

國立交通大學

電子工程學系 電子研究所

博士論文

降低晶片上匯流排雜訊之匯流排編碼技術
之研究

**On-Chip Bus Encoding for *LC* Crosstalk
Reduction**

研究生：涂尚瑋

指導教授：周景揚 博士

中華民國九十五年八月

降低晶片上匯流排雜訊之匯流排編碼技術之研究
On-Chip Bus Encoding for *LC* Crosstalk Reduction


研究生：涂尚瑋 **Student: Shang-Wei Tu**

指導教授：周景揚博士 **Advisors: Jing-Yang Jou**

國立交通大學

電機學院 電子工程學系 電子研究所

博士論文



**A Dissertation Submitted to
Department of Electronics Engineering
and Institute of Electronics
College of Electrical and Computer Engineering
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in Electronics Engineering
August 2006
Hsinchu, Taiwan, Republic of China**

中華民國九十五年八月

降低晶片上匯流排雜訊之匯流排編碼技術 之研究

學生：涂尚璋

指導老師：周景揚

國立交通大學

電機學院 電子工程學系 電子研究所



摘 要

隨著製程演進至奈米科技，長導線之電路延遲與功率消耗，已經成為晶片設計上之棘手難題，此外，導線間，電感與電容耦合效應，也在電路上造成極嚴重問題，如串音，雜訊造成的電路延遲與多餘功率消耗等，因此，在現今高速晶片匯流排之設計上，設計者必須面對比以往更加嚴重的耦合效應所造成之電路延遲與功率消耗。

現今已經發表之晶片上匯流排編碼技術，大部分單純考慮電容耦合效應（造成最長延遲之匯流排輸入組態為由電容耦合效應所造成）從而導出其編碼系

統，因此，這些編碼技術有可能無法適用於超深次微米製程下之電路，尤其是在電感耦合效應非常明顯之狀況。

在本篇論文中，我們首先透過一系列實驗，呈現出在只考慮電容與同時考慮電感電容耦合效應下，所得到之最差輸入轉變組態，並發覺這兩種不同考量下所得到之結果極為不同，從而證實，前人所提出的編碼技術，由於單純考慮電容耦合效應，所以應用於電感耦合效應強烈的電路設計時，可能無法改善匯流排電路延遲，甚或造成額外的電路延遲。

在發現此一現象後，針對強烈電感耦合效應造成之匯流排延遲，我們提出以 bus-invert 方法來降低電路延遲，根據我們實驗結果，此一方法確實可有效降低電感耦合所造成之匯流排電路延遲。

接著，根據模擬結果，在同時考量導線電容與電感效應下，我們發現匯流排之最差輸入組態，會依據電路設計者所使用之參數，而有所不同，然而 bus-invert 方法只適用於降低電感效應為主之雜訊，應用上會有所限制，所以為了改善 bus-invert 使用上的限制，我們另外提出一種有彈性之匯流排編碼技術，此種編碼技術會依據設計者輸入之設計參數，產生出不同之編碼結果，以符合使用者需求，此外，經過適當修改，依據所給定之限制，此一編碼系統可應用於預測與增長訊號傳輸距離，接著我們也以實驗結果來支持我們的論點。

接下來，為了可更有效地降低導線耦合雜訊與改善我們編碼系統之效能，我們提出一個同時利用編碼與插地線之技術，以適用於未來奈米製程下之匯流排

設計，此一技術可依據使用者所給定之匯流排參數、工作頻率與限制，產生一組可有效降低電容電感耦合效應之匯流排編碼與架構，從而降低電路延遲，實驗結果也證實，此項技術確實可有效降低電路延遲。

由於導線之功率消耗業已成為現今高效能電路設計之棘手難題，有鑑於此，在考量使用者給定之延遲限制下，我們首先提出利用有彈性之匯流排編碼技術，以降低匯流排功率消耗；為得到更好的改善效果，接著我們也提出一個同時利用編碼與插地線之技術，在給定匯流排參數、工作頻率與延遲限制下，用以降低匯流排功率消耗，模擬結果證實，此兩項技術確實可在考量各種參數與延遲限制下，產生可達成匯流排功率消耗極小化之匯流排編碼，除此之外，比較有彈性之匯流排編碼技術，同時利用編碼與插地線之技術也確實能得到更好之匯流排功率改善效果。



On-Chip Bus Encoding for *LC* Crosstalk Reduction

Student: Shang-Wei Tu Advisors: Jing-Yang Jou

Department of Electronics Engineering

Institute of Electronics

National Chiao Tung University



Abstract

As technology advances, the global interconnect delay and the power consumption of long wires become crucial issues in nanometer technologies. In particular, both inductive and capacitive coupling effects between wires result in serious problems such as crosstalk delay, coupling noise, and power consumption. Hence, the strong coupling effects between wires make the delay and power consumption of on-chip buses worse than before. Therefore, it is crucial to reduce the

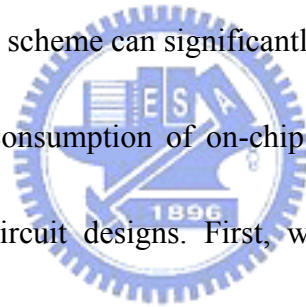
delay and power consumption of the on-chip bus to improve the circuit performance in DSM designs.

