國立交通大學

電子工程學系 電子研究所碩士班

碩士論文

考慮可變省電工作模式之繞線研究

On Power-State-Aware
Routing and Buffer Insertion

研究生： 吳敏華

指導教授： 江蕙如 博士

中華民國 九十六年七月

考慮可變省電工作模式之繞線研究
On Power-State-Aware Routing and Buffer Insertion

研究生：吳敏華　　Student：Ming-Hua Wu

指導教授：江蕙如　　Advisor：Dr. Iris Hui-Ru Jiang

國立交通大學
電子工程學系 電子研究所碩士班
碩士論文

A Thesis
Submitted to Department of Electronics Engineering & Institute of Electronics
College of Electrical and Computer Engineering
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in
Electronics Engineering

July 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年七月

考慮可變省電工作模式之繞線研究

研究生：吳敏華　　　　指導教授：江蕙如 博士

國立交通大學

電子工程學系　電子研究所碩士班

摘要

進入奈米世代，導線延遲和低功耗皆是重要課題。繞線時安插緩衝器可以改善導線延遲；而可調整的省電工作模式和多重供電電壓皆是省電效果極佳的技術。然而，如果沒有將省電工作模式間的切換納入考量，不當安插緩衝器可能會喪失信號的完整性。本篇論文是文獻中首篇考慮省電工作模式的繞線研究，此議題是目前工業界實際面臨且迫切需要的。藉由階層化的動態規劃演算法，同時達成最佳化功率消耗的目地、滿足信號傳遞時間的要求、並且維持信號的完整性。實驗數據說明本篇論文所提出之演算法不僅效果優異，且相較於前人，也大幅縮短所需的執行時間。

ON POWER-STATE-AWARE ROUTING AND BUFFER INSERTION

Student: Ming-Hua Wu   Advisor: Dr. Iris Hui-Ru Jiang

Institute of Electronics
Department of Electronics Engineering
National Chiao Tung University

Abstract

Interconnect delay and low power are two of the main issues in nanotechnology. Buffer insertion during routing can reduce interconnect delay; power state management and multiple supply voltage can lower power consumption. However, buffering without considering power states may cause the signal integrity problem. In this thesis, we first consider power states into routing and buffer insertion. Based on a hierarchical approach combined with dynamic programming, we can simultaneously minimize power, satisfy timing constraints and maintain signal integrity. Compared with previous works, the experimental results show this method is promising.

Acknowledgements

I would like to express my sincerest appreciation to my advisor, Prof. Iris Hui-Ru Jiang, for her advice and guidance in research and life during my master program at NCTU. Her enthusiasm and inspiration motivated me. I do enjoy our interaction during discussion and cooperation. Without her efforts, I cannot complete the thesis smoothly and delightedly.

I am grateful to the members of my thesis committee, Prof. Jing-Yang Jou, Prof. Hung-Ming Chen and Prof. Mango Chia-Tso Chao for their invaluable advices and suggestions.

Special thanks go to Mr. Bruce Tseng for his kind assistance and members in EDA family of Institute of Electronics for their counsel and help. I would like to appreciate my classmates, Chia-I Chen, Chia-Wei Cheng, Ai-Hsin Liu and Li-Ya Wang for their friendship and encouragement during my life at NCTU. I am also thankful to my friends, Yvette Lai, Rita Min and Rio Huang for their friendship and company.

Last but not least, my deepest gratitude goes to my parents, Shuei-Fu Wu and Jin-Ji Lin, my sister Pei-Shan Wu, and my grandmother Bing Wu for their unflagging love, constant support and encouragement throughout my life.

Ming-Hua Wu

National Chiao Tung University

July 2007

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

In nanotechnology, the interconnect delay and the power consumption are two issues of most significant importance. Many interconnect delay reduction techniques have been proposed, such as gate sizing [1], wire sizing [9, 10, 14], and buffer insertion [4, 7, 8, 9, 10, 13, 14, 15, 19]; among them, buffer insertion is an effective way [2]. Traditional methodology applies buffer insertion in the post-layout stage [7, 8, 9], but improves delay little due to the limitation of routing resources. Therefore, buffer insertion during routing has become the mainstream in the past decade.

On the other hand, low power issues have been extensively studied in literature: multiple supply voltage (MSV) [1, 12, 13, 15, 16], power state management [3, 11], etc. Table 1.1 lists the impacts of popular power reduction techniques on power, timing, area, and methodology [5]. It can be seen that both multiple supply voltage and power shut-off techniques have large benefits on power. In a multiple supply voltage design, each cell can be driven by one from several voltage levels. This method can reduce power most directly, because power consumption is proportional to the square of the supply voltage. However, a signal going through differently leveled cells induces leakage currents and raises the integrity issue. To overcome these problems, a level converter should be inserted if a lower voltage cell drives a higher one. Moreover, arbitrarily assigning voltage levels may make power/ground planning difficult. Therefore, we can cluster cells with the same supply source into a group, thus partitioning a design into voltage islands. As shown in Figure 1.1(a), [17] clustered 317k cells into 17 voltage islands in Figure 1.1(b). In addition, circuits are not always running at the high performance mode, e.g., several voltage islands may idle sometimes, but still consume power. With power state management, we can turn off these idle islands. However, in this case, we shall carefully do routing and buffer insertion. Figure 1.2 shows an MSV design with two voltage islands turned off under a given power state. The left-hand sided path costs a low voltage buffer and a level converter. The right-hand sided one costs only one high voltage buffer, but it conducts an incorrect signal because of no power supply. Therefore, it is desirable to find a buffering algorithm with power state consideration. Moreover, if we consider multiple power states for a net, the routing results should be feasible for all power states. As demonstrated in Figure 1.3, given a source s and two sinks $t1$ and $t2$, the net should work at two power states $P1$ and $P2$. Voltage islands $VI5$ and $VI6$ are turned off at $P1$, and $VI1$ and $VI5$ are turned off at $P2$. To maintain signal integrity for both $P1$ and $P2$, the net is routed as Figure 1.3(a) and 1.3(b). However, this long wire may violate timing constraints. Then, a buffer is inserted in $VI1$; this buffer solves timing violation at $P1$ as Figure 1.3(c) but fails at $P2$ as Figure 1.3(d) because of no power supply. Hence, if we can create a new voltage island $VI7$ as Figure 1.3(e) and 1.3(f), then the net will maintain signal integrity and satisfy timing constraints for both $P1$ and $P2$.

Table 1.1: The comparison between several power reduction techniques. Multiple supply voltage and power shut-off has large and huge reduction on power consumption, respectively. [5]

Power-reduction technique	Power Benefit	Timing Penalty	Area Penalty	Methodology Impact			
				Architecture	Design	Verification	Implementation
Multi-Vt optimization	Medium	Little	Little	Low	Low	None	Low
Clock gating	Medium	Little	Little	Low	Low	None	Low
Multi-supply voltage	Large	Some	Little	High	Medium	Low	Medium
Power shut-off	HUGE	Some	Some	High	High	High	High
Dynamic and adaptive voltage frequency scaling	Large	Some	Some	High	High	High	High
Substrate biasing	Large	Some	Some	Medium	None	None	High

1.2 Previous Work

Several previous works focused on simultaneous routing and buffer insertion for single supply voltage designs [4, 10, 14, 19], most of them used dynamic programming. [13] and [15] extended buffered tree construction to dual supply voltages. We briefly introduce these works as follows.

