Discharge-Path-Based Antenna Effect Detection and Fixing for X-Architecture Clock Tree

Journal:	*IEEE Transactions on Very Large Scale Integration Systems*
Manuscript ID:	TVLSI-00341-2009
Manuscript Type:	Regular Paper
Date Submitted by the Author:	01-Sep-2009
Complete List of Authors:	Tsai, Chia-Chun; Nanhua University, Computer Science and Information Engineering Kuo, Chung-Chieh; National Taipei University of Technology, Computer and Communication Engineering Lee, Trong-Yen; National Taipei University of Technology, Electronic Engineering Hsu, Feng-Tzu; National Taipei University of Technology, Computer and Communication Engineering
Key Words:	Antenna effect, jumper insertion, layer assignment, via timing impact, X-architecture

Discharge-Path-Based Antenna Effect Detection and Fixing for X-Architecture Clock Tree

Chia-Chun Tsai, Chung-Chieh Kuo, Trong-Yen Lee, and Feng-Tzu Hsu

This work was supported in part by the National Science Council of Taiwan under Grants NSC 96-2221-E-343-007 and NSC 97-2221-E-343-008.

Affiliation of author

C. C. Tsai is with Department of Computer Science and Information Engineering, Nanhua University, Chia-Yi, 62248, Taiwan, R.O.C. (e-mail: chun@mail.nhu.edu.tw).

C. C. Kuo is with Graduate Institute of Computer and Communication, National Taipei University of Technology, Taipei, 106, Taiwan, R.O.C. (e-mail: s4419014@ntut.edu.tw).

T. Y .Lee is with Department of Electronic Engineering and Institute of Computer and Communication, National Taipei University of Technology, Taipei, 106, Taiwan, R.O.C. (e-mail: tylee@ntut.edu.tw).

F. T. Hsu is with Graduate Institute of Computer and Communication, National Taipei University of Technology, Taipei, 106, Taiwan, R.O.C. (e-mail: nhalex@yahoo.com).

Discharge-Path-Based Antenna Effect Detection and Fixing for X-Architecture Clock Tree

***Abstract*—The antenna effect is a phenomenon in the plasma-based nanometer process. This effect directly influences manufacturability, yield, and reliability of VLSI circuits. Because the metal wires directly exposed to the plasma are able to accumulate sufficient charges to damage thin gate oxides. For X-architecture clock trees those connect clock sinks with diagonal and Manhattan metal wires, this work proposes a discharge-path-based antenna detection algorithm and fixes antenna violations with jumper insertion technique. In addition, via timing impact is considered in delay calculation. For reducing the number of jumpers, layer assignment technique is integrated into the proposed algorithm. Give an X-architecture clock tree with *n* clock sinks, the layer configuration, and the upper bound of antenna effect, the proposed algorithm, called as PADJI, runs in $O(n^2)$ to achieve antenna violation free. In terms of the number of inserted jumpers, delay, power consumption, and the total vias listed in the experimental results on benchmarks, PADJI achieves improvements of 46.12%, 0.0932%, 0.0913%, and 46.18%, respectively, compared with other previous antenna fixing algorithms for X-architecture clock trees, and further reduction of 48.21%, 0.0242%, 0.0186%, and 20.36%, respectively, with integrating layer assignment technique.**

***Index Terms*—Antenna effect, clock tree, design for manufacturability, jumper insertion, layer assignment, via timing impact, X-architecture.**

I. Introduction

With the continuous and rapid increasing complexity of very-large-scale-integration (VLSI) designs in present nanometer fabrication technologies, manufacturing yield and product reliability have become important issues in design and manufacture fields [1-2]. The fine feature size of modern IC technology is typically achieved by using plasma-based processes. In nanometer era, more stringent process requirements cause some advanced high-density plasma reactors applied in the production lines for achieving fine-line patterns. However, the plasma-based process contains a potential behavior to charge conducting components of the fabricated structures. During wafer manufacturing, the metal wires those are not covered by the oxide shielding layer are directly exposed to the plasma and able to accumulate charges. The announced experimental results report that the sufficient accumulated charges may affect the quality of thin gate oxide [3]. This phenomenon is called the antenna effect or the plasma-induced gate oxide damage. Chen and Koren [4] proved that the amount of such charging is proportional to the plasma-exposed area. They defined the ratio of plasma-exposed area, $A_{s,metal}$, to gate oxide area, A_{poly}, as an antenna ratio (AR) for predicting the antenna effect. The definition of AR is given as follows.

$$AR = \frac{\text{plasma - exposed area}}{\text{gate - oxide area}} = \frac{A_{s,metal}}{A_{poly}} \le k_{th}, \quad (1)$$

where k_{th} is the threshold of AR.

Figure 1(a) shows a metal wire with width w_M, length l_M, and thickness t_M. $A_{s,metal}$ and A_{poly} are defined as $2(w_M+l_M)t_M$ and $w_M l_M$, respectively [4]. TSMC 0.18μm technology file [5] indicates that the required k_{th} is 400. Consequently, the upper bound of antenna effect, L_{max}, can be derived as 200μm. Either the metal wire length or the concatenated metal wires' accumulated length is longer than L_{max}; an antenna violation will be occurred. The upper bound could be a limited length, a limited exposed area, a ratio of antenna strength (length or area) to gate oxide area, or a model of antenna strength caused by conductors.

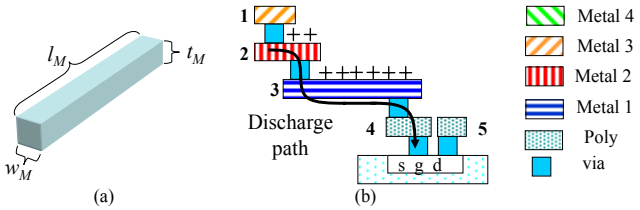

Fig. 1. (a) A metal wire and (b) a cross-section view for illustrating antenna effect.

As shown in Fig. 1(b), the manufacturing process supports four metal layers and all the interconnects are manufactured in the order of poly, metal 1, metal 2, metal 3, and metal 4. Obviously, metal 3 is the topmost layer of this structure. The total exposed wire areas of the gate oxide are counted from poly, metal 1 to metal 2 [6-7]. The exposed wires are able to accumulate enough charges to damage the gate oxide with Fowler-Nordheim (F-N) tunnelling current through the discharge path. The wire area of/above the topmost metal layer is irrelevant to the antenna consideration [6]. Hence, the accumulated discharge path length is the sum of the segments 2-4 lengths. If the accumulated length is longer than or equal to L_{max}, an antenna violation will be occurred.

In order to prevent the damaged gates in the plasma-based processes, antenna effect should be considered in the routing rules. The rules follow the allowable maximum antenna size or antenna ration [4-5] to determine antenna violations. For fixing violations, three general solutions are introduced as follows.

1) Insert a jumper to break the wire which has antenna violation, as shown in Fig. 2(a). This method can shorten the antenna critical wire, but each jumper needs extra vias to connect different layers. The accumulated length of gate is the sum of the segments 4-5 lengths after inserting a jumper. Therefore, the discharge path length is shorter than that of Fig. 1(b). Ho *et al.* [8-9] proposed an antenna avoidance full-chip routing with jumper insertion for spanning trees. Su and Chang [10-13] used an optimal greedy jumper insertion algorithm to minimize the number of jumpers in a spanning tree without/with obstacles. For 90nm to 250nm CMOS technologies, the range of L_{max} is given from 50μm to 200μm [10-13].

2) Assign the antenna-critical wire with higher layer, as shown in Fig. 2(b). Wu *et al.* [14-15] applied layer assignment technique and a Steiner-tree partition algorithm to fix antenna violations.

3) During layout design, embed a protection diode on each input port of a standard cell to prevent charge damage, as shown in Fig. 2(c). Huang *et al.* [16-17] transformed the antenna-critical wires, routing grids, and feasible positions into a flow network for placing protection diodes. They applied minimum-cost network-flow algorithm to solve the diode insertion and routing problems. Jiang and Chang [18-19] inserted both diodes and jumpers to achieve high antenna fixing rate due to limited routing resource in a chip. As reported in TSMC 0.25μm technology file and the experiments, the range of L_{max} is from 50μm to 100μm.

In modern high-density VLSI routings, the protection diode insertion solution may consume cell area resources and increase manufacturing cost. Moreover, the diode is always embedded even the wire which connects the cell is antenna violation free. Thus, the jumper insertion and layer assignment solutions are more popular than the diode insertion solution. In addition, the router or physical synthesis tools can be guided by the two solutions to achieve antenna violation free and higher yield.

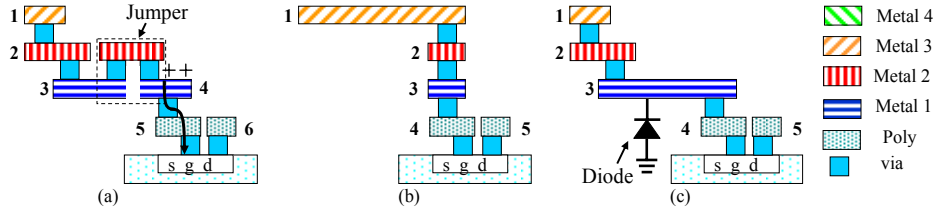

Fig. 2. Fix antenna violation with (a) jumper insertion, (b) layer assignment, and (c) embedded diode.

As VLSI fabrication technology gets into deep-submicron (DSM) era and faces process variations, interconnect delay gradually dominates chip performance. In order to prevent fault functions induced by different arrival times, minimizing clock delay and clock skew of synchronize designs are important. Moreover, the antenna effect should be considered during clock tree synthesis. Many previous works [6-17] proposed their methods applying one or more of the above three solutions to fix antenna violations for general routings. Tsai *et al.* [20] first solved the antenna effect for X-clock trees with jumper insertion and layer assignment methods (JILA). However, the approach detects all the antenna violations with the topmost layer of layer configuration instead of the discharge paths those connect gates. Consequently, the number of antenna violations will be over-estimated to result in unnecessary inserted jumpers, worse clock delay, more vias, and extra power consumption.

This work proposes a discharge-path-based antenna effect detection algorithm for a given X-architecture clock tree, and then, inserts jumpers to fix antenna violations. The inserted jumpers need extra vias to connect different metal wires and degrade the timing performance of clock tree. Besides, via timing impact cannot be neglected in modern VLSI technologies due to via delay is increasingly large. Therefore, via timing impact is considered in delay calculation. Furthermore, a wire sizing technique is applied to achieve zero skew after inserting jumpers. Experimental results indicate that the proposed algorithm can significantly achieve more improvements in the number of inserted jumpers, clock delay, skew, total vias, and power consumption compared with JI and JILA [20], respectively.

The organization of this paper is as follows. Section II formulates the problem of antenna detection and fixing for a given X-clock tree. Section III discusses the delay models considering via timing impact. Section IV presents the proposed algorithm and discusses each procedure in detail. Section V talks about the integration of the proposed algorithm and layer assignment technique for reducing the number of inserted jumpers and required vias. Experimental results on benchmarks are reported in Section VI. Finally, the conclusion and extensions of this work are given in Section VII.

II. Problem Formulation

With a view to detecting and fixing antenna violations, the metal wires not only connecting gates but also collecting charges are defined as the discharge paths and should be visited. Next, the total lengths of discharge paths are compared with the upper bound of antenna effect, L_{max}. The objective of this work is to detect antenna violations in a given X-clock tree with discharge paths, and then, all the violations can be fixed by jumper insertion method. After inserting jumpers, a wire sizing technique is used to compensate clock skew. This work applies PMXF algorithm [21] to construct the initial clock tree based on X-architecture [22]. In the layer configuration of the adopted X-architecture, the horizontal and vertical wires are assigned with metal 1 (M1) and metal 2 (M2), as well as, ±45° wires are assigned with metal 3 (M3) and metal 4 (M4), respectively.

Figure 3 illustrates antenna effect and antenna fixing with jumper insertion. In a part of X-architecture clock tree shown in Fig. 3(a), both s_i and s_j represent clock sinks. A sink is the clock input port of a standard cell or an intellectual property (IP). $P_B(s_i, s_j)$ is the bending point and requires one or two more vias for connecting different metal layers. $P_T(s_i, s_j)$ is the tapping point [23] of s_i

and s_j for achieving zero skew clock tree. Figure 3(b) is the cross-section view of Fig. 3(a). Two discharge paths, DP_1 and DP_2, through clock sinks, s_i and s_j, are presented, respectively.

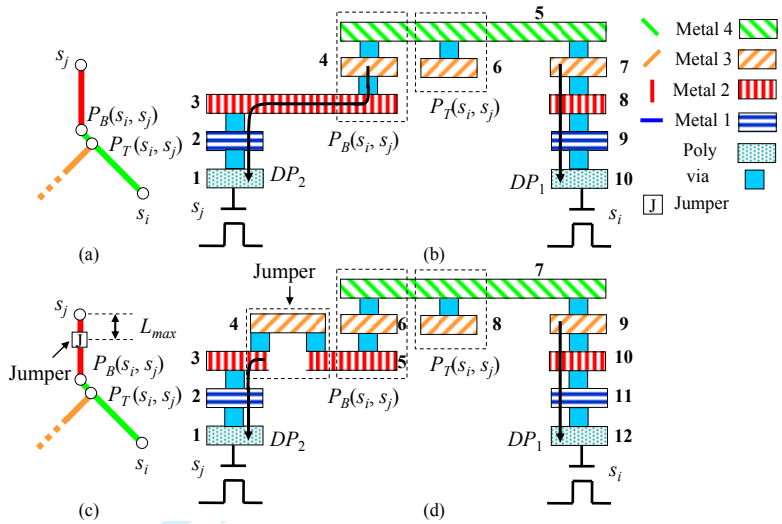

Fig. 3. Illustrations of antenna effect and antenna fixing with jumper insertion. (a) X-based wires connect two sinks, s_i and s_j, (b) the cross-section view of (a), (c) a jumper is inserted to fix the antenna violation, and (d) the cross-section view of (c).

