Tree

In subsections (Section 4.1 $\sim$ 4.3), we have formulated the SNDR problem, and proposed an iterative method to optimize a subtree of clock (a single net). To optimize the entire clock tree, we perform our SNDR method from the downstream to upstream subnets of clock tree. Algorithm 1 presents pseudocode of our SNDR flow for

[6] We note that R and C are respectively proportional to, and inversely proportional to, w_e. Taking logs transforms multiplier and divider into first-order terms.

[7] The maximum errors that we have seen are -22% for R and -4% for C, for the SNDR sets that we study.

[8] We do not perform buffer sizing since it can lead to larger delay variations and degrade EM when buffers are upsized.

[9] We have compared the runtime with testcase *wb_dma_top*; the overall runtimes of the QCP-based method and LP-based method are 170 minutes and 29 minutes respectively with a similar quality of solution.

the entire clock tree. We apply SNDR to each subtree with EM, slew constraints, skew constraints and delay margin to generate a tapered clock tree. $D_{init(u,v)}$ is the initial delay from node u to v before applying SNDR. We collect the set of delay constraints (Line 6) with the wire delays. To calculate clock skew, we define $D_{min(u)}$ and $D_{max(u)}$, which are the minimum and maximum path delay from node u to its leaf nodes (flip-flops). $path\_delay_{min}(u,v)$ and $path\_delay_{max}(u,v)$ are the minimum and maximum path delay from u to leaf nodes through node v, and they are used for the skew constraints (Line 17). After collecting the delay and skew constraints for each subtree, we apply SNDR ($SNDR_{sub}$) to the subtree t (Line 22). $SNDR_{sub}$ finds an optimal NDR for each wire segment in the subtree by solving the problem of Equation (1) using the Iterative LP described in Section 4.3. Then, we update the $D_{min(u)}$ and $D_{max(u)}$, which are calculated recursively from downstream subnets to upstream subnets (Line 23-26). For $cell\_delay(v)$, we get the worst buffer delay which is calculated with the initial capacitances before optimization. EM and slew constraints are checked net by net independently in $SNDR_{sub}$.

Algorithm 1 SNDR Wire Sizing for Clock Tree

```
Procedure $SNDR(T, \{E\}, U_K, U_S, M)$
Input : clock tree $T$, a set of EM constraints $\{E\}$, skew constraints $U_K$, slew constraints $U_S$ and delay margin $M$;
Output : tapered clock tree with SNDR
 1: $T_n \leftarrow$ all subtrees (= routed subnets) in $T$;
 2: while $T_n \neq \emptyset$ do
 3:     Pick a subtree $t$ with maximum level;
 4:     $u \leftarrow$ source node of $t$;
 5:     for all sink nodes $v$ of $t$ do
 6:         $DelayConst_n \leftarrow DelayConst_n \cup \{D_{u,v} < D_{init(u,v)} + M\}$;
 7:         $SlewConst_n \leftarrow SlewConst_n \cup \{S_v < U_S\}$;
 8:         if $v$ is a clock port of flip-flop then
 9:             $path\_delay_{max}(u,v) \leftarrow D_{u,v}$;
10:             $path\_delay_{min}(u,v) \leftarrow D_{u,v}$;
11:         else
12:             $path\_delay_{max}(u,v) \leftarrow D_{max(v)} + cell\_delay(v) + D_{u,v}$;
13:             $path\_delay_{min}(u,v) \leftarrow D_{min(v)} + cell\_delay(v) + D_{u,v}$;
14:         end if
15:     end for
16:     for all two sink nodes $v$ and $w$ pair of $t$ do
17:         $SkewConst_n \leftarrow SkewConst_n \cup \{path\_delay_{max}(u,v) - path\_delay_{min}(u,w) < S\}$;
18:     end for
19:     for all edges $e$ of $t$ do
20:         $EMConst_n \leftarrow EMConst_n \cup \{I_e < E_{e,w_e}\}$;
21:     end for
22:     $SNDR_{sub}(t, DelayConst_n, SkewConst_n, SlewConst_n, EMConst_n)$;
23:     for all sink nodes $v$ of $t$ do
24:         $D_{max(u)} \leftarrow max(D_{max(u)}, path\_delay_{max}(u,v))$;
25:         $D_{min(u)} \leftarrow min(D_{min(u)}, path\_delay_{min}(u,v))$;
26:     end for
27:     $T_n \leftarrow T_n - t$;
28: end while
```

Algorithm 1 can be used to minimize skew by setting the skew constraints to near zero and/or adding weights for aggressive optimization in the objective function. *SNDR* then finds solutions which minimize skew under given EM, slew and delay constraints.

