重複器插入位置尋求多源多汲匯流排傳遞延遲的最小化

Propagation Delay Minimization for Multi-source Multi-sink Bus with Located Repeater Insertion

研　究　生：江侃紘　　Student : Kuang-Hung Chiang

指導教授：蔡加春　　Advisor : Dr. Chai-Chun Tsai

南華大學

資訊管理學系

碩士論文

A Thesis

Submitted to Department of Information Management
College of Science and Technology
Nanhua University
in partial Fulfillment of the Requirements
for the Degree of
Master of Information Management
June 2008
Chaiyi, Taiwan, Republic of China

中華民國　97　年　6　月

南　華　大　學

資訊管理學系

碩 士 學 位 論 文

插入多個重複器達到多源多汲匯流排傳遞延遲的最小化

研究生：汪佐紘

經考試合格特此證明

口試委員：何宗易

晏光明

高仕斌

指導教授：高仕斌

系主任(所長)：鍾鳳嬌

口試日期：中華民國 97 年 6 月 26 日

誌　　謝

首先誠摯的感謝指導教授蔡加春博士，在過去的三年裡，給予我課業上與研究上的細心指導與關懷，老師細心的教導使我在這三年中獲益良多。同時也感謝吳光閔博士，在這三年裡，不斷給予鼓勵與課外教導，在此一併感謝、感恩兩位老師的指導與鼓勵。

在研究所三年的日子裡，感謝眾位學長與同學的砥礪，讓我能安然渡過低潮期，最後感謝我的雙親，兄姊和女友，在我感到灰心時，適時給予我鼓勵，讓我更有信心的完成研究所的學業。在此將本論文獻給所有關心我的人，將這一份喜悅與榮耀分享給大家。

重複器插入位置尋求多源多汲匯流排傳遞延遲的最小化

學生：江佻紘　　指導教授：蔡加春 博士

南　華　大　學　資訊管理學系碩士班

摘　　要

自從超大型積體電路的製程技術進入深次微米之後，影響系統效能的因素已經由原先的閘級延遲改變為連線延遲，因此如何減少連線延遲便成為提昇系統效能的一個重要目標。以往對於連線延遲的計算方式是採用RC或者RLC模型，但隨著晶片的工作頻率不斷提昇，使得原本不被考慮的電感效應越來越明顯。在本論文中則使用與HSPICE誤差值極小的fitted Elmore delay(FED)模型用來評估與計算連線延遲。

匯流排普遍存在於一顆晶片內，其連線延遲也直接影響晶片內部電路的執行效能，如何減少匯流排上的訊號延遲將是影響電路執行效能的

關鍵因素。本論文提出三個演算法來降低多源多汲(multi-source multi-sink)匯流排上的訊號延遲，此演算法是以FED 模型來計算連線延遲，我們在匯流排上找出臨界路徑上每一個線段後，平均插入適當的雙向訊號重複器同時調整其大小，並繼續找出臨界路徑中訊號重複器插入的最佳位置來改善連線延遲，反覆此程序，直到臨界路徑的連線延遲不能再改善為止。根據實驗結果顯示，我們所提出的演算法與文獻比較對於不同製程參數0.18微米和0.13微米的連線延遲至少可以改善1.8%和3.7%。

Propagation Delay Minimization for Multi-source Multi-sink Bus with Located Repeater Insertion

Student：Kuang-Hung Chaing　　Advisors：Dr. Chai-Chun Tsai .

Department of Information Management
The M.I.M. Program
Nanhua University

ABSTRACT

Since the advance of deep submicron meter technology in VLSI, the performance dominating factor is changed from gate delay to interconnect delay. Therefore, how to reduce interconnection delay becomes a critical goal for improving system performance. The RC and RLC delay models are two widely used models for calculating the interconnection delay in the past. But the increment of working frequency of chip leads the designer to re-exam the effect of inductance. In this thesis the fitted Elmore delay (FED) model which has less simulation error compared to HSPICE is used for computing and evaluating interconnection delay.

Bus is an important transmission media inside a chip and its wire connection also significantly influences the performance of circuit. Eliminating the propagation delay of signal on the bus helps us to increase the performance of circuit. In this thesis we also proposed a greedy algorithm to reduce the signal transmission delay for multi-source and multi-sink

structures on the bus. In our proposed algorithm, the bidirectional repeaters are averagely inserted into the critical path and the size of repeaters is also adjusted. Afterward, the best position where the repeater should be inserted is found to improve the delay. The above steps are repeatedly executed until the minimum delay is stable. Experimental results show that our proposed algorithm can at least reduce 1.8% and 3.7% propagation delay time for the processes of 0.18μm and 0.13μm, respectively, while compared to the literatures respectively.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

In the past, the logic delay dominates the overall circuit performance in designing the VLSI system. In other word, the circuit performance is significantly restricted by the logic delay. Since the advance of deep submicron technology, a new challenge is arisen. In the process of deep submicron, propagation delay plays an important role which significantly affects the performance of VLSI system. There are several major buses exist in a chip such as data, control and address buses which significantly consumed by interconnection delay, especially for the bidirectional data bus. Figure 1 shows the circumstance of two data buses of a chip. A data can be transferred from a module to other modules via the data bus in a different time period, e.g., an n-bit data is propagated from the memory to the microprocessor at the time period 1 and another n-bit data is transferred from the FPGA to the D/A converter at the time period 2. It is noted that the D/A converter can be as a source or sink in a non-overlap time period. Thus, the data bus is a multi-source multi-sink bidirectional bus. In general, data run on the data bus frequently while a chip works up and their propagation delays dominate the chip performance. Therefore, maximizing reduction for the propagation delay on a bus has the potential to investigate.

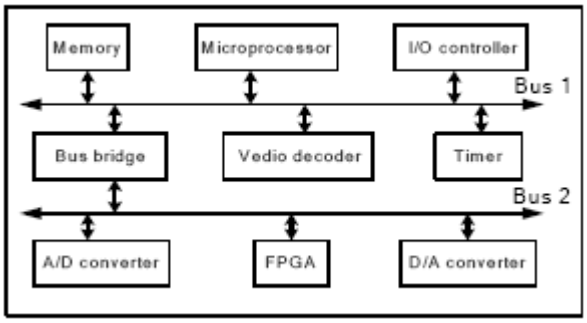


Figure 1 A typical bus structure in a chip

1.1 Motivation

In the era of the chip working at low frequency, the RC model [1] proposed by Elmore is the popular interconnection delay calculation model. However, the Elmore model only considers the resistances and capacitances impact on interconnection delay. Since the calculation error between the Elmore's model and HSPICE is very small and the fast computational property of Elmore mode, the Elmore model becomes the standard interconnection delay calculation method in the field of VLSI design. In addition, with the increase of the working frequency of chip, the inductance effect which was ignored in the past should be re-examined in this new era. In the second order RLC circuit, it will result in oscillating effect and this effect leads the RC and RLC model to produce imprecise interconnection delay calculation. Therefore, the fitted Elmore delay (FED) model [2] instead of RC and RLC models is adopted in this thesis since the properties of fast computation and less computation error.

After the introduction of interconnection delay mode, we describe how to reduce the interconnection delay in the following. Since the magnitude of interconnection delay significantly influences the performance of circuit, how to efficiently eliminate the interconnection delay becomes the critical problem in modern chip design. There are many related researches [6-8, 10-11] had been proposed recently. In general, the commonly ways to reduce interconnection delay are the methods of adjusting the source driving resistance, resizing the wire size, and buffer insertion. In the early periods, adjusting the source driving resistance is widely used method to reduce the interconnection delay. Unfortunately, the effect of interconnection delay reduction by adjusting the source driving resistance is not significant. Therefore, the method of resizing wire size was proposed. The basis of this method is that the resistance of wire is proportional to the area of wire. The

larger wire area contributes more wire resistance. In other word, reducing the length of wire can also reduce the wire resistance. In the method of buffer insertion and resizing, if the delay of inserted buffer is less than that of the interconnection delay of half-length, the buffer insertion can reduce full-length interconnection delay. By incorporating the buffer resizing mechanism, the optimal interconnection delay reduction can be achieved. Therefore, the method of buffer insertion and resizing is used in this thesis. In general, a signal repeated is usually composed by a uni-directional buffer. Nevertheless, the bidirectional signal repeater [3, 4] is used in the thesis due to the bidirectional signal repeater can eliminate the inconvenience of adding extra control signal which is unavoidable in uni-directional signal repeater. Although using bidirectional signal repeater can eliminate the problem of control signal, it also complicates the delay reduction problem. Therefore, resizing for the wire is not considered in this thesis to simplify the optimization problem.