During metallization processes, chips are manufactured "layer-by-layer." When an interconnect is being assembled, it will contain several pieces of metals. Long interconnects act as temporary capacitors and accumulate charges. The charges are gained from the energy provided by fabrication steps such as plasma etching or chemical mechanical polishing (CMP). When a metal wire connects a clock sink, the accumulated charges will discharge through the discharge path (*DP*). Meanwhile, if the length of *DP* is longer than or equal to L_{max}, the clock sink that *DP* flows through will be damaged. As shown in Fig. 3(b), DP_1 is the discharge path of clock sink s_i. The accumulated length of DP_1 of s_i, denoted as $L_{acu}(s_i)$, is the sum of the segments 9-12 lengths due to metal 4 is the topmost layer, L_{top}, of DP_1 [8-9]. Since a sink directly connects the topmost layer of discharge path, like s_i, this sink will not suffer antenna damage [9]. In contrast, s_j has another discharge path, DP_2, and $L_{acu}(s_j)$ is the sum of the segments 1-4 lengths due to metal 4 is L_{top} of DP_2. Because $L_{acu}(s_j)$ is longer than L_{max}, s_j has an antenna violation. Hence, this work applies the technique illustrated in Fig. 2(a) to insert a jumper for shortening the effective conductor length of s_j. Fig. 3(c) shows that a jumper is inserted on the wire which connects s_j and $P_B(s_i, s_j)$ to break the original discharge path. The distance between s_j and the inserted jumper is L_{max}. Figure 3(d) shows the cross-section view of Fig. 3(c). After inserting a jumper, DP_2 can be shortened to the sum of the segments 1-3 lengths. In addition, the floating segments 5 and 6 will not collect charges [9]. As aforementioned, jumpers those are implemented with vias and metal wires will consume extra routing resource and increase clock delay. Therefore, minimizing the number of inserted jumpers during fixing antenna violations is important.

Based on the above discussion, this work first visits each sink in the given X-clock tree. Then, the discharge paths of each sink are applied to determine antenna violations. Next, jumpers are inserted on the antenna-critical metal wires to fix all the violations. Meanwhile, the number of inserted jumpers should be minimized to prevent worse clock delays and power consumption. Thus, the problem of antenna detection and fixing for an X-clock tree can be formulated as follows.

Given an X-clock tree with a set of n clock sinks, S={s_1, s_2, …, s_n}, layer configuration, and the upper bound of antenna effect L_{max}, this work detects antenna effect in the given X-clock tree with discharge paths and fixes all the violations with jumper insertion.

III. Delay Model of Metal Wire Considering Via Timing Impact

Elmore delay (ED) model [24] is widely applied to calculate wire delay in clock tree synthesis. Based on ED model, the equivalent circuit of a wire e_i with width w_i and length l_i is shown in Fig. 4(a). The sheet resistance and the unit area capacitance are denoted as r and c_a, respectively. Because ED model may overestimate wire delay and limit accuracy, fitted Elmore delay (FED) model [25] is alternately proposed. Figure 4(b) shows the equivalent circuit of e_i based on FED model, where c_f is the fringing capacitance.

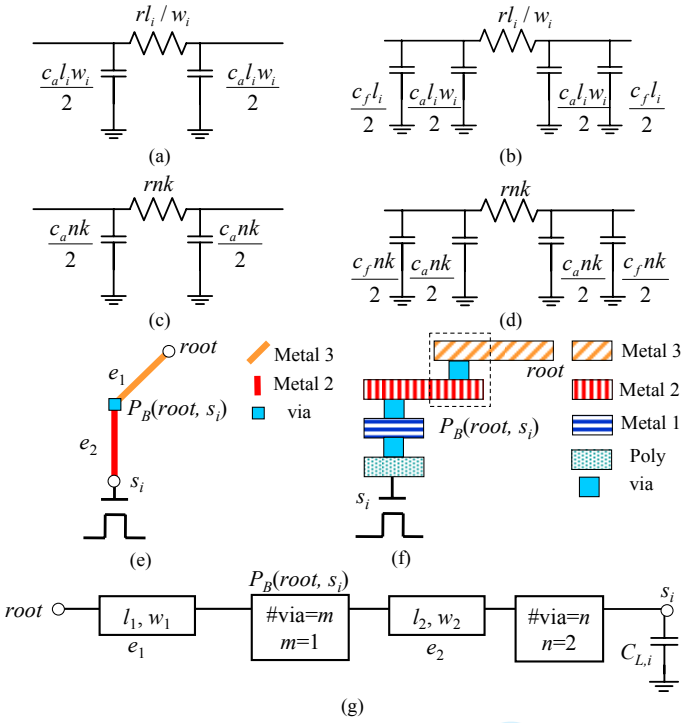

Fig. 4. The equivalent circuits of wire e_i are based on (a) ED and (b) FED models, respectively. The equivalent circuits of n-vias are based on (c) ED and (d) FED models, respectively. (e) The X-based wires connect *root*, bending point, and sink s_i, (f) the cross-section view of (e), and (g) the equivalent model of (e).

Based on ED and FED models, the delays of e_i which connects a load capacitance at sink i, denoted as $C_{L,i}$,are respectively formulated as follows.

$$Delay(i) = r\left(\frac{l_i}{w_i}\right)\left(\frac{c_a w_i l_i}{2} + C_{L,i}\right) \text{ based on ED model} \tag{2}$$

and

$$Delay(i) = r\left(\frac{l_i}{w_i}\right)\left[\frac{(Dc_a w_i + Ec_f)l_i}{2} + FC_{L,i}\right] \text{ based on FED model,} \tag{3}$$

where the coefficients D, E, and F, are obtained by curve fitting technique [25]. The load capacitance, $C_{load,i}$, is defined as the capacitance of a tree rooted at sink i or node i. The $C_{load,i}$ based on ED and FED models are respectively defined as bellow.

$$C_{load,\,i} = \begin{cases} \sum_{j \in T(i)} (c_a l_j w_j + C_{load,\,j}) \text{ for node } i \text{ based on ED model} \\ \sum_{j \in T(i)} \left[\frac{(Dc_a w_j + Ec_f) l_j}{F} + C_{load,\,j} \right] \text{ for node } i \text{ based on FED model} \\ C_{L,\,j} \text{ for sink } i \text{ based on ED and FED models,} \end{cases} \quad (4)$$

where $T(i)$ is the set of tree edges at the downstream of node i. Based on ED and FED models, the clock delays from *root* to sink k can be respectively calculated using

$$Delay(root,\,k) = \sum_{i \in P(k)} r\left(\frac{l_i}{w_i}\right)\left[\frac{c_a l_i w_i}{2} + C_{load,\,i}\right] \text{ based on ED model} \quad (5)$$

and

$$Delay(root, k) = \sum_{i \in P(k)} r\left(\frac{l_i}{w_i}\right)\left[\frac{(Dc_a w_i + Ec_f) l_i}{2} + FC_{load,\,i}\right] \text{ based on FED model,} \quad (6)$$

where $P(k)$ is the set of tree edges those are along the path from *root* to sink k.

When vias are required by a sink/node or a bending point for connecting different metal layers, delay calculation should considers via timing impact in clock tree synthesis of the present complexity VLSI design. The reason is that via delay is increasingly large and cannot be neglected any more. A via is modeled as a π-model circuit. In addition, both via resistance and capacitance are k-times of wire resistance and capacitance. The bias k is a constant and set as 2 [26]. Consequently, the equivalent circuits of n-vias based on ED and FED models are shown in Figs. 4(c) and 4(d), respectively, where n is the number of vias.

Figure 4(e) shows that the wire e_1 connects *root* and the bending point, $P_B(root, s_i)$, as well as, $P_B(root, s_i)$ connects s_i with e_2. Figure 4(f) shows the cross-section view of Fig. 4(e). The figure demonstrates that $P_B(root, s_i)$ and s_i respectively require one and two vias. Figure 4(g) shows the equivalent model of Fig. 4(e). Based on ED model, this work derives the total clock delay of wires connecting s_i up to *root* and considers via timing impact as follows.

$$\begin{aligned} &Delay(root, s_i) \text{ based on ED model} \\ &= \sum_{i \in P(s_i)} r\left(\frac{l_i}{w_i}\right)\left[\frac{c_a l_i w_i}{2} + C_{load,\,i}\right] \\ &= mkr\left(\frac{mkc_a}{2} + C_{L,i}\right) + \frac{rl_2}{w_2}\left(\frac{c_a l_2 w_2}{2} + mkc_a + C_{L,i}\right) + nkr\left(\frac{nkc_a}{2} + c_a l_2 w_2 + mkc_a + C_{L,i}\right) + \frac{rl_1}{w_1}\left(\frac{c_a l_1 w_1}{2} + nkc_a + c_a l_2 w_2 + mkc_a + C_{L,i}\right) \\ &= \frac{rc_a}{2}\left[l_2^{\,2} + 2nkl_2 w_2 + (m+n)^2 k^2 + \frac{2mkl_2}{w_2} + l_1^{\,2} + 2l_2 l_1 + \frac{2(m+n)kl_1}{w_1}\right] + \left[r\left(\frac{l_2}{w_2} + \frac{l_1}{w_1}\right) + (m+n)kr\right] C_{L,i} \\ &= \frac{rc_a}{2}\left[\frac{l}{w} + (m+n)k\right]\left[lw + (m+n)k\right] + r\left[\frac{l}{w} + (m+n)k\right] C_{L,i}\Big|_{w=w_1=w_2,\, l=l_1+l_2} \\ &= r\left[\frac{l}{w} + (m+n)k\right]\left\{\frac{c_a\left[lw + (m+n)k\right]}{2} + C_{L,i}\right\} \\ &= r\left(\frac{l}{w} + via_i k\right)\left[\frac{c_a(lw + via_2 k)}{2} + C_{L,i}\right]\Big|_{via_i = m+n}, \end{aligned} \quad (7)$$

where l_1 and w_1 are the length and width of e_1; l_2 and w_2 are the length and width of e_2; m and n are the number of vias required by $P_B(root, s_i)$ and s_i, respectively; and via_i is the total vias of s_i. Based on FED model, this work also derives the clock delay, $Delay(root, s_i)$, as follows.

$$\begin{aligned} & Delay(root, s_i) \text{ based on FED model} \\ &= \sum_{i \in P(s_i)} r\left(\frac{l_i}{w_i}\right)\left[\frac{(Dc_a w_i + Ec_f) l_i}{2} + FC_{load,\,i}\right] \\ &= r\left[\frac{l}{w} + (m+n)k\right]\left\{\frac{Dc_a\left[lw + (m+n)k\right] + Ec_f\left[l + (m+n)k\right]}{2} + FC_{L,i}\right\} \\ &= r\left(\frac{l}{w} + via_i k\right)\left[\frac{Dc_a(lw + via_i k) + Ec_f(l + via_i k)}{2} + FC_{L,i}\right]_{via_i = m+n}. \end{aligned} \tag{8}$$

IV. Discharge-Path-Based Antenna Detection and Fixing with Jumper Insertion

In order to detect and fix antenna effects in the given X-clock tree, the proposed algorithm is presented in Fig. 5 and called as PADJI (discharge-path-based antenna effect detection and fixing with jumper insertion). For a given X-clock tree with a set of n clock sinks, $S=\{s_1, s_2, \ldots, s_n\}$, the layer configuration, and the upper bound of antenna effect L_{max}, the objective of this work is to detect antenna effect with discharge paths, and then, fix antenna violations with minimum jumpers.

Algorithm: ***PADJI*** */*Discharge-path-based effect antenna detection and fixing with jumper insertion */*
Input: *A given X-clock tree with a set of sinks, S, layer configuration, and the upper bound of antenna effect, L_{max}*
Output: *A zero skew X-clock tree with antenna violation free and minimum jumpers*

```
1  for each s_i in S
2  { if( e(s_i).layer = the L_top of layer configuration )
3    { s_i is antenna violation free; }
4    else
5    { DP(s_i) = DischargePath(s_i); /* Obtain a set of discharge paths of s_i, DP(s_i). */
6      for each dp_k in DP(s_i)
7      { e_total(s_i) = e_total(s_i) + dp_k – Duplicate(∀e_m∈dp_k, e_total(s_i));
         /* Obtain a set of segments those accumulate charges and may damage s_i. */
8        if( |e_total(s_i)| ≥ L_max )
9        { InsertJumper(s_i); /* Insert a jumper to fix antenna violation occurred on s_i. */
10         break;
11       }
12     }
13   }
14   WireSizing(s_i); /* Adjust wire width to maintain zero skew. */
15 }
```

Fig. 5. The proposed PADJI algorithm.