Most existing works consider only **RC** effects (the worst-case switching pattern resulting from coupling capacitance) to develop their encoding schemes to reduce the bus delay and/or the bus power consumption. Therefore, their works may not be suitable for the very deep-submicron designs when the inductive coupling becomes significant.

In this dissertation, we first show that the worst-case switching pattern that incurs the longest bus delay while considering the **RLC** effect is quite different from that while considering **RC** effect alone. It implies that the existing encoding schemes based on the **RC** model may not improve or possibly worsen the delay when the inductance effects become dominant. Then we propose a *bus-invert* method to reduce the worst-case on-chip bus delay with the dominance of the inductance coupling effect. Simulation results have shown that our encoding method can significantly reduce the worst coupling delay of the buses with strong inductive coupling.

From our simulation results, we observe that the worst-case switching pattern could vary with given design parameters considering the **RLC** effects of interconnects. However, the proposed *bus-invert* method can be utilized to reduce the bus coupling delay only when the inductance effects dominate. Therefore, we propose a flexible

encoding scheme for on-chip buses that can consider the given parameters to reduce the *LC* coupling delay. In addition, with some modification, this new encoding scheme can be utilized to predict and lengthen the signal propagation length of a bus under the given constraints. Simulation results are also given to support our claims. Next, to further reduce the coupling noise, we propose a joint shield insertion and bus encoding scheme for global bus design in nanometer technologies. With the user-given bus parameters, the working frequency, the scheme can effectively reduce the *LC* coupling effects and hence, minimize the bus coupling delay. Simulation results show that the proposed scheme can significantly reduce the coupling delay.



Recently, the power consumption of on-chip interconnect is another crucial issue in high performance circuit designs. First, we propose a flexible encoding scheme to minimize the power consumption of buses under given delay constraints. To further improve the performance of the encoding scheme (further reduce the power consumption), we also utilize the joint shield insertion and bus encoding method to solve the problem. With the user-given bus parameters, the working frequency, and the delay constraint, both schemes can minimize the bus power consumption subject to the delay constraint by effectively reducing the *LC* coupling effects. Simulation results show that the proposed schemes can significantly reduce the coupling delay and the power consumption of a bus according to the delay constraint. In addition, the

joint shield insertion and bus encoding method can gain more power reduction than that gained by only using the flexible bus encoding scheme.



誌 謝

首先，我要感謝周景揚教授，沒有周老師的鼓勵、督促、與指導，相信我不可能順利完成這篇論文。周老師不只在我的研究上給予許多指導，在學習與生活態度上也給予我良好的典範，使我不只在學業上精進，在身體健康與待人處事上也都能有長足的進步。

另外，我也要感謝張耀文教授，由於張教授細心指導，讓我得以大幅改善英文寫作與組織功力，對於日後論文的順利發表助益甚多。

我也非常感謝父母與家人，在博士班漫長的攻讀旅程中，除了經濟上的支援，對於我的鼓勵與關懷也從沒間斷過，家人不只是我疲累時的避風港，更是不斷支持我往前衝的加油站，謝謝你們，我完成博士班學位了！另外，我也要感謝我的女朋友—吳郁秀，雖然在一起還不滿一年，不過在我博士班的最後一年，稱職地扮演了我專屬的最佳後援會，使我能繼續全力衝刺完最後的這段修業行程。

我同時也要感謝實驗室許多學長姐、同學、學弟妹的幫助，尤其是王成業的許多幫忙—不管是學術上的還是生活上的。也非常感謝兩位願意跟我一起做研究的學弟—黃俊盛與林子為，他們是我研究上的重要夥伴，少了他們的幫忙，這篇論文肯定失色不少。除此，許智揚、王俊堯、黃俊達、黃恆亮學長等的幫忙，也是一股股支持我的重要助力。最後，感謝那一票在我苦悶時陪我一起打屁聊天的實驗室同仁，沒有你們我還真不曉得該如何度過這個漫長的修業行程。

謹將這篇論文獻給所有曾經對我博士論文有過幫助的人，與在我博一時不幸過世的沈文仁教授。

Contents

摘 要	i
Abstract	iv
誌 謝	viii
Contents	ix
List of Tables	xii
List of Figures	xiii
List of Figures	xiii
Chapter 1 INTRODUCTION	1
1.1 Coupling Noise on Deep-Submicron Interconnect.....	1
1.1.1 Coupling Capacitance	4
1.1.2 Coupling Inductance	9
1.2 Bus Encoding.....	22
1.2.1 Basic Concept	22
1.2.2 Previous Works	25
1.2.3 Our Contribution and Proposed Bus Encoding Schemes	26
1.3 Organization.....	30
Chapter 2 BUS-INVERT CODING SCHEME	31
2.1 Motivation.....	31
2.2 The Bus-invert Scheme.....	32