1.2 Related Works

In 1948, Elmore [1] proposed the first interconnect delay computation model which used first order equation to derive RC model. Henceforward, the RC delay model is widely adopted for calculating the delay due to its properties of fast computation and exactitude. Till 1987, Wyatt [5] improved the Elmores' RC model to further increase the precision of propagation delay calculation.

There are many papers [6-8] which talking about propagation delay reduction were proposed in the past. However, the most effect way to reduce delay is by doing the buffer or signal repeater insertion. In addition, the process of buffer resizing is also adopted for further eliminating the

interconnection delay. The literatures [8, 9] used the mechanism of buffer insertion and resizing to reduce interconnection delay. In [9], the author not only inserted the buffer to lower the interconnection delay but adjusted the size of wire to achieve optimal solution. The advantage of signal repeater is that it can detect the signal direction automatically without extra control signal. Previous works of [10, 11] are two papers which used signal repeater insertion and resizing method to reduce interconnection delay. In [12], the authors designed a low power and low delay signal repeater for optimally application. The papers [14-20] mentioned above are all based on the RC and RLC model to compute the interconnection delay. But with the increase of the working frequency and manufacture process, a more precise model called [10] FED model has been proposed for calculating the interconnection delay. However, several simulation software proved that the FED model can achieve minimum calculation error and hence promoting the application of FED model. The literature [1] is applied in this thesis to calculate the delay because the FED model was derived from the Elmore delay mode so it has the capacities of fast computation speed and less calculation error while compared to HSPICE. Therefore, we tried to propose a greedy algorithm which inserts proper amount of bidirectional signal repeaters into wire and resizes it to solve the problem of minimum delay optimization. In addition, the FED model is adopted in our proposed algorithm to compute the interconnection delay. Finally, our proposed algorithm can be used for the multi-source and multi-sink bus structures.

1.3 Thesis Organization

This thesis is organized as follows: In Chapter 2, the problem we want to solve will be precisely described and some mathematical symbols are also defined in this chapter. The interconnection delay model and the equivalent signal repeater model are introduced completely in Chapter 3. Chapter 4 completely describes the greedy algorithm which we adopted in this thesis to solve the interconnection delay minimization problem. In addition, the time complexity analysis is also mentioned in chapter 4. In the implementation chapter, Chapter 5, the interconnection delay after bidirectional signal repeater insertion will be computed and several comparisons are also resulted to show the improvement of our proposed algorithm. Chapter 6 gives the conclusion and the future works.

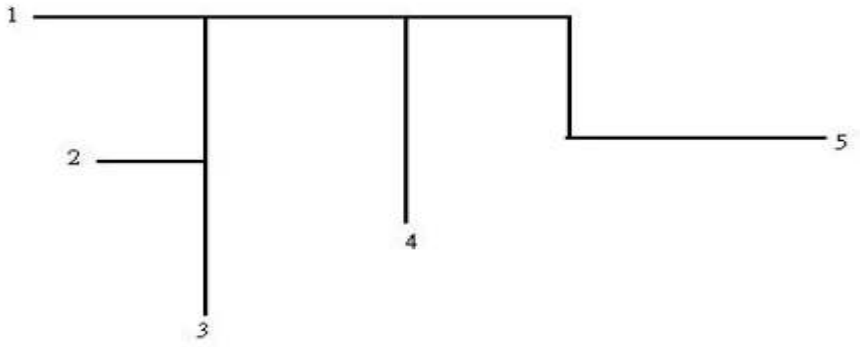
Chapter 2 Problem Formulation

With the increase of working frequency, the characteristic of interconnection should be taken into account while computing interconnection delay. In the previous RC model, only wire resistance and capacitance had been considered for calculating interconnection delay. But when working at high frequency speed, the inductance effect becomes essential so that leads to the traditional RC model useless. Therefore, the traditional RC model should be modified to include the influence of inductance effect. The modified RC model is the so called RLC model. Since the interconnection delay significantly affects the performance of overall system, we should try to reduce the interconnection delay as small as possible. The general method to reduce interconnection delay is breaking the length of wire by inserting a uni-directional buffer. But the drawback of uni-directional buffer is that it reduces the interconnection delay by only one direction. To support bidirectional interconnection delay reduction, the additional control signals should be added to the uni-directional buffer to do so. However, in order to avoid the additional control signal inclusion, the bidirectional buffer is adopted in this thesis. The main difference between uni-directional buffer and bidirectional buffer is that the bidirectional buffer not only can achieve bidirectional delay reduction but it can also automatically determine the direction of current by detecting the voltage level so that the extra control signal is unnecessary in this type of buffer. Since the method to transmit signal has two types: uni-directional and bidirectional transmission, the multi-source and multi-sink bus structure is applied in this thesis.

Assume that a bus structure contains m terminals and n wires and each terminal is represented by unequal Source or Sink at different timing instance. In other word, a terminal represented by source at certain timing instance

might be represented by sink at other timing instance. Therefore, only one source and multiple sinks composed a circuit at any timing instance. At any timing instance, it will have signal flows through several wires and the collection of these wires called path. It is possible that the signal flows through the same wires at different timing instance and each timing instance has different interconnection delay from source terminal to sink terminal. Therefore, all timing instances together compose the multiple sources and multiple sinks bus structure. By our definition, the critical path is the path which has the longest interconnection delay.

In the deep submicron manufacture process, the interconnection delay is usually longer than gate delay. Therefore, the goal is to minimize the interconnection delay on critical path. However, reducing the interconnection delay of critical path may introduce new critical path in practice. Therefore, the order for placing the signal repeater on the wire becomes the noticeable problem. In order to solve this problem, we introduce a signal repeater insertion algorithm in Chapter 4. Figure 2.1 (a) shows a bus structure which contains 9 wires and 5 terminals and Figure 2.1(b) shows the corresponding direction of signal flow on each wire at different timing instance. For example, there are two signals transmitted from source terminal 1 to sink terminals 3 and 4 and so forth. From the above explanation, we found that it will be only one source terminal and multiple sinks at any timing instance. Hence, the problem we would like to solve is to find out the critical path from these 8 timing instances and reduce the interconnection delay of critical path so that the interconnection delay as small as possible. As a result, the performance of overall system can be improved.



(a) Five terminals and nine bus-segments

Time interval	Source	Sinks
Period 1	1	3 · 4
Period 2	2	1 · 5
Period 3	3	4 · 5
Period 4	4	1 · 3

(b) Four timing periods

Figure 2.1 Bus architecture with five terminals

Assume that t_{ij} represents the interconnection delay from source i to sink j without the insertion of signal repeater. Our goal is to find the path which has maximum t_{ij} and insert a signal buffer to minimize t_{ij}. Such as follows:

$$min(max\ t_{ij}) \quad \text{.....(1)}$$

From the problem description, several assumptions are made as follows:

1. All places where signal repeater can be inserted are already known in bus structure.
2. All wires have the same wire width in bus structure.

In summary, the problem about how to reduce interconnection delay by

inserting signal repeater can be briefly described as follows. Given a multiple sources and multiple sinks bus structure which has n places where the signal repeaters can be inserted into and p paths. t_{ij} is the interconnection delay on the path from source i to sink j. Our target is to find out the maximum t_{ij} and insert a signal repeater to minimize interconnection delay.

