Clock Tree Synthesis Based on RC Delay Balancing

Fumihiro Minami and Midori Takano

ULSI Research Center, TOSHIBA Corporation

Abstract

This paper presents a novel clock tree synthesis method based on top down binary tree construction, optimal buffer insertion, and bottom up wiring with precise RC delay balancing. The proposed method has achieved near-zero clock skews with minimal clock delays while achieving similar total wire lengths in comparison with previous heuristics.

1 Introduction

The circuit speed in a synchronous VLSI design is limited by the clock skew which is defined as the maximum difference in the delays from the clock source to the synchronous components. The system speed is also limited by the inter-chip clock skew which is the clock delay difference among chips. In general, most systems require a clock skew of less than 10% of the system clock period. Thus, clock skew minimization and clock delay reduction are needed for high performance chips.

Several approaches for clock routing using a tree structure have been reported as follows: the H-tree method [1], the Method of Means and Medians (MMM) [2], and the Path Length Balancing method (PLB) [3]. However, the H-tree structure is not applicable for asymmetric distributions of synchronous components, and MMM and PLB are not aimed at path delay balancing which is the primary objective. Therefore, these techniques are insufficient in minimizing the clock skew.

This paper presents a novel approach to clock routing that minimizes the clock skew by constructing a binary tree with RC delay balancing. The proposed Path Delay Balancing method (PDB) always yields near-zero clock skews. The key point of PDB is that the branching point of a binary tree is so determined as to equilibrate the RC delay for each subtree calculated by Elmore's formula [4]. The proposed method uses a hierarchical tree structure divided by buffer cells to prevent the delay increase caused by the quadratic term of the wire length for a circuit with too many synchronous components. The synchronous components are so clustered in the lowest tree level as to balance the load capacitance within each cluster where the wire resistance is negligible, and thus the clock skew among clusters is minimized. As a result, the clock skew and clock delay are perfectly minimized with the above mentioned techniques.

This paper is organized as follows: in Section 2, the keys to minimize the skew and delay are presented. Section 3 presents the definition of the problem. The algorithm for clock tree synthesis is presented in Section 4, and practical considerations are presented in Section 5. The experimental results are presented in Section 6, and the paper is concluded in Section 7.

2 Skew and Delay Minimization

In this section, a delay calculation model is presented and then the keys to minimize the skew and delay are presented.

2.1 Delay calculation model

For a lumped RC tree network, Elmore's formula [4][6] is commonly used for delay calculation and the authors have also adopted it in this paper. According to this formula, the delay from a root to a leaf in a RC tree is defined as the sum of the products of both the resistances R_i and its downstream capacitances C_i along the path, which is defined as follows:

$$Delay = w \Sigma R_i C_i$$

where w is a constant coefficient.

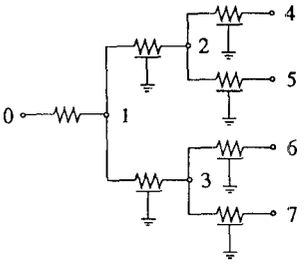


Figure 1: Binary clock tree

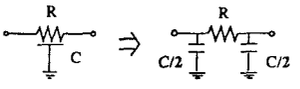


Figure 2: π-model subcircuit

In a binary tree for clock routing, Elmore's delay is calculated in the following bottom up manner.

Let the highest branch node of the tree be labeled *1* and two child nodes of node k be labeled $2k$ and $2k+1$, and the driver output pin be labeled *0*, as shown in Figure 1. Let $C(k)$ be the downstream capacitance from node k and $L(k)$ be the wire length between node k and its parent node. Let r and c be the unit length resistance and capacitance, and let R_{on} and dt be the on-resistance and the intrinsic delay of a driver cell. Then, the delay from a driver input pin to leaf node e is expressed as follows.

$$Delay(e) = dt + \Sigma d(k) \quad (1)$$
$$d(k) = w\, r\, L(k) \cdot (c\, L(k) / 2 + C(k)) \quad \text{if } k > 1$$
$$d(1) = w\, R_{on}\, C(1)$$
$$C(k) = C(2k) + C(2k+1) + c\, (L(2k) + L(2k+1))$$
$$C(e) = \text{input pin capacitance of leaf } e$$

In this expression, each wire segment which is expressed by a distributed RC line is assumed to be approximately equivalent to a π-model subcircuit as shown in Figure 2.