4.5 Further Optimization with More SNDRs

Recall that our proposed methodology focuses on transparency to existing CTS flows, i.e., we perform the wire sizing after routing, potentially just before design closure and signoff. Thus, we limit our selection of SNDRs to constant track cost, so as not to harm the original design with noise coupling or with routing ECOs. This being said, if we allow additional options for NDRs, for example, smaller spacing rules, then other optimizations such as reduction of wire congestion with ECO routing become possible. Figure 6 shows the capacitance per μm with various SNDRs (the values are extracted from the capacitance tables used in *Cadence Encounter DIS V10.1* [24].). Though coupling capacitance increases as spacing decreases, the total capacitance of each SNDR is below that of the original NDR (4W5S) while maintaining the original spacing without (3W3S) case. Clearly, there is an available tradeoff between area and coupling noise along with various spacing options. Given this, it is possible to tune the objective function to maximize the benefits from SNDR across a range of power-, area- and noise-constrained designs.

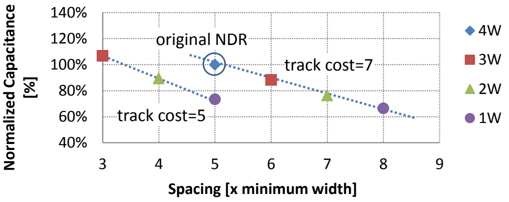

Figure 6: % reduction of capacitance per unit length with various SNDRs.

4.6 Implementation Flow

We propose a practical flow that implements SNDRs using a commercial place-and-route tool, as illustrated in Figure 7. For an input design, we perform clock tree synthesis with a given fixed NDR. After routing, we extract the locations of buffers and flip-flops, which are the sources and sinks of clock nets, along with all clock wire segments, from the routed DEF [30] of the implemented design. From this information, we construct the optimization instances (Equation (1)) with appropriate timing (delay, skew, slew) and EM constraints. Solving each optimization instance entails (i) calculating a matrix of coefficients for the equations governing the solution of width variables (one variable for every edge in each given subnet) with timing, EM reliability and tapering constraints, and (ii) applying Iterative LP to solve these problem instances and obtain wire sizing solutions for every edge of each given subnet. We then update the original DEF file with the obtained wire sizing solutions for each subnet. The modified DEF is read back into the P&R tool, and RC values are recalculated. With the updated RC values, we check for any electrical/timing or EM reliability violations; if there are any violations caused by an SNDR on a wire segment, we revert that segment to its original NDR.

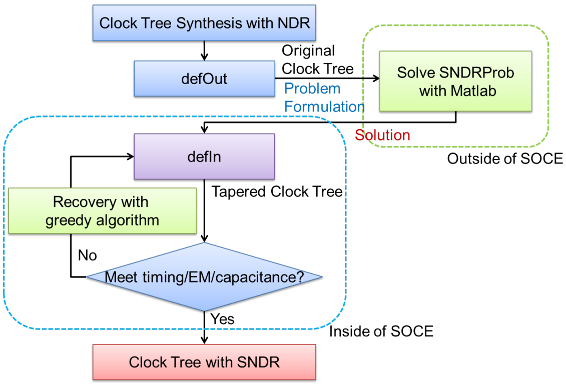

Figure 7: Overall implementation flow.