From the problem description of previous sections, we observed that there are three common ways that can be used to reduce interconnection delay. These three methods are:

1. Adjusting driving resistance of source terminal.
2. Wire size resizing.
3. Buffer or Signal Repeater insertion and its size adjusting.

The most common and general way to reduce interconnection delay is to adjust the driving resistance of source terminal. Since the driving resistance of source terminal is usually the output resistance of logic gate, the increasing in logic gate size also reducing the output resistance of logic gate and thus the interconnection delay can also be reduced. Unfortunately, this method reduces the interconnection delay slightly. Therefore, additional incorporating of other method to further reduce the interconnection delay is required.

Proper resizing for wire size can efficiently reduce interconnection delay. As we know that the RC model is a popular way to compute interconnection delay. In this RC model, the R represents wire resistance and the C is wire capacitance. In other word, resizing the size of wire also refers to change the area of wire. Nevertheless, the resistance R is proportional to the area of wire. Therefore, the interconnection delay can be reduced by resizing the area of wire. Before describing how signal repeater reduces the interconnection delay, we should explain the RC delay model first. The equation [11] is

$$t_D = rcl^2 / 2 \quad \text{.....(2)}$$

where t_D indicates wire delay, r is the wire resistance per unit length wire, c refers to the wire capacitance per unit square, l is the length of wire. If a buffer has been inserted into the middle of wire, the length of the wire after buffer insertion will become the half of the original wire length. In this situation, the wire delay becomes as follows.

$$t_D = 2 * rc(1/2)^2 / 2 + t_{buf} = 0.5 * rcl^2 / 2 + t_{buf} \quad \text{.....(3)}$$

where t_{buf} indicates the intrinsic delay of buffer. From (3), if t_{buf} is less than the half of the original interconnection delay, the total interconnection delay can be reduced by buffer insertion.

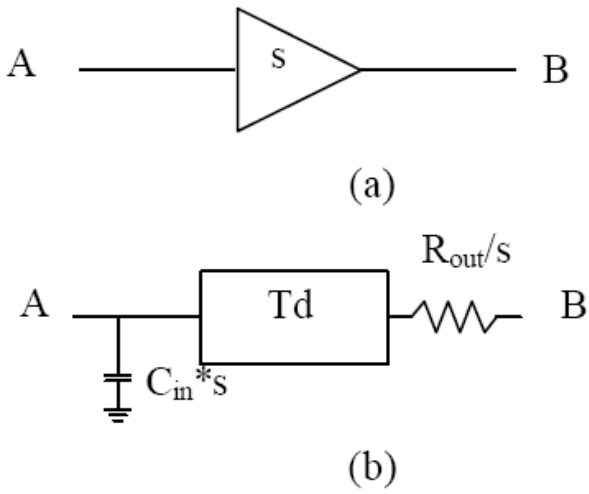


Figure 2.2 (a) A repeater model and (b) its equivalent delay model.

From (3) we observed that the interconnection delay can definitely reduced by buffer insertion. However, the buffer inserted here is the unit size buffer. In some case, the unit size buffer may useless for interconnection delay

reduction. Therefore, adjusting the size of inserted buffer is necessary to reduce interconnection delay. Figure 2.2 shows the equivalent circuit of a buffer. In this equivalent circuit, C_{in} indicates the input capacitance of unit size buffer, R_{out} is the output resistance of unit size buffer, Td refers to the intrinsic delay of buffer, and s indicates the size of buffer. With the increase of buffer size, the input capacitance of buffer is increased and the output resistance of buffer is decreased as well. In this situation, the intrinsic delay of buffer remains unchanged. Therefore, adjusting the buffer size results in more output resistance reduction and less input capacitance increasing at the beginning, and hence reduces the interconnection delay. Once the input capacitance increasing over the output resistance reduction, the interconnection delay will increase contrarily. Therefore, efficient resizing for the buffer size can significantly reduce the interconnection delay.

The three methods mentioned above are the common ways to reduce interconnection delay. Mixed using of these three methods can result better performance. But the uni-directional buffer only supports one directional signal flow. The drawback of uni-directional buffer is that it cannot transmit two-directional signals. To support two-directional signal transmission, additional control signal is required for uni-directional buffer to determine the direction of signal flow. However, extra control signal contributes additional hardware cost. Therefore, the bidirectional signal repeater which automatically determines the direction of signal flow without control signal supporting is used in this thesis to reduce interconnection delay. The detail structure of bidirectional signal repeater will be described completely in Chapter 3.

Using bidirectional signal repeater indicates that signal flow has two directions. Therefore, adjusting the size of wire size is not considered here since adjusting the wire size must satisfy the driving rule. For uni-directional

buffer, it is easy to achieve interconnection delay minimization by considering the adjustment of wire size since the property of single directional signal flow. But for the bidirectional signal buffer, if we take the mechanism of adjusting the wire size into account, the problem becomes very difficult to solve since nobody guarantees that the minimum interconnection delay for one direction is also the minimum interconnection delay for another direction. So, this is the reason that why the method of adjusting the wire size is not adopted in this thesis. In this thesis, we only use the methods of bidirectional signal buffer insertion and signal buffer resizing to achieve interconnection delay minimization.

After deciding to adopt the methods of bidirectional signal repeater insertion and signal repeater size adjusting, the next step is to decide the place where signal buffer should be inserted. Although the interconnection delay can be reduced by inserting the bidirectional signal repeater, the place where the signal buffer to be inserted significantly affects the performance of interconnection delay minimization. Therefore, deciding the place where the signal repeater to be inserted such that the interconnection delay can be minimized becomes the critical problem in this thesis.

Chapter 3 Delay Model

3.1 FED Delay Model

Abou-Seido [10] proposed a new model called fitted Elmore delay (FED). FED is significantly more accurate than the Elmore delay model. The maximum error in delay estimation is at most 2% for FED model, compared to 8.5% for the scaled Elmore delay model. The average error is less than 0.8%.

The notations of technology parameters in our study are listed below.

W_{min} : the minimum wire width

r_g : the output resistance of a minimum device

c_g : the input capacitance of a minimum device

r : the sheet resistance

c_a : the unit area capacitance

c_f : the unit fringing capacitance

l : the length of a wire

For an interconnect wire of length l and width w connecting a driver with driver resistance rd and a load with load capacitance cl. In our study, we adopt $rd = rg/100$, $cl = cg*100$ and $w = 6* Wmin$ on 0.18μm technology.

The fitted Elmore delay is given by:

$$\begin{aligned}&\mathrm{FED}(r_d, c_l, l, w)\\&= \mathrm{A}\cdot r_d c_a l w + \mathrm{B}\cdot r_d c_f l + \mathrm{C}\cdot r_d c_l + \mathrm{D}\cdot r c_a l^2/2 + \mathrm{E}\cdot r c_f l^2/2w\\&\quad + \mathrm{F}\cdot r l c_l / w \qquad (4)\end{aligned}$$

The technology parameters and coefficients of the Fitted Elmore models for all technologies are given in Table 3.1 and Table 3.2.