The proposed PADJI algorithm consists of several procedures in the *for* loop, and its overview is given as follows. For each sink $s_i \in S$ in the given X-clock tree, if the layer of a metal wire which connects s_i is the top layer, L_{top}, of layer configuration, s_i is antenna violation free. Otherwise, the procedure $DischargePath(s_i)$ will visit all the discharge paths those relate s_i to obtain $DP(s_i)$. $DP(s_i)$ is a set of discharge paths of s_i and may collect charges to damage s_i. Next, the proposed algorithm moves the metal wires of each discharge path, $dp_k \in DP(s_i)$, into the accumulated wires of s_i, $e_{total}(s_i)$. The length of $e_{total}(s_i)$, $|e_{\text{total}}(s_i)|$, is used to detect antenna violations. Because each dp_k may contain several pieces of edges, $e_m \in dp_k$, the duplicate edge is excluded to avoid length overestimation and unnecessary jumpers. If $|e_{total}(s_i)|$ is equal to or longer than L_{max}, the procedure $InsertJumper(s_i)$ will insert a jumper to fix the antenna violation of s_i. Finally, the procedure $WireSizing(s_i)$ is employed to adjust the width of wire connecting s_i

for compensating clock skew. After introducing the overview of the proposed PADJI algorithm, each procedure will be described in the following subsections in detail.

A. Discharge-Path-Based Antenna Detection – DischargePath(s_i)

Figure 6(a) shows a part of X-clock tree. For a pair of sinks s_i and s_j, the procedure *DischargePath*(s_i) visits all the discharge paths those relate the given sink s_i. Because a discharge path contains several metal edges, the procedure then classify these edges and inserts the effective ones into $DP(s_i)$. The notations those are adopted by the *DischargePath* procedure are introduced first as follows.

This work applies the binary tree structure to model the X-clock tree. Figure 6(b) shows the binary topology of Fig. 6(a). For calculating the number of inserted vias and estimating via delays, this work adopts two via definitions proposed in [27]. In a binary X-clock tree, the via which is required by a leaf node (sink) or an internal node is defined as a node via (NV) and represented by a square. When a via locates at a bending point of a tree edge, the via is defined as an edge via (EV) and represented by a triangle. The number of NVs or EVs is equal to the difference of metal layers.

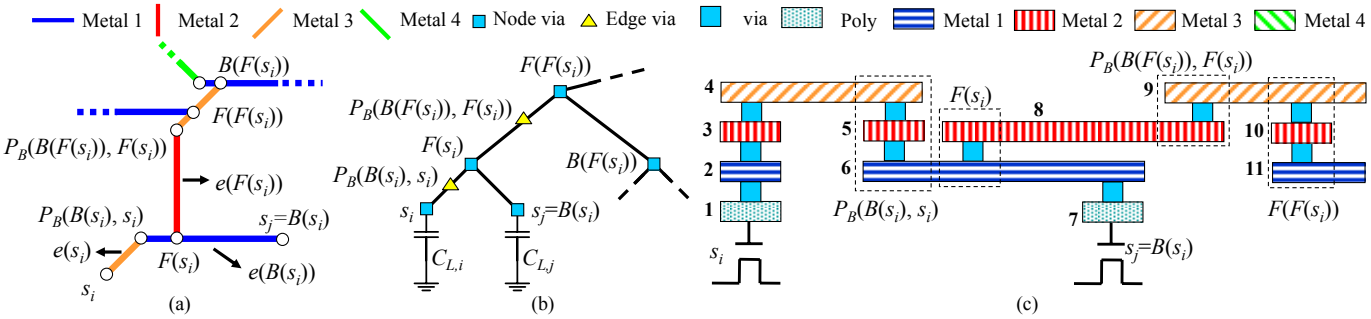

Fig. 6. (a) A part of X-clock tree, (b) the binary topology of (a) with two via definitions, and (c) the cross-section view of (a).

Based on the properties of binary X-clock tree shown in Fig. 6(b), sink s_j is s_i's brother and denoted as $B(s_i)$. The tapping point of s_i and s_j is their father and denoted as $F(s_i)$. The bending point of s_i and s_j (also called $B(s_i)$) is denoted as $P_B(B(s_i), s_i)$. In order to detect antenna effects with discharge paths, this work defines the connecting edges and their layers as follows. The edge connecting s_i and $P_B(B(s_i), s_i)$ is denoted as $e(s_i)$, and $e(s_i)$.layer represents the layer of $e(s_i)$. The edge connecting $B(s_i)$ and $P_B(B(s_i), s_i)$ is denoted as $e(B(s_i))$, and the layer of $e(B(s_i))$ is denotes as $e(B(s_i))$.layer. The edge connecting $F(s_i)$ and its bending point $P_B(B(F(s_i)), F(s_i))$ is denoted as $e(F(s_i))$, and $e(F(s_i))$.layer represents the layer of $e(F(s_i))$. Figure 6(c) shows the cross-section view of Fig. 6(a). Because $e(s_i)$.layer is M3, the number of NVs is three for connecting poly layer up to M3. For $P_B(B(s_i), s_i)$, the number of EVs is two for connecting M1 ($e(B(s_i))$.layer) up to M3 ($e(s_i)$.layer).

After introducing the adopted notations, the proposed *DischargePath* procedure is presented in Fig. 7. For a given sink s_i, the top layer of s_i, $L_{top}(s_i)$, is determined by the highest layer of $e(s_i)$.layer and $e(B(s_i))$.layer. Then, this work classifies and presents three cases of s_i as follows for getting a set of discharge paths of s_i, $DP(s_i)$.

Case1: $e(s_i)$.layer > $e(B(s_i))$.layer (lines 2-7): In the case, the charges those are accumulated on $e(B(s_i))$ will not damage s_i. Hence, the procedure have to check the layer and location of $F(s_i)$. When $e(F(s_i))$.layer is higher than $L_{top}(s_i)$ and $F(s_i)$ locates on $e(s_i)$, $e(s_i)$ will be inserted into dp_k of $DP(s_i)$. In addition, the exceptional s_i is antenna violation free.

Case2: $e(s_i)$.layer = $e(B(s_i))$.layer (lines 8-15): In the case, both $e(s_i)$ and $e(B(s_i))$ may collect charges to damage s_i, so the procedure has to check $F(s_i)$.layer. If $e(F(s_i))$.layer is higher than $L_{top}(s_i)$, only $e(s_i)$ and $e(B(s_i))$ will be inserted into dp_k of $DP(s_i)$. When

$e(F(s_i))$.layer is equal to $L_{top}(s_i)$, the number of effective conductors will be increasing and collect more charges. Consequently, the procedure not only inserts $e(s_i)$ and $e(B(s_i))$ into dp_k of $DP(s_i)$ but also gives $F(s_i)$ to the *DischargePath* procedure for determining the extra edges. In addition, the exceptional s_i is antenna violation free.

Case3: $e(s_i)$.layer < $e(B(s_i))$.layer (lines 16-27): When $e(F(s_i))$.layer is higher than $L_{top}(s_i)$, the procedure will check the location of $e(F(s_i))$. If $F(s_i)$ locates on $e(s_i)$, only $e(s_i)$ will be inserted into dp_k of $DP(s_i)$. Otherwise, both $e(s_i)$ and $e(B(s_i))$ will be inserted into dp_k of $DP(s_i)$. When $e(F(s_i))$.layer satisfies the constrain, [$e(s_i)$.layer, $L_{top}(s_i)$) and $F(s_i)$ locates on $e(s_i)$, the procedure not only inserts $e(s_i)$ into dp_k of $DP(s_i)$ but also gives $F(s_i)$ to the *DischargePath* procedure for determining the extra edges. The constrain means that $e(F(s_i))$.layer is higher than or equal to $e(s_i)$.layer and $e(F(s_i))$.layer is lower than $e(B(s_i))$.layer. In addition, only $e(s_i)$ will be inserted into dp_k of $DP(s_i)$ for the exceptional s_i.

```
Procedure: DischargePath(s_i)
Input: A sink or node s_i
Output: A set of discharge paths of s_i
1  L_top(s_i) = Max( e(s_i).layer, e(B(s_i)).layer );  /* Determine the L_top of s_i. */
2  if( e(s_i).layer > e(B(s_i)).layer ) /* Case 1 */
3  {  if( e(F(s_i)).layer > L_top(s_i) and F(s_i) locates on e(s_i) )
4     {  Insert e(s_i) into dp_k of DP(s_i);  }
5     else
6     {  s_i is antenna violation free;  }
7  }
8  else if( e(s_i).layer = e(B(s_i)).layer )  /* Case 2 */
9  {  if( e(F(s_i)).layer > L_top(s_i) )
10    {  Insert e(s_i) and e(B(s_i)) into dp_k of DP(s_i);  }
11    else if( e(F(s_i)).layer = L_top(s_i) )
12    {  Insert e(s_i), e(B(s_i)), and DischargePath(F(s_i)) into dp_k of DP(s_i);  }
13    else
14    {  s_i is antenna violation free;  }
15 }
16 else  /* Case 3, e(s_i).layer < e(B(s_i)).layer */
17 {  if( e(F(s_i)).layer > L_top(s_i) )
18    {  if( F(s_i) locates on e(s_i) )
19       {  Insert e(s_i) into dp_k of DP(s_i);  }
20       else
21       {  Insert e(s_i) and e(B(s_i)) into dp_k of DP(s_i);  }
22    }
23    else if( e(s_i).layer ≤ e(F(s_i)).layer < L_top(s_i) and F(s_i) locates on e(s_i) )
24    {  Insert e(s_i) and DischargePath(F(s_i)) into dp_k of DP(s_i);  }
25    else
26    {  Insert e(s_i) into dp_k of DP(s_i);  }
27 }
```

Fig. 7. The proposed *DischargePath* procedure for determining a set of discharge paths of the given sink s_i.

For clearly explaining the proposed *DischargePath* procedure, Fig. 8 presents a simple case. Figure 8(a) shows a part of initial X-clock tree without considering antenna effect, where s_9, s_{10}, s_{13}, and s_{14} are sinks, s_{21}, s_{23}, and s_{27} are the tapping points of (s_9, s_{10}), (s_{13}, s_{14}), and (s_{21}, s_{23}), respectively. First, s_{13} is taken as an example and given to the *DischargePath* procedure for detecting the antenna violation. Meanwhile, s_{14} and s_{23} are respectively denoted as $B(s_{13})$ and $F(s_{13})$ due to s_{14} is paired with s_{13} and s_{23} is the tapping point of (s_{13}, s_{14}). Because $e(s_{13})$.layer and $e(B(s_{13}))$.layer are M1 and M3, the procedure *DischargePath*(s_{13}) can determine that $L_{top}(s_{13})$ is M3 and s_{13} satisfies the condition of Case3. Next, $e(F(s_{13}))$.layer is M2, and $F(s_{13})$ not only satisfies the interval constrain of [M1, M3) but also locates on $e(s_{13})$. Hence, the procedure inserts $e(s_{13})$ into dp_1 of $DP(s_{13})$ and gives $F(s_{13})$ to the *DischargePath* procedure for determining the extra edges. Then, the procedure *DischargePath*($F(s_{13})$) (or denotes as *DischargePath*(s_{23})) gets that $e(s_{23})$.layer is M2, $e(B(s_{23}))$.layer is M4, and $e(F(s_{23}))$.layer is M1. Therefore, *DischargePath*($F(s_{13})$) just returns $e(s_{23})$. Finally, we get that $DP(s_{13})$ contains $e(s_{13})$ and $e(s_{23})$. For the other sink s_{14}, Fig. 8(a) shows that $e(s_{14})$.layer is M3, $e(B(s_{14}))$.layer is M1, and $F(s_{14})$ does not locates on $e(s_{14})$. Consequently, s_{14} satisfies the condition of Case1 and has no

antenna violation. The antenna effects of other two sinks s_9 and s_{10} can be detected by the same process, respectively. After launching the *DischargePath* procedures, Fig. 8(b) shows that two jumpers are respectively inserted on $e(s_{13})$ and $e(s_{10})$ to fix the antenna violations. Notably, the distance between the antenna-critical sink s_{13} (or s_{10}) and the inserted jumper is L_{max}.

For verifying the proposed *DischargePath* procedure, Fig. 8 (c) presents the cross-section view of Fig. 8(a). The set of discharge paths of s_{13}, $DP(s_{13})$, contains two discharge paths, dp_1 and dp_2. In this work, each segment length is X-architecture distance not Manhattan distance. For $DP(s_{13})$, dp_1 length is the sum of the segments 5-7 lengths, as well as, dp_2 length is the sum of the segments 6-8 and 10 lengths. Therefore, $|e_{total}(s_{13})|$ is the sum of the segments 5-8 and 10 lengths whereas the duplicate segments have been removed to avoid length overestimation. Next, segments 5, 7, and 10 can be ignored due to they are metallised for vias. Finally, $|e_{total}(s_{13})|$ can be simplified as the sum of the segments 6 and 8 lengths. Note that the lengths of segments 6 and 8 are equal to $|e(s_{13})|$ and $|e(s_{23})|$, respectively. Because $|e_{total}(s_{13})|$ is longer than L_{max}, an antenna violation is occurred. After inserting a jumper to fix the violation, the original segment 6 shown in Fig. 8(c) is broken into two segments 6 and 9 shown in Fig. 8(d). It is obvious that the length of $e_{total}(s_{13})$ is short enough to avoid antenna violation. For the other sink s_{14}, it has no antenna violation due to $e(s_{14})$.layer is M3 which is the L_{top} of dp_1 in $DP(s_{14})$. For sinks s_9 and s_{10}, another jumper is inserted on $e(s_{10})$ to fix the antenna violation.

