第 25 卷第 11 期　　半 导 体 学 报　　Vol. 25, No. 11
2004 年 11 月　　CHINESE JOURNAL OF SEMICONDUCTORS　　Nov., 2004

Timing Optimization by Inserting Minimum Buffers with Accurate Delay Models*

Zhang Yiqian, Hong Xianlong, Zhou Qiang and Cai Yici

(*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084, *China*)

Abstract: An algorithm of minimizing the number of buffers under certain delay constraint with accurate delay models is presented. Given a two-terminal net, the algorithm can minimize the total number of buffers inserted to meet the delay constraint. A high order delay model is applied to estimate interconnect delay and a nonlinear delay model based on look-up table is for buffer delay estimation. The experimental results show that the algorithm can efficiently achieve the trade-offs between number of buffers and delay, and avoid needless power and area cost. The running time is satisfactory.

Key words: buffer insertion; interconnect optimization; layout; VLSI
EEACC: 2570　　**CCACC**: 7410D
CLC number: TN47　　**Document code**: A　　**Article ID**: 0253-4177(2004)11-1409-07

1 Introduction

For very deep sub-micron technology, it is well known that interconnect delay has become more dominant than gate delay in determining the overall circuit performance. Thus, interconnect optimization has become a critical step in the high performance VLSI design. Some interconnect optimization techniques, such as topology optimization, device sizing, buffer insertion, and wire sizing, have gained widespread acceptance in deep submicron design. In particular, buffer insertion techniques have been successful in reducing interconnect delay.

Buffer insertion has been an active area of study in recent years. Closed formed solutions have been proposed in Refs. [1~3] for inserting buffers on a 2-pin net. The authors of Ref. [4] insert buffers on a tree by iteratively finding the best buffer location. Van Ginneken[5] proposed a dynamic programming algorithm which finds the optimal solution under the Elmore wire delay model and a linear gate delay model. Several variants of this algorithm have been proposed[1,6,7]. These algorithms above use both simplified gate and wire delay models. However, it is well known that both the Elmore delay model and the linear gate model are not accurate enough. Furthermore, both models are not aware of the input waveform, which is becoming increasingly important in today's very deep sub-micron technology. The optimal solution under these simple models may be not good enough. Thus, the author of Ref. [8] extends Van Ginneken's algo-

* Project supported by National Natural Science Foundation of China (No. 60176016), National High-Tech Research & Development Program of China (No. 2002AAIZ1460)

Zhang Yiqian female, was born in 1977, PhD candidate. Her research interests focus on interconnect planning in VLSI layout design.
Hong Xianlong male, was born in 1940, professor. His research interests focus on VLSI layout algorithms and DA systems.
Zhou Qiang male, was born in 1961, associate professor. His research interests focus on VLSI layout design.

Received 8 January 2004, revised manuscript received 8 June 2004

rithm by using both accurate interconnect and gate delay models, and the signal waveform is considered in buffer insertion. Gao *et al.*[9] presents a graph-based algorithm for optimal buffer insertion to minimize delay time under accurate delay models. The algorithm shows that the buffer insertion problem can be reduced to the shortest path problem, and it is simpler and easier to understand and implement than the algorithm in Ref. [8].

As the intrinsic delay of a buffer becomes smaller and the chip dimension gets larger, there may be more and more buffers inserted for high-performance designs in future technology generations (e.g., close to 800000 for the 70nm technology as estimated in Ref. [10]). The insertion of so many buffers will cause great power and area cost. On the other hand, it may be not necessary to minimize delay time in circuit design, and only to satisfy certain delay constraint which is enough for general VLSI design. In order to avoid needless power and area cost, the number of buffers should be minimized so long as the delay constraint is satisfied.