Table 3.1 Technology parameters

	$0.25\mu m$	$0.18\mu m$	$0.13\mu m$	$0.07\mu m$
$W_{min}(\mu m)$	0.25	0.18	0.13	0.07
$r_g\,(\Omega)$	16200	17100	22100	22100
$c_g(fF)$	0.282	0.234	0.135	0.066
$R\,(\Omega/\square)$	0.073	0.068	0.081	0.095
$c_a(fF/\mu m^2)$	0.059	0.060	0.046	0.056
$c_f(fF/\mu m)$	0.082	0.064	0.043	0.040

Table 3.2 Coefficients for the fitted Elmore delay model

	$0.25\mu m$	$0.18\mu m$	$0.13\mu m$	$0.07\mu m$
A / ln 2	1.00724	1.00962	1.01258	1.01863
B / ln 2	1.02993	1.03047	1.03010	1.02619
C / ln 2	1.00332	1.00426	1.00511	1.00530
D / ln 2	1.12520	1.12524	1.12673	1.13639
E / ln 2	1.10598	1.10582	1.10463	1.09722
F / ln 2	1.04665	1.04468	1.04836	1.06471

A simple tree as shown in Figure 3.1

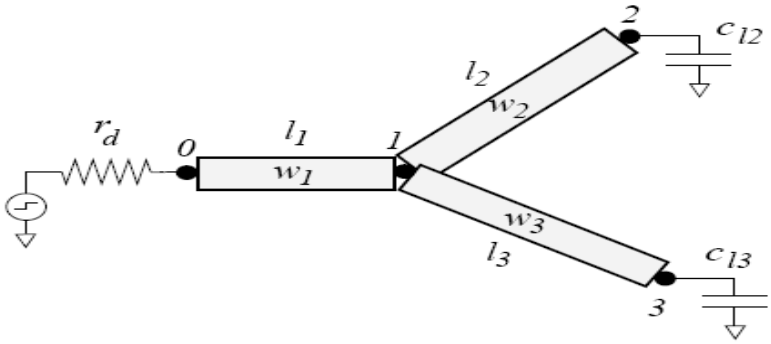


Figure 3.1 An example of a routing tree

The fitted Elmore delay model for interconnect trees is obtained by scaling the six terms above by the constants A, B, C, D, E, and F found by multiple linear regression for a single wire. There is no need to perform curve fitting again.

Fitted Elmore delay for node 2,

FED delay2= A · $r_d\, c_a\,(l_1\, w_1 + l_2\, w_2 + l_3\, w_3)$

+ B · $r_d\, c_f(l_1 + l_2 + l_3)$

+ C · $r_d\,(c_{l\,2} + c_{l3})$

+ D · $r\, c_a$ / 2 (l_1^2 + $2\, l_1\, l_2\, w_2 / w_1$ + $2\, l_1\, l_3\, w_3 / w_1$ + l_2^2)

+ E · $r\, c_f$ / 2 (l_1^2 / w_1 + $2\, l_1\, l_2 / w_1$ + $2\, l_1\, l_3 / w_1$ + l_2^2 / w_1)

+ F · r ($l_1 / w_1 * c_{l\,2}$ + $l_1 / w_1 * c_{l\,3}$ + $l_2 / w_2 * c_{l\,2}$)

For a general tree, let T be the set of indices of all tree edges. Let $T(i)$ be the set of indices of tree edges at the downstream of edge i. Let S be the set of indices of all sinks. Let $S(i)$ be the set of indices of sinks at the downstream of edge i. Let $P(k)$ be the set of indices of tree edges along the path from the driver to node k.

Fitted Elmore delay for node k,

$$= \mathrm{A} \cdot r_d \sum_{i \in T} c_a l_i w_i + \mathrm{B} \cdot r_d \sum_{i \in T} c_f l_i + \mathrm{C} \cdot r_d \sum_{j \in S} c_{lj}$$

$$+ \mathrm{D} \cdot \sum_{i \in P(k)} r l_i / w_i (c_a l_i w_i / 2 + \sum_{j \in T(i)} c_a l_j w_j)$$

$$+ \mathrm{E} \cdot \sum_{i \in P(k)} r l_i / w_i (c_f l_i / 2 + \sum_{j \in T(i)} c_f l_j)$$

$$+ \mathrm{F} \cdot \sum_{i \in P(k)} r l_i / w_i (\sum_{j \in S(i)} c_{lj}) \quad \text{.....(5)}$$

3.2 Delay Valuation with Repeater Insertion

Figure 3.2 shows a typical bus with an inserted bidirectional repeater [13]. From the figure, two repeaters S_{RU} and S_{LD} are inserted into the middle of a two-source two-sink bus and, *a* and *b* are drivers and sinks used on different timing periods. That is, a signal can be propagated from the source *a* to the sink *b* via the repeater S_{RU} and another signal is propagated from the source *b* to the sink *a* via the repeater S_{LD}. The propagation delay from the source *a* to the sink *b*, t_{ab}, can be derived as below.

$$t_{ab} = t_{ab'} + T_d + t_{b'b} \quad \text{.....(6)}$$

R_{da} is the output resistance of the source driver *a* and C_{Lb} is the loading capacitance of the sink *b*. $r_{aa'}$ is the resistance of the bus segment *aa'* and $c_{b'b}$ is the capacitance of the bus segment *b'b*.

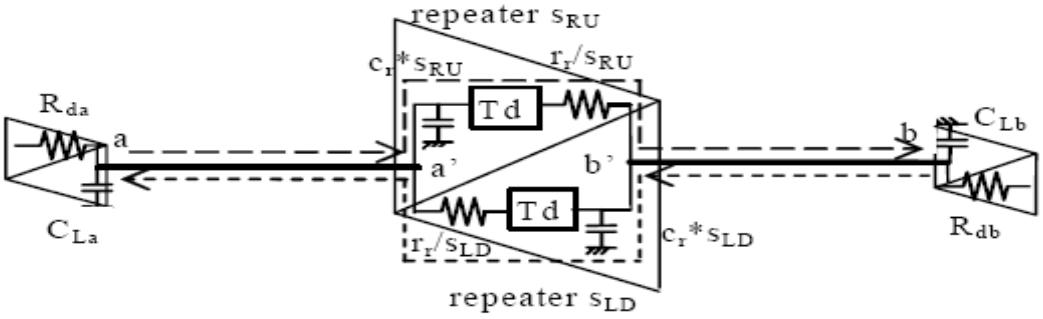


Figure 3.2 Two-source two-sink bus with an inserted bidirectional repeater

In one-source one-sink bus with an inserted repeater, the position where the repeater inserted would effect the delay of the bus. Figure 3.3 shows an illustration of our implementation which demonstrates a one-source one-sink

with repeater insertion case. Table 3.3 shows the parameters of 0.18μm technology.

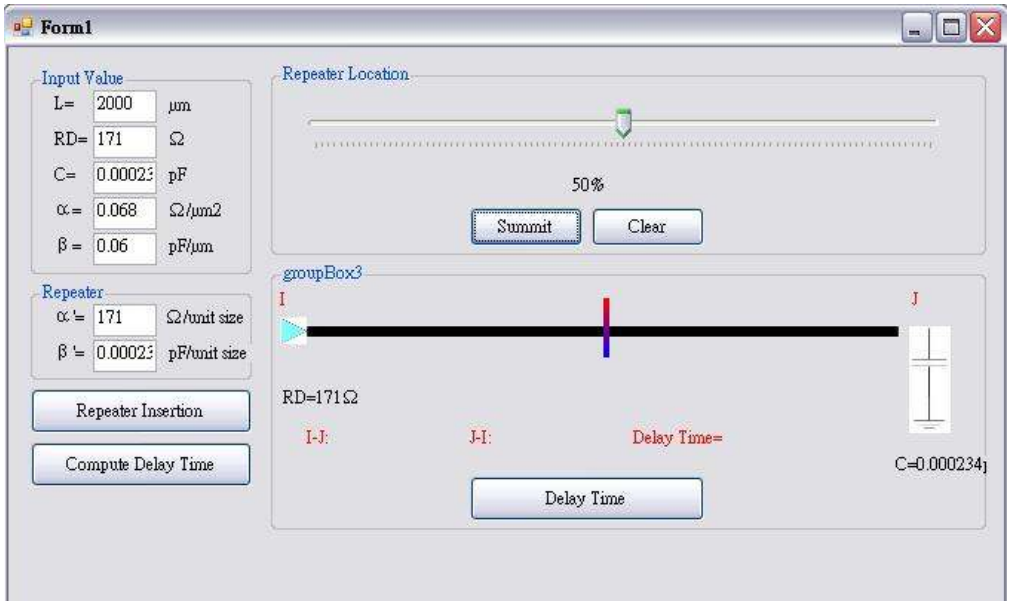
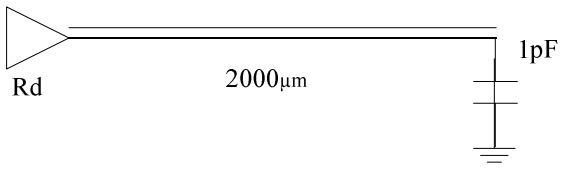