2.2 Skew minimization

The skew between two different paths is equal to the difference of the sum of RC products from the last common branch to each leaf because the RC products along a common path is the same. According to this, the local skew $S(k)$ at each node k can be defined as the maximum delay difference for all paths from node k to its descendant leaves. Let $T_L(k)$ and $T_S(k)$ be the maximum and minimum delays of all paths from node k to its descendant leaves, then $S(k)$ is expressed as follows:

$$\begin{aligned} S(k) &= T_L(k) - T_S(k) \\ &= \max (S(2k),\ S(2k+1), \\ &\qquad T_L(2k) + d(2k) - T_S(2k+1) - d(2k+1), \\ &\qquad T_L(2k+1) + d(2k+1) - T_S(2k) - d(2k)) \\ &= \max (S(2k),\ S(2k+1),\ \mu(k) + |\lambda(k)|) \end{aligned} \quad (2)$$

IEEE 1992 CUSTOM INTEGRATED CIRCUITS CONFERENCE

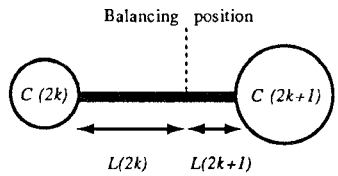


Figure 3: RC delay balancing position

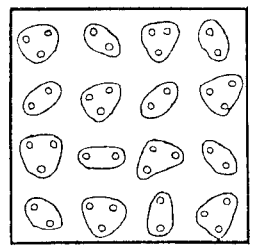


Figure 4: Clustering

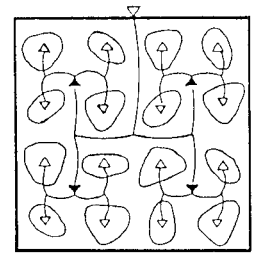


Figure 5: Tree construction

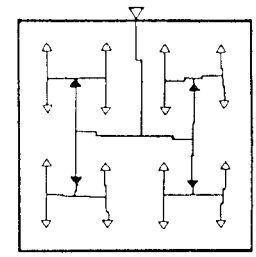


Figure 6: Balanced routing

where

$$\mu(k) = (S(2k) + S(2k+1)) / 2$$
$$\lambda(k) = d(2k) + \tau(2k) - d(2k+1) - \tau(2k+1)$$
$$\tau(j) = (T_L(j) + T_S(j)) / 2 .$$

Then, the clock skew in a tree is defined as $S(1)$.

$\lambda(k)$ means the skew between the delay median of the left side descendants and that of the right side descendants. Because $\mu(k)$ is less than either $S(2k)$ or $S(2k+1)$, if $\lambda(k)$ is forced to be equal to zero, then $S(k)$ is rewritten as follows:

$$S(k) = \max (S(2k), S(2k+1)).$$

This equation suggests that the local clock skews do not exceed the descendant clock skews. Consequently, if a position of each node k is so determined that $\lambda(k)$ is zero in a bottom up manner, the clock skew will be equal to the maximum skew at the leaves and will normally be zero. This is the key to minimize the clock skew in this paper.

The condition that $\lambda(k)$ is zero is as follows:

$$d(2k) + \tau(2k) = d(2k+1) + \tau(2k+1)$$

or

$$r\,L(2k) \cdot (c\,L(2k) / 2 + C(2k)) + \varepsilon(k) = r\,L(2k+1) \cdot (c\,L(2k+1) / 2 + C(2k+1)) \quad (3)$$

where

$$\varepsilon(k) = (\tau(2k) - \tau(2k+1)) / w .$$

As shown in this equation, delay balancing is analogous to moment balancing of a mobile where the capacitance corresponds to the weight and the resistance corresponds to the arm length. Figure 3 shows the image.

2.3 Delay minimization

For a circuit with too many synchronous components, the clock delay increases in proportion to the square of the wire length. In such a case, a hierarchical clock tree structure by multistage buffering is commonly used to reduce the clock delay.