1.2.1 Routing Algorithms for Single Supply Voltage Designs

Both [10] and [19] solved simultaneous routing and buffer insertion for two-pin nets, and [10] also dealt with wire sizing at the same time. The authors in [19] proposed an algorithm extended from Dijkstra's shortest path algorithm working on the grid graph. Each node was labeled with a set of capacitance-delay pairs to represent a partial solution. A partial solution with minimum delay was updated to

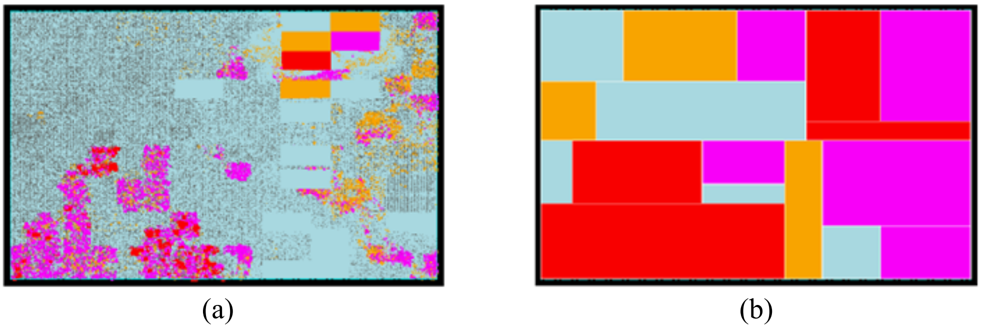


Figure 1.1: Clustering 317k cells in (a) with the same supply source into group and partitioning this design into 17 voltage islands in (b). [17]

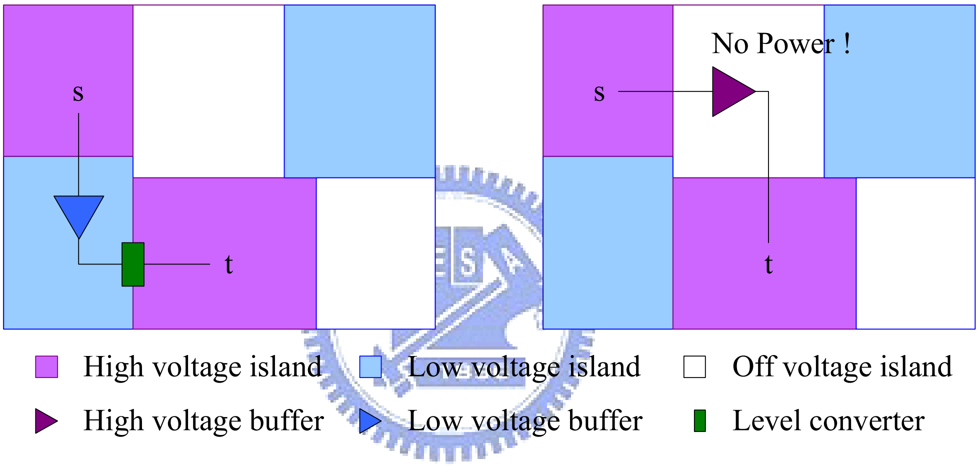


Figure 1.2: Under a given power state, the left-hand sided path is feasible, but the right-hand sided one is not because of no power supply.

its neighbor nodes until reaching the source of the net. In addition, buffer locations were restricted because of macro blocks, but wiring was fine. On the other hand, the authors in [10] used graph-theoretic shortest path formulation to solve the problem. [10] constructed a buffer planning graph to represent possible buffer choices and a looked-up table for wire sizing, and then updated partial solutions from the sink until reaching the source. However, both [10] and [19] handled two-pin nets only, so their applications were limited. Therefore, [4] constructed a multi-terminal buffered tree under fix buffer locations by dynamic programming in a bottom-up fashion. The algorithm constructed subtrees from each sink first, and then recursively merged subtrees and pruned inferior partial solutions until completing a tree.

1.2.2 Routing Algorithms for Dual Supply Voltage Designs

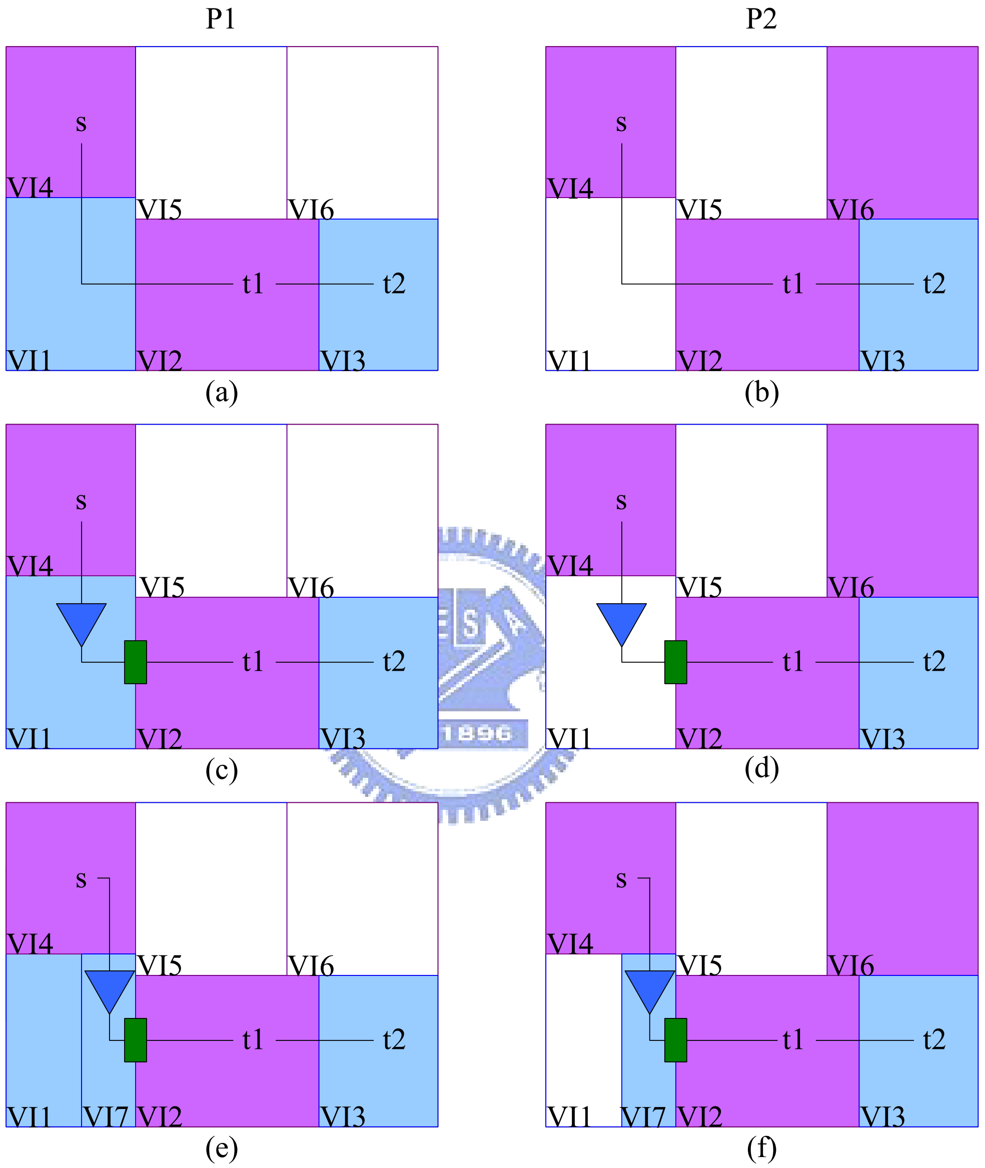
[13] and [15] used dual voltage buffers to construct a buffered tree. [13] recursively visited children nodes from the source first, and then enumerated all possible solutions in a bottom-up fashion. However, [13] assumed the source was driven by high voltage, and prohibited low voltage buffers to drive high ones. Thus, level converters were not used inside the interconnect tree, and only few level converters were needed right before high voltage sinks. To remove this impractical restriction, [15] proposed an algorithm to a simplified dual voltage design with only one low voltage island. Because the low voltage island may be turned off sometimes, no buffer is allowed to be placed inside this island, if the source and any sink are outside it. As shown in Figure 1.4, the source s and three sinks $t1$, $t2$ and $t5$ are outside the island (indicated by the dotted block); therefore, no buffer is inserted inside it. Even so, when this island is turned off, signal sent from sinks $t3$ and $t4$ may float making sink $t2$ possibly obtain a signal ruined with noise. Because no buffers can be inserted inside this low voltage island, sinks inside it may violate timing constraints and incur long transition and large loading. Moreover, considering only one low voltage island is somewhat over-simplified. Therefore, considering power states of voltage islands, we can maintain signal integrity better, insert buffers more flexibly and save more power.