Figure 3.3 Implementation demonstrates a one-source one-sink bus with a repeater insertion

Table 3.3 The parameters based on 0.18μm technology

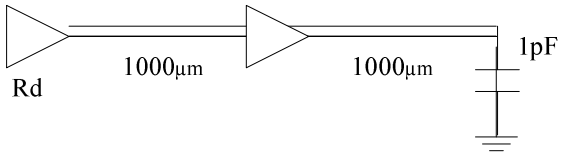
Tecnology	$Rl\ (\Omega/\square)$	$(fF/\mu m^2)$	$r_r(\Omega)$	$c_r(fF)$	$Td(ps)$
0.18μm	0.068	0.118	180	23.4	36.4

Figure 3.4 shows the case of no repeater insertion between source and sink on the bus which has 2000μm in length. The calculated delay of this case is 375ps. Figure 3.5 shows another case which one repeater is inserted into the half position between source and sink. The calculated delay of this case is 340ps. In addition, the case of one repeater inserted into the one/fourth position between source and sink is shown in Figure 3.6. The delay is 340ps. From above three cases, we can observe that the delay is varied with the repeater position. Therefore, adjusting repeater position may sometime reduce or increase delay.



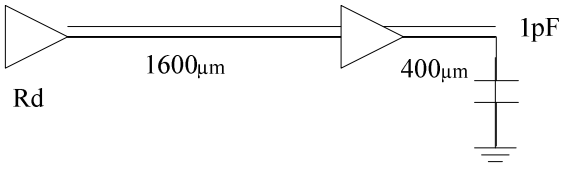
delay =375ps

Figure 3.4 No repeater between source and sink



delay =340ps

Figure 3.5 One repeater is inserted into the half position between source and sink



delay =313ps

Figure 3.6 One repeater is inserted into the one/fourth position between source and sink

Chapter 4 Proposed Algorithms

In this chapter, we will describe signal repeater insertion algorithm [13] and three proposed algorithms in detail.

Tsai [13] proposed a greedy algorithm, Bus_Repeater_Insertion, to minimize the maximum propagation delay of the bus. Always find a new critical path and a number of c segments along the current critical path and try to insert proper sized repeaters into the segments along the critical path. This process is repeated until no any improvement in delay reduction. The detailed algorithm is stated as below.

Bus_Repeater_Insertion(*n,p***)**/* n and p are number of different segments and periods, respectively. */
{repeat
critical_delay **= Find_Critical_Path(***n,p,c***);** /*c is the return value that is the number of segments along the current critical path. */
{ *pre_delay=critical_delay; pre_max_delay=MAX;*
for(*x=1 to c***)**
{ *Let rn is the number of repeaters inserted into the current critical path;*
if(*the segment x have not inserted any repeater yet***)**
rn =rn+1;
max_delay=**Adjust_Previous_Repeater(***rn***);**
if(*max_delay<pre_max_delay***)**
pre_max_delay=max_delay;
}//end of for
critical_delay=pre_max_delay;
if(*critical_delay<pre_delay***)**
Store the repeater information including inserted repeater locations, sizes, and directions.
}
until *critical_delay >= pre_delay*
}

In the algorithm, the function of Find_Critical_Path is to find the new critical path with the maximum propagation delay and to get the total number of c segments along the critical path. Insert repeaters *rn* into the number of c locations and size the repeaters to minimize the bus critical delay.

The function of Adjust_Previous_Repeater shown as following is used to iteratively size the number of repeaters *rn* until the current critical delay does not improve anymore. First insert a unit-size repeater into the middle of a segment y, and then size the repeater until the delay of adjust_critical_delay cannot be reduced. The inserted repeater in the segment y will be unidirectional that depends only the signal propagation direction to the segment toward the left, down, right, or top. If the propagation direction to the segment y includes the left and right (top and down) in different periods, the inserted repeater would be bidirectional. Consequently, two unidirectional repeaters used for the bidirectional propagation are required to size alternatively for minimizing the delay.

```
Adjust_Previous_Repeater(rn) /* number of repeater rn will be sized for reducing the critical delay */
{repeat
    for (y=1 to rn) /* Size a repeater that is possible to insert into the middle of segment y for reducing the critical delay */
    { y.LD_size=0; y.RU_size=0;
        repeat
            if(segment y has the direction of flowing to Left or Down)
            { repeat
                adjust_critical_delay=Bus_Repeater_Sizing(LD_size);
              until adjust_critical_delay cannot be reduced
              y.LD_size=LD_size;
            }


            if(segment y has the direction of to Right or Up)
            { repeat
                adjust_critical_delay=Bus_Repeater_Sizing(RU_size);
              until adjust_critical_delay cannot be reduced
              y.RU_size=RU_size;
            }
        until adjust_critical_delay cannot be reduced
        Store y.LD_size and y.RU_size;
    } //end for
 until adjust_critical_delay cannot be reduced
 return (adjust_critical_delay);
}
```

The function, Adjust_Previous_Repeater, includes another major call of Bus_Repeater_Sizing. According to [21], Bus_Repeater_Sizing takes the time of $O(p*c*n^3)$, where p is the number of different periods, c is the segments along the critical path, and n is the number of possible repeater locations. Since $c<<n$, thus the running time of Bus_Repeater_Sizing can be reduced to be $O(p*n^3)$. The function Adjust_Previous_Repeater has three *repeat* loops

and one *for* loop. These *repeat* loops run at most s times, s is the maximum repeater size. Thus the time complexity of the function is $O(s^3*p*n^4)$. Since the size s is always less than n, the execution time of Adjust_Previous_Repeater can thus be reduced to be $O(p*n^4)$. The main procedure, Bus_Repeater_Insertion, has a *repeat* loop and a *for* loop. The *repeat* loop runs at most n times and the *for* loop runs at most c times. Therefore, the time complexity of the algorithm is $O(p*n^5)$.

4.1 Multiple Repeater Insertion Algorithm

The first proposed algorithm is called Multiple Repeater Insertion Algorithm (MRIA). The main idea of proposed MRIA is that it inserts more than one repeaters into circuit to achieve delay minimization. The proposed MRIA use function of Bus_Repeater_Insertion [13] and finds out the critical path first. Afterward, the repeater is inserted into the segment of current critical path and the size of repeater is adjusted as well, the function of Find_Delay_Time_for_Critical_Path is to compute delay for critical path. After the repeater insertion and resizing procedures, the maximum delay and critical path are re-found. If the re-found critical path is the same as the critical path found previously, the repeater insertion and resizing processes are performed for the next segment except the previous processed one. Otherwise, the repeater insertion and resizing processes are performed for the segments sequentially. The above processes are operated repeatedly until the maximum delay can’t be further improved. The pseudo code of proposed MRIA is described as follows.

```
Multiple Repeater Insertion
{
    Bus_Repeater_Insertion(n,p); // reference [13]
    repeat
    {
        critical_delay = Find_Critical_Path(n,p,c,i) /*c is the return value
        that is the number of segments along the current critical path and I is
        the index of critical path*/
        for(x=1 to c)
        {
            repeat
            {
                previous_critical_delay=Find_Delay_Time_for_Critical_P
                ath(i);
                Insert one repeater into current segment x and evenly
                arranges the inserted repeaters on segment x;
                Adjust_Previous_Repeater(rn);/*Resizing rn repeaters*/
                current_critical_delay=Find_Delay_Time_for_Critical_Pa
                th(i);
                if(current_critical_delay < previous_critical_delay)
                    Store the repeater information including the number of
                    inserted repeaters, locations, and sizes.
            } while (current_critical_delay < previous_critical_delay)
        }
    } while (current_critical_delay < previous_critical_delay)
}
```