For multistage buffering, the delay is influenced by the positions of the inserted buffer cells. Buffer insertion close to the primary driver cell causes a large delay in the buffer stage, and buffer insertion close to the leaf nodes causes a large delay in the primary driver stage, so there exists an optimal position for buffer cells to be inserted.

In the proposed method, buffer cells are so inserted as to minimize the total clock delay by a greedy method.

In addition, the clock pins in the lowest trees are routed by the Steiner-tree method which minimizes the wire length and thus reduces the delay. This is because the wire resistances in the lowest tree are very small in comparison with the on-resistances of the buffers and RC delay balanced routing is unnecessary.

3 Problem definition

Given a circuit placement, the clock pin positions, the number of buffering stages and the performance data for buffer cell types, the clock layout optimization problem can be defined as follows: construct a hierarchical clock tree by multi-stage buffering which minimizes the clock skew, delay, and wire length subject to the clock pin positions. In this paper, it is assumed that two layers are available for clock layout.

4 Algorithm

The proposed method consists of three steps, which are synchronous component clustering, hierarchical tree topology construction, and RC delay balanced routing. In tree topology construction, the optimal combination of the buffer types and levels to be inserted into the tree is obtained by a greedy method. The main process flow is shown as follows.

```
procedure clock_tree_synthesis
Begin
  B : set of sequentially inserted buffer types and
      their insertion depths in the tree
  for b ∈ B , do
    synchronous component clustering
    hierarchical tree topology construction
      /* RC delay balanced routing */
    for each level of hierarchical tree, do
      branch position determination for each subtree
      buffer positions adjustment for delay equalization
      tree level decrement
    end
    saving the routing result if the delay is the smallest
  end
End
```

4.1 Synchronous component clustering

First, synchronous components are clustered by recursive bi-sectioning until the wire resistance within each cluster becomes negligibly small, as shown in Figure 4. At each division, the dividing line is so determined as to balance the load capacitance of two regions which is defined as the sum of the estimated wire capacitance [5] and the input gate capacitance, or

$$LoadCap = c\,h\,f(fanout) + \Sigma\, C_g \quad (4)$$

where

- c : capacitance per unit wire length
- h : half perimeter of the minimal bounding box
- $f(fanout)$: statistical coefficient function of wire length
- C_g : input pin capacitance for each gate .

After clustering, the clock pins in each cluster are routed by the Steiner-tree method to minimize the wire length.

By this clustering, the inter-cluster skew is minimized because of the uniform load capacitance and negligible wire resistance for each cluster.

4.2 Hierarchical tree topology construction

Secondly, the binary tree topology whose root is a primary clock driver and whose leaves are clusters is constructed by top down bi-sectioning [2], and then buffer cells are inserted to construct the hierarchical tree. In each buffering stage, buffer cells of the same type are topologically inserted at the same level of the binary tree, as shown in Figure 5. The lowest buffer cells are placed at the center of each cluster region, and middle buffer cells are placed at or near the highest branch points of the subtrees, which are determined by the process explained in Section 4.3.

4.3 RC delay balanced routing

Lastly, each subtree of the hierarchical tree is recursively routed in a bottom up manner, as shown in Figure 6. In the subtree which is driven by a middle buffer cell or a primary driver cell, the branch position determination and the buffer position adjustment are so performed as to minimize skew as follows.

1) Branch position determination

In the subtree, the branch position for each tree node is so determined as to minimize its local skew $S(k)$, or to satisfy Equation (3).

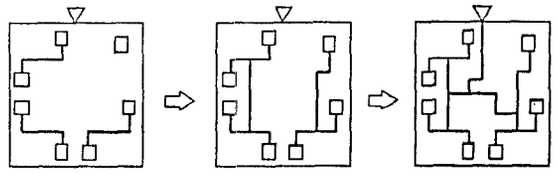
Let l be $L(2k) + L(2k+1)$ and x be the ratio of $L(2k)$ to l, then x is solved as follows:

$$x = \frac{C(2k+1) + c\,l/2 - \varepsilon(k)/r\,l}{C(2k) + C(2k+1) + c\,l} \qquad (5)$$

To minimize the wire length and clock delay, l is assumed to be the Manhattan length between node $2k$ and $2k+1$. If x is not in the range from 0 to 1, x is set at 0 or 1 so that $S(k)$ is minimized.