In this paper, we present a graph based algorithm of minimizing the number of buffers subject to a given delay constraint with accurate delay models. The main contributions of this paper are:

(1) Our algorithm solves the problem by transforming it into a series of the shortest path problems. As we know, the dynamic programming algorithm proposed by Van Ginneken has gained popularity with various buffer insertion problem, but it is based on the hierarchical nature of the Elmore delay model. However, for accurate delay models the RC delay are not additive, this would result in a large number of partial sub-trees needed to be analyzed during the bottom-up dynamic programming algorithm. Furthermore, when a signal waveform is considered, the input transition time is unknown in the bottom-up dynamic programming procedure, we must calculate delay under many possible input transition times for a RC tree, this will deserve huge memory requirements and runtime cost.

(2) Accurate interconnect and buffer delay models are adopted. A high order delay model is applied to calculate the interconnect delay more accurately. Nonlinear delay model based on look-up table is used to calculate the delay of buffer. The look-up tables are all from an industrial circuit library. And signal rise/fall time is also taken into account in the delay calculation for interconnect and buffer.

2 Delay calculation for interconnect and buffer

As shown in Fig. 1, we approximate each signal as a finite ramp which is characterized by two parameters: delay time D and transition time Tr, where D is defined as the time required for the response to reach half its final value, and Tr is taken to be the time required for the response to increase from 10 to 90 percent of its final value.

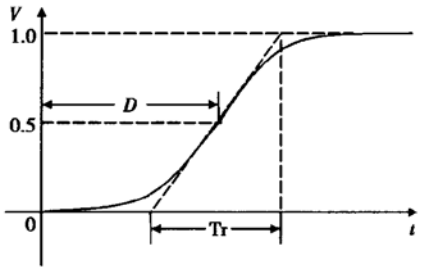

Fig. 1 Curve illustrating the definition of delay time and transition time

A high order delay model is applied to estimate interconnect delay. The built-in delay evaluation engine uses congruence transformation technique[11] to reduce the order of a large RC net-list while stability and passivity can be guaranteed. To be an efficient timing analysis engine with good trade-offs between accuracy and speed, an adaptive method is used to control the order of net-list reduction. In benchmark testing, the accuracy can be within 1% of SPICE simulation. The output of the delay calculator is the delay time D and the transition time Tr.

We use a nonlinear delay model based on look-up table for buffer delay estimation. The propagation and transition tables are provided in the industrial circuit library. The value of D is computed by performing table look-up and interpolation within the propagation table that is a two-dimensional table indexed by total output capacitance and input transition time. The total output capacitance is the sum of pin and wire capacitance on the net connected with the output pin of the buffer. The transition time Tr is computed by performing simple linear interpolation within the transition table that is a one-dimensional table indexed by total output capacitance.

We build two lookup tables for delay calculation. The first table is for interconnect delay calculation, as shown in Fig. 2(a), the value of D and Tr from p to q will be calculated, here, p and q are respectively the start and the end point of the wire segment. The second table is for buffer and interconnect delay calculation, as shown in Fig. 2(b), the value of D and Tr from a to q will be calculated, here, a is the input of the buffer, and q is the end point of the wire segment.

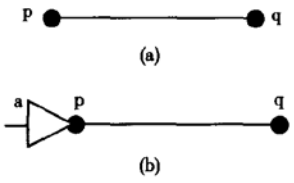

Fig. 2 (a) Interconnect; (b) Buffer and interconnect

Suppose a net has N candidate buffer locations which are uniformly distributed. For any possible buffer insertion scheme, the length of any segment can be any from 1 to $N+1$ in some units. According to our delay calculation procedure, we can easily conclude that either table is a two-dimensional table indexed by two parameters: the length of the wire segment and the input transition time (We assume that there are Tr_{max} possible transition times). Given the two parameters, we can get the corresponding value of D and Tr by table lookup.

3 Problem formulations

The problem we want to solve can be stated as follows:

Given: a two-terminal net, N candidate buffer insertion locations in the net.

Objective: find a buffer insertion scheme, so that the number of buffers inserted in the net is minimized, and the total delay of the net does not exceed the delay bound T_0.