In our proposed algorithm, the literature [13] is adopted in the first step to produce the result with single repeater insertion. Therefore, the computational complexity of proposed algorithm should additional include the computational complexity of literature [13]. According to [13], Bus_Repeater_Sizing takes the time of $O(p*c*n^3)$, where p is the number of

different periods, c is the segments along the critical path, and n is the number of possible repeater locations. Since $c<<n$, thus the running time of Bus_Repeater_Sizing can be reduced to be $O(p*n^3)$. The function Adjust_Previous_Repeater has three *repeat* loops and one *for* loop. These *repeat* loops run at most s times, s is the maximum repeater size. Thus the time complexity of the function is $O(s^3*p*n^4)$. Since the size s is always less than n, the execution time of Adjust_Previous_Repeater can thus be reduced to be $O(p*n^4)$. The main procedure, Bus_Repeater_Insertion, has a *repeat* loop and a *for* loop. The *repeat* loop runs at most n times and the *for* loop runs at most c times. Therefore, the time complexity of the algorithm is $O(p*n^5)$.

In this algorithm, it inserts some repeaters into the circuit and each repeater is sized. Assume there are m repeaters are inserted in maximum. Since there are m addition repeater positions are introduced, the computational complexity becomes $O(p*(m*n^5))$ in contrast with the algorithm of single repeater insertion. Nevertheless, since the m and n have the same meaning physically, the computational complexity of proposed MRIA is $O(p*n^5)$.

4.2 Repeater Position Adjustment Algorithm

The second proposed algorithm is called Repeater Position Adjustment Algorithm (RPAA). The main idea of proposed RPAA is on the repeater position adjustment. The proposed RPAA places the repeater on different position along the critical path and the best position which results in minimum delay is selected as the final result. The operations of proposed RPAA are described as follows briefly. We use function of Bus_Reapeter_Insertion [13] and the critical path is found for the following RPAA operation. The repeater is placed on the segment of critical path with the position which decided by binary search. After the best position is decided, the critical path and maximum delay are re-found. If the re-found critical path is the same as the critical path found previously, the repeater position adjustment and resizing processes are performed for the next segment except the previous processed one. Otherwise, the repeater position adjustment and resizing processes are performed for the segments sequentially. The above processes are operated repeatedly until the maximum delay can't be further improved. The pseudo code of proposed RPAA is listed as follows.

```
Repeater_Postion_Adjustment
{
    Bus_Repeater_Insertion(n,p) // reference [13]
    repeat
    {
        critical_delay = Find_Critical_Path (n,p,c,i); /*c is the return value
        that is the number of segments along the current critical path and i is the index of
        critical path*/
        for(x = 1 to c)
        {
            Repeater_Position_Adjustment(x,i);
            current_critical_delay = Find_Critical_Path (n,p,c1,i1); /*i1 is the index
            of critical path*/
            if(i != i1) /*Determine whether the critical path is the same*/
            break; /*Exit the for loop*/
        }
    }until current_critical_delay >= critical_delay
    Repeater_Position_Adjustment(x,i)
    {
        critical_delay_t = Find_Delay_Time_for_Critical_Path (i);/*Find the delay for
        critical path i*/
        repeat
        {
            y = Binary_Search(x,l,r); /*Binary search between sub-segment l and r for
            segment x, until segment length of l,r less than 50μm */
            Place the repeater at the position y on segment x;
            Adjust_Previous_Repeater(1); /*Resizing the repeater*/
            current_critical_delay_t = Find_Delay_Time_for_Critical_Path (i);
            if(current_critical_delay_t<critical_delay_t)
            {
                critical_delay_t = current_critical_delay_t;
                best_position = y;
                Record the best repeater size;
            }
        }until current_critical_delay_t >= critical_delay_t;
        Place the repeater at the best_position on segment x;
    }
}
```

The computational complexity of this algorithm is proportional to the number of positions that the repeater should be placed. Since the position that the repeater should be placed is decided by binary search, it results in $O(logn)$ additional computational complexity. Therefore, the additional introduced computational complexity is $O(logn)$ possible repeater positions. Since $logn << n$, the computational complexity of proposed RPAA is $O(p*n^5)$.

4.3 Multiple Repeater Insertion with Position Adjustment Algorithm

The third proposed algorithm is called Multiple Repeater Insertion with Position Adjustment Algorithm (MRIPAA). The main idea of proposed MRIPAA is to combine MRIA and RPAA algorithms to reduce maximum delay. The procedure of proposed MRIPAA is described as follows. First, the proposed MRIA algorithm is performed to insert proper amount of repeaters into circuit to achieve first-pass delay minimization. Afterward, the proposed RPAA algorithm is applied to the circuit for aiming at best delay optimization. The pseudo code of proposed MRIPAA is listed as follows.

```
Multiple_Repeater_Insertion_Position_Adjustment
{
    Multiple Repeater Insertion();
    repeat
    {
        critical_delay = Find_Critical_Path (n,p,c,i); /*c is the return value
        that is the number of segments along the current critical path and i is
        the index of critical path*/
        for(x = 1 to c)
        {
            Repeater_Position_Adjustment(x,i,bc);/*bc is the number of
            repeaters on segment x
            current_critical_delay = Find_Critical_Path (n,p,c1,i1); /*i1 is t
            he index of critical path*/
            if(i != i1) /*Determine whether the critical path is the same*/
                break; /*Exit the for loop*/
        }
    }until current_critical_delay >= critical_delay
```

```
Repeater_Position_Adjustment(x,i,bc)
{
    critical_delay_t = Find_Delay_Time_for_Critical_Path (i);/*Find
    the delay for critical path i*/
    repeat
    {
        for(z = 1 to bc)
        {
            y = Binary_Search(x,l,r); /*Binary search between
            sub-segment l and r for segment x, until segment length of l,r
            less than 50μm */
            Place the repeater at the position z on segment x;
            Adjust_Previous_Repeater(z); /*Resizing the repeater*/
            current_critical_delay_t=Find_Delay_Time_for_Critical_
            Path (i);
            if(current_critical_delay_t<critical_delay_t)
            {
                critical_delay_t = current_critical_delay_t;
                best_position = y;
                Record the best repeater size;
            }
        }
    }until current_critical_delay_t >= critical_delay_t;
    Place the repeater at the best_position on segment x;
}
}
```

In this algorithm, the main idea of proposed MRIPAA is to combine MRIA and RPAA algorithms. In MRIA algorithm and RPAA algorithm, the computational complexity of proposed MRIA and RPAA are $O(p*n^5)$. Therefore, the computational complexity of proposed MRIPAA is $O(p*n^5)$.

Chapter 5 Experimental Results

We implement the greedy algorithm in Java language and run on a PC Pentium-4 2.8GHz and 512MB memory under MS-Windows XP. Table 5.1 shows the parameters of 0.18μm and 0.13μm [10] technologies. Since no standard benchmarks are given, we create seven bus cases shown in Table 5.2 to evaluate our algorithm. *Td* is the intrinsic delay of a unit-size repeater. And we assumed that the source driving resistance *rd* and the sink load capacitance *cl* are referred that of a unit-size repeater. *Cpath* is the longest path of a bus, and *#Term*, *#Sour*, *#Sink*, *#Peri*, and *#Loc* are the number of terminals, sources, sinks, timing periods, and segment locations, respectively. For all the test cases, we assume that $rd = rg /100$, $cl = cg*100$ and $w = 6*$ *Wmin* on 0.18μm and 0.13μm technology.