After each branch position determination, node k is connected to each child node with a minimum length, and $C(k)$, $T_L(k)$, and $T_S(k)$ are calculated. These are recursively processed by the following depth first procedure, as shown in Figure 7.

```
procedure delay_balance( P )
Begin
  if P is a leaf then
    F.pos = input pin position of leaf
    F.c   = input pin capacitance
    F.TL  = max delay of lower tree
    F.TS  = min delay of lower tree
    return( F )
  else
      PL : left side subtree of current node
      PR : right side subtree of current node
    FL = delay_balance( PL )
    FR = delay_balance( PR )
    decide branch position so as to minimize the λ(k)
    route from the current node to two child nodes
      l : wire length between one child node and another
    F.pos = branch position for current node
    F.c   = FL.c + FR.c + c l
    F.TL  = max ( FL.TL + d(2k), FR.TL + d(2k+1) )
    F.TS  = min ( FL.TS + d(2k), FR.TS + d(2k+1) )
    return( F )
  endif
End
```



(a) Routing image

(b) Equivalent mobile

Figure 7: Bottom up routing in a single tree

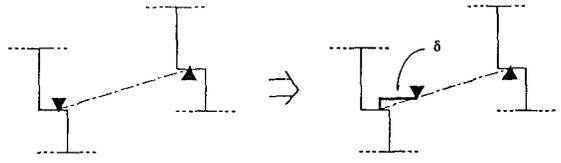


Figure 8: Buffer position adjustment

2) Buffer positions adjustment

After routing at each tree level, the buffer cell positions are so adjusted as to balance the downstream delays by the following technique.

Initially, each buffer cell is placed at the highest branch point of its subtree. Next, the buffer cell is moved to the point with a Manhattan distance δ from the initial position, where the downstream delay is equal to the largest one at the same tree level. A new position is searched along the line which connects the initial position with its brother buffer position of the upper subtree, as shown in Figure 8.

Let Δ be the difference between $\tau(l)$ of the current subtree and the largest one of the subtree at the same level, then δ is required to satisfy the following equation.

$$\Delta = w\,R_{on}\,c\,\delta + w\,r\,\delta\,(c\,\delta/2 + C(l)) \qquad (6)$$

Such δ is solved as follows:

$$\delta = \frac{\sqrt{b^2 + 2\,rc\Delta/w} - b}{rc} \qquad (7)$$

where

$$b = R_{on}\,c + r\,C(l)\,.$$

In most cases, small δ is sufficient for equalizing the delays at the same level because $C(l)$ is relatively large. And such selection of the moving direction leads to a wire length reduction at an upper level which cancels out the wire length increase at the current level. Consequently, the adjustment for buffer cell positions is so effective that subtree delays of the same level can be balanced without elongating the total wire length.

As mentioned above, RC delay balanced routing for a hierarchical tree results in a near-zero clock skew and minimal clock delay.

5 Practical considerations

In practice, one would like to minimize the skew exactly. In this case, we can modify the RC delay balanced routing method by adding a process for the elongating the wire length in order to balance the delay exactly at each branch and at each cluster. Branch position determination just after routing between two child nodes is also effective for exact delay balancing.

Another practical concern is the many possible combinations for buffer insertion to be checked. To reduce the combinations and the computational time, the authors propose to use the upper driving limit of the tree depth for each driver type.

6 Experimental results

The proposed PDB method was implemented in C on a SUN4 system, and was tested on single and hierarchical tree examples. Table 1 shows the statistics for the test examples and the coefficient w was set at 0.7 for delay calculation.

In the single tree case, the routing performance for PDB was compared with that for MMM, and PLB for three industrial data and the MCNC benchmark data *Primary2*, under the condition of the same topology. Table 2 shows the computational results of the clock skew, delay, and wire length. The results showed that PDB yields a zero clock skew with a slight small delay and almost the same wire length.

In the hierarchical tree case, PDB was tested on two examples with one and two stage buffering. Table 3 shows the excellent results which had an extremely small delay compared with the single tree and a sufficiently small skew. The skew was mostly caused by the inter-cluster skew, or the maximum capacitance difference between the clusters which was always less than 10% of the average capacitance.

Figure 9 and 10 show the global and detail routing results of *Data3* by the PDB method.