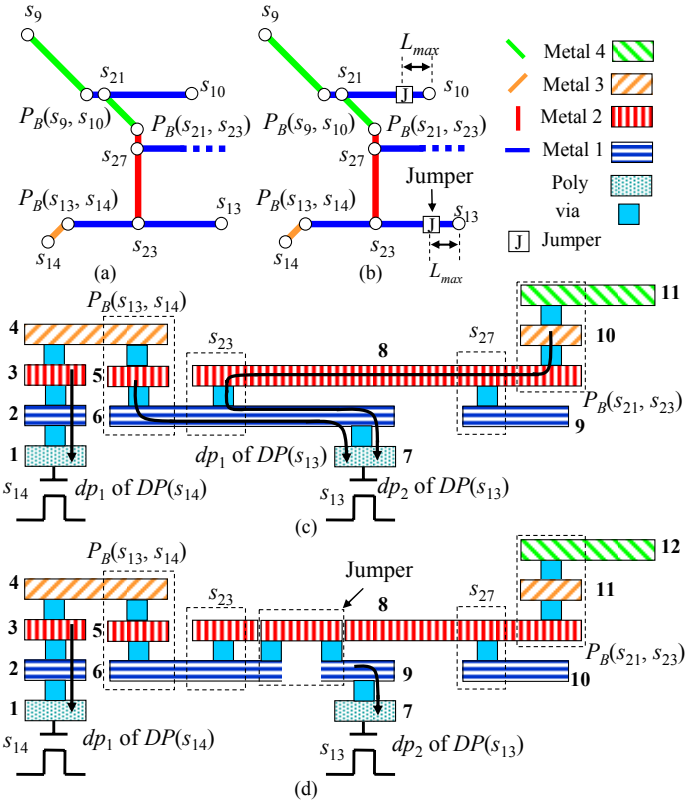

Fig. 8. (a) A part of initial X-clock tree without considering antenna violation, (b) two jumpers are respectively inserted to fix the antenna violations of (a), (c) the cross-section view of (a), and (d) the cross-section view of (b) with jumper insertion.

B. *Adjust Wire Width after Jumper Insertion – WireSizing(s_i)*

After inserting jumpers to fix antenna violations, the number of vias is increased and results in clock skew. Hence, the proposed PADJI algorithm applies a wire sizing technique to achieve zero skew. For a pair of sinks (s_{13}, s_{14}) shown in Fig. 9(a), their tapping point s_{23} and the inserted jumper are taken as an example to explain the *WireSizing* procedure. After inserting a jumper to fix the

antenna violation of s_{13}, the number of required vias of s_{13}, $\#via_{13}$, is increased from 1 to 3 (see the cross-section view in Fig. 8(d)). The reason is that s_{13} and the inserted jumper respectively require one and two vias. In order to size the width of wire $e(s_{23}, s_{13})$ and achieve zero skew, the increased vias should be considered in delay calculation.

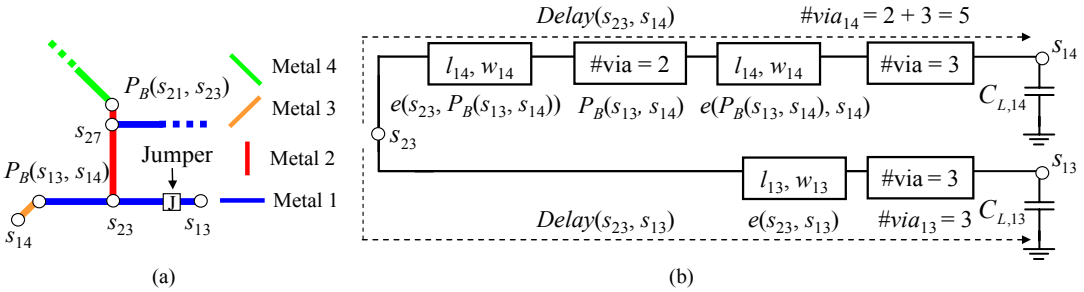

Fig. 9. (a) A part of X-clock tree with jumper insertion and (b) the equivalent model of (a) for determining the width of wire $e(s_{23}, s_{13})$.

This work shows the equivalent model of Fig. 9(a) in Fig. 9(b) for explaining how to derive the width of $e(s_{23}, s_{13})$. First, s_{14} is antenna violation free requires no jumper. Therefore, the number of vias of s_{14}, $\#via_{14}$, is fixed and equal to the sum of three NVs required by sink s_{14} and two EVs required by bending point $P_B(s_{13}, s_{14})$. Based on the ED model, $Delay(s_{23}, s_{14})$ that considers via timing impact can be calculated by (7). Next, this work formulates $Delay(s_{23}, s_{13})$ and lets the two delays be equal for deriving the width of $e(s_{23}, s_{13})$, w_{13}, as follows.

$$
\begin{aligned}
&Delay(s_{23},s_{13}) = Delay(s_{23},s_{14}), \text{ based on ED model} \\
&r\left(\frac{l_{13}}{w_{13}} + via_{13}k\right)\left[\frac{c_a(l_{13}w_{13} + via_{13}k)}{2} + C_{L,13}\right] - Delay(s_{23},s_{14}) = 0 \\
&\frac{rvia_{13}kc_al_{13}}{2}w_{13} + \left(\frac{c_avia_{13}k}{2} + C_{L,13}\right)\frac{rl_{13}}{w_{13}} + \left[\frac{rc_a(via_{13}k)^2}{2} + \frac{rc_al_{13}^2}{2} + rvia_{13}kC_{L,13} - Delay(s_{23},s_{13})\right] = 0 \qquad (9) \\
&\frac{rvia_{13}kc_al_{13}}{2}w_{13}^2 + \left\{r\left[\frac{c_al_{13}^2}{2} + via_{13}k\left(\frac{c_avia_{13}k}{2} + C_{L,13}\right)\right] - Delay(s_{23},s_{14})\right\}w_j + \left[rl_{13}\left(\frac{c_avia_{13}k}{2} + C_{L,13}\right)\right] = 0 \\
&w_{13} = \left(-b \pm \sqrt{b^2 - 4ac}\right)/2a,
\end{aligned}
$$

where

$$
\begin{aligned}
a &= \frac{rvia_{13}kc_al_{13}}{2} \\
b &= r\left[\frac{c_al_{13}^2}{2} + via_jk\left(\frac{c_avia_{13}k}{2} + C_{load,13}\right)\right] - Delay(s_{23},s_{13}) \\
c &= rl_{13}\left(\frac{c_avia_{13}k}{2} + C_{load,13}\right).
\end{aligned}
$$

The JI (Jumper Insertion) algorithm proposed in the study in [20] applies FED model to calculate delay but does not achieve zero skew. In order to fairly compare JI with the proposed PADJI algorithm, this work also derives w_{13} as (10) to make $Delay(s_{23}, s_{13})$ and $Delay(s_{23}, s_{14})$ be equal under FED model.

$$Delay(s_{23}, s_{13}) = Delay(s_{23}, s_{14}), \text{ based on FED model}$$
$$r\left(\frac{l_{13}}{w_{13}} + via_{13}k\right)\left[\frac{Dc_a(l_{13}w_{13} + via_{13}k) + Ec_f(l_{13} + via_{13}k)}{2} + FC_{L,13}\right] = Delay(s_{23}, s_{14}) \quad (10)$$
$$w_{13} = \left(-b \pm \sqrt{b^2 - 4ac}\right)/2a,$$

where

$$a = \frac{rvia_{13}kDc_a l}{2}$$
$$b = r\left[\frac{Dc_a l + Ec_f via_{13}k}{2} l + via_{13}k\left(\frac{Dc_a + Ec_f}{2} via_{13}k + FC_{L,13}\right)\right] - Delay(s_{23}, s_{14})$$
$$c = rl\left[\frac{Dc_a via_{13}k + EC_f(l + via_{13}k)}{2} + C_{L,13}\right].$$

C. Power Estimation

This work estimates the power consumption for a whole clock tree by charging the capacitance $C_{load,\,i}$ defined in (4). Therefore, we can get that both the wire capacitances and load capacitances are charged. Finally, the total power consumption in a clock tree is formulated as follows.

$$\text{Power} = \sum_{\forall e_i} C_{load,\,i} F_{clk} V_{dd}{}^2, \quad (11)$$

where F_{clk} and V_{dd} are the clock frequency and supplying voltage, respectively.

D. Time Complexity Analysis

For a given X-clock tree with a set of n-sinks, each sink will be visited by the proposed PADJI algorithm. Hence, the *for* loop runs in $O(n)$. In the *for* loop, the *DischargePath* procedure obtains a set of discharge paths for each sink. Because the worst case is that each sink contains k paths, the procedure runs in $O(k)$. The rest of procedures in the *for* loop can insert a jumper to fix antenna violation and adjust wire width to achieve zero skew with *InsertJumper* and *WireSizing* in constant time, respectively. Finally, the time complexity of the proposed PADJI algorithm is $O(n^2)$.

E. Verification of the Proposed PADJI Algorithm

For comparative study, PMXF [21] algorithm is first applied to construct an initial X-clock tree without antenna effect consideration. The X-clock tree contains a set of 16 clock sinks, $S = \{s_1, s_2, \ldots, s_{16}\}$, dumped from the IBM benchmark *r*1 [30]. The routing result of X-clock tree constructed by PMXF is shown in Fig. 10(a). Next, JI [20] and the proposed PADJI algorithms respectively load the initial X-clock tree to calculate the accumulated length of each sink for detecting antenna violations. Both JI and PADJI use the FED model with 130nm fabrication parameters to calculate delay. In addition, the upper bound of antenna effect L_{max} is given as 200μm. Finally, the two algorithms insert jumpers to fix all the antenna violations in the initial X-clock tree. Figures 10(b) and 10(c) respectively present the antenna violation free X-clock trees.

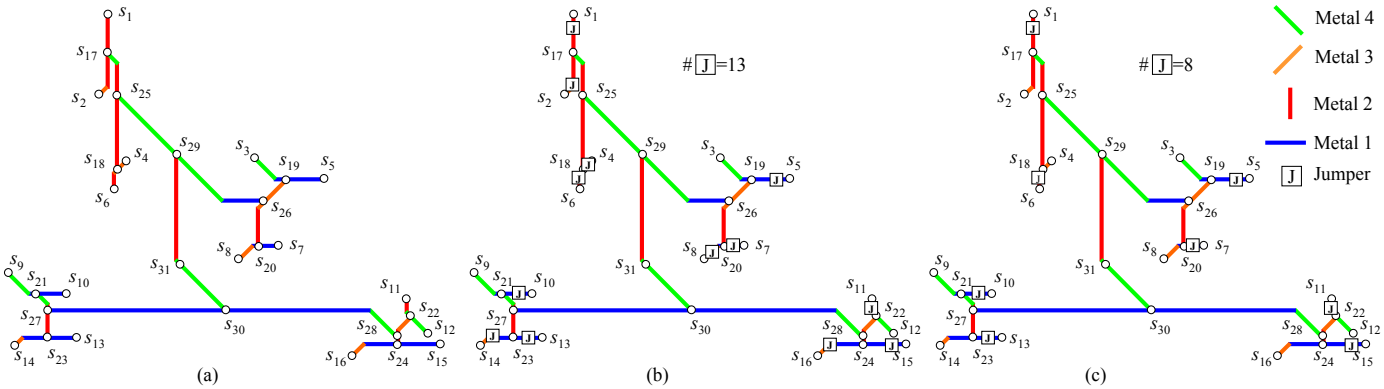

Fig. 10. (a) The initial 16-sink X-clock tree is constructed by PMXF. Antenna violation free X-clock trees are achieved by (b) JI [20] and (c) PADJI algorithms, respectively.

Table I lists the comparison of JI and PADJI algorithms in the accumulated length of each sink, $|e_{total}(s_i)|$. Columns 1, 2, 7 and 8 give the label and coordinate of each sink and node. Column 3 gives the load capacitance, C_L, of each sink. Columns 4, 5 and the last two columns give the label and coordinate of bending point, $P_B(B(s_i), s_i)$, and tapping point, $F(s_i)$, of each pair of sinks and nodes. Note that $|e_{total}(s_i)|$ listed in column 6 are determined by JI and PADJI, respectively. As shown in Table I, JI algorithm detected that there are 13 accumulated lengths of sinks longer than L_{max}. In contrast, PADJI detected eight.