1.3 Our Contribution

In this thesis, we first consider power states into buffered routing tree construction for dual supply voltage designs. The goal is to minimize total power consumption under timing constraints. We adopt a hierarchical approach combined with dynamic programming. First of all, we construct a local buffered tree within each voltage island. Secondly, we connect these local trees into a global one with power state consideration. If no feasible solution meets timing constraints, we will create new voltage islands around the existing ones. To our best knowledge, no existing works considered power states during routing and buffer insertion so far. Power state diagrams are available in system level [3, 11], and voltage islands are planned at floorplanning. Therefore, we can combine them into a power state table that indicates each state with active islands. Our method can easily be extended to multiple supply voltage designs and sophisticated delay models. The experimental results show that our algorithm can not only minimize power but also maintain signal integrity effectively and efficiently.

1.4 Organization of the Thesis

The remainder of this thesis is organized as follows. Chapter 2 presents some preliminaries and the problem formulation, Chapter 3 describes our algorithm, Chapter 4 shows our experimental results, and finally Chapter 5 gives conclusions.



(a) (b) (c) (d) (e) (f)

Figure 1.3: The net with the source s and two sinks $t1$ and $t2$ should work at power states $P1$ and $P2$. The routed path in (a) and (b) can maintain signal integrity but may violate timing constraints. In (c), the buffered path fixes the timing violation for $P1$, but makes signals incorrect for $P2$ as in (d). Creating a new voltage island $VI7$, the buffered path has correct signals and timing for $P1$ in (e) and $P2$ in (f).

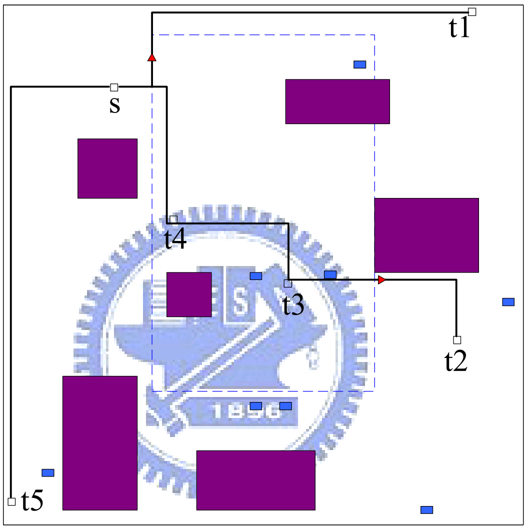


Figure 1.4: The dotted block is the low voltage island and the small rectangles are available buffer locations. The source s and sinks $t1$, $t2$, $t5$ are outside the island, so buffers cannot be placed within the island [15]. Even so, signal sent from sinks $t3$ and $t4$ may float making sink $t2$ possibly obtain a signal ruined with noise while the island is turned off.

Chapter 2

Preliminaries and Problem Formulation

In this Chapter, we introduce delay and power models used in this thesis, and then give the problem formulation.

2.1 Delay and Power Models

Figure 2.1 shows the circuit models of a wire, a buffer, and a level converter. Two types of buffers–low voltage one and high voltage one, and one type of level converter are used throughout this thesis. Table 2.1 lists the parameter settings in our experiments. Using the Elmore delay model [6], the delay of a wire D_w and that of a buffer D_b are computed as follows.

$$\begin{aligned} D_w(L) &= r_w \cdot L \cdot (\frac{1}{2} \cdot c_w \cdot L + C_l), \\ D_b &= D_i + R_b \cdot C_l, \end{aligned}$$

where c_w is unit length capacitance, r_w is unit length resistance, L is wire length, C_l is the downstream capacitance, D_i is the intrinsic delay of a buffer, C_b is the input capacitance of a buffer, R_b is the output resistance of a buffer. The delay model can easily be extended to consider the inductance effect. A level converter is modeled in a similar way. The wire power consumption P_w is measured by energy per switch,

$$P_w(L) = \frac{1}{2} \cdot c_w \cdot L \cdot {V_{dd}}^2,$$

where V_{dd} is the supply voltage.

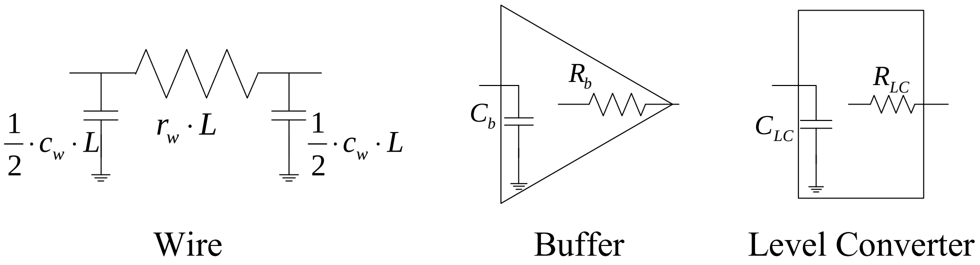


Figure 2.1: The circuit models of a wire, a buffer and a level converter.

Table 2.1: Parameters used in this thesis.

Model		**Value**
Wire		$c_w = 0.15\ fF/\mu m$, $r_w = 1.0\ \Omega/\mu m$
Buffer	High	$C_b = 3.4\ fF$, $R_b = 1.0\ k\Omega$, $D_i = 36.4\ ps$
	Low	$C_b = 3.4\ fF$, $R_b = 1.2\ k\Omega$, $D_i = 40.0\ ps$
Level converter		$C_{LC} = 3.4\ fF/\mu m$, $R_{LC} = 1.2\ k\Omega$, $D_{LC} = 100.0\ ps$

We use a net with one source s and two sinks $t1$ and $t2$ as an example to demonstrate how to compute delay as in Figure 2.2. Assume the source has 1 $k\Omega$ driving resistance and both sinks have 10 fF loading capacitances. Two sinks are merged at node A. The distance between A and $t1$ is 50 μm, and that between A and $t2$ is 75 μm. In addition, a high voltage buffer B is inserted between s and A, and the distance between s and B is 100 μm, and that between B and A is 20 μm. The delay of the net can be partitioned into three parts–s to B, B to A and A to sinks. The total delay is the sum of these three parts and computed as follows.

The delay of A to $t1$:

$$\begin{aligned} D(A,t1) &= r_w \cdot L(A,t1) \cdot (\frac{1}{2} \cdot c_w \cdot L(A,t1) + C_l) \\ &= 1.0 \cdot 50 \cdot (\frac{1}{2} \cdot 0.15 \cdot 50 + 10) \\ &= 0.6875\ ps. \end{aligned}$$

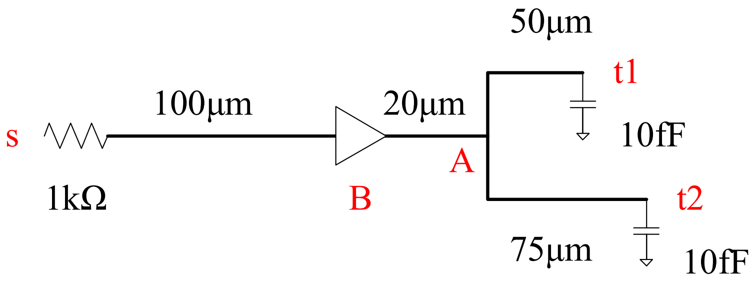


Figure 2.2: An net with two sinks $t1$ and $t2$.