2.3	Simulation Results	38
2.3.1	Bus Coupling Delay Reduction.....	38
2.3.2	Delay Overhead of the Bus Encoder.....	42
2.4	Summary.....	46
 Chapter 3 FLEXIBLE BUS ENCODING SCHEME		47
3.1	Motivation.....	47
3.2	Proposed Flexible Encoding Scheme.....	49
3.2.1	Flexible Bus Encoding Flow.....	49
3.2.2	Simulation Results	62
3.2.2	Summary of Proposed Flexible Encoding Scheme.....	70
3.3	Flexible Encoding Scheme for Increasing Bus Propagation Length	71
3.3.1	Motivation.....	71
3.3.2	Signal Propagation Length Increasing Flow	73
3.3.3	Simulation Results	77
3.3.4	Summary of Signal Propagation Length Increasing Flow	85
 Chapter 4 JOINT SHIELD INSERTION AND BUS ENCODING SCHEME		87
4.1	Motivation.....	87
4.2	Optimized Shield Insertion	88
4.3	Proposed Joint Shield Insertion and Bus Encoding Scheme	91
4.3.1	Signal Propagation Length Increasing Flow with Joint Shield Insertion and Bus Encoding Scheme	92
4.3.2	Simulation Results	94
4.4	Summary.....	99

Chapter 5	FLEXIBLE ON-CHIP BUS ENCODING FOR	
POWER	MINIMIZATION	UNDER
		DELAY
CONSTRAINT.....		100
5.1	Motivation.....	100
5.2	Flexible Bus Encoding Scheme for Power Minimization Under	
	Delay Constraint	103
5.2.1	Encoding Flow for Power Minimization.....	103
5.2.2	Simulation Results	108
5.2.3	Summary of Flexible Bus Encoding Scheme for Power Minimization	
	Under Delay Constraint.....	114
5.3	Joint Shield Insertion and Bus Encoding Scheme for Power	
	Minimization Under Delay Constraint	115
5.3.1	Motivation.....	115
5.3.2	Joint Shield Insertion and Bus Encoding Flow for Power Minimization	
	116
5.3.3	Simulation Results	118
5.3.4	Summary.....	121
Chapter 6	CONCLUSIONS & FUTURE WORKS	122
Bibliography		126

List of Tables

Table 1: Simulation results of a 5-bit bus considering only <i>RC</i> effects (0: no transition).....	8
Table 2: Simulation results of a 5-bit bus considering <i>RLC</i> effects. ($V_{dd} = 1.2V$)	17
Table 3: Simulation results of the 5-bit bus when wire capacitance becomes dominant (10X wire capacitance).....	19
Table 4: The transition delays of the 2-bit bus shown in Figure 13 . (↑: transit from “0” to “1”; ↓: transit from “1” to “0”; –: no transition).....	24
Table 5: The transition delays of the 3-bit bus shown in Figure 14	24
Table 6: Reduction of worst-case noise by using the <i>bus-invert</i> method for bus widths ranging from 2 to 11.	41
Table 7: Interconnect and device parameters used.	42
Table 8: The simulation results of the coupling delay reduction by using our encoding method, and the delay overhead of the encoder for different technology nodes.	45
Table 9: The runtime improvement of our method.	64
Table 10: The wire overheads versus different delay constraints for a 4-bit data bus.	66
Table 11: Delay overheads of codecs.	84
Table 12: Area overheads of codecs.	85
Table 13: Simulation results of all bus structures for a 6-bit bus with m varying from 6 to 11. ($E\#$ denotes # of additional signal wires added into the original bus; $S\#$ denotes # of shielding wires inserted into the original bus)	95
Table 14: Transition currents obtained form HSPICE simulations and the superposition method.....	106
Table 15: The simulation results of the delay and power minimization for $n = 5$ and $m = 6\sim 7$. (↑: switching from “0” to “1”; ↓: switching from “1” to “0”; _: no switching)	109
Table 16: The simulation results of the delay and power minimization for $n = 6$ and $m = 7\sim 8$	110
Table 17: The simulation results of the delay and power minimization for $n = 5$ and $m = 7$ with the working frequency varying from 100MHz to 5GHz.....	112
Table 18: The simulation results of the power minimization only for $n = 5\sim 6$ and $m = 7\sim 8$	113