5. EXPERIMENTAL SETUP AND RESULTS

5.1 Experimental Setup

We extract wire resistance and capacitance for different wire NDRs, starting from capacitance table, technology LEF and ITF standard inputs from the relevant PDK. For our experiments, we use nine open-source designs from the *OpenCores* website [29]. We use the *Synopsys 32/28nm PDK* [26] cell library for the design implementation. We synthesize the designs using *Synopsys DesignCompiler vF-2011.09* [27] and perform place-and-route with

Cadence Encounter DIS v10.1 [24]. We solve the wire sizing problem formulated above using *Mathworks MATLAB R2012b* [25]. Insertion delay constraints are set by adding a small (5%) margin to the original insertion delay of each sink; this improves flexibility of the SNDR assignment with negligible impact on the final insertion delay, as shown in Table 4. We use the maximum skew of the original design as the skew constraint, and the maximum transition time of the original design with a margin (less than 5% of clock period) as the maximum transition time (slew) constraint. EM constraints are obtained from the Synopsys 32/28nm PDK.[10] For simplicity, we assume that the track constraint is a constant, i.e., we perform an "iso-area" optimization; the corresponding set of NDRs is (4W5S), (3W6S), (2W7S) and (1W8S). We allow the optimization to use only smaller-width NDRs to replace the original NDR of the clock tree, i.e., we only taper the wire. The clock tree is initially routed with the maximum wire width NDR (4W5S). We note that in this particular technology, use of a smaller-width NDR in the initial implementation can result in EM violations after CTS.

5.2 Experimental Results

We have performed the SNDR optimization on nine benchmark designs to assess capacitance and power reductions afforded by use of SNDRs. Table 2 shows the number of instances, sinks and clock buffers for each testcase.

Table 2: Testcases: number of instances, sinks and clock buffers.

testcase	#instances	#sinks	#clock buffers
aes_cipher_top	21766	530	11
eth_top	11342	1743	37
jpeg_encoder	67185	4512	100
mc_top	6525	934	27
mpeg2_top	13026	2931	59
tv80s	6478	359	25
usbf_top	11446	1515	31
wb_conmax_top	32559	770	25
wb_dma_top	3035	523	15

Figure 8 shows the proportions of total clock wirelength routed with each NDR after applying our wire sizing optimization. From the results, we see that more than 80% of the wiring is replaced by smaller NDRs.

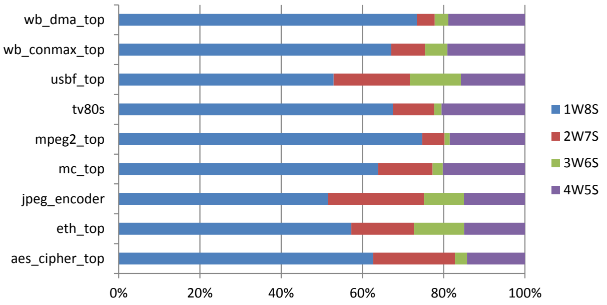

Figure 8: Proportions of total clock tree wirelength routed with different NDRs after optimization.

Table 4 shows the post-optimization implemented results and clock network power reduction versus the original (conventional) designs which use fixed NDRs of (4W5S) in Table 4(a) and (2W4S) in Table 4(b). Runtime for the *MATLAB* implementation of our algorithm is between 10 seconds and ∼100 minutes per subnet on a 2.5GHz Intel Xeon processor; this runtime varies widely depending on the number of sinks, the number of wire segments, and the number of constraints. Each testcase requires sequential optimization of subnets, but it is simple to parallelize the solution of subnets that are independent of each other. Importing modified DEF back into the P&R tool never requires more than 10 seconds for any testcase. The "delta" values in the rightmost three columns of the table are reported in *picoseconds*, i.e., the insertion delay, slew and skew changes are negligible (moreover, negative delta values represent improvements in these parameters). We see that our SNDR flow can reduce clock wire capacitance over the conventional designs by up to 10.3% (average 9.2%) when (4W5S) is used as the fixed NDR. With the reduced wire capacitance, we achieve up to 5.7% (average 4.9%) and 4.2% (average 3.4%) clock switching power and total power savings[11] with essentially zero EM violations or timing degradations. We have also applied SNDRs for the case that (2W4S) is used as the fixed NDR. In this case, SNDRs have just two options, i.e., (2W4S) or (1W5S). Wire capacitances reduce by up to 5.9% (4.7% on average), which yields up to 2.8% (average 2.2%) and 2.1% (average 1.5%) clock switching power and total power savings. We note that our results correspond to a viable flow implemented in a widely-used commercial EDA tool: all of the post-optimization SNDR-based clock routing is DRC-clean, and while the optimization is guided by our abstractions of delay and slew, all of the reported extraction and timing results, EM checks, etc. are according to the same commercial tool.