The delay of A to $t2$:

$$\begin{aligned} D(A,t2) &= r_w \cdot L(A,t2) \cdot (\frac{1}{2} \cdot c_w \cdot L(A,t2) + C_l) \\ &= 1.0 \cdot 75 \cdot (\frac{1}{2} \cdot 0.15 \cdot 75 + 10) \\ &= 1.172\ ps. \end{aligned}$$

The delay of A to sinks:

$$\begin{aligned} D(A,t) &= max\{D(A,t1), D(A,t2)\} \\ &= 1.172\ ps. \end{aligned}$$

The downstream loading at A:

$$\begin{aligned} C(A,t) &= C(A,t1) + C(A,t2) \\ &= (c_w \cdot L(A,t1) + C_l) + (c_w \cdot L(A,t2) + C_l) \\ &= (0.15 \cdot 50 + 10) + (0.15 \cdot 75 + 10) \\ &= 38.75\ fF. \end{aligned}$$

The delay of B to A due to wire:

$$\begin{aligned} D_w(B,A) &= r_w \cdot L(B,A) \cdot (\frac{1}{2} \cdot c_w \cdot L(B,A) + C(A,t)) \\ &= 1.0 \cdot 20 \cdot (\frac{1}{2} \cdot 0.15 \cdot 20 + 38.75) \\ &= 0.805\ ps. \end{aligned}$$

The delay of B to A due to buffer:

$$\begin{aligned} D_b(B,A) &= D_i + R_b \cdot (c_w \cdot L(B,A) + C(A,t)) \\ &= 36.4 + 1.0k \cdot (0.15 \cdot 20 + 38.75) \\ &= 78.15\ ps. \end{aligned}$$

The delay of B to sinks:

$$\begin{aligned} D(B,t) &= D(B,A) + D(A,t) \\ &= D_w(B,A) + D_b(B,A) + D(A,t) \\ &= 0.805 + 78.15 + 1.172 \\ &= 80.127\ ps. \end{aligned}$$

The delay of s to B:

$$\begin{aligned} D(s,B) &= R_s \cdot (c_w \cdot L(s,B) + C_b) \\ &\quad + r_w \cdot L(s,B) \cdot (\frac{1}{2} \cdot c_w \cdot L(s,B) + C_b) \\ &= 1k \cdot (0.15 \cdot 100 + 3.4) \\ &\quad + 1.0 \cdot 100 \cdot (\frac{1}{2} \cdot 0.15 \cdot 100 + 3.4) \\ &= 19.49\ ps. \end{aligned}$$

The delay of s to sinks:

$$\begin{aligned} D(s,t) &= D(s,B) + D(B,t) \\ &= 19.49 + 80.127 \\ &= 99.617\ ps. \end{aligned}$$

2.2 Problem Formulation

We formulate the power-state-aware routing and buffer insertion problem as follows.

The Power-State-Aware Routing and Buffer Insertion Problem: Given a multi-terminal net, blockages, voltage island planning, power states, and buffer and level converter libraries, construct a buffered routing tree with minimum power and modify voltage island planning and power states (if new voltage islands are created), such that timing constraints are satisfied.

In addition, low voltage buffers can only be inserted in low voltage islands; high voltage buffers and level converters can only be inserted in high voltage islands. Moreover, if no feasible solutions exist, new voltage islands are introduced and thus power states and voltage island planning should be modified accordingly.

Chapter 3

Algorithm

To solve the power-state-aware routing and buffer insertion problem, we propose a hierarchical approach combined with dynamic programming. Because sinks may be spread over several voltage islands, we handle voltage islands one by one. First of all, sinks within each voltage island are connected to a local tree. Secondly, local trees are connected to a global one with power state consideration. If no feasible solutions exist, new voltage islands will be created at global tree construction. Figure 3.1 gives the overview of our flow.

3.1 Local Tree Construction

The procedure of local tree construction is listed in Figure 3.2. First of all, the pseudo sink is found in line 1. The pseudo sink is an artificial sink representing all sinks within a voltage island, and also the root of a local tree. Secondly, grid lines are constructed in line 2. Thirdly, grid nodes are initialized in line 3. Finally, in lines 5–15, we iteratively propagate solutions from each sink and prune redundant solutions if necessary until finding a feasible solution. We use a priority queue Q_s and a working queue Q_w in these while loops. Q_w maintains the ordering of sinks according to the distances between sinks and the pseudo sink. Q_w records grid nodes which should propagate solutions to neighbors. We detail the procedure as follows.

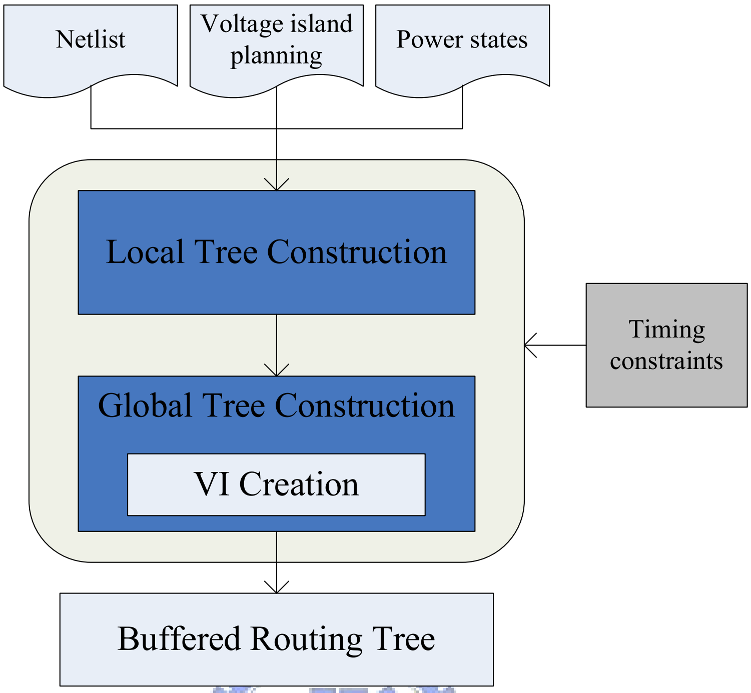


Figure 3.1: The overview of our flow.

3.1.1 Finding the Pseudo Sink

We connect sinks within a voltage island at this local level. However, for sinks outside the island of the source, we do not know where is the source or the upstream net. Therefore, the pseudo sink is to guide the direction of local tree construction. We expect that the upstream of the local trees can approach to the source of the net to reduce delays. Therefore, we consider several nodes which are relatively close to the source within the current island to be the pseudo sink.

According to the geometry relation between the voltage island of the source VI_{src} and the current island, a set of pseudo sink candidates are selected. If the current island is overlapped with the horizontal or vertical centerlines of VI_{src}, grid

```
Algorithm: Local Tree Construction
Input:   Voltage island $VI_i$
         Timing constraints
Output: A local buffered routing tree

Find the pseudo sink $T_i$ of $VI_i$
Construct grid lines within $VI_i$
Initialize grid nodes
Push sinks within $VI_i$ into priority queue $Q_s$
while $Q_s$ is not empty do
    Select sink $t_j$ with max. distance to $T_i$ from $Q_s$
    Push sink $t_j$ into queue $Q_w$
    while $Q_w$ is not empty and
            no feasible solution is found do
        Select node $w$ from $Q_w$
        Propagate solution from $w$ to its neighbor $u$
        if a feasible solution is found then
            Update solution
        else
            Prune redundant solutions on $u$
        Push neighbor $u$ into $Q_w$
```

Figure 3.2: The procedure of local tree construction.

nodes on the side closed to VI_{src} are selected (indicated by red lines in Figure 3.3). Otherwise, candidates are selected as shown in Figure 3.4. Considering obstacle penalties, the pseudo sink is the candidate with the shortest distance to the source. The obstacle penalty is estimated by the smaller overhead on these two L-shaped paths in the worst case as shown in Figure 3.5 [18]. In Figure 3.5(a), the obstacle penalty of the left-up path between x and y is zero, and that of the right-bottom one is $H1$. Thus, the obstacle penalty is zero. Moreover, in Figure 3.5(b), the obstacle penalty of left-up path is $W1$, and that of right-bottom one is $H2$. Thus, the obstacle penalty is $\min\{W1, H2\}$. The total distance between x and y is the Manhattan distance plus the obstacle penalty.

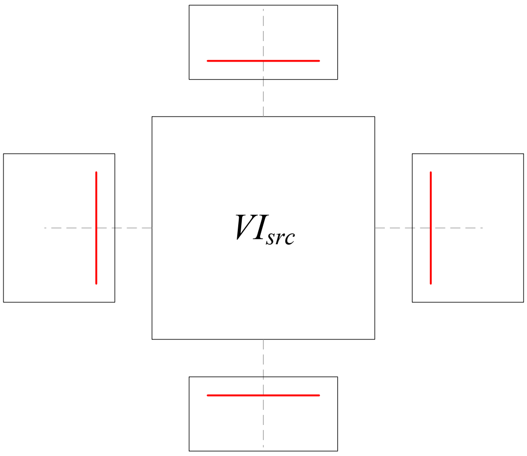


Figure 3.3: Dotted gray lines are centerlines of the source voltage island VI_{src}. Red lines indicate the set of pseudo sink candidates if the current island is overlapped with centerlines.