TABLE I
COORDINATES, BENDING POINTS, TAPPING POINTS, AND ACCUMULATED LENGTHS OF THE 16 SINKS AND NODES IN THE GIVEN X-CLOCK TREE

Sink	Coordinate	C_L(pF)	$P_B(B(s_i), s_i)$	$F(s_i)$	$\|e_{total}(s_i)\|$ (μm)		Node	Coordinate	$P_B(B(s_i), s_i)$	$F(s_i)$
					JI [20]	PADJI				
s_1	(18596.0, 36216.0)	0.078	$P_B(s_1, s_2)$	s_{17}	6337.0	6337.0	s_{17}	(18596.0, 32834.6)	$P_B(s_{17}, s_{18})$	s_{25}
s_2	(17841.0, 29124.0)	0.035	(18596.0, 29879.0)	(18596.0, 32834.6)	1067.7	0.0	s_{18}	(19444.0, 22366.0)	(19444.0, 31986.6)	(19444.0, 29044.8)
s_4	(20313.0, 23235.0)	0.078	$P_B(s_4, s_6)$	s_{18}	1418.4	0.0	s_{19}	(34327.3, 21556.0)	$P_B(s_{19}, s_{20})$	s_{26}
s_6	(19310.0, 20716.0)	0.035	(19310.0, 22232.0)	(19444.0, 22366.0)	1516.0	1516.0	s_{20}	(31950.0, 15681.0)	(31950.0, 19178.7)	(32445.4, 19674.1)
s_3	(31657.0, 23412.0)	0.046	$P_B(s_3, s_5)$	s_{19}	0.0	0.0	s_{21}	(12226.2, 11439.0)	$P_B(s_{21}, s_{23})$	s_{27}
s_5	(37908.0, 21556.0)	0.037	(33513.0, 21556.0)	(34327.3, 21556.0)	4395.0	4395.0	s_{23}	(13276.8, 7520.0)	(13276.8, 10388.4)	(13276.8, 10004.2)
s_7	(33860.0, 15681.0)	0.044	$P_B(s_7, s_8)$	s_{20}	2111.0	5608.7	s_{22}	(45633.5, 9408.5)	$P_B(s_{22}, s_{24})$	s_{28}
s_8	(30550.0, 14482.0)	0.045	(31749.0, 15681.0)	(31950.0, 15681.0)	1695.6	0.0	s_{24}	(44404.8, 6943.0)	(44404.8, 8179.9)	(44404.8, 7608.0)
s_9	(10014.0, 13282.0)	0.058	$P_B(s_9, s_{10})$	s_{21}	0.0	0.0	s_{25}	(19444.0, 29044.8)	$P_B(s_{25}, s_{26})$	s_{29}
s_{10}	(14975.0, 11439.0)	0.075	(11857.0, 11439.0)	(12226.2, 11439.0)	3118.0	3118.0	s_{26}	(32445.4, 19674.1)	(28814.6, 19674.1)	(24724.4, 23764.3)
s_{13}	(15850.0, 7520.0)	0.040	$P_B(s_{13}, s_{14})$	s_{23}	4778.0	7646.4	s_{27}	(13276.8, 10004.2)	$P_B(s_{27}, s_{28})$	s_{30}
s_{14}	(10822.0, 7270.0)	0.041	(11072.0, 7520.0)	(13276.8, 7520.0)	353.6	0.0	s_{28}	(44404.8, 7608.0)	(42008.6, 10004.2)	(29162.4, 10004.2)
s_{11}	(45204.0, 10920.0)	0.067	$P_B(s_{11}, s_{12})$	s_{22}	1082.0	1082.0	s_{29}	(24724.4, 23764.3)	$P_B(s_{29}, s_{30})$	s_{31}
s_{12}	(47147.0, 7895.0)	0.032	(45204.0, 9838.0)	(45633.5, 9408.5)	0.0	0.0	s_{30}	(29162.4, 10004.2)	(24724.4, 14442.2)	(24760.9, 14405.6)
s_{15}	(48326.0, 6943.0)	0.064	$P_B(s_{15}, s_{16})$	s_{24}	6913.0	8149.8	s_{31}	(24760.9, 14405.6)	-	-
s_{16}	(40398.0, 5928.0)	0.032	(41413.0, 6943.0)	(44404.8, 6943.0)	1435.4	0.0				
				#Violations	13	8				

In order to clearly demonstrate the difference between JI and the proposed PADJI algorithms, this work extracts two sinks, (s_7, s_8), from Fig. 10 and takes them as an example. Figures 11(a) and 11(d) show the local view and cross-section view of (s_7, s_8) in the initial X-clock tree constructed by PMXF.

JI algorithm is first discussed as follows. When a sink connects the L_{top} of layer configuration, JI thinks that the sink is antenna violation free. In contrast, if the total length of wires connecting a sink is longer than L_{max}, JI will insert a jumper connecting L_{top} to fix the violation. In this example, JI takes M4 as L_{top}. In Fig. 11(d), JI obtains that the accumulated length of s_7, $|e_{total}(s_7)|$, is the sum of the segments 5-7 lengths. Because segments 5 and 7 are metallised for vias, Table I lists that $|e_{total}(s_7)|$ is as the segment 6 length

as 2110μm. Next, JI inserts a jumper to fix the antenna violation of s_7 due to $|e_{total}(s_7)|$ is longer than L_{max}. The fixed routing result is shown in Fig. 11(b). Furthermore, the cross-section view of Fig. 11(b) is presented in Fig. 11(e). The jumper required by s_7 can shorten $|e_{total}(s_7)|$ but needs three vias to connect segments 8 and 11, as well as, the other three vias to connect segments 11 and 9. For the other sink s_8 listed in Table I, $|e_{total}(s_8)|$ is the sum of the segments 1-4 lengths and approximated as the segment 4 length as 1695.6μm. JI also inserts another jumper to fix the antenna violations of s_8 due to $|e_{total}(s_8)|$ is longer than L_{max}. As shown in Fig. 11(e), the jumper inserted on segment 4 can also shorten $|e_{total}(s_8)|$ but needs two extra vias to connect segments 4-5 and segments 5-6, respectively. Finally, JI totally inserts two jumpers to fix antenna violations and consumes extra eight vias for (s_7, s_8).

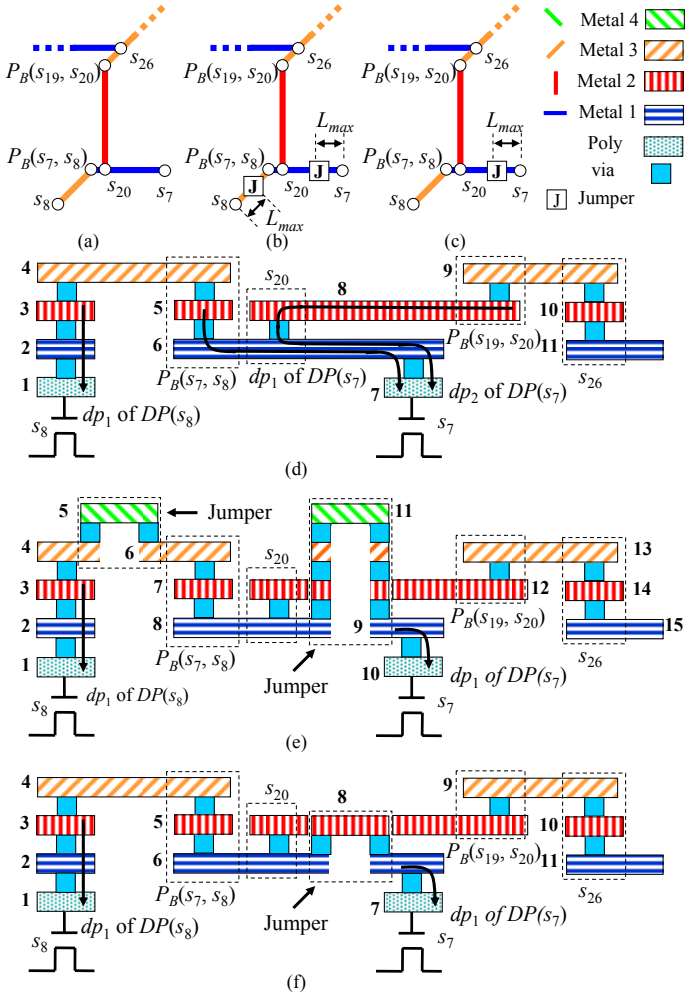

Fig. 11. Sinks s_7 and s_8 are taken as an example. (a) The local view of initial X-clock tree which has antenna violations, (b) two jumpers required by JI algorithm are inserted to fix antenna violations of (a), (c) only one jumper is required and inserted by the proposed PADJI algorithm. (d) The cross-section view of (a), where s_7 and s_8 respectively contain two and one discharge path, (e) the cross-section view of (b), and (f) the cross-section view of (c).

Second, the proposed PADJI algorithm which applies discharge paths to detect antenna violations are discussed as follows. As shown in Fig. 11(d), sinks s_7 and s_8 respectively have two and one discharge path. For $DP(s_7)$, dp_1 is the first discharge path of s_7 and consists of segments 5-7. The second one, dp_2, consists of segments 6-8. Segment 8 connecting s_{20} and $P_B(s_{19},s_{20})$ can collect charges to damage s_7. However, JI does not consider the segment, but PADJI does. For sink s_8, the top layer of segments which connect s_8 is M3. The discharge path of s_8, dp_1 of $DP(s_8)$, only contains segments 1-3. In contrast, segment 4 is overestimated by JI when detecting s_8. So, JI inserts an unnecessary jumper to fix the antenna violations of s_8. As aforementioned, that is the reason

PADJI detects each discharge path of a given sink to precisely obtain the accumulated length and determine antenna violation.

After using the cross-section view to explain antenna effect and antenna fixing with jumper insertion, the proposed *DischargePath* procedure will be discussed from the viewpoint of routing result. For a pair of sinks s_7 and s_8, their bending point and tapping point s_{20} are respectively denoted as $B(s_7)$ and $F(s_7)$. As shown in Fig. 11(a), $e(s_7)$.layer is M1 and $e(B(s_7))$.layer is M3. The procedure *DischargePath*(s_7) in PADJI obtains that $L_{top}(s_7)$ is M3 and s_7 satisfies the condition of Case3. Because $e(F(s_7))$.layer is M2 and $F(s_7)$ not only satisfies the interval constraint of [M1, M3) but also locates on $e(s_7)$, the procedure inserts $e(s_7)$ into $DP(s_7)$ and gives $F(s_7)$ to the *DischargePath* procedure for determining the extra edges. Next, the procedure *DischargePath*($F(s_7)$) (or denoted as *DischargePath*(s_{20})) gets that $e(s_{20})$.layer is M2, $e(B(s_{19}))$.layer is M3, and $e(F(s_{26}))$.layer is M1. Therefore, s_{20} also satisfies the condition of Case3, and *DischargePath*($F(s_{20})$) returns $e(s_{20})$. We get that $DP(s_7)$ contains $e(s_7)$ and $e(s_{20})$, as well as, $|e_{total}(s_7)|$ can be approximated as the sum of $|e(s_7)|$ and $|e(s_{20})|$ as 5608.7μm. Notably, $e(s_7)$ and $e(s_{20})$ are segments 6 and 8, respectively. For sink s_8, the procedure *DischargePath*(s_8) in PADJI obtains that s_8 satisfies the condition of Case1 and has no antenna violation. Finally, only one jumper is inserted to fix the antenna violation of s_7, as shown in Fig. 11(c). The cross-section view of Fig. 11(c) is presented in Fig. 11(f). The proposed PADJI algorithm just inserts a jumper with two extra vias to connect M2. Compared with JI algorithm, PADJI takes less jumpers and vias for fixing antenna violation.

Table II lists the experimental results on the 16-sink X-clock trees those are respectively constructed and fixed by PMXF, JI, and PADJI algorithms. Note that the ratio is defined as JI/PMXF or PADJI/PMXF. Total vias is the sum of node vias and edge vias. The upper bound of antenna effect, L_{max}, is given as 200μm. In the table, JI inserted five more jumpers than PADJI to fix antenna violations. For the column "Skew" in Table II, PMXF constructed an X-clock tree and considered via timing impact. However, the clock skew induced by via was ignored by PMXF and JI. Consequently, this work applies the wire sizing technique to compensate clock skew. As listed in Table II, applying discharge path to detect antenna violation can avoid accumulated length over-estimation and unnecessary inserted jumpers. It is not surprising that PADJI had better performance of 0.0604%, 0.048%, and 39.1304% than JI in terms of delay, power, and total vias, respectively.

TABLE II

COMPARISON OF PMXF [21], JI [20], AND THE PROPOSED PADJI ALGORITHMS ON THE 16-SINK X-CLOCK TREES BASED ON 130NM FED MODEL AND L_{MAX}=200μM

#Sinks	#Inserted Jumpers		Delay (μs)			Skew (μs)			Power (W)			Total vias		
	JI	PADJI	PMXF	JI	PADJI	PMXF	JI	PADJI	PMXF	JI	PADJI	PMXF	JI	PADJI
16	13	8	0.021500	0.021518	0.021505	0.000009	0.000011	0	0.002083	0.002085	0.002084	92	144	108
Ratio	-	-	-	1.000837	1.000233	-	-	-	-	1.000960	1.000480	-	1.565217	1.173913

V. INTEGRATION OF PADJI AND LAYER ASSIGNMENT TECHNIQUE

Layer assignment technique [14-15] is another method for fixing antenna violations. An antenna-critical wire has to be assigned with a different layer to avoid antenna effect. For example, the initial X-clock tree shown in Fig. 11(a) can be totally routed with single layer M4 which is the L_{top} of layer configuration. Then, each sink needs four NVs to connect poly layer up to M4. In addition, the number of EVs is zero due to all the metal wires connecting sinks or nodes are routed with single layer. Therefore, the number of required vias is 64 and much less than 144 and 108 required by JI and PADJI algorithms, respectively. We can also get that layer assignment technique can obtain the optimal routing result of X-clock tree without antenna violations. However, the single layer routing result is not practical, especially for modern complex multi-level VLSI designs. Hence, this work proposes another layer assignment method. This method can provide acceptable layer candidates for an antenna-critical wire and be integrated into the proposed PADJI algorithm for decreasing the number of required vias. On the other hand, this work will insert jumpers to fix

antenna violations when the routing layer is too congested to be assigned. The proposed algorithm which integrates PADJI and layer assignment technique is presented in Fig. 12 and called as PADJILA (discharge-path-based antenna detection and fixing with jumper insertion and layer assignment). For a given X-clock tree with a set of n-sinks, $S = \{s_1, s_2, \ldots, s_n\}$, the layer configuration, and the upper bound of antenna effect, L_{max}, the objective of the proposed PADJILA algorithm is not only to fix antenna violations in the X-clock tree but also to minimize the number of inserted jumpers and required vias.

```
Algorithm: PADJILA /*Discharge-path-based antenna effect detection and fixing with jumper insertion and
                     layer assignment */
Input: A given X-clock tree with a set of sinks, S, layer configuration, and the upper bound of antenna effect, L_max
Output: An antenna violation free zero skew X-clock tree with minimum number of inserted jumpers and required vias
1  for each s_i in S
2  { if( e(s_i).layer = the L_top of layer configuration )
3    { s_i is antenna violation free; }
4    else
5    { DP(s_i) = DischargePath(s_i);  /* Obtain a set of discharge paths of s_i, DP(s_i). */
6      for each dp_i in DP(s_i)
7      { e_total(s_i) = e_total(s_i) + dp_i – Duplicate(∀e_i∈dp_i, e_total(s_i));
         /* Obtain a set of segments those accumulate charges and may damage s_i. */
8        if( |e_total(s_i)| ≥ L_max )
9        { InsertJumper(s_i);  /* Insert a jumper to fix antenna violation occurred on s_i. */
10         via_original = #via(s_i, B(s_i));  /* Store #vias before launching LayerAssignment(s_i). */
11         Layer_candidates(s_i) = LayerAssignment(s_i);  /* Obtain the layer candidates of e(s_i).layer. */
12         via_modified = #via(s_i, B(s_i));
           /* Count #vias when assign Layer_candidates(s_i) to e(s_i).layer without the inserted jumper.*/
13         if( via_modified < via_original and no ping-pong effect )
14         { Remove the inserted jumper and let e(s_i).layer be one of Layer_candidates(s_i) to achieve less #via(s_i, B(s_i)); }
15         break;
16       }
17     }
18   }
19   WireSizing(s_i);  /* Adjust wire width to maintain zero skew. */
20 }
```

Fig. 12. The proposed PADJILA algorithm.