Like the algorithm in Ref. [9], we construct graph G (shown in Fig. 3). Each column represents a level. Level 0 and level 4 represent respectively the source and the sink position. Levels 1~3 represent three candidate buffer locations (for notational convenience, we assume that there are three candidate buffer locations). Level 5 represents the position where a sink node f is inserted. Starting from left to right, the current node is connected to all the nodes on its right by directed edges. We assume that there are Tr_{max} possible transition times from Time[1] to Time[Tr_{max}] in ascending order. And corresponding Tr_{max} nodes indexed from 1 to Tr_{max} are created at each level.

First an input transition time Tr_{in} is assumed for the source. Then a node i is picked as the source node at level 0 such that Time[i] is the closest to Tr_{in}. The delay time and the output transition time Tr_{out} of the output waveform are calculated for each assumed connection. The node index k at each level is determined so that Time[k] matches Tr_{out}. By repeating a similar process to nodes level by level, all the nodes except for those at level 4 can be connected to some nodes on its right levels. Finally, all dots at level 4 are connected to the sink node f. It is clear that each path from a to f represents a buffer insertion scheme.

Each edge e is annotated with two labels: the delay time x and the number of buffers k associated with the edge. Obviously, for the edges connecting from level 0 to other levels, x is valued by the delay time in Fig. 2(a), and k is valued by 0; for the edges

connecting from level 4 to level 5, x and k are both valued by 0; for other edges, x is valued by the delay time in Fig. 2(b), and k is valued by 1.

Given such a graph theoretic interpretation, our problem can be stated as follows:

Formulation 1. Given: the above graph G and corresponding interpretation on G.

Objective: find a path p connecting a and f in G, so that K is minimized subject to T not exceeding the delay bound T_0. Here, K is the total number of buffers in the path, that is, $K = \sum_{e \in p} k$, and T is the total delay of the path, that is, $T = \sum_{e \in p} x$.

The problem can be solved by dynamic programming technique. However, firstly, for accurate delays models the RC delay are not additive, this would require that for each node we trace back the first buffer and construct the RC sub-tree, as shown in Fig. 3. Thus, there are much more partial sub-trees needed to be analyzed during bottom-up dynamic programming algorithm compared with Van Ginneken's hierarchical algorithm. Secondly, when a signal waveform is considered, the input transition time is unknown in the bottom-up procedure, we must calculate delay under Tr_{max} possible input transition times for a RC tree, this will deserve huge memory requirements and runtime cost.

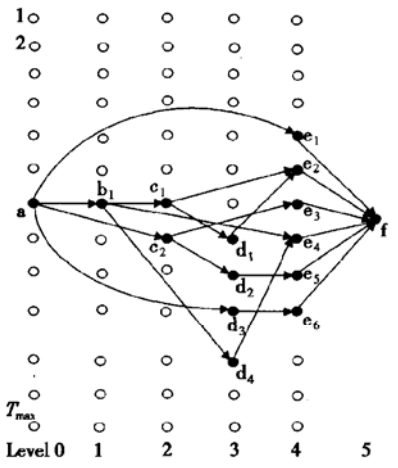

Fig. 3 Graph G for buffer insertion under accurate delay models

In our algorithm, we solve the problem by transforming it into a series of the shortest path problems. Firstly, we propose a new formulation, called balance of buffer number and delay minimization(BBNDM) problem:

Formulation 2(BBNDM). Given: the above graph G and corresponding interpretation on G.

Objective: find a path p connecting a and f in G, such that $K + \alpha T$ is minimized, here, α is the coefficient.

Obviously, we have $K + \alpha T = \sum_{e \in p} k + \alpha \sum_{e \in p} x = \sum_{e \in p} (k + \alpha x)$, thus, each edge e is weighted by $k + \alpha x$, and the BBNDM problem can be described as the shortest path problem.

4 Properties

In this section, we explain some important properties that our algorithm lies on.