List of Figures

Figure 1: Technology scaling versus the interconnect delay [2].	2
Figure 2: A three-wire bus.	4
Figure 3: A 5-bit coplanar bus structure.	5
Figure 4: Our simulation flow.	7
Figure 5: Magnetic field \vec{B} created by a time-varying current flowing through a conductor loop.	10
Figure 6: Electric voltage created by time-varying magnetic field passing through a conductor loop. The integral of the magnetic field over the loop area is referred to as the magnetic flux.	11
Figure 7: Magnetic field created by the time-varying current in loop i induces voltage in victim loop j , since some of the magnetic field passes through j .	12
Figure 8: An LC cross-coupled 5-bit bus structure. (a) The switching pattern of the worst-case delay of the central wire in the RL model. (b) The switching pattern of the best-case delay of the central wire in the RL model. (\uparrow : switch from “0” to “1”. \downarrow : switch from “1” to “0”).	15
Figure 9: The delays (% of that of pattern $00\uparrow 00$) of the worst-case switching pattern with various wire capacitances.	18
Figure 10: The inductive noise is less significant as wire capacitance increases.	18
Figure 11: The delays (% of that of the pattern $00\uparrow 00$) of the worst-case switching pattern with various signal rise times.	21
Figure 12: The overall bus structure including the encoder and decoder.	23
Figure 13: A 2-bit bus example.	24
Figure 14: The encoded bus after encoding the 2-bit bus in Figure 13 to a 3-bit one.	24
Figure 15: (a) A 4-bit bus encoder for the <i>bus-invert</i> scheme.	34
Figure 16: The worst-case $\#$ -bit bus delay (% of the delay of only one transition pattern of the $\#$ -bit bus) with $\#$ varying from 2 to 11.	39
Figure 17: The reduction of worst-case delay for a $\#$ -bit bus by using the <i>bus-invert</i> method and the shield insertion technique with the $\#$ varying from 2 to 11.	39
Figure 18: Our flexible bus encoding flow.	49
Figure 19: Extract RLC and generate the corresponding SPICE file.	51
Figure 20: HSPICE simulations with the <i>basis vectors</i> of the <i>minimum basis vector</i>	

<i>set</i>	55
Figure 21: An example of superposition.	56
Figure 22: The voltage waveforms of the signal wires of a 3-bit bus obtained (a) by directly conducting HSPICE simulation with $(- \uparrow \uparrow)$ and (b) by using superposing the results of $(- - \uparrow)$ and $(- \uparrow -)$	57
Figure 23: An example of the transition graph of a 3-bit bus.	58
Figure 24: Our greedy algorithm to find a maximum clique.	60
Figure 25: The greedy search result of Figure 22	61
Figure 26: The number of extended clique size by using the second loop in Figure 24	62
Figure 27: The runtime of our method and pure HSPICE simulation.	63
Figure 28: Transition delay of the bus (6 bit) before encoding.	64
Figure 29: Transition delay of the bus (7 bit) after encoding.	65
Figure 30: The bus structures after (a) using our encoding method and (b) applying the shield insertion technique.	68
Figure 31: The worst-case transition delay by using our method and the shield insertion technique.	69
Figure 32: The signal propagation length increasing flow. (D : bus propagation length; D_{max} : maximum bus propagation length).....	73
Figure 33: Curve-fitting with different functions.....	76
Figure 34: The maximum propagation length vs. different wire overheads for a 4-bit data bus.	78
Figure 35: The maximum propagation length vs. different wire overheads for a 6-bit data bus.	79
Figure 36: The maximum propagation length by using our flow and the shield insertion technique under different frequencies.....	81
Figure 37: The maximum propagation length by using our flow and the shield insertion technique for different bus width.....	82
Figure 38: All combinations of two-shield-inserted bus structures for a 6-bit bus. In this figure, the black lines denote the signal wire and the grey and longer lines denote the inserted power/ground wires.	89
Figure 39: The optimized bus structures with different number of shield inserted in a 6-bit bus. In this figure, the black lines denote the signal wire and the grey and longer lines denote the inserted power/ground wires.	89
Figure 40: The worst-case current return paths of a 6-bit bus with one shield inserting in different position.....	90
Figure 41: The signal propagation length increasing flow with joint shield insertion and bus encoding scheme.	91

Figure 42: The bus structures generated by our bus structure generation scheme..... 94

Figure 43: Increasing length of different bus structures with $m - n = 4$ 97

Figure 44: Increasing length of different bus structures with $m - n = 5$ 97

Figure 45: Increasing length of a 6-bit bus with m varying from 6 to 9 by using only bus encoding, only shield insertion, and joint shield insertion and bus encoding..... 98

Figure 46: The proposed flexible encoding flow for bus power minimization..... 105

Figure 47: Pseudo code of finding a code set with minimal edge weight..... 108

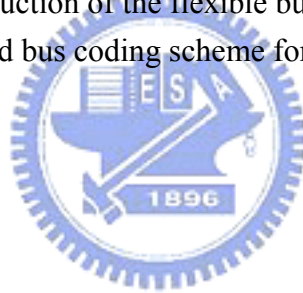
Figure 48: Power reduction of $n= 5$ and m varying from 6 to 10. 114

Figure 49: The proposed joint shield insertion and bus encoding flow for bus power minimization. 117

Figure 50: Average power reduction of the flexible bus encoding scheme and the joint shield insertion and bus coding scheme for 0.5 GHz working frequency. 119