Table 3: Results using SNDRs with an upper bound on track cost of 7.

testcase	track cost reduction (%)	capacitance reduction (%)
aes_cipher_top	53.8	12.15
eth_top	46.8	13.14
jpeg_encoder	48.1	12.37
mc_top	49.1	12.45
mpeg2_top	52.5	12.14
tv80s	48.0	11.49
usbf_top	48.1	11.11
wb_conmax_top	49.7	13.16
wb_dma_top	52.7	14.28

Last, to assess the potential for area recovery using various SNDRs, we perform an additional study using a set of SNDRs which maintains track cost less than the original fixed NDR. That is, we explore "reduced-area" SNDRs of (1W2S), (2W4S) and (3W6S). While the original set of SNDRs all require 7 tracks, the new set of SNDRs requires ≤ 7 tracks. For example, a (1W2S) SNDR might be used where a (1W8S) SNDR was used in the previous experiment. We perform the same optimization using characterized RC models for the new set of SNDRs, with results summarized in Table 3. We see in the table that the new set of SNDRs actually achieves slightly improved capacitance reduction, while also saving up to 53.8% of routing tracks (49.9% on average), calculated as a weighted average along the entire routed wirelength of the clock tree.

6. CONCLUSIONS

The notion of wire tapering for clock tree (area, power, skew) optimization has been studied in the literature for nearly 20 years. However, tapering flows have not achieved traction in production IC implementation methodologies, possibly due to the lack of perceived benefit in combination with the complexity of flow development. For several reasons, we believe that wire width selection in CTS has become worth revisiting: clock power reduction is increasingly critical in a power-limited world; variability has driven CTS optimizations toward larger drivers and high-fanout clock nets; and EM reliability limits in particular (along with the desire to avoid time-consuming EM-fix steps in physical design) have motivated the widespread use of NDRs in modern CTS methodologies.

[10] Maximum skew constraints of 3-18ps are used except for jpeg_encoder (110ps). Slew constraints of 47-100ps are used except for eth_top, jpeg_encoder, mpeg2_top (> 200ps). EM constraints are calculated using Equation (11).

[11] Clock buffers and FFs are not changed. The power savings are achieved only with wire capacitance reduction.

Table 4: SNDR results: Capacitance, clock power, insertion delay, skew, and maximum slew. Δ values < 0 are improvements.

(a) With the default NDR (4W5S)

test case	wire cap. (pF)	switching power (mW)	total power (mW)	wire cap. reduction	switching power reduction	total power reduction	Δ clock latency (ps)	Δ clock skew (ps)	Δ max slew (ps)	EM vio.
aes_cipher_top	0.62	0.742	0.986	**8.69%**	**5.41%**	4.22%	-6.00	**-3.00**	**0.40**	1
eth_top	1.26	1.883	2.589	**8.88%**	**4.46%**	3.24%	-1.70	**0.00**	**-0.30**	0
jpeg_encoder	4.07	5.068	6.926	**9.72%**	**5.43%**	4.16%	-9.70	**-19.00**	**-12.80**	1
mc_top	0.68	0.933	1.445	**9.40%**	**4.61%**	3.11%	-4.80	**-2.00**	**0.10**	0
mpeg2_top	2.47	4.116	5.633	**7.76%**	**4.20%**	3.18%	-3.90	**-2.00**	**0.70**	0
tv80s	0.36	0.403	0.821	**9.31%**	**5.35%**	2.75%	-4.10	**-2.00**	**0.80**	0
usbf_top	1.16	2.016	2.780	**8.44%**	**4.37%**	3.24%	-1.70	**-1.00**	**0.50**	0
wb_conmax_top	0.71	1.031	1.587	**10.11%**	**5.65%**	3.72%	-10.10	**-4.00**	**0.40**	0
wb_dma_top	0.34	0.918	1.447	**10.32%**	**4.94%**	3.25%	-3.50	**-2.00**	**0.50**	0