Table 3.1: Parameters used in a solution.

Parameter	**Description**
R	Driving resistance
C	Loading capacitance
rat	Required arrival time
POW	Power
HW	Hardware cost
BV	Voltage level
PP	Propagated path

3.1.2 Grid Line Construction

Horizontal and vertical grid lines are constructed not only at sinks and the pseudo sink but also around blockages. As shown in Figure 3.6, T is the pseudo sink, $t1$, $t2$ and $t3$ are three sinks, and blue lines are grid lines of given voltage island. These grid lines are bounded by the voltage island for local tree construction to reduce the time complexity.

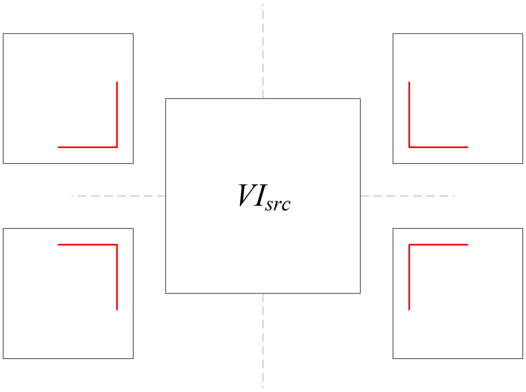


Figure 3.4: Dotted gray lines are centerlines of the source voltage island VI_{src}. Red lines indicate the set of pseudo sink candidates if the current island is not overlapped with centerlines.

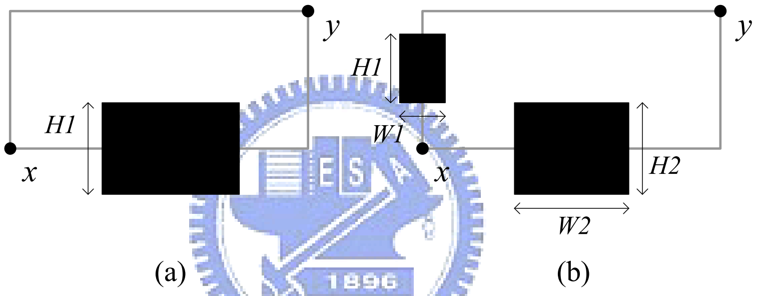


Figure 3.5: The obstacle penalty between x and y in (a) is zero, and that in (b) is $\min\{W1, H2\}$.

3.1.3 Grid Node Initialization

A seven-tuple $(R, C, rat, POW, HW, BV, PP)$ is used to represent a solution. Parameters are listed in Table 3.1. Each grid node is initialized according to the type of the node.

- For being a sink, *node i* is initialized with a loading capacitance: $Solution(0, C_l, rat_i, POW_{C_l}, 0, BV_i, \{node\ i\})$.
- Otherwise, *node i* is initialized with a buffer: $Solution(R_b, C_b, \infty, POW_b, HW_b, BV_i, \{node\ i\})$,

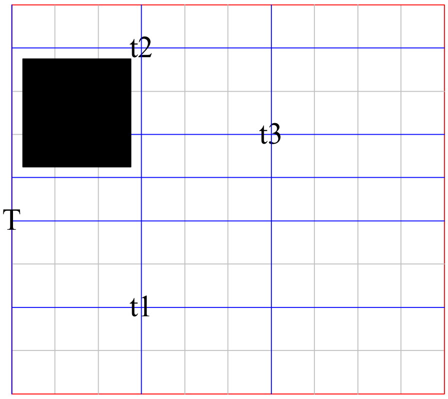


Figure 3.6: Blue lines are grid lines of the pseudo sink T, three sinks $t1$–$t3$ and a blockage (black block). All intersections on grid lines are grid nodes.

and without buffer:

$Solution(0, 0, \infty, 0, 0, BV_i, \{node\ i\})$.

Each grid node is allowed to insert a buffer. For the case with restricted buffer locations, a grid node on an infeasible buffer location is initialized without buffer only. In addition, if a pseudo sink is located at a high voltage island, it is also initialized with a level converter, in case it would be driven by low voltage cells.

3.1.4 Solution Propagation

For local tree construction, solutions are propagated from sinks to the pseudo sink. Solutions of a grid node w are propagated to its neighbors until reaching the pseudo sink or the partially routed tree. For a neighbor grid node u, the current solutions of u and those of w are combined to a new one for u. If a solution is propagated to a routed grid node and the required arrival time is met, the result should be updated to the root, i.e., the pseudo sink. Otherwise, solutions are kept propagating to other grid nodes for forming a feasible solution. We detail how to generate a new solution as follows.

For propagating to an un-routed node u:

- If node u is inserted a buffer:

$$\begin{aligned} rat_{new} &= rat_w - D_i - R_u \cdot (c_w \cdot L + C_w) \\ &\quad -r_w \cdot L \cdot (\frac{1}{2} \cdot c_w \cdot L + C_w). \end{aligned}$$

- If node u is without buffer:

$$rat_{new} = rat_w - r_w \cdot L \cdot (\frac{1}{2} \cdot c_w \cdot L + C_w).$$

For propagating to a routed node u:

- If node u is inserted a buffer:

$$\begin{aligned} rat_1 &= rat_u - R_u \cdot (c_w \cdot L + C_w), \\ rat_2 &= rat_w - D_i - R_u \cdot (C_x + c_w \cdot L + C_w) \\ &\quad -r_w \cdot L \cdot (\frac{1}{2} \cdot c_w \cdot L + C_w), \\ rat_{new} &= \min\{rat_1, rat_2\}, \end{aligned}$$

where C_x is loading from other branches.

- If node u is without buffer:

$$\begin{aligned} rat_1 &= rat_u, \\ rat_2 &= rat_w - r_w \cdot L \cdot (\frac{1}{2} \cdot c_w \cdot L + C_w), \\ rat_{new} &= \min\{rat_1, rat_2\}. \end{aligned}$$

3.1.5 Redundancy Pruning

During solution propagation, we only store some prior solutions to save memory space. Therefore, if solution A has smaller required arrival time, and larger power and capacitance than solution B, then A is pruned. However, if solution A has smaller required arrival time, but smaller power and/or capacitance than solution B, then A is kept. In addition, in a high voltage island, a solution with level converter at the pseudo sink is kept for maintaining signal integrity in the global buffered tree.

We summary this subsection with an example. Figure 3.7, demonstrates how to propagate solutions for the case in Figure 3.6. First of all, we select sink $t3$ which has the maximum distance to the pseudo sink T. Then, a solution with capacitance loading of $t3$ is propagated to its neighbor grid nodes as shown in Figure 3.7(a), and then solutions of these neighbors are propagated to their neighbors in Figure 3.7(b). Gray circles indicate the progress of propagations. Figure 3.7(c) shows the result of two more propagation steps after Figure 3.7(b). The propagation is repeated until a feasible solution from $t3$ to T is found, then solutions are updated as in Figure 3.7(d). Secondly, sink $t2$ is selected, as demonstrated in Figure 3.8(a), solutions are propagated in the similar manner with sink $t3$ until the partially routed tree indicated by red circle is reached. If the required arrival time of the pseudo sink is satisfied, solutions are updated to T. Figure 3.8(b) is the resulting tree of connecting $t2$ and $t3$.