The overview of the proposed PADJILA algorithm is given as follows. To simplify the presentation, this work just discusses how to fix the antenna-critical sink with jumper insertion and layer assignment method (lines 8-16 of PADJILA algorithm). As aforementioned, if the accumulated length of sink s_i, $|e_{total}(s_i)|$, is longer than or equal to L_{max}, s_i is antenna-critical and should be fixed by inserting a jumper. For a pair of sinks s_i and its brother $B(s_i)$, $via_{original}$ includes not only the node vias required by s_i, $B(s_i)$, and $F(s_i)$ but also the edge vias inserted on the wires connecting s_i and $B(s_i)$. In addition, the extra vias required by the inserted jumpers are also included in $via_{original}$. Then, the procedure *LayerAssignment*(s_i) obtains the acceptable layer candidates of $e(s_i)$.layer and gives them to $Layer_{candidates}(s_i)$. Next, $via_{modified}$ acts as $via_{original}$ to count the total required vias, where $e(s_i)$.layer is one of the $Layer_{candidates}(s_i)$. Meanwhile, this work assumes the jumper which is inserted on $e(s_i)$ to be removed. If $via_{modified}$ is less than $via_{original}$ and there is no ping-pong effect, the inserted jumper will be exactly removed and $e(s_i)$.layer can also be obtained with less required vias. The ping-pong effect is defined as that when assigning one of layer candidates may cause another antenna violation and require an extra jumper for fixing. This work will provide an example to discuss the effect in this section. After introducing the overview of PADJILA, the proposed *LayerAssignment* procedure is presented in Fig. 13 in detail.

```
Procedure: LayerAssignment(s_i)
Input: A sink s_i
Output: A set of layer candidates of e(s_i).layer
1  if( e(B(s_i)).layer = L_top of layer configuration )
2  {  return e(B(s_i)).layer;  }
3  else
4  {  if( e(B(s_i)).layer > e(F(s_i)).layer )
5     {  return e(B(s_i)).layer to L_top of layer configuration;  }
6     else if( e(B(s_i)).layer = e(F(s_i)).layer )
7     {  return e(B(s_i)).layer+1 to L_top of layer configuration;  }
8     else
9     {  if( F(s_i) locates on e(s_i) )
10       {  return e(F(s_i)).layer to L_top of layer configuration;  }
11       else
12       {  return e(B(s_i)).layer+1 to L_top of layer configuration;  }
13    }
14 }
```

Fig. 13. The proposed *LayerAssignment* procedure.

For a sink s_i which is given to the proposed *LayerAssignment* procedure, the acceptable layer candidates of $e(s_i)$.layer can be easily and immediately obtained. When $e(B(s_i))$.layer equals the L_{top} of layer configuration, the candidate will be $e(B(s_i))$.layer. Otherwise, the procedure *LayerAssignment*(s_i) will compare the layers of $e(B(s_i))$ and $e(F(s_i))$ (lines 3-14).

When $e(B(s_i))$.layer is higher than $e(F(s_i))$.layer, the procedure will return $e(B(s_i))$.layer up to the L_{top} of layer configuration. If $e(B(s_i))$.layer is equal to $e(F(s_i))$.layer, the layer candidates of $e(s_i)$ will be $e(B(s_i))$.layer+1 up to the L_{top} of layer configuration. For the exceptional cases, the procedure will check $F(s_i)$ does locate on $e(s_i)$ or not. If it does, the procedure will return $e(F(s_i))$.layer up to the L_{top} of layer configuration. Otherwise, the layer candidates of $e(s_i)$.layer will be $e(B(s_i))$.layer+1 up to the L_{top} of layer configuration.

In order to verify the proposed PADJILA algorithm and to compare with other previous works, Fig. 14(a) shows an initial 16-sink X-clock tree which is identical to the tree shown in Fig. 10(a). Figure 14(b) shows the antenna violation free X-clock tree fixed by the JILA [20] algorithm. PADJILA integrates layer assignment method into PADJI for decreasing the number of inserted jumpers and total required vias. The routing result of X-clock tree which is fixed by PADJILA is shown in Fig. 14(c).

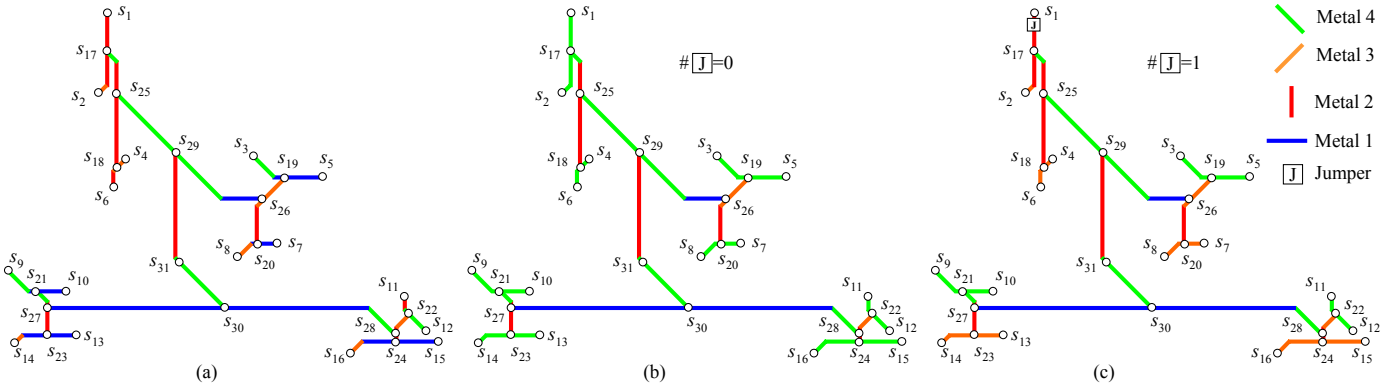

Fig. 14. (a) The initial 16-sink X-clock tree is constructed by PMXF. In order to reduce the number of inserted jumpers and total required vias, the X-clock trees are fixed by (b) JILA [20] and (c) the proposed PADJILA algorithms, respectively.

Table III lists the experimental results on the 16-sink X-clock trees those are respectively constructed and fixed by PMXF, JILA, and PADJILA algorithms. This work first compares the routing results of JI and JILA respectively shown in Fig. 10(b) and Fig. 14(b). The later algorithm assigns all the 13 wires those are inserted jumpers by the former one with the L_{top} of layer configuration (M4). As reported in Table III, JILA completes an antenna violation free X-clock tree with less vias and no inserted jumper. It is

not surprising that JILA can improve the total vias, delay, and power consumption compared with JI. However, assigning all the antenna-critical wires to the L_{top} of layer configuration may result in layer congestion, especially for modern high-density multi-level VLSI routings. Consequently, the proposed PADJILA algorithm provides several layer candidates to antenna-critical wires for increasing layer flexibility. As reported in Table III, PADJILA inserts one jumper and gets improvements in delay, powers, and total vias compared with JILA. Furthermore, this work confirms that layer assignment method can decrease vias. That is the reason PADJILA requires less total vias than that of the initial X-clock tree constructed by PMXF.

TABLE III
COMPARISON OF PMXF [21], JI LA[20], AND THE PROPOSED PADJILA ALGORITHMS ON THE 16-SINK X-CLOCK TREES BASED ON 130NM FED MODEL AND L_{MAX}=200μM

#Sinks	#Inserted Jumpers		Delay (μs)			Skew (μs)			Power (W)			Total vias		
	JILA	PADJILA	PMXF	JILA	PADJILA	PMXF	JILA	PADJILA	PMXF	JILA	PADJILA	PMXF	JILA	PADJILA
16	0	1	0.021500	0.021503	0.021499	0.000009	0.000010	0	0.002083	0.002083	0.002082	92	100	90
Ratio	-	-	-	1.000139	0.999968	-	-	-	-	1.000000	0.999519	-	1.086956	0.978260

This work takes two sinks, s_4 and s_6, as an example from the X-clock trees those are respectively fixed by PADJI and PADJILA. For PADJI algorithm, Figs. 15(a) and 15(b) present the local view and cross-section view of the pair of sinks (s_4, s_6). From the viewpoint of cross-section shown in Fig. 15(b), a jumper is inserted on $e(s_6)$ and needs two extra vias. Sink s_4 needs three NVs to connect poly layer up to M3. Both s_6 and s_{18} need two and one via. In addition, one EV is required by the bending point $P_B(s_4, s_6)$ for connecting M2 and M3. Therefore, (s_4, s_6) totally needs nine vias when applying PADJI to fix antenna violations. Before using the procedure *LayerAssignment*(s_6) to obtain the layer candidates of $e(s_6)$.layer, $via_{original}$ is set as nine.

In *LayerAssignment*(s_6), s_4 and s_{18} are denoted as $B(s_6)$ and $F(s_6)$, respectively. As shown in Fig. 15(a), $e(B(s_6))$.layer is M3 and higher than $e(F(s_6))$.layer, M2. Therefore, the procedure obtains that the layer candidates of $e(s_6)$.layer are M3 and M4. When assigning $e(s_6)$.layer with M3, the jumper inserted on $e(s_6)$ is assumed to be removed. Then, $via_{modified}$ is obtained as seven due to s_4, s_6, and s_{18} need three, three, and one NVs, respectively. Besides, the number of EVs is zero. When assigning $e(s_6)$.layer with M4, the other $via_{modified}$ is obtained as same as $via_{original}$. Because setting the layer candidate of $e(s_6)$.layer be M3 can save two vias compared with $via_{original}$, the jumper inserted on $e(s_6)$ is exactly removed and $e(s_6)$.layer is assigned with M3. Figures 15(c) and (d) are the local view and cross-section view of (s_4, s_6) fixed by PADJILA algorithm.

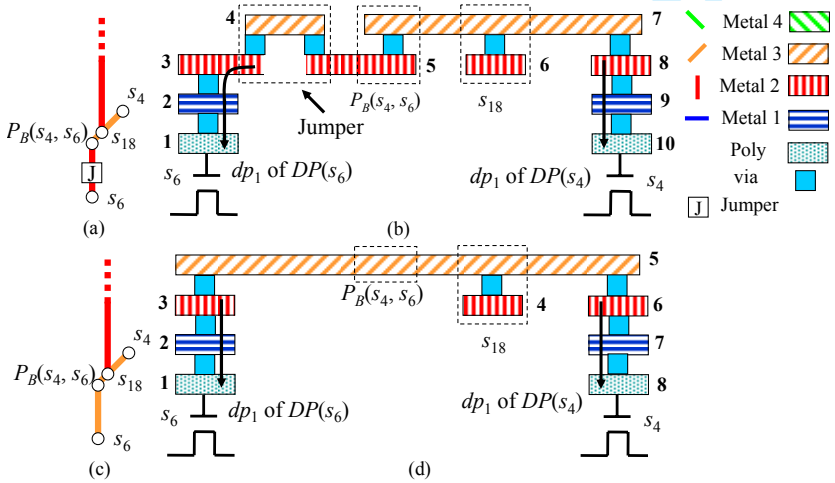

Fig. 15. Take s_4 and s_6 as an example to prove the effectiveness of the proposed *LayerAssignment* procedure. (a) The local view of (s_4, s_6) which is fixed by PADJI, and a jumper is inserted on $e(s_6)$. (b) The cross-section view of (a), (c) the local view (s_4, s_6) which is fixed by PADJILA, where $e(s_6)$.layer is assigned with M3 for decreasing the number of required vias. (d) The cross-section view of (c).

When the number of inserted jumpers and required vias are not able be decreased by layer assignment method, jumper insertion method will be applied. This work takes two sinks, s_1 and s_2, as an example from the X-clock trees those are respectively fixed by PADJI and PADJILA algorithms. The example is used to present the observation of ping-pong effect. For PADJI algorithm, Figs. 16(a) and 16(b) present the local view and cross-section view of the pair of sinks (s_1, s_2). From the viewpoint of cross-section shown in Fig. 15(b), a jumper is inserted on $e(s_1)$ and needs two extra vias. Sink s_1 requires two NVs to connect poly layer up to M2, as well as, s_2 and s_{17} also need three and two vias, respectively. In addition, one EV is required by the bending point $P_B(s_1, s_2)$ for connecting M2 and M3. Therefore, (s_1, s_2) totally needs ten vias when applying PADJI to fix antenna violations. Before using the procedure *LayerAssignment*(s_1) to obtain the layer candidates of $e(s_1)$.layer, $via_{original}$ is set as ten.