Figure 51: Average power reduction of the flexible bus encoding scheme and the joint shield insertion and bus coding scheme for 1 GHz working frequency. 120

Figure 52: Average power reduction of the flexible bus encoding scheme and the joint shield insertion and bus coding scheme for 5 GHz working frequency. 120



Chapter 1

INTRODUCTION

1.1 Coupling Noise on Deep-Submicron

Interconnect



The trend of technology scaling (shrinking feature sizes) and the increase of clock frequencies are predicted to continue. Shrinking feature sizes imply shorter gate lengths, decreasing interconnect pitch (more congested interconnect), and lower device threshold voltage. With aggressive scaling of transistor gate lengths, interconnect delay increasingly dominates chip performance in deep-submicron designs [1 – 3]. As **Figure 1** shows, further technology scaling in deep-submicron indicates that interconnect delay will continue to dominate overall chip performance. Therefore, it is crucial to optimize or reduce interconnect delay and noise for improving circuit performance.

With the decreasing of interconnect pitch and the increasing of the line aspect ratio, coupling capacitance will dominate total wire capacitance in deep-submicron designs. For example, the ratio of total inter-line (coupling) capacitance to total line-to-ground (area) capacitance in standard 0.18 μm technologies (minimum pitch space) is six to one, and the ratio can be higher than eight to one in 0.13 μm technologies [4]. Hence, the coupling capacitance of interconnect should specially be considered while considering the **RC** delay and crosstalk noise of interconnects.

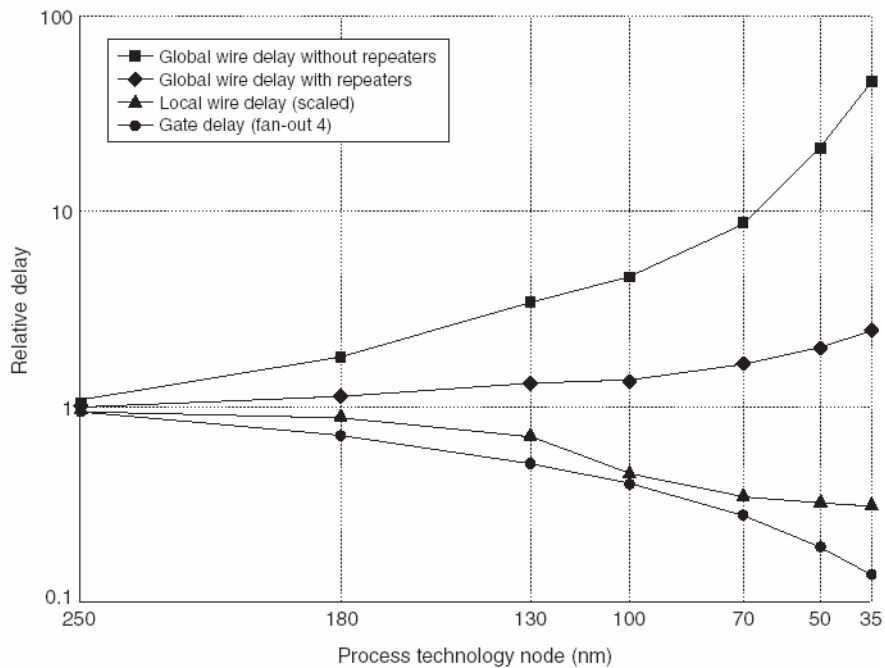
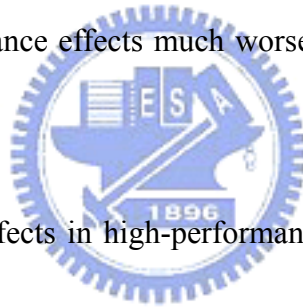


Figure 1: Technology scaling versus the interconnect delay [2].

As the process technology advances and the clock frequency increases over GHz, the inductance effects of on-chip interconnect structures have become

increasingly significant [2, 5]. This phenomenon is caused by the following factors: (1) the introduction of copper and wider upper-layer metal wires reduces the resistance of interconnect, (2) the use of low- k dielectric decreases the capacitance effects, and (3) high clock frequency (short rise/fall time) decreases the impedance of wire capacitance which is $1/j\omega C$ ($\omega = 2\pi f$), and increases the impedance of wire inductance which is $j\omega L$. All of these trends make inductance effects much more significant than before, especially on global interconnects. Further, the use of new dielectric materials (ultra-low- k or extreme low- k materials [2]) and the continuing increase of clock frequencies will cause inductance effects much worse than capacitance effects in the near future.



On-chip inductance effects in high-performance circuit designs might impact interconnect in many ways. The performance of a circuit will be reduced due to the increase of wire delay [6, 7]. The long-range inductive crosstalk can cause serious signal integrity related problems [7, 8]. Signal overshoots and undershoots due to wire inductance may damage devices. Finally, inductance in power and ground grids can increase the noise in the supply and ground voltages when large currents flow. This is also known as the *ground-bounce* problem. Therefore, inductance effects cannot be neglected in today's high-performance circuit designs, especially for global interconnects such as clock wires and signal buses.