3.2 Global Tree Construction

The procedure of global tree construction is similar to that of local one, for lines 4–7 (see Figure 3.9). Global tree construction may need to create new

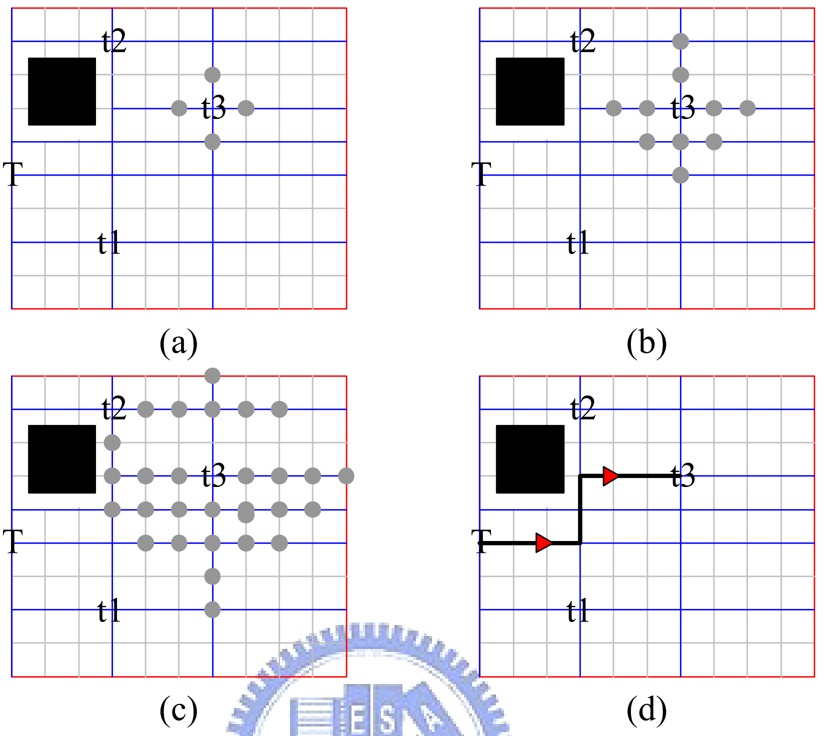


Figure 3.7: Solution propagation from sink $t3$ to the pseudo sink T. (a) and (b) are the results of the first and the second propagation steps. (c) is the results after four steps. (d) is a feasible solution of connecting $t3$ to T.

voltage islands in lines 8–10. During global tree construction, we have to consider the power state table for signal integrity. Because voltage islands may be turned off, we cannot place any buffer within voltage islands of different power state groups excepting those of the always-on group, but pure wiring is fine. In addition, new voltage islands and level converters should properly be added. Isolation cells are not modeled here, but it can be easily extended.

3.2.1 Finding Power State Groups

Voltage islands are clustered into several groups according to their power states. As shown in Figure 3.10, $VI2$ and $VI4$ are active at all power states, viewed as an always-on group $G0$. $G1$ contains $VI5$ and $VI6$ which are only active at

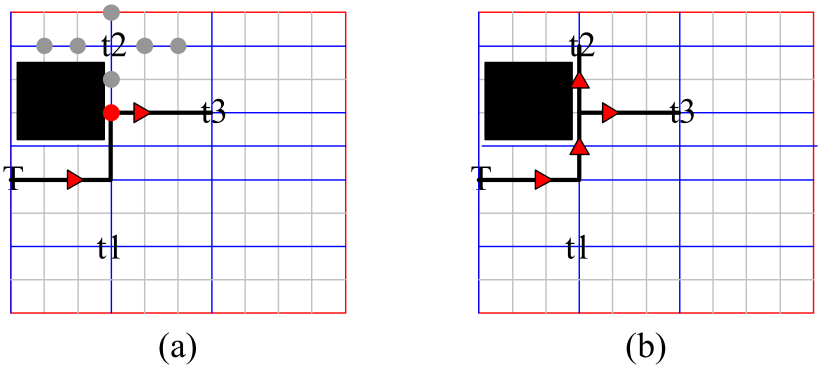


Figure 3.8: Solution propagation from sink $t2$ to the partially routed tree. The red circle in (a) indicates the propagation reaching the partially routed tree. (b) is the resulting buffered tree for $t2$ and $t3$.

$P1$; $G2$ contains $VI1$ which is active at $P2$ and $P3$; $G3$ contains $VI3$ which is only active at $P2$. Please note that we only gather voltage islands with the same active behavior. Therefore, the complexity of global tree construction is bounded by the number of voltage islands, not exponential with that of the power states. In other words, the maximum number of power state groups is the number of voltage islands as demonstrated in Figure 3.11. In this case, each pair of voltage islands have different active behavior.

We detail how to generate power state groups as follows. We encode each power state to a distinct binary digit. If an island is turned on at some power states, we assign one's to the corresponding digits, and zero's to others. Hence, the active behavior of each island can be encoded by a binary sequence. By converting the binary sequence to a decimal number, we have the signature of an island. Because of uniqueness, islands with the same signature possess the same active behavior. Therefore, they should be grouped together. For example, in Figure 3.10, we map $P1$ to 2^0, $P2$ to 2^1, and $P3$ to 2^2. $VI1$ is turned on at $P2$ and $P3$, and its signature is $2^1 + 2^2 = 6$. $VI2$ is turned on at all power states, and its signature is $2^0 + 2^1 + 2^2 = 7$. Finally, we obtain the signature of $VI3$, $VI4$, $VI5$ and $VI6$ is 4, 7, 1 and

```
Algorithm: Global Tree Construction
Input:  Power states
        Timing constraints
Output: A global buffered routing tree
Find power state groups $G$
foreach group $G_j$ in $G$ do
  foreach voltage island $VI_k$ in $G_j$ do
    Construct grid lines
    Initialize grid nodes
    Propagate solution from
     the pseudo sink $T_k$ of $VI_k$
    Prune redundant solutions
    if no feasible solutions then
      Create new voltage islands
      Update solutions and power states
```

Figure 3.9: The procedure of global tree construction.

1, respectively. Therefore, $VI2$ and $VI4$ form the first power state group; $VI5$ and $VI6$ form the second; $VI1$ forms the third; $VI3$ forms the fourth as listed in Figure 3.10.

Power state table	**Power state group**
P1: VI2 VI4 VI5 VI6	G0: VI2 VI4
P2: VI1 VI2 VI3 VI4	G1: VI5 VI6
P3: VI1 VI2 VI4	G2: VI1
	G3: VI3

Figure 3.10: Assume the power state table has three power states. Six voltage islands are clustered into four power state groups.

3.2.2 Solution Propagation

During global tree construction, solutions are propagated from the pseudo sink of each voltage island to a partially routed global tree in the following two conditions.

- Partially routed global tree of voltage islands in the always-on group.

Power state table	**Power state group**
P1: VI1 VI2 VI3 VI5	G0: VI2
P2: VI2 VI3 VI4 VI6	G1: VI1
P3: VI1 VI2 VI4	G2: VI3
	G3: VI4
	G4: VI5
	G5: VI6

Figure 3.11: The case with the maximum number of power state groups. Assume the power state table has three power states. Six voltage islands are clustered into six power state groups.

- Partially routed global tree of voltage islands in the same power state group.

Propagation is in the same manner with local tree construction. For example, Figure 3.12 is the case with the power state table in Figure 3.10. Dotted triangles represent local trees, and gray circles indicate propagations. As shown in Figure 3.12(a), $VI2$ and $VI4$, voltage islands in the always-on group, form a partially routed global tree. For $VI5$, solutions of the pseudo sink is propagated to the partially routed global tree of the always-on group. For $VI6$ in Figure 3.12(b), solutions of the pseudo sink may be propagated to the tree of the always-on group, or to $VI5$ in the same group.

To maintain signal integrity, solutions propagate from grid node w to its neighbor grid node u can be classified into the following four conditions.

- Both w and u are at high voltage island(s):

  solution of w with or without high voltage buffer can be combined with that of u, but that of w with level converter cannot.

- w is at low voltage island and u is at high one:

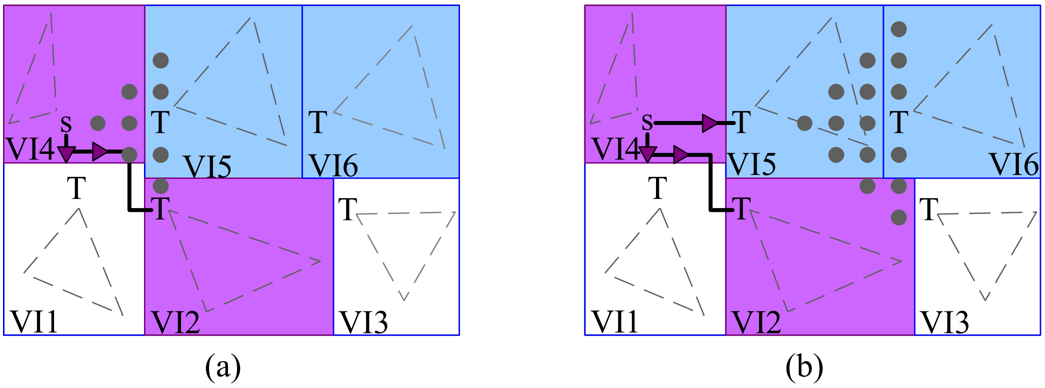


Figure 3.12: Dotted triangles represent resulting local trees rooted at the pseudo sinks. Solutions of T in $VI5$ are propagated to always-on group islands $VI2$ and $VI4$ in (a). Solutions of T in $VI6$ are propagated to always-on group island $VI4$ and the same group island $VI5$ in (b).

solution of w can be combined with solution of u with or without high voltage buffer, but cannot be combined with that of u with level converter.