(b) With the default NDR (2W4S)

test case	wire cap. (pF)	switching power (mW)	total power (mW)	wire cap. reduction	switching power reduction	total power reduction	Δ clock latency (ps)	Δ clock skew (ps)	Δ max slew (ps)	EM vio.
aes_cipher_top	0.49	0.651	0.892	**4.85%**	**2.76%**	2.10%	-2.30	**-2.00**	**3.10**	0
eth_top	1.24	1.866	2.657	**4.13%**	**2.04%**	1.54%	-1.80	**-11.00**	**-0.80**	0
jpeg_encoder	3.23	4.478	6.121	**5.15%**	**2.57%**	1.83%	-6.20	**-15.00**	**-3.30**	0
mc_top	0.53	0.831	1.338	**4.28%**	**1.78%**	1.12%	-1.80	**-1.00**	**0.00**	0
mpeg2_top	1.88	3.586	5.045	**3.88%**	**1.84%**	1.35%	-2.40	**-4.00**	**-2.10**	0
tv80s	0.27	0.346	0.760	**3.71%**	**1.82%**	0.88%	-1.20	**0.00**	**0.50**	0
usbf_top	0.89	1.773	2.537	**5.88%**	**2.65%**	1.85%	-5.00	**-3.00**	**0.30**	0
wb_conmax_top	0.57	0.919	1.471	**4.86%**	**2.37%**	1.56%	-3.60	**-2.00**	**-0.20**	0
wb_dma_top	0.27	0.829	1.354	**5.10%**	**2.13%**	1.40%	-1.50	**0.00**	**0.30**	1

In this work, we have assessed the potential for capacitance and power reduction from "smart NDRs" that substitute narrower-width NDRs for selected clock segments while maintaining all skew, slew, insertion delay and EM reliability criteria. We formulate a wire sizing problem of choosing per-segment NDRs to minimize the capacitance of the clock tree subject to electrical and reliability constraints, and we propose an effective solution to this problem for each subnet. We then extend the SNDR approach to the entire clock tree by propagating skew constraints from downstream to upstream subnets. We also propose a practical methodology to apply SNDRs without disruption of standard clock network synthesis flows.

With a 32/28nm library and open-source benchmark netlists, an "iso-area" execution of our SNDR methodology (scripted in a widely-used commercial EDA tool, with all extraction and analysis performed in the same tool) achieves up to 5.7% clock switching power savings (4.9% savings on average) and up to 10.3% (9.2% on average) clock network capacitance reduction versus the fixed-NDR (4W5S) methodology. A "reduced-area" execution of the SNDR methodology shows that substantial wiring resource (i.e., track usage) reductions are possible with lesser capacitance savings. Our ongoing work adds more detailed signal integrity analyses to both the optimization and validation aspects of our SNDR methodology. We are also extending our implementation to work with other commercial P&R (CTS) tools, and extending validations to additional technology libraries and design testcases. Additional research directions include the use of SNDRs not only to reduce power and routing costs, but also to control local skews at all levels of the buffered clock tree hierarchy for improved chip timing and robustness.

7. REFERENCES

[1] H. B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI*, Addison-Wesley, 1990.

[2] K. D. Boese and A. B. Kahng, "Zero-Skew Clock Routing Trees With Minimum Wirelength", *Proc. ASIC Conf.*, 1992, pp. 17-21.

[3] T.-H. Chao, Y.-C. Hsu and J.-M. Ho, "Zero Skew Clock Net Routing", *Proc. DAC*, 1992, pp. 518-523.

[4] C.-P. Chen, C. C. N. Chu and D. F. Wong, "Fast and Exact Simultaneous Gate and Wire Sizing by Lagrangian Relaxation", *IEEE TCAD* 18(7) (1999), pp. 1014-1025.

[5] J. Cong, L. He, C.-K. Koh and D. Z. Pan, "Interconnect Sizing and Spacing with Consideration of Coupling Capacitance", *IEEE TCAD* 20(9) (2001), pp. 1164-1169.

[6] J. Cong, D. Z. Pan and P. V. Srinivas, "Improved Crosstalk Modeling for Noise Constrained Interconnect Optimization", *Proc. ASP-DAC*, 2001, pp. 373-378.