1.1.1 Coupling Capacitance

In this section, we describe how the coupling capacitance affects on-chip buses. **Figure 2** shows an example of a three-wire bus. Assume all inputs transit simultaneously. Then the effective capacitance C_{eff} of the center wire i can be represented as the following equation [9].

$$C_{eff} = C_0 + C_c \left| \frac{\Delta V_{i-1} - \Delta V_i}{V_{dd}} \right| + C_c \left| \frac{\Delta V_i - \Delta V_{i+1}}{V_{dd}} \right|, \quad (1)$$

where C_0 is the capacitance between the wire and the substrate, C_c is the capacitance between adjacent wires, ΔV_i is the voltage variation of wire i , and V_{dd} is the supply voltage (equals to rail-to-rail signal voltage in CMOS circuits). This equation shows C_{eff} varies according to the transition direction of the adjacent wires.

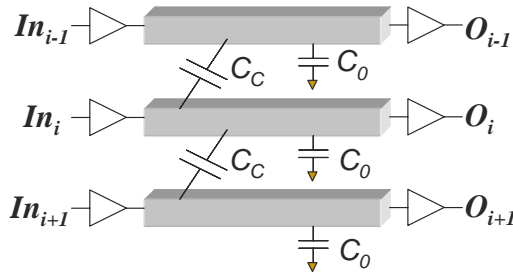


Figure 2: A three-wire bus.

When wire i switches alone and both neighbors are quiet, C_{eff} is $C_0 + 2C_c$.

While three wires simultaneously switch in the same direction, C_{eff} corresponds to C_0 . In this case, the coupling capacitance has no effect on the central wire. Hence, the transition delay of wire i is smaller than the case of solitary transition. However, as the central wire i and both neighboring wires $i+1$ and $i-1$ simultaneously switch for opposite transition direction, C_{eff} becomes the maximum value $C_0 + 4C_c$. Therefore, the bus transition delay is maximized in this case. This worst-case delay determines the bus clock cycle time and bus performance.

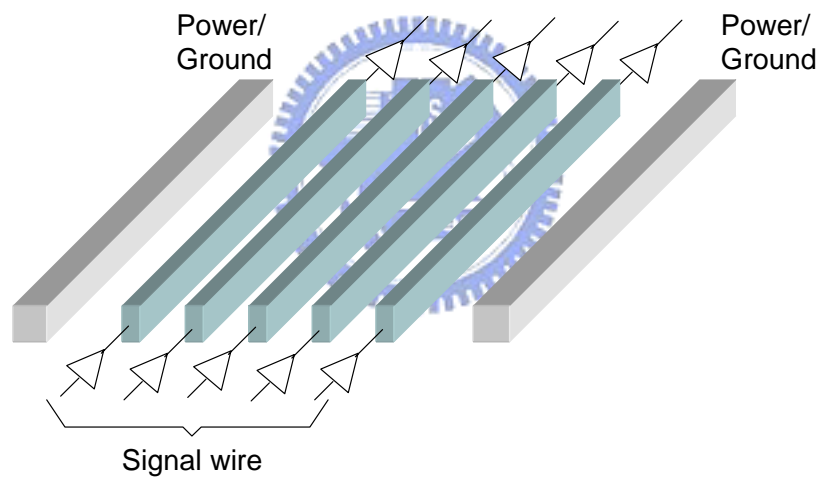
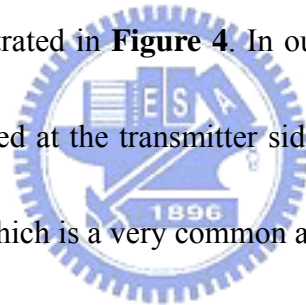


Figure 3: A 5-bit coplanar bus structure.

In this dissertation, we used the bus structure shown in **Figure 3** to conduct our simulations. We assume that all drivers (receivers) have a uniform size and all signal wires have a uniform width, spacing, and length. In this experiment, the length, width, and pitch of the signal wire were $2000\mu\text{m}$, $0.8\mu\text{m}$, and $2\mu\text{m}$, respectively. The

respective width and pitch of the power/ground were $2\mu\text{m}$ and $13\mu\text{m}$. The heights of all wires are set to $2\mu\text{m}$. The signal rise/fall time was set to 100ps . With these feasible parameters [2, 5, 10], we used the famous 3D field-solver FastCap [11] to extract the self and coupling capacitance and FastHenry [12] to extract the resistance, self inductance, and coupling inductance. Then with these extracted *RLC* parameters, we can construct the coupling *RLC* and *RC* circuit models. Both circuit models were constructed as π -segments using series resistance (or series resistance and inductance for *RL*) and shunt capacitance. Finally, the circuits were simulated by using HSPICE. The overall flowchart is illustrated in **Figure 4**. In our simulations, we assumed that synchronous latches are located at the transmitter side. Thus all the signals switch at the same time on the buses, which is a very common assumption for buses [13].