Table 5.1 The parameters based on 0.18μm and 0.13μm technologies

	$0.18\mu m$	$0.13\mu m$
$W_{min}(\mu m)$	0.18	0.13
$r_g\,(\Omega)$	17100	22100
$c_g(fF)$	0.234	0.135
$R\,(\Omega/\square)$	0.068	0.081
$c_a(fF/\mu m^2)$	0.060	0.046
$c_f(fF/\mu m)$	0.064	0.043
Td(ps)	36.4	26.2

Table 5.2 Data of test cases

Case	Cpath	#Term	#Sour	#Sink	#Peri	#Loc
1	11000μm	3	2	2	3	4
2	15500μm	5	4	4	8	9
3	20550μm	7	5	5	11	12
4	13000μm	9	6	7	11	18
5	16400μm	8	8	8	8	13
6	22000μm	8	6	7	12	16
7	26800μm	10	6	7	15	24

Tables 5.3, 5.4 and 5.5 show the experimental results for FED-based buses with 0.18μm and 0.13μm technologies, respectively. *Tcri* and *Tmax* are two critical delays without and with inserted repeaters, respectively. *Uloc/loc* is the ratio of number of inserted repeater locations and number of available locations. Size is the sum of all the inserted repeater sizes. Saving is the percentage of (*Tmax*-*Tcri*)/*Tmax*. The *Improved* is the improvement of proposed algorithms when compared with [13]. Table 5.3 (a) and (b) show the simulation results of proposed MRIA for the parameters of 0.18μm and 0.13μm, respectively. The columns of *TmaxMRIA*(ns) and *Improved* indicate the maximum delay and the improved ratio of proposed MRIA algorithm, respectively. From these two tables, we can observe that our proposed MRIA can further reduce the maximum delay about 1.8% at least when compared to [13]. Furthermore, since the proposed MRIA algorithm inserts and resizes the repeaters into circuit to reduce the maximum delay, the repeater size should be analyzed to show the effect of our proposed MRIA. The column of *SizesMRIA* indicates the repeater size after the proposed MRIA algorithm. From these two tables, we can observe that the increasing of repeater size is 6.8 on average. This small repeater size increasing implies that the proposed MRIA algorithm can further reduce the maximum delay with the cost of small hardware overhead.

Table 5.3(a) Results of MRIA algorithm based on 0.18μm parameters

Case	Tcri(ns)	Tmax(ns)[13]	Uloc/loc	Sizes	Saving
1	0.827	0.442	4/4	16	46.6%
2	1.83	0.613	8/9	30	66.5%
3	3.375	0.864	12/12	45	74.4%
4	1.677	0.496	15/18	48	70.4%
5	2.07	0.623	13/13	50	69.9%
6	4.147	0.94	15/16	44	77.3%
7	6.948	1.079	21/24	69	84.5%
Average				43.1	69.9%

Case	Tmax(ns)[13]	MRIA(ns)	Sizes	**Improved**
1	0.442	0.433	21	**2.0%**
2	0.613	0.611	35	**0.3%**
3	0.864	0.864	45	**0.0%**
4	0.496	0.496	48	**0.0%**
5	0.623	0.623	51	**0.0%**
6	0.94	0.897	64	**4.6%**
7	1.079	1.021	79	**5.4%**
Average			49.0	**1.8%**

Table 5.3(b) Results of MRIA algorithm based on 0.13μm parameters

Case	Tcri(ns)	Tmax(ns)[13]	Uloc/loc	Sizes	Saving
1	0.779	0.391	4/4	14	49.8%
2	1.753	0.535	8/9	28	69.5%
3	3.263	0.758	12/12	45	76.8%
4	1.585	0.414	15/18	48	73.9%
5	1.976	0.509	13/13	46	74.2%
6	4.018	0.844	15/16	43	79.0%
7	6.786	0.967	21/24	63	85.8%
Average				41.0	72.7%

Case	Tmax(ns)[13]	MIRA(ns)	Sizes	**Improved**
1	0.391	0.371	26	**5.1%**
2	0.535	0.523	34	**2.2%**
3	0.758	0.75	49	**1.1%**
4	0.414	0.414	48	**0.0%**
5	0.509	0.509	46	**0.0%**
6	0.844	0.768	61	**9.0%**
7	0.967	0.885	77	**8.5%**
Average			48.7	**3.7%**

Table 5.4 (a) and (b) show the simulation results of proposed RPAA for the parameters of 0.18μm and 0.13μm, respectively. The columns of *TmaxRPAA(ns)* and Improved indicate the maximum delay and improved ratio of proposed RPAA algorithm, respectively. From these two tables, we can observe that our proposed RPAA can further reduce the maximum delay about 6.2% at least when compared to [13]. Furthermore, the column of *SizesRPAA* indicates the repeater size after the proposed RPAA algorithm. From these two tables, we can observe that the increasing of repeater size is 2.3 on average.

Table 5.4(a) Results of RPAA algorithm based on 0.18μm parameters

Case	Tcri(ns)	Tmax(ns)[13]	Uloc/loc	Sizes	Saving
1	0.827	0.442	4/4	16	46.6%
2	1.83	0.613	8/9	30	66.5%
3	3.375	0.864	12/12	45	74.4%
4	1.677	0.496	15/18	48	70.4%
5	2.07	0.623	13/13	50	69.9%
6	4.147	0.94	15/16	44	77.3%
7	6.948	1.079	21/24	69	84.5%
Average				43.1	69.9%

Case	Tmax(ns)[13]	RPAA(ns)	Sizes	**Improved**
1	0.442	0.407	16	**7.9%**
2	0.613	0.568	34	**7.3%**
3	0.864	0.775	43	**10.3%**
4	0.496	0.486	49	**2.0%**
5	0.623	0.59	51	**5.3%**
6	0.94	0.878	57	**6.6%**
7	1.079	1.037	62	**3.9%**
Average			44.6	**6.2%**

Table 5.4(b) Results of RPAA algorithm based on 0.13μm parameters

Case	Tcri(ns)	Tmax(ns)[13]	Uloc/loc	Sizes	Saving
1	0.779	0.391	4/4	14	49.8%
2	1.753	0.535	8/9	28	69.5%
3	3.263	0.758	12/12	45	76.8%
4	1.585	0.414	15/18	48	73.9%
5	1.976	0.509	13/13	46	74.2%
6	4.018	0.844	15/16	43	79.0%
7	6.786	0.967	21/24	63	85.8%
Average				41.0	72.7%

Case	Tmax(ns)[13]	RPAA(ns)	Sizes	**Improved**
1	0.391	0.347	16	**11.3%**
2	0.535	0.498	35	**6.9%**
3	0.758	0.647	44	**14.6%**
4	0.414	0.402	47	**2.9%**
5	0.509	0.487	46	**4.3%**
6	0.844	0.766	56	**9.2%**
7	0.967	0.922	65	**4.7%**
Average			44.1	**7.7%**

Table 5.5 (a) and (b) show the simulation results of proposed MRIPAA for the parameters of 0.18μm and 0.13μm, respectively. The columns of *TmaxMRIPAA(ns)* and Improved indicate the maximum delay and improved ratio of proposed MRIPAA algorithm, respectively. From these two tables, we can observe that our proposed MRIPAA can further reduce the maximum delay about 7.6% at least when compared to [13]. Furthermore, the column of *SizesMRIPAA* indicates the repeater size after the proposed MRIPAA algorithm. From these two tables, we can observe that the increasing of repeater size is 6.9 on average.