7 Conclusion

This paper has presented a novel clock tree synthesis method based on top down binary tree construction, optimal buffer insertion, and bottom up wiring with precise RC delay balancing. The proposed method has achieved near-zero clock skews with minimal clock delays for a wide range of clock pin counts and distributions on industrial chips.

Future work will address wire width optimization and give consideration to internal path delay in a large macro with its blockage area during the construction of the clock tree.

References

[1] H. B. Bakoglu, J. T. Walker and J. D. Meindl, "A symmetric clock-distribution tree and optimized high-speed interconnections for reduced clock skew in ULSI and WSI circuits", *Proc. IEEE Int. Conference on Computer Design*, pp. 118-122, 1986.

[2] M. A. B. Jackson, A. Srinivasan, and E. S. Kuh, "Clock Routing for High-Performance ICs", *Proc. 27th Design Automation Conference*, pp. 573-579, 1990.

[3] A. Kahng, J. Cong, and G. Robins, "High-Performance Clock Routing Based on Recursive Geometric Matching", *Proc. 28th Design Automation Conference*, pp. 322-327, 1991.

[4] J. Rubinstein, P. Penfield and M. A. Horowitz, "Signal Delay in RC Tree Networks", *IEEE Trans. on CAD*, CAD-2(3), pp. 202-211, 1983.

[5] S. Boon, S. Butler, R. Byrne, B. Setering, M. Casalanda, and Al Scherf, "High Performance Clock Distribution for CMOS ASICs", *IEEE Custom Integrated Circuits Conference*, pp. 15.4.1-15.4.5, 1989.

[6] H. B. Bakoglu: Circuits, Interconnections, and Packaging for VLSI, Chapter 5, Addison-Wesley Publishing Co., Reading, Mass., 1990.

	d_I (nsec)	R_{on} (psec/LU)	C_g (LU)
Driver	0.79	3.6	
Buffer1	0.70	31.4	1
Buffer2	0.68	15.7	2
Flipflops			1

(a) Performance data for used cells

	r (psec/LU mm)	c (LU/mm)
Horizontal wire	1.22	1.09
Vertical wire	2.14	1.50

(b) Performance data for wires

	Pin counts	Chip width (mm)	Chip height (mm)
Data1	410	6.6	6.6
Primary2	603	12.0	12.0
Data2	1026	14.7	14.7
Data3	1661	14.7	14.7

(c) Test examples

Table 1: Statistics of experiments

		MMM	PLB	PDB
Data1	Skew	0.37	0.33	0.00
	Delay	5.31	5.10	5.07
	Length	178	175	176
Primary2	Skew	0.68	0.39	0.00
	Delay	13.73	13.32	12.95
	Length	432	430	432
Data2	Skew	0.47	0.24	0.00
	Delay	33.33	32.50	32.41
	Length	726	721	723
Data3	Skew	0.71	0.49	0.00
	Delay	48.16	46.96	46.81
	Length	927	918	920

Table 2: Skew (nsec), maximum delay (nsec), and total wire length (mm) comparison in a single tree

		Without buffering	With buffering
Primary2	Skew	0.00	0.07
	Delay	12.95	3.03
	Length	432	232
Data3	Skew	0.00	0.05
	Delay	46.81	3.94
	Length	920	544

Table 3: Skew (nsec), maximum delay (nsec), and total wire length (mm) comparison between without buffering and with buffering

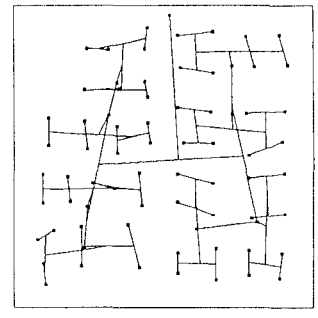


Figure 9: Global routing example

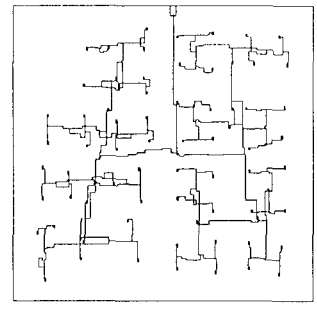


Figure 10: Detail routing example