In this experiment, we assume that all signal wires only switch at the same time and may have arbitrary switching patterns, i.e., switching high or switching low. Comparing to [14], authors assume that aggressors can switch at arbitrary moments and victims are quiet. In addition, [14] tries to find the switching pattern and switching time resulting in the worst-case noise (*WCN*) defined as the maximum crosstalk noise peak on a quiet victim net. However, our work tries to find the simultaneously switching pattern that causes the maximum transition delay on a switching victim. Therefore, our work is significantly different from [14].

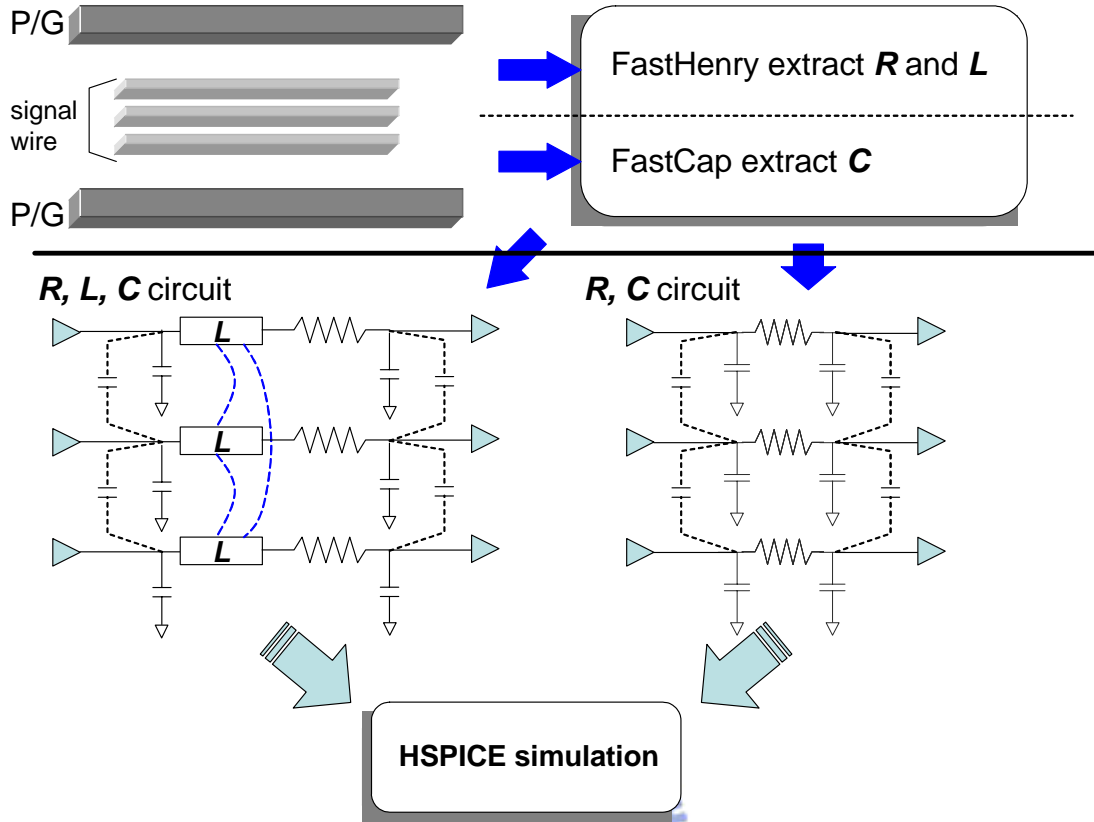


Figure 4: Our simulation flow.

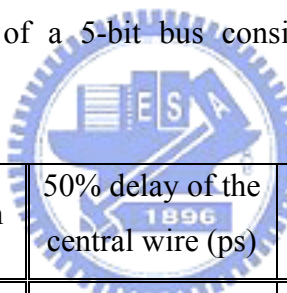
In this dissertation, ramp signal is given to the input of the driver and the transition time τ_r can be calculated from the working frequency f by the following equation [15]:

$$\tau_r = \frac{0.35}{f} . \quad (2)$$

In this subsection, we simulate all switching patterns on the 5-bit bus structure considering only RC effects. The simulation results are listed in **Table 1**. We should note that the number of total switching patterns is $2^5 = 32$ (without considering

non-transition cases). However, switching from “0” to “1” is symmetric to switching from “1” to “0” for the bus delay computation. Therefore, the complete switching patterns can be reduced to $2^5/2 = 16$. Besides, since the 5-bit bus structure is also a symmetric structure with respect to the central signal wire. For example, the switching patterns $\downarrow\downarrow\uparrow\uparrow$ and $\uparrow\uparrow\downarrow\downarrow$ have the same delay effect on the central signal wire. Hence, the complete switching patterns can further be reduced to 10 patterns as listed in **Table 1** (the first 10 patterns).