In *LayerAssignment*(s_1), s_2 and s_{17} are denoted as $B(s_1)$ and $F(s_1)$, respectively. As shown in Fig. 16(a), $e(B(s_1))$.layer is M3 and lower than $e(F(s_1))$.layer, M4, as well as, $F(s_1)$ locates on $e(s_1)$. Therefore, the procedure obtains that the layer candidates of $e(s_1)$.layer is M4. When assigning $e(s_1)$.layer with M4, the jumper inserted on $e(s_1)$ is assumed to be removed. Then, $via_{modified}$ is obtained as eight due to s_1 and s_2 need four and three NVs, respectively. In addition, one EV is required by $P_B(s_1, s_2)$. However, this process results that s_2 becomes antenna-critical and another jumper with two vias is required for fixing. Moreover, $via_{modified}$ is increased from eight to ten, and the number of inserter jumpers cannot be decreased. The phenomenon is called ping-pong effect. The effect is defined as that when assigning one of layer candidates may cause another antenna violation and require an extra jumper for fixing. When the effect is occurred, this work will skip layer assignment method. Finally, this work keeps the jumper which is inserted on $e(s_1)$ for fixing. Figures 16(c) and 16(d) are the local view and cross-section view of (s_1, s_2), respectively, when applying layer assignment method to result in ping-pong effect.

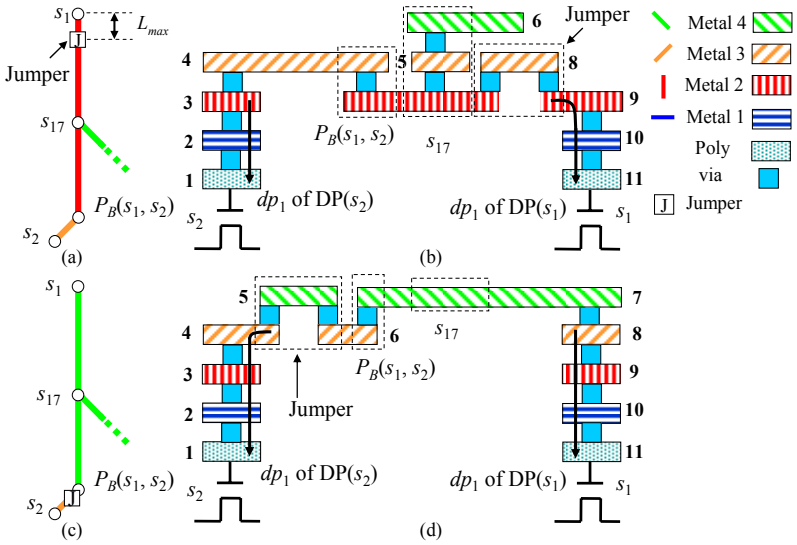

Fig. 16. Take s_1 and s_2 as an example to explain the ping-pong effect. (a) The local view of X-clock tree fixed by PADJI where a jumper is inserted on $e(s_1)$, (b) the cross-section view of (a), (c) the local view of X-clock tree when assigning $e(s_1)$ with M4, and (d) the cross-section view of (c).

After discussing the proposed PADJILA algorithm in detail, the time complexity analysis is given as follows. As aforementioned, every sink in the given X-clock tree will be visited to determine whether it is antenna critical or not. Hence, the *for* loop shown in Fig. 12 runs in $O(n)$. In the *for* loop, the *DischargePath* procedure obtains a set of discharge paths of each sink. The worst case is that each sink contains k paths. Therefore, the procedure runs in $O(k)$. The *LayerAssignment* procedure shown in Fig. 13 can obtain the acceptable layer candidates in constant time. Furthermore, both the *InsertJumper* and *WireSizing* procedures also

run in constant time. Finally, the time complexity of the proposed PADJILA algorithm is $O(n^2)$.

VI. Experimental Results

The proposed PADJI and PADJILA algorithms were implemented with C++. This work performed the two algorithms on a Windows machine with 1.7GHz Pentium-4 processor and 512MB memory. The fabrication parameters of 70nm and 130nm processes under ED and FED delay models are respectively listed in Table IV. In the table, r, c_a, and c_f are the sheet resistance, unit area capacitance, and fringing capacitance, respectively. The three coefficients, D, E, and F are the biases of FED delay model. In power estimation, F_{clk} and V_{dd} are the clock frequency and supplying voltage. Finally, L_{max} is the upper bound of antenna effect.

For evaluating the proposed algorithms and comparing with other works, three sets of benchmarks were downloaded from the GSRC bookshelf [28]: (i) Primary1 and Primary2 (MCNC benchmarks [29]), (ii) *r*1-*r*5 (IBM benchmarks [30]), and (iii) *s*1423, *s*5378, and *s*15850 (ISCAS89 benchmarks [31]). Table V lists the number of sinks, chip size, and the range of load capacitances of each benchmark.

Table IV
Technology Parameters for the Proposed PADJI/PADJILA Algorithms and Other Prior Works

Algorithms	Tech. (nm)	#Layers	Delay Model	r (Ω/μm)	c_a (fF/μm)	c_f (fF/μm)	D	E	F	F_{clk} (Hz)	V_{dd} (v)	L_{max}(μm)
PADJI/PADJILA	70	4	ED	1.357	0.00392	-	-	-	-	2.5G	0.75	100
PMXF/JI/JILA	130	4	FED	0.623	0.00598	0.043	1.12673ln2	1.10463ln2	1.04836ln2	100M	1.2	200

Table V
Related Information of Benchmarks

Benchmarks		#Sinks	Chip size (μm x μm)		Load capacitance (fF)
			Height (μm)	Width (μm)	
MCNC	Primary1	269	6000	6000	500
	Primary2	603	10500	10500	500
IBM	*r*1	267	70000	69984	30 – 80
	*r*2	598	93134	94016	30 – 80
	*r*3	862	98500	97000	30 – 80
	*r*4	1903	126988	126970	30 – 80
	*r*5	3101	145224	142920	30 – 80
ISCAS89	*s*1423	74	11000	14000	50
	*s*5378	179	13000	13000	50
	*s*15850	597	15000	16000	50

In order to make fair comparison, both JI [20] and the proposed PADJI algorithms apply PMXF [21] algorithm to construct the initial X-clock trees of all the benchmarks without antenna effect consideration. Furthermore, FED model with 130nm fabrication parameters is used to calculate delay, and the upper bound of antenna effect L_{max} is set as 200μm. Next, JI and PADJI algorithms respectively load these X-clock trees to detect antenna effects, and then, insert jumpers to fix antenna violations. Table VI lists the experimental results of all the benchmarks. The ratio is defined as JI/PMXF or PADJI/PMXF. From the table, PADJI inserted 46.12% less jumpers than JI on average. Consequently, PADJI had better performance of 0.0932%, 0.0913%, and 46.18% on average in terms of delay, power, and total vias, respectively, compared with JI. In addition, the proposed wire sizing technique can maintain the zero skew property of given X-clock trees. Comparing with PMXF which ignored antenna effect, PADJI achieved antenna violation free and slightly increased 0.0672% in delay, 0.0749% in power, and 17.21% in total vias. The reason is that fixing antenna violations with jumper insertion needs extra vias.

TABLE VI

COMPARISON OF PMXF [21], JI [20], AND THE PROPOSED PADJI ALGORITHMS BASED ON 130NM FED MODEL AND L_{MAX}=200μM

Bench marks	#Sinks	#Inserted Jumpers		Delay (μs)			Skew (μs)			Power (W)			Total vias		
		JI	PADJI	PMXF	JI	PADJI	PMXF	JI	PADJI	PMXF	JI	PADJI	PMXF	JI	PADJI
Primary1	269	211	79	0.059456	0.059495	0.059483	0.000806	0.000816	0	0.174272	0.174341	0.174322	1194	2014	1352
Primary2	603	353	305	0.254020	0.254135	0.254113	0.000688	0.000694	0	0.412771	0.412895	0.412874	2350	3720	2960
*r*1	267	207	115	0.310696	0.311018	0.310928	0.000360	0.000389	0	0.063340	0.063405	0.063359	1220	2022	1450
*r*2	598	439	256	1.125454	1.126724	1.125910	0.000859	0.000975	0	0.156019	0.156188	0.156067	2846	4646	3358
*r*3	862	642	378	1.803820	1.805865	1.804441	0.001217	0.001301	0	0.210124	0.210368	0.210197	4020	6562	4776
*r*4	1903	1421	815	4.804431	4.810863	4.806244	0.002762	0.003736	0	0.487032	0.487646	0.487208	9158	14866	10788
*r*5	3101	2283	1292	8.585707	8.597276	8.588974	0.003049	0.004162	0	0.809903	0.810941	0.810201	14585	23555	17169
*s*1423	74	54	20	0.007489	0.007509	0.007492	0.000038	0.000041	0	0.006009	0.006039	0.006028	351	559	391
*s*5378	179	137	70	0.017915	0.017965	0.017930	0.000033	0.000027	0	0.015470	0.015508	0.015490	772	1308	912
*s*15850	597	414	166	0.049893	0.050067	0.050013	0.000124	0.000159	0	0.058674	0.058830	0.058722	2711	4399	3043
Average Ratio		-	0.5388		1.001604	1.000672	-	-	-	-	1.001662	1.000749	-	1.6339	1.1721

The proposed PADJILA algorithm integrated PADJI and layer assignment method to reduce the number of inserted jumpers and required vias in advance. Table VII lists the experimental results of all the benchmarks constructed by PMXF, as well as, fixed by JILA [20] and PADJILA algorithms. The ratio is defined as JILA/PMXF or PADJILA/PMXF. As reported in Tables VI and VII, the number of inserted jumpers required by JILA and PADJILA can be effectively reduced by layer assignment method compared with JI and PADJI. Moreover, PADJILA inserted 48.21% less jumpers than JILA on average and respectively achieved reductions of 0.0242%, 0.0186%, and 20.36%, in delay, power, and total vias, compared with JILA. Comparing with PMXF, PADJILA slightly increased 0.0297% in delay, 0.0536% in power, and 0.87% in total vias. It is obvious that the proposed PADJI and PADJILA algorithms outperformed other previous works significantly.

TABLE VII

COMPARISON OF PMXF [21], JILA [20], AND THE PROPOSED PADJILA ALGORITHMS BASED ON130NM FED MODEL AND L_{MAX}=200μM

Bench marks	#Sinks	#Inserted Jumpers		Delay (μs)			Skew (μs)			Power (W)			Total vias		
		JILA	PADJILA	PMXF	JILA	PADJILA	PMXF	JILA	PADJILA	PMXF	JILA	PADJILA	PMXF	JILA	PADJILA
Primary1	269	12	3	0.059456	0.059472	0.059461	0.000806	0.000813	0	0.174272	0.174299	0.174281	1194	1520	1200
Primary2	603	8	7	0.254020	0.254089	0.254059	0.000688	0.000700	0	0.412771	0.412839	0.412813	2350	3107	2364
*r*1	267	13	7	0.310696	0.310805	0.310784	0.000360	0.000399	0	0.063340	0.06336	0.063355	1220	1470	1234
*r*2	598	22	12	1.125454	1.125812	1.125730	0.000859	0.000909	0	0.156019	0.156061	0.156049	2846	3299	2870
*r*3	862	37	20	1.803820	1.804302	1.804236	0.001217	0.001165	0	0.210124	0.210189	0.210175	4020	4702	4060
*r*4	1903	80	32	4.804431	4.806045	4.805609	0.002762	0.002551	0	0.487032	0.487185	0.487146	9158	10597	9222
*r*5	3101	145	71	8.585707	8.588937	8.588136	0.003049	0.003488	0	0.809903	0.810178	0.810109	14585	17000	14727
*s*1423	74	0	0	0.007489	0.007494	0.007491	0.000038	0.000041	0	0.006009	0.006029	0.006027	351	399	351
*s*5378	179	13	8	0.017915	0.017937	0.017926	0.000033	0.000030	0	0.015470	0.015486	0.015480	772	1003	788
*s*15850	597	32	13	0.049893	0.049958	0.049921	0.000124	0.000143	0	0.058674	0.058732	0.058698	2711	3350	2737
Average Ratio		-	0.5179	-	1.000539	1.000297	-	-	-	-	1.000722	1.000536	-	1.2123	1.0087

In order to present the effectiveness of the proposed algorithms under the advanced manufacturing process, Table VIII respectively lists the experimental result of all the benchmarks fixed by PADJI and PADJILA with 70nm technology. The ratio is defined as PADJILA/PADJI. From the table, PADJILA made improvements of 95.4%, 0.0043%, 0.0019%, and 6.27% on average in the number of inserted jumpers, delay, power, and total vias, respectively. Comparing the experimental results based on 130nm technology, more jumpers were required to fix antenna violations. Figure 17(a) shows the full chip and local views of benchmark *r*4 fixed by the proposed PADJI algorithm. Figure 17(b) presents the other routing result and views of *r*4 fixed by the proposed PADJILA algorithm. In the local views, the jumpers those were inserted to fix antenna violations were represented with the squares labeled with J. Comparing the three local views respectively shown in the right sides of Figs. 17(a) and 17(b), layer assignment method can effectively reduce the number of inserted jumpers to improve delay and power consumption.