[7] M. A. El-Mousy and E. G. Friedman, "Exponentially Tapered H-tree Clock Distribution Networks", *IEEE TVLSI* 13(8) (2005), pp. 971-975.

[8] W. C. Elmore, "The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers", *J. Applied Physics* 19(1) (1948), pp. 55-63.

[9] M. R. Guthaus, D. Sylvester and R. B. Brown, "Clock Buffer and Wire Sizing Using Sequential Programming", *Proc. DAC*, 2006, pp. 1041-1046.

[10] S. Hu, C. J. Alpert, J. Hu, S. Karandikar, Z. Li, W. Shi and C. N. Sze, "Fast Algorithms For Slew Constrained Minimum Cost Buffering", *IEEE TCAD* 26(11) (2007), pp. 2009-2022.

[11] A. B. Kahng, S. Muddu, E. Sarto and R. Sharma, "Interconnect Tuning Strategies for High-performance ICs", *Proc. DATE*, 1998, pp. 471-478.

[12] A. B. Kahng, S. Nath and T. S. Rosing, "On Potential Design Impacts of Electromigration Awareness", *Proc. ASP-DAC*, 2013, to appear.

[13] R. Kay and L. T. Pileggi, "EWA: Efficient Wiring-sizing Algorithm for Signal Nets and Clock Nets", *IEEE TCAD* 17(1) (1998), pp. 40-49.

[14] I.-M. Liu, T.-L. Chou, A. Aziz and D. F. Wong, "Zero-Skew Clock Tree Construction by Simultaneous Routing, Wire Sizing and Buffer Insertion", *Proc. ISPD*, 2000, pp. 33-38.

[15] S. Pullela, N. Menezes and L. T. Pillage, "Reliable Non-Zero Skew Clock Tree Using Wire Width Optimization", *Proc. DAC*, 1993, pp. 165-170.

[16] S. Pullela, N. Menezes and L. T. Pillage, "Low Power IC Clock Tree Design", *Proc. CICC*, 1995, pp. 263-266.

[17] S. Pullela, N. Menezes and L. T. Pillage, "Moment-Sensitivity-Based Wire Sizing for Skew Reduction in On-Chip Clock Net", *IEEE TCAD* 16(2) (1997), pp. 210-215.

[18] T. Sakurai, "Approximation of Wiring Delay in MOSFET LSI", *IEEE JSSC* 18(4) (1983), pp. 418-426.

[19] J.-L. Tsai, T.-H. Chen and C.-P. Chen, "Zero Skew Clock-Tree Optimization With Buffer Insertion/Sizing and Wire Sizing", *IEEE TCAD* 23(4) (2004), pp. 565-572.

[20] S. X. D. Tan, C. J. R. Shi and J.-C. Lee, "Reliability-Constrained Area Optimization of VLSI Power/Ground Networks via Sequence of Linear Programmings", *IEEE TCAD* 22(12) (2003), pp. 1678-1684.

[21] R. A. Thisted, *Elements of Statistical Computing: Numerical Computation*, Chapman & Hall / CRC, 1988.

[22] X. Wu, X. Hong, Y. Cai, C. K. Cheng, J. Gu and W. Dai, "Area Minimization of Power Distribution Network Using Efficient Nonlinear Programming Techniques", *Proc. ICCAD*, 2001, pp. 153-157.

[23] Q. Zhu and W. W.-M. Dai, "High-Speed Clock Network Sizing Optimization Based on Distributed RC and Lossy RLC Interconnect Models", *IEEE TCAD* 15(9) (1996), pp. 1106-1118.

[24] *Cadence Encounter DSI User's Manual*, http://www.cadence.com .

[25] *Mathworks MATLAB User's Manual*, http://www.mathworks.com .

[26] *Synopsys 32/28nm Open PDK*, http://www.synopsys.com .

[27] *Synopsys DesignCompiler User's Manual*, http://www.synopsys.com .

[28] *Synopsys HSPICE User's Manual*, http://www.synopsys.com .

[29] *OpenCores: Open Source IP-Cores*, http://www.opencores.org .

[30] *LEF DEF Guide*, http://www.si2.org.openeda.si2.org/projects/lefdef .