Table 1: Simulation results of a 5-bit bus considering only *RC* effects (0: no transition).



Switching pattern	50% delay of the central wire (ps)	Delay comparison with that of 00↑00
↑↑↑↑↑	33	-35.29%
↑↓↑↑↑	49	-3.92%
↑↓↑↑↓	51	0.00%
↑↓↑↓↑	73	43.14%
↑↓↑↓↓	75	47.06%
↓↑↑↑↑	35	-31.37%
↓↑↑↑↓	37	-27.45%
↓↑↑↓↓	54	5.88%
↓↓↑↑↑	52	1.96%
↓↓↑↓↓	76	49.02%
00↑00	51	0.00%

From **Table 1**, the three patterns, $\downarrow\downarrow\uparrow\downarrow$, $\uparrow\downarrow\uparrow\downarrow$, and $\uparrow\downarrow\uparrow\downarrow$, result in significantly larger delays on the central signal wire. Obviously, as discussed

previously, when we consider the resistance, self capacitance, and coupling capacitance of interconnects, the worst-case switching patterns that incur the largest delay is when adjacent wires simultaneously switch in opposite transition directions. On the other hand, the pattern $\uparrow\uparrow\uparrow\uparrow$ results in the minimum delay on the central signal wire. In other words, the best-case switching pattern that incur the minimum delay is when adjacent wires simultaneously switch in the same transition direction with that of the victim wire.

1.1.2 Coupling Inductance



1.1.2.1 Magnetic Induction

The process of inductive interaction between conductors carrying currents can be decomposed into three effects which take place concurrently [16, 17]:

1. Currents flowing through conductors create magnetic fields (*Ampere's Law*);
2. Magnetic fields varying with time create induced electric fields (*Faraday's Law*);
3. Induced electric fields exert forces upon the electrons in the conductors

and cause electric voltage (Electric Potential) drops.

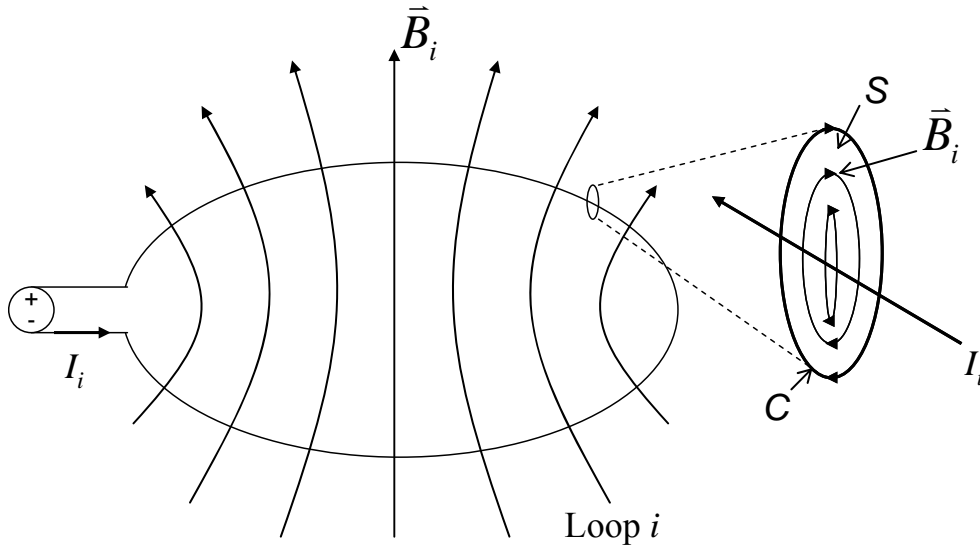
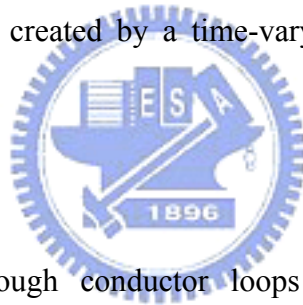


Figure 5: Magnetic field \vec{B} created by a time-varying current flowing through a conductor loop.



Currents flowing through conductor loops create magnetic fields. This relationship between current density j and the resulting magnetic field \vec{B}_i is

Ampere's Law:

$$\int_C \vec{B}_i \cdot d\vec{l} = \mu \int_S \vec{J}_i \cdot d\vec{s} = \mu I_i \quad (3)$$

where the path C for the line integral is the contour bounding the surface S (see **Figure 5**), I is the total current through S , and μ is the magnetic permeability of the insulator surrounding the wire. Here we assume *quasistatic* condition (neglect the

displacement current). The orientation of \vec{B}_i can be determined from the *right-hand rule*: if the thumb of the right hand points in the current direction, the other fingers point in the direction of the magnetic field. The sense of tracing C and the direction of current flow also follow the *right-hand rule*.