Figure 14 shows the routing result of a small scale X-clock tree, and the congestions of high metal layers (M3 and M4) are lower than those of benchmarks. Therefore, the decreasing degree of total vias listed in Table III is higher than that listed in Table VIII.

TABLE VIII

COMPARISON OF THE PROPOSED PADJI AND PADJILA ALGORITHMS BASED ON 70NM ED MODEL AND L_{MAX}=100μM

Bench marks	#Sinks	#Inserted Jumpers		Delay (μs)			Skew (μs)			Power (W)			Total vias		
		PADJI	PADJILA	PMXF	PADJI	PADJILA	PMXF	PADJI	PADJILA	PMXF	PADJI	PADJILA	PMXF	PADJI	PADJILA
Primary1	269	103	6	0.182017	0.182036	0.182026	0.002284	0	0	1.660459	1.660469	1.660469	1081	1287	1193
Primary2	603	332	8	0.598993	0.599016	0.599001	0.004528	0	0	4.028535	4.028721	4.028604	1803	2467	2241
*r*1	267	120	3	0.298881	0.298922	0.298903	0.000297	0	0	0.230762	0.230781	0.230771	1214	1454	1389
*r*2	598	278	14	0.646688	0.646757	0.646733	0.000438	0	0	0.552051	0.552109	0.552090	2708	3264	3115
*r*3	862	405	26	1.210099	1.210222	1.210183	0.000208	0	0	0.808037	0.808115	0.808096	4134	4944	4730
*r*4	1903	864	48	3.409288	3.409644	3.409538	0.000969	0	0	1.961904	1.962100	1.962041	8986	10714	10205
*r*5	3101	1417	78	6.757963	6.758670	6.758445	0.001092	0	0	3.370986	3.371318	3.371221	14608	17442	16553
*s*1423	74	30	1	0.013947	0.013949	0.013948	0.000029	0	0	0.035684	0.035684	0.035684	359	419	382
*s*5378	179	73	3	0.029410	0.029414	0.029412	0.000080	0	0	0.102754	0.102764	0.102764	782	928	910
*s*15850	597	270	15	0.122842	0.122857	0.122856	0.000113	0	0	0.413496	0.413535	0.413535	2600	3140	3112
Average Ratio		-	0.046	-	1.000110	1.000067	-	-	-	-	1.000073	1.000054	-	1.2106	1.1479

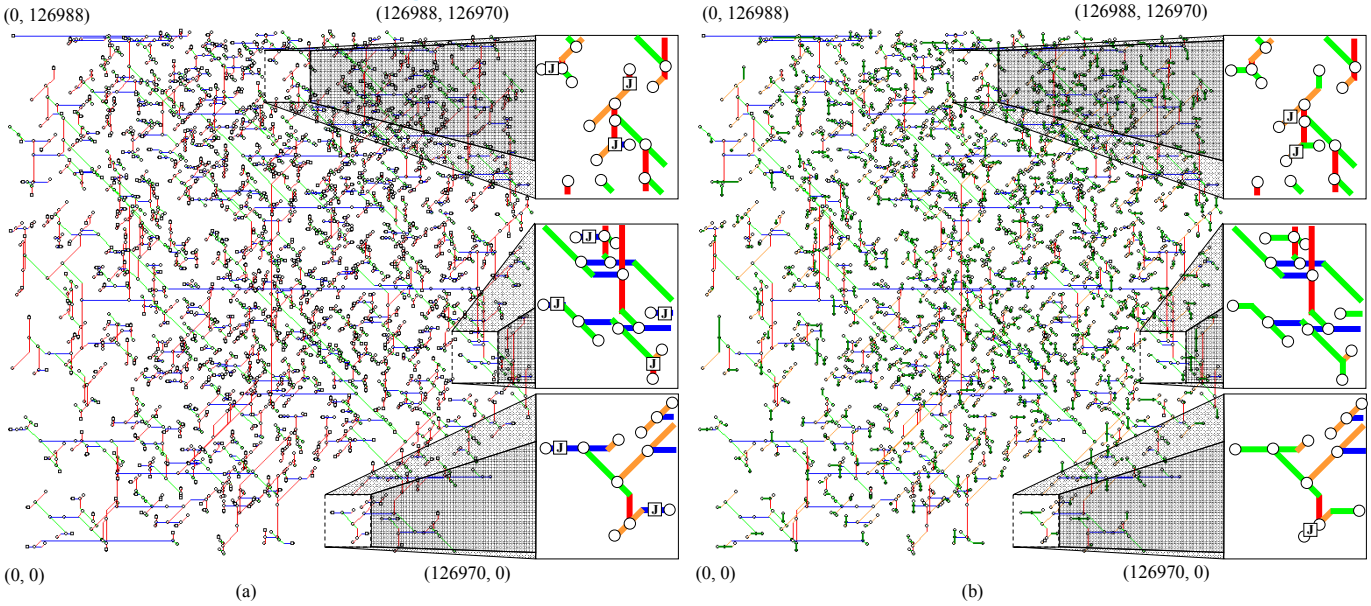

Fig. 17. Full chip views and local views of benchmark *r*4 fixed by (a) PADJI and (b) PADJILA algorithms.

VII. CONCLUSION

Antenna effect is one of the design for manufacturing (DFM) problems in nanometer process. This work proposes a discharge-path-based antenna detection algorithm and achieves antenna violations free X-clock trees with jumper insertion. Furthermore, the proposed algorithm integrates layer assignment method to reduce the number of inserted jumpers and required vias. Therefore, the clock delay and power consumption can be improved in advance. For via timing impact, the experimental results indicated that delay and power consumption are slightly proportioned to total vias. However, modern complex multi-level VLSI designs wherein the number of vias is on the order of 50 million and even higher. Hence, not only clock tree construction but also jumper insertion and layer assignment methods for fixing antenna violations should consider via timing impact. For other DFM issues, this work can be extended to combine with optical proximity correction (OPC) and double via insertion into X-clock trees. For the clock tree optimization, inserting buffers on interconnections is a general and efficient method for improving clock delay. On the other hand, the metal wires connecting the input ports of buffers may accumulate sufficient charges to damage buffers. Consequently, the buffered clock trees should require extra jumpers to fix antenna violations compared with bufferless clock trees.

ACKNOWLEDGMENT

REFERENCES

[1] L. Riviere-Cazaux, K. Lucas, and J. Fitch, "Integration of Design For Manufacturability (DFM) Practices In Design Flows," in *Proc. International Symposium Quality Electronic Design*, March 2005, pp. 102-106.

[2] J. T. Kong, "CAD for Nanometer Silicon Design Challenges and Success," *IEEE Trans. on Very Large Scale Integration Systems*, vol. 12, No. 11, Nov. 2004, pp. 1132-1146.

[3] S. Krishnan, S. Rangan, S. Hattangady, G. Xing, K. Brennan, M. Rodder and S. Ashok, "Assessment of charge-induced damage to ultra-thin gate MOSFETs," in *IEDM Technology Dig*est, 1997, pp. 445-448.

[4] Z. Chen and I. Koren, "Layer reassignment for antenna effect minimization in 3-layer channel routing," in *Proc. IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, Nov. 1996, pp. 77-85.

[5] *TSMC 0.18μm Design Manual*, Document number TA-10A5-4001, pp. 67-68.

[6] P. H. Chen, S. Malkani, C. M. Peng, and J. Lin, "Fixing antenna problem by dynamic diode dropping and jumper insertion," in *Proc. International Symposium Quality Electronic Design*, March 2000, pp. 275-282.

[7] H. Shirota, T. Sadakane, M. Terai, and K. Okazaki, "A new router for reducing antenna effect in ASIC design," in *Proc. IEEE Custom Integrated Circuits Conference*, May 1998, pp. 601-604.

[8] T. Y. Ho, Y. W. Chang, and S. J. Chen, "Multilevel routing with antenna avoidance," in *Proc. ACM International Symposium Physical Design*, Apr. 2004, pp. 33-40.

[9] T. Y. Ho, Y. W. Chang, and S. J. Chen, "Multilevel routing with jumper insertion for antenna avoidance," *Integration: VLSI J*ournal, vol. 29, no. 4, pp. 420-432, Jul. 2006.

[10] B. Y. Su and Y. W. Chang, "An exact jumper insertion algorithm for antenna effect avoidance/fixing," in *Proc. ACM/IEEE Design Automation Conference*, Jun. 2005, pp. 597-602.

[11] B. Y. Su, Y. W. Chang, and J. Hu, "An optimal jumper insertion algorithm for antenna avoidance/fixing on general routing trees with obstacles," in *Proc. ACM International Symposium Physical. Des*ign, Apr. 2006, pp. 56-63.

[12] B. Y. Su, Y. W. Chang, and J. Hu, "An exact jumper insertion algorithm for antenna violation avoidance/fixing considering routing obstacles," *IEEE Trans. on Computer-Aided-Design Integrated Circuits and Syst*ems, vol. 26, no. 4, pp. 719-734, Apr. 2007.

[13] B. Y. Su and Y. W. Chang, "An optimal jumper-insertion algorithm for antenna avoidance/fixing," *IEEE Trans. on Computer-Aided Design Integrated Circuits and Syst*ems, vol. 26, no. 10, pp. 1818-1829, Oct. 2007.

[14] D. Wu, J. Hu, and R. Mahapatra, "Coupling aware timing optimization and antenna avoidance in layer assignment," in *Proc. ACM International Symposium Physical Design*, Apr. 2005, pp. 20-27.

[15] D. Wu, J. Hu, and R. Mahapatra, "Antenna avoidance in layer assignment," *IEEE Trans. on Computer.-Aided Design Integrated Circuits and Syst*ems, vol. 25, no. 4, pp. 734-738, Apr. 2006.

[16] L. D. Huang, X. Tang, H. Xiang, D. F. Wong, and I. M. Liu, "A polynomial time optimal diode insertion/routing algorithm for fixing antenna problem," in *Proc. ACM/IEEE Design Automation Test in Europe*., Mar 2002, pp. 470-475.

[17] L. D. Huang, X. Tang, H. Xiang, D. F. Wong, and I. M. Liu, "A polynomial time optimal diode insertion/routing algorithm for fixing antenna problem," *IEEE Trans. on Computer-Aided Design Integrated Circuits and Syst*ems, vol. 23, no. 1, pp. 141-147, Jan. 2004.

[18] Z. W. Jiang and Y. W. Chang, "An optimal simultaneous diode/jumper insertion algorithm for antenna fixing," in *Proc. IEEE/ACM International Conference Computer-Aided Des*ign, Nov. 2006, pp. 669-674.

[19] Z. W. Jiang and Y. W. Chang, "An optimal network-flow-based simultaneous diode and jumper insertion algorithm for antenna fixing," *IEEE Trans. on Computer-Aided Design Integrated Circuits and Systems*, vol. 27, no. 6, pp. 1055-1065, June 2008.

[20] C. C. Tsai, F. T. Hsu, C. C. Kuo, J. O. Wu, and T. Y. Lee, "Antenna effect consideration for X-clock routing," in *Proc. IEEE International Conf. on Solid-State and Integrated-Circuit Technology*, Oct. 2008, pp. 2248-2251.

[21] C. C. Tsai, C. C. Kuo, J. O. Wu, T. Y. Lee, and R. S. Hsiao, "X-clock routing based on pattern matching," in *Proc. IEEE International SOC Conf.*, Sept. 2008, pp. 357-360.

[22] S. L. Teig, "The X-architecture: not your father's diagonal wiring," in *Proc. Int. Workshop on System-Level Interconnect Prediction*, Apr. 2002, pp. 33-37.

[23] R. S. Tsay, "Exact zero skew," in *Proc. International Conf. Computer-Aided Design*, Nov. 1991, pp. 336-339.

[24] W. C. Elmore, "The transient response of damped linear network with particular regard to wideband amplifiers," *Journal Applied Physics*, vol. 19, pp. 55-63, 1948.

[25] Arif Ishaq Abou-Seido, Brian Nowak, and Chris Chu, "Fitted Elmore Delay: a simple and accurate interconnect delay model," *IEEE Trans. on VLSI*, vol. 12, no. 7, pp. 691-696, 2004.

[26] T. Y. Ho, Y. W. Chang, S. J. Chen, and D. T. Lee, "Crosstalk- and performance-driven multilevel full-chip routing," *IEEE Trans. on Computer-Aided Design Integrated Circuits and Syst*ems, vol. 24, no. 6, pp. 869-878, June 2005.

[27] C. H. Wang and W. K. Mak, "λ-Geometry clock tree construction with wire length and via minimization," in *Proc. IEEE International Symposium VLSI-DAT*, Apr. 2007, pp. 124-127.

[28] VLSI CAD Laboratory, University of California, San Diego, "Bounded-skew clock tree routing version 1.0," May 2000, http://vlsicad.ucsd.edu/GSRC/bookshelf/Slots/BST/.

[29] M. A. B. Jackson, A. Srinivasan and E. S. Kuh, "Clock routing for high-performance ICs," in *Proc. ACM/IEEE Design Automation Conf.*, June 1990, pp. 573-579.

[30] R. S. Tsay, "An exact zero-skew clock routing algorithm," *IEEE Trans. on Computer-Aided Design Integrated Circuits and Syst*ems, vol. 12, no. 2, pp. 242-249, Feb. 1993.

[31] J. G. Xi, and Wayne W. M. Dai, "Useful-skew clock routing with gate sizing for low power design," in *Proc. ACM/IEEE Design Automation Conf.*, June 1996, pp. 383-388.