- w is at high voltage island and u is at low one:

  only solution of w with level converter can be combined with solution of u.

- Both w and u are at low voltage island(s):

  solution of w with or without low voltage buffer can be combined with solution of u.

In other words, level converters are only inserted at grid nodes in high voltage islands, receiving signal from low voltage nodes.

3.2.3 New Voltage Island Creation

If no feasible solutions can be found after line 7, we create a new voltage island and update solutions. In addition, the power state table and groups are modified accordingly. In order to reduce the difficulty of power/ground planning and to minimize the modification of voltage island planning, we prefer to create new voltage islands at the peripheral of original ones. Therefore, for an infeasible solution, some buffers are temporarily allowed to be inserted at the peripheral of islands in the different power state groups to satisfy the timing constraints. New voltage islands are then created to cover these buffers. As shown in Figure 3.13, local trees of $VI2$ and $VI3$ are connected the source s. Assume $VI1$ and $VI4$ are turned on, but $VI2$ and $VI3$ are turned off. In Figure 3.13(a), red lines indicate an infeasible solution of connecting the local tree of $VI1$ to the partially routed global tree. The signal, from s to T, going through a turned-on island $VI4$ and a turned-off one $VI2$ violates timing constraints. Therefore, if several buffers can be inserted on $VI2$ as in Figure 3.13(b), the violation may be removed. Then, a new voltage island is created which is indicated by blue rectangle.

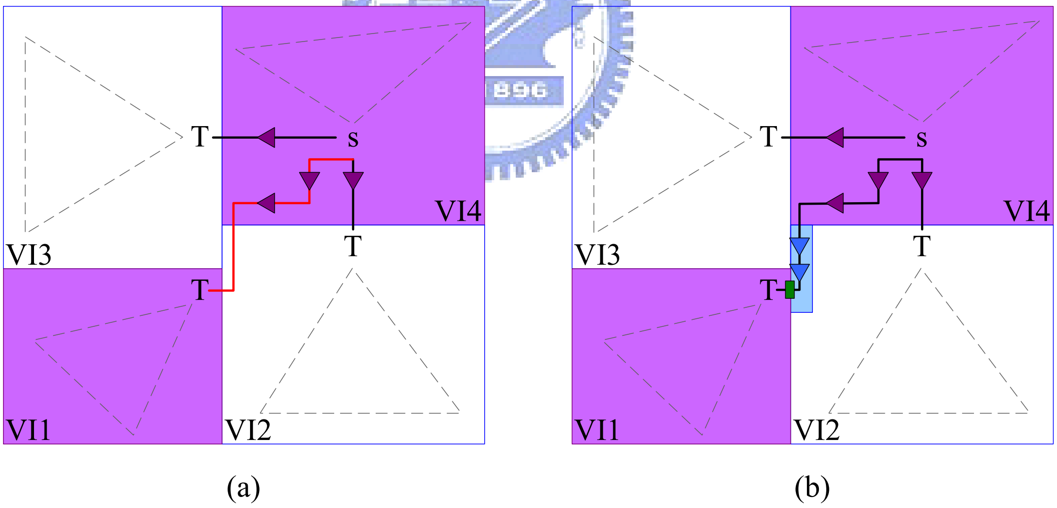


Figure 3.13: We show a case with a new created voltage island. $VI1$ and $VI4$ are turned on, and $VI2$ and $VI3$ are turned off. In (a), red lines indicate a solution violates timing constraints. If several buffers can be inserted in $VI2$ as in (b), the timing violation may be fixed. The blue rectangle is the new created voltage island.

Chapter 4

Experimental Results

We implemented our algorithms in C++ language on a 2.4GHz AMD Opteron$^{\mathrm{TM}}$ platform with 4GB memory.

We randomly created eight benchmarks listed in Table 4.1. The number specified in each case represents the number of sinks. The second column is the dimension of each case (the grid size is 0.5mm*0.5mm); the third column is the number of power states; the fourth column is the number of voltage islands; the fifth column is the delay; the sixth and seventh columns are the maximum and the average power consumption; the eighth column is the runtime. The ninth to eleventh columns are the number of high voltage buffers, low voltage buffers and level converters used in the resulting buffered tree. Each grid node is available for buffering. The maximum power consumption is the total power consumption while all voltage islands are turned on. Without loss of generality, we measured average power consumption by averaging the power of each power state. For example, power states and power consumption information are listed in Table 4.2. Assume two power states and five voltage islands in the case. Total power consumption of the global tree is 12300 pJ, is also the maximum power. Power consumption of voltage islands are measured at the pseudo sinks, representing the summation of the downstream power. Thus, power consumption of $P1$ is total power excluding power consumption of $VI2$ and $VI5$, and that of $P2$ is excluding power consumption of $VI1$ and $VI3$.

The average power consumption is computed as follows:

$$\begin{aligned} power(P1) &= power(total) - power(VI2) - power(VI5) \\ &= 12300 - 1850 - 2700 \\ &= 7750\ pJ. \\ power(P2) &= power(total) - power(VI1) - power(VI3) \\ &= 12300 - 1200 - 2310 \\ &= 8790\ pJ. \\ average\ power &= \frac{1}{2} \cdot (power(P1) + power(P2)) \\ &= \frac{1}{2} \cdot (7750 + 7890) \\ &= 7820\ pJ. \end{aligned}$$

In addition, the accurate power consumption can be computed by extracting the operating time slots of each power state from simulations. Compared with previous works, our method is effective and efficient.

To demonstrate the impacts of power states on routing and buffer insertion, Figure 4.1 shows our results of the case in Figure 1.4. In [15], sink $t2$ may receive an incorrect signal while the very low voltage island is turned off. In Figure 4.1, $VI1$ is the low voltage island, and $VI2$ is the high one. In addition, assume two power states, $VI1$ is turned on in one state, and turned off in the other. The resulting buffered tree shows even if $VI1$ is turned off, sinks $t1$, $t2$, $t5$ outside $VI1$ still receive correct signals. (The correctness is independent of how available buffer locations distribute.) Thus, our work maintains signal integrity well while considering power states.

In Figure 4.2, we show Case17 which the source s is at low voltage island $VI2$, and level converters (indicated by green rectangles) are properly inserted at

Table 4.1: The experimental results of eight benchmarks.

Benchmark	Dimension	#PS	#VI	Delay	Max. Power	Avg. Power	Runtime	Hardware Cost		
				(ns)	(pJ)	(pJ)	(s)	BUF_H	BUF_L	LC
Case12	30*20	2	4	7.6	6.6	5.7	0.21	22	15	0
Case17	30*20	2	6	6.1	6.6	4.9	0.17	18	18	3
Case20	40*40	3	6	8.5	12.8	7.1	1.06	36	44	3
Case26	50*50	3	6	9.7	18.2	9.5	2.38	52	68	3
Case37	50*50	3	6	9.7	23.2	12.8	4.50	73	77	3
Case56	100*100	2	6	20.7	57.4	48.9	36.03	197	158	0
Case101	200*200	2	5	58.5	150.8	106.2	507	513	391	2
Case200	200*250	2	5	52.4	236.9	191.6	2064	829	655	1

Table 4.2: Information for computing average power consumption.