Table 5.5(a) Results of MRIPAA algorithm based on 0.13μm parameters

Case	Tcri(ns)	Tmax(ns)	Uloc/loc	Sizes	Saving
1	0.827	0.442	4/4	16	46.6%
2	1.83	0.613	8/9	30	66.5%
3	3.375	0.864	12/12	45	74.4%
4	1.677	0.496	15/18	48	70.4%
5	2.07	0.623	13/13	50	69.9%
6	4.147	0.94	15/16	44	77.3%
7	6.948	1.079	21/24	69	84.5%
Average				43.1	69.9%

Case	Tmax(ns)[13]	MRIPAA(ns)	Sizes	**Improved**
1	0.442	0.399	29	**9.7%**
2	0.613	0.568	34	**7.3%**
3	0.864	0.775	43	**10.3%**
4	0.496	0.486	49	**2.0%**
5	0.623	0.59	51	**5.3%**
6	0.94	0.826	68	**12.1%**
7	1.079	1.011	62	**6.3%**
Average			48.0	**7.6%**

Table 5.5(b) Results of MRIPAA algorithm based on 0.13μm parameters

Case	Tcri(ns)	Tmax(ns)[13]	Uloc/loc	Sizes	Saving
1	0.779	0.391	4/4	14	49.8%
2	1.753	0.535	8/9	28	69.5%
3	3.263	0.758	12/12	45	76.8%
4	1.585	0.414	15/18	48	73.9%
5	1.976	0.509	13/13	46	74.2%
6	4.018	0.844	15/16	43	79.0%
7	6.786	0.967	21/24	63	85.8%
Average				41.0	72.7%

Case	Tmax(ns)[13]	MRIPAA(ns)	Sizes	**Improved**
1	0.391	0.328	26	**16.1%**
2	0.535	0.489	42	**8.6%**
3	0.758	0.647	44	**14.6%**
4	0.414	0.402	47	**2.9%**
5	0.509	0.487	46	**4.3%**
6	0.844	0.695	66	**17.7%**
7	0.967	0.892	77	**7.8%**
Average			49.7	**10.3%**

From the analysis of previous section, we can conclude that the proposed RPAA algorithm is performed better than MRIA algorithm, MRIPAA algorithm is performed better than RPAA algorithm. Previous analyses show the maximum delay improvement ratio of proposed MRIA, RPAA and MRIPAA is 1.8%, 6.2% and 7.6% at least, respectively. For the increase of hardware overhead, the proposed MRIA, RPAA and MRIPAA contribute 6.8, 3.1 and 6.9 additional repeater size on average, respectively.

Chapter 6 Conclusion and Future Works

In this thesis, in order to include the influence of inductance effect at high work frequency, we try to use the FED model to calculate interconnection delay. In addition, the methods of bidirectional signal repeater insertion and the repeater size adjusting are adopted in this thesis to reduce the interconnection delay on the bus. At the same time, we proposed an algorithm which inserts the signal repeater by Greedy method. The proposed algorithm selects a critical path to insert the signal repeater and adjusts its size such that the interconnection delay can be minimized on the critical path. Afterwards, the proposed algorithm is iteratively executed until the minimum interconnection delay becomes stable. In the experimental result section, the process parameters of 0.13 and 0.18μm are applied for our design to generate simulation results. The simulation results show that our proposed algorithm can at least reduce the interconnection delay by 1.8% and 3.7% in average when compared to literature [13]. We can conclude that the proposed RPAA algorithm is performed better than MRIA algorithm, MRIPAA algorithm is performed better than RPAA algorithm.

Since the demands of SoC system are increased rapidly, the delay of long wire dominates the overall SoC system. Therefore, significantly reduces the interconnection line delay can extremely improve the system performance. Extended works can associate the alternate repeater insertion and sizing with reducing the effects of propagation delay, crosstalk, and power consumption simultaneously.

References

[1] W. C. Elmore, “The transient response of damped linear networks,” Journal of Applied Physics, vol. 19, pp. 55-63, January 1948.

[2] A. I. Abou-Seido, B. Nowak, and C. Chu, “Fitted Elmore Delay: A Simple and Accurate Interconnect Delay Model,” IEEE Transactions on VLSI Systems, vol. 12, Issue 7, pp. 691-696, July 2004.

[3] D.-Y. Kao, C.-C. Tsai, C.-K. Cheng, and T.-T. Lin, “New design and implementation for signal repeaters,” The Sixth VLSI Design/CAD Workshop, pp. 173-176, 1995.

[4] C.-C. Tsai, Y.-J. Jiang, and S.-H. Kou, “Design and implementation of bus repeater,” The 13th Technological and Vocational Education Conference, pp. 297-306, May 1998.

[5] J. L. Wyatt, Circuit Analysis, Simulation and Design. North-Holland, The Netherlands：Elsevier Science, 1987.

[6] J. Lillis, C.-K. Cheng, and T.-Y. Lin, “Simultaneous routing and buffer insertion for high performance interconnect,” in Proceedings of Sixth Great Lakes Symposium on VLSI, pp. 148-153, 1996.

[7] J. Cong, and D. Z. Pan, ”Interconnect delay estimation models for synthesis and design planning,” in Proceedings of ASP-DAC, pp. 97-100, 1999.

[8] C.-C. Tsai, D.-Y. Kao, and C.-K. Cheng, “Performance driven bus buffer insertion,” IEEE Transactions on CAD of Integrated Circuits and Systems, pp. 429-437, 1996.

[9] T. Okamoto and J. Cong, “Buffered Steiner Tree Construction with Wire Sizing for Interconnect Layout Optimization,” in proceedings of IEEE/ACM International Conference on Computer-Aided Design, pp. 44-49, 1996.

[10] P. Sarkar and C. K. Koh, “Routability-driven repeater block planning for interconnect-centric floorplanning,” IEEE Transactions on CAD of Integrated Circuits and Systems, pp. 660-671, May 2001.

[11] J. Lillis, and C.-K. Cheng, “Timing Optimization for Multisource Nets: Characterization and Optimal Repeater Insertion,” IEEE Transactions on Computer-Aided Design of Integrated Circuit and Systems, pp. 322-331, March 1999.

[12] V. Adler and E. G. Friedman, “Repeater Design to Reduce Delay and Power in Resistive Interconnect,” IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, vol. 45, pp. 607-616, May 1998.

[13] C.-C. Tsai, J.-O. Wu and T.-Y. Lee, “Propagation Delay Minimization for RLC-Based Multi-source Multi-sink Bus with Repeater Insertion,” in proceedings of IEEE Asia-Pacific Conference on Circuits and Systems, pp. 1287-1290, December 4-7 2006.

[14] T.-C. Chen, S.-R. Pan, and Y.-W. Chang, “Performance optimization by wire and sizing under the transmission line model,” Proc. of International Conference on Computer Design, pp. 23-26, September 2001.

[15] T.-C. Chen, S.-R. Pan, and Y.-W. Chang, “Timing modeling and optimization under the transmission line model,” IEEE Trans. On VLSI systems, vol. 12, no. 1, January 2004.

[16] S.-L. Wang and Y.-W. Chang, “Accurate delay formulae for buffer RLC trees,” The 14th VLSI Design /CAD symposium, paper A2-2, August 2003.

[17] Y. I. Ismail and E. G. Friedman, "Effects of inductance on the propagation delay and repeater insertion in VLSI circuits," IEEE Transactions on Very Large Scale Integration Systems, vol.8, no.2, pp.195-206, April 2000.

[18] Y. I. Ismail, E. G. Friedman, and J. L. Neves, "Repeater insertion in tree structured inductive interconnect," IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, , vol.48, no.5, pp.471-481, May 2001.

[19] R. Venkatesan, J. A. Davis, and J. D. Meindl, "Compact distributed RLC interconnect models - part IV: unified models for time delay, crosstalk, and repeater insertion," IEEE Transactions on Electron Devices, vol.50, no.4, pp. 1094-1102, April 2003.

[20] C.-A. Lin and C.-H. Wu, “Second-order approximations for RLC trees,” IEEE Trans. CAD of Integrated Circuits and Systems, Vol. 23, No. 7, pp. 1124-1128, July 2004.

[21] C.-C. Tsai, "Timing Driven Based on Signal Repeater Insertion,” Journal of National Taipei University of Technology, Vol. 31-2, pp. 111-133, September 1998.