Dominance property: Since paths are characterized by two parameters K and T, they may be compared with a partial order. A path p: a~ f is said to be non-dominated if all other paths p': a~ f have $\{K' > K$ or $T' > T\}$ or $\{K' = K$ and $T' = T\}$.

Lemma 1: For any value of α, the optimal solution of the BBNDM problem is not dominated by any other solution.

Proof: Let (K, T) be the optimal path found for the BBNDM problem. Suppose there exists a path (K', T') that dominates (K, T), thus, we have $\{K' < K$ and $T' < T\}$ or $\{K' < K$ and $T' = T\}$ or $\{K' = K$ and $T' < T\}$. Therefore, $K + \alpha T > K' + \alpha T'$. Since (K, T) is the optimal path found for the BBNDM problem, we have $K + \alpha T \leqslant K' + \alpha T'$. So, there is a contradiction.

Lemma 2: Let (K, T) be the path p found for the BBNDM problem under α. As the value of α decreases, the value of K decreases, and the value of T increases.

Proof: Let (K_1, T_1) be the path p_1 found for the BBNDM problem when the coefficient α is valued by α_1, and (K_2, T_2) be the path p_2 found for the

BBNDM problem when the coefficient α is valued by α_2.

Without losing generality, let

$$\alpha_1 < \alpha_2 \tag{1}$$

Since (K_1, T_1) is found for the BBNDM problem under α_1,

$$K_1 + \alpha_1 T_1 \leqslant K_2 + \alpha_1 T_2 \tag{2}$$

Similarly,

$$K_2 + \alpha_2 T_2 \leqslant K_1 + \alpha_2 T_1 \tag{3}$$

From Eqs. (2) and (3), we get

$$\alpha_2(T_2 - T_1) \leqslant K_1 - K_2 \leqslant \alpha_1(T_2 - T_1) \tag{4}$$

Hence,

$$(\alpha_2 - \alpha_1)(T_2 - T_1) \leqslant 0 \tag{5}$$

From Eqs. (1) and (5), we get

$$T_1 \geqslant T_2 \tag{6}$$

From Eqs. (4) and (6), we get

$$K_1 - K_2 \leqslant \alpha_1(T_2 - T_1) \leqslant 0 \tag{7}$$

Thus,

$$K_1 \leqslant K_2 \tag{8}$$

5 Our algorithm

5.1 Algorithm description

According to the properties in Section 4, we assign α with various values, and get the corresponding optimal solutions by the shortest path algorithm. Firstly, α is assigned a initial value, if the total delay T exceeds the delay bound T_0, we repeatedly keep doubling α, until we find the two successive values α_1 and α_2 such that the corresponding T_1 exceeds T_0, and T_2 is less than T_0. On the other hand, if starting from the initial α and the corresponding T is less than T_0, we repeatedly keep halving α, until we find α_1 and α_2.

Then we do a binary search in the range $[\alpha_1, \alpha_2]$ to identify the final value of α so that the corresponding optimal solution of the BBNDM problem satisfies the condition that the total number of buffers K is as minimal as possible subject to the total delay T not exceeding the delay bound T_0. The flow of the algorithm is shown in Fig. 4. In the algorithm, when the value of α_1 is modified, it will increase, and the total number of buffers (denoted by BufNum1) of the optimal solution under α_1 will remain unchanged or get larger; when the value of α_2 is modified, it will decrease, and the total number of buffers (denoted by BufNum2) of the optimal solution under α_2 will remain unchanged or get smaller. When the values of BufNum1 and BufNum2 both remain unchanged in at least m modifications, the algorithm ends and we get the final optimal solution. In our algorithm, we set m= 3, and the experimental results are satisfactory.

```
Main Routine
1. Find α1 and α2;
2. BufNum1= the total number of buffers of path p1 under α1;
3. BufNum2= the total number of buffers of path p2 under α2;
4. α= (α1+ α2)/2;
5. Do
6. {
7.        Find the shortes a~ f path p in G under α;
8.        if (the total delay of path p exceeds the delay bound T0)
9.        {
10.           α1= α;
11.           BufNum1= the total number of buffers of path p;
12.           p1←p;
13.       }
14.       else
15.       {
16.           α2= α;
17.           BufNum2= the total number of buffers of path p;
18.           p2←p;
19.       }
20.       α= (α1+ α2)/2;
21. }
22. until (the values of BufNum1 and BufNum2 both remain unchanged in at least m modifications)
23. return p2.
```