Power state table	**Power consumption** (pJ)	
P1: VI1 VI3 VI4 P2: VI2 VI4 VI5	total	12300
	$VI1$	1200
	$VI2$	1850
	$VI3$	2310
	$VI4$	1590
	$VI5$	2700

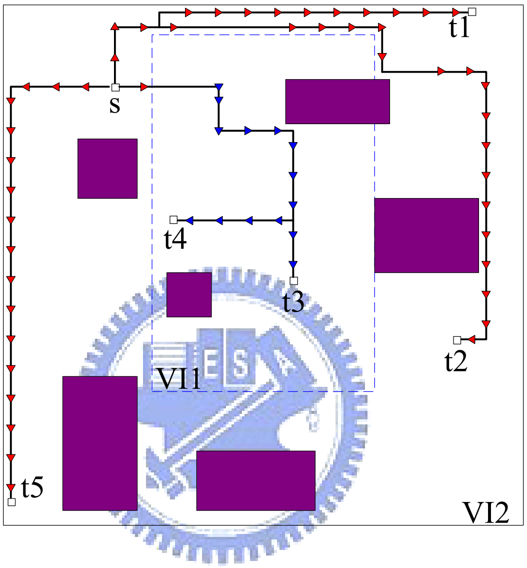


Figure 4.1: The resulting buffered tree generated by our method on the case in Fig. 1.4. $VI1$ is the low voltage island, $VI2$ is the high one. It can be seen while $VI1$ is turned off, sinks $t1$, $t2$, $t5$ outside $VI1$ still receive correct signals.

grid nodes of high voltage islands $VI1$, $VI3$ and $VI5$. The power states of Case17 are detailed in Figure 4.3. According to the power states, for $P1$, $VI4$ and $VI6$ are turned off, pseudo sinks of $VI1$, $VI3$ and $VI5$ are connected to $VI2$ directly. In addition, for $P2$, $VI1$ and $VI5$ are turned off, the pseudo sink of $VI4$ is connected through $VI1$, but no buffer is inserted within it, and that of $VI6$ is connected through $VI5$ without buffering. Therefore, it can be seen that signal integrity can be maintained for both power states.

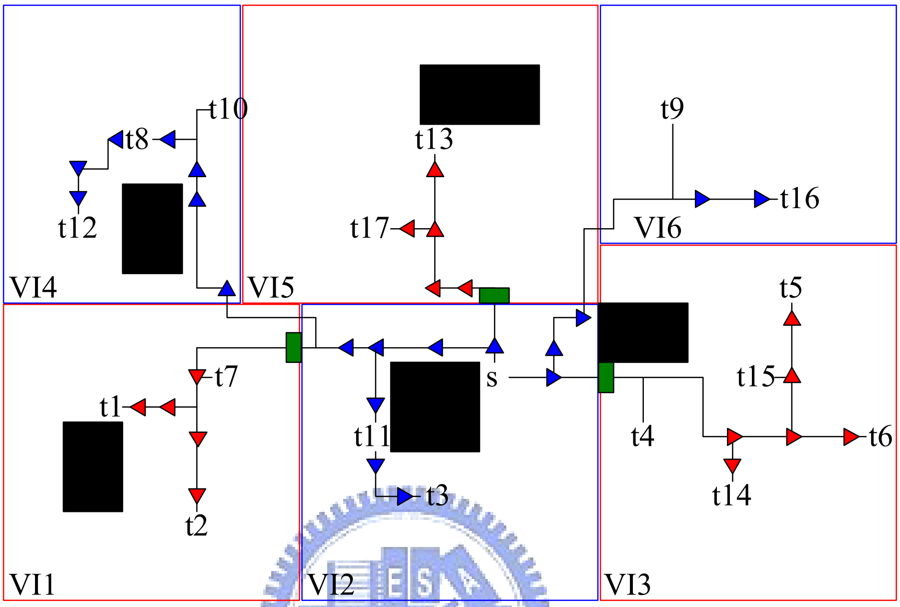


Figure 4.2: The resulting buffered tree of Case17. $VI1$, $VI3$ and $VI5$ are high voltage islands, and $VI2$, $VI4$ and $VI6$ are low ones. Because the source s is at low voltage island, some level converter are inserted properly.

Power state table	**Power state group**
P1: VI1 VI2 VI3 VI5	G0: VI2 VI3
P2: VI2 VI3 VI4 VI6	G1: VI1 VI5
	G2: VI4 VI6

Figure 4.3: Assume two power states and six voltage islands in Case17. Voltage islands are clustered into three power state groups.

Chapter 5

Conclusion

This thesis created a new research direction on considering power states into routing and buffer insertion. We proposed a hierarchical approach combined with dynamic programming to construct a buffered routing tree considering power states for dual supply voltage designs. Our method can easily be extended to multiple supply voltage designs and to more sophisticated delay models. The experimental results show we can not only minimize power but also maintain signal integrity effectively and efficiently.

Bibliography

[1] S. Augsburger and B. Nikolić, "Combining dual-supply, dual-threshold and transistor sizing for power reduction," in *Proc. International Conference on Computer Design*, pp. 316–321, 2002.

[2] H. Bakoglu, *Circuits, interconnections, and packaging for VLSI,* Addison-Wesley, 1990.

[3] R. Bergamaschi and Y. Jiang, "State-based power analysis for Systems-on-Chip," in *Proc. Design Automation Conference*, pp. 638–641, 2003.

[4] J. Cong and X. Yuan, "Routing tree construction under fixed buffer locations," in *Proc. Design Automation Conference*, pp. 379–384, 2000.

[5] A. Eliopoulos, P. Chen, and Q. Wang, "How to architect, design, implement, and verify low-power digital integrated circuits," in EDA DesignLine, 2007 (http://www.powermanagementdesignline.com/197001220).

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

[7] L. P.P.P. van Ginneken, "Buffer placement in distributed RC-tree networks for minimal Elmore delay," in *Proc. International Symposium on Circuits and Systems*, pp. 865–868, 1990.

[8] L. N. Kannan, P. R. Suaris, and H.-G. Fang, "A methodology and algorithms for post-placement delay optimization," in *Proc. Design Automation Conference*, pp. 327–332, 1994.

[9] J. Lillis, C.-K. Cheng, and T.-T. Y. Lin, "Optimal and efficient buffer insertion and wire sizing," in *Proc. Custom Integrated Circuits Conference*, pp. 259–262, 1995.

[10] M. Lai and D.F. Wong, "Maze routing with buffer insertion and wiresizing," *IEEE Trans. on Computer-Aided Design*, Vol. 12, pp. 1205-1209, 2002.

[11] A. Naveh et al., "Power and thermal management in the Intel® Core$^{\text{TM}}$ Duo processor," *Intel Technology Journal*, Vol. 10, 2006 (http://developer.intel.com/technology/itj/2006/volume10issue02/art03_Power_and_Thermal_Management/p01_abstract.htm).

[12] A. Srivastava and D. Sylvester, “Minimizing total power by simultaneous Vdd/Vth assignment,” in *Proc. Asia and South Pacific Design Automation Conference*, pp. 400–403, 2003.

[13] K. H. Tam and L. He, “Power optimal dual-vdd buffered tree considering buffer stations and blockages,” in *Proc. Design Automation Conference*, pp. 497–502, 2005.

[14] X. Tang, R. Tian, H. Xiang, and D.F. Wong, “A new algorithm for routing tree construction with buffer insertion and wire sizing under obstacle constraints,” in *Proc. International Conference on Computer-Aided Design*, pp. 49–56, 2001.

[15] B. Tseng, “An efficient algorithm for voltage island aware buffered routing tree construction,” Master’s Thesis, National Chiao Tung University, Hsinchu, 2006.

[16] K. Usami and M. Horowitz, “Clustered voltage scaling technique for low-power design,” in *Proc. International Symposium on Low Power Electronics and Design*, pp. 3–8, 1995.

[17] H. Wu, I-M. Liu, M. D.F. Wong, and Y. Wang, “Post-placement voltage island generation under performance requirement,” in *Proc. International Conference on Computer-Aided Design*, pp. 309–316, 2005.

[18] P.-C. Wu, J.-R. Gao, and T.-C. Wang, “A fast and stable algorithm for obstacle-avoiding rectilinear steiner minimal tree construction,” in *Proc. Asia and South Pacific Design Automation Conference*, pp. 262–267, 2007.

[19] H. Zhou, D.F. Wong, I-M. Liu, and A. Aziz, “Simultaneous routing and buffer insertion with restrictions on buffer locations,” *IEEE Trans. on Computer-Aided Design*, Vol. 19, pp. 819–824, 2000.