Fig. 4 Main routine of the algorithm

To find the shortest path, we use an efficient algorithm[9], which can avoid the waste of lots of memory. The details on the algorithm can be found in Ref. [9].

5.2 Remark on the algorithm

Our algorithm solves the problem by trans-

forming it into a series of the shortest path problems. We find the corresponding optimal solutions by the shortest path algorithm under various values of α until the algorithm converges at the final value of α so that the corresponding optimal solution of the BBNDM problem satisfies that the total number of buffers K is as minimal as possible subject to the total delay T not exceeding the delay bound T_0. As we know, the running time complexity of the shortest path algorithm is polynomial: $O(\mathrm{Tr}_{\max}^2 N^2)$. Moreover, the experimental results show that the algorithm converges very quickly in practice. Therefore, our algorithm can efficiently achieve the trade-offs between number of buffers and delay.

6 Experimental results

We have implemented our algorithm in C language on Sun WorkStation V880 and tested it with many nets of industrial circuits under 0.13μm technology. The experimental results of one of the nets are shown in Table 1. The wire length of the net is 257400μm. The delay time without inserting buffer is 99.8891ns.

In Table 1, the numbers of the candidate buffer locations are listed in column 1, the given delay constraints for the net are listed in column 2, the various values of α are listed in column 3, the numbers of buffers inserted are listed in column 4, the corresponding values of delay time and transition time of the net after buffer insertion are listed in columns 5 and 6, and in the last column, the related running times are given. Furthermore, in the remaining three rows, we give the minimized delay time and the corresponding transition time when buffers can be inserted as many as possible.

From the table, we can see that our algorithm can efficiently achieve the trade-offs between number of buffers and delay, and the running time is satisfactory. Furthermore, when the value of N gets larger, the solution gets better, while the running time is increasing.

Table 1 Experimental results for the net

N	Con /ns	α	Num	Delay time/ns	Transition time/ns	Running time/s
11	52	0.8809	7	51.2874	7.1105	0.39
	55	0.2308	4	54.6929	14.1210	0.30
	60	0.1846	3	59.0246	24.1361	0.27
	70	0.0925	2	64.4481	38.1571	0.25
	80	0.0691	1	75.3046	75.2127	0.34
	Minimized delay time= 50.0792ns, Transition time= 2.1030ns, the number of buffers= 10					
23	50	1.3500	8	49.9965	7.1105	1.78
	52	0.8897	6	51.4892	7.1105	2.07
	55	0.2927	4	54.6929	14.1210	2.29
	60	0.1664	3	58.1106	14.1210	1.97
	70	0.0914	2	64.1309	31.1466	1.92
	80	0.0690	1	75.1930	65.1977	2.12
	Minimized delay time= 47.8900ns, Transition time= 2.1030ns, the number of buffers= 18					
29	48	3.9750	18	46.5995	5.1075	4.44
	50	1.0031	7	49.8029	5.1075	3.64
	52	0.6960	6	50.8006	9.1135	4.40
	55	0.2777	4	54.7316	12.1180	4.04
	60	0.1737	3	58.3333	16.1240	5.17
	70	0.0914	2	64.0929	29.1436	4.05
	80	0.0679	1	75.0477	71.2067	5.51
	Minimized delay time= 46.0152ns, Transition time= 3.1045ns, the number of buffers= 22					

7 Conclusion

In this paper, we present a graph-based algorithm of minimizing the number of buffers subject to a given delay constraint with accurate delay models. The algorithm solves the problem by transforming it into a series of the shortest path problems. The experimental results show that the algorithm can efficiently achieve the trade-offs between number of buffers and delay, and avoid needless power and area cost. And the running time is satisfactory.

References

[1] Alpert C J, Devgan A. Wire segmenting for improved buffer insertion. IEEE/ACM DAC, 1997: 588

[2] Chu C C N, Wong D F. Closed form solution to simultaneous buffer insertion/sizing and wire sizing. International Symposium on Physical Design, 1997: 192

[3] Dhar S, Franklin M A. Optimum buffer circuits for driving long uniform lines. IEEE J Solid-State Circuits, 1991, 26(1):

32

[4] Lin S, Sadowska M M. A fast and efficient algorithm for determining fanout trees in large networks. Proc Euro Conf on Design Automation, 1991: 539

[5] Van Ginneken L P P P. Buffer placement in distributed RC-tree networks for minimal elmore delay. Intl Symp Circuits and Systems, 1990: 865

[6] Lillis J, Cheng C K, Lin T T Y. Optimal wire sizing and buffer insertion for low power and a generalized delay model. IEEE J Solid-State Circuits, 1996, 31(3): 437

[7] Okamoto T, Cong J. Interconnect layout optimization by simultaneous steiner tree construction and buffer insertion. ACM/SIGDA Physical Design Workshop, 1996: 1

[8] Menezes N, Chen C P. Spec-based repeater insertion and wire sizing for on-chip interconnect. Proc of the 12th Intl Conf on VLSI Design, 1999: 476

[9] Gao Youxin, Wong D F. A graph based algorithm for optimal buffer insertion under accurate delay models. Proceedings of Design, Automation and Test in Europe, 2001

[10] Cong J. An interconnect-centric design flow for nanometer technologies. Proceedings of the Workshop on Synthesis and System Integration of Mixed Technologies, 2001: 199

[11] Odabasioglu A, Celik M, Pileggi L T. PRIMA: passive reduced-order interconnect macromodeling algorithm. In: Proceedings of ACM/IEEE ICCAD, San Jose, CA, USA, 1997: 58

[12] Zhang Yiqian, Zhou Qiang, Hong Xianlong, et al. Path-based timing optimization by buffer insertion with accurate delay model. Chinese Journal of Semiconductors, 2004, 25(5): 520

精确时延模型下满足时延约束的缓冲器数目最小化的算法*

张轶谦　洪先龙　周　强　蔡懿慈

(清华大学计算机科学与技术系，北京　100084)

摘要：提出了在精确时延模型下，满足时延约束的缓冲器数目最小化的算法. 给出一个两端线网，该算法可以求出满足时延约束的最小缓冲器数目. 运用高阶时延模型计算互连线的时延，运用基于查找表的非线性时延模型计算缓冲器的时延. 实验结果证明此算法有效地优化了缓冲器插入数目和线网的时延，在二者之间取得了较好的折中. 算法的运行时间也是令人满意的.

关键词：缓冲器插入；互连优化；布图；VLSI

EEACC：2570　　**CCACC**：7410D

中图分类号：TN47　　**文献标识码**：A　　**文章编号**：0253-4177(2004) 11-1409-07

* 国家自然科学基金(批准号：60176016) 及国家高技术研究与发展计划(批准号：2002AAIZ1460) 资助项目

张轶谦　女，1977 年出生，博士研究生，主要从事超大规模集成电路布图设计中互连优化算法的研究.

洪先龙　男，1940 年出生，教授，博士生导师，主要从事超大规模集成电路物理设计算法的研究以及集成电路设计自动化的研究.

周　强　男，1961 年出生，副教授，主要从事超大规模集成电路物理设计算法的研究.

2004-01-08 收到，2004-06-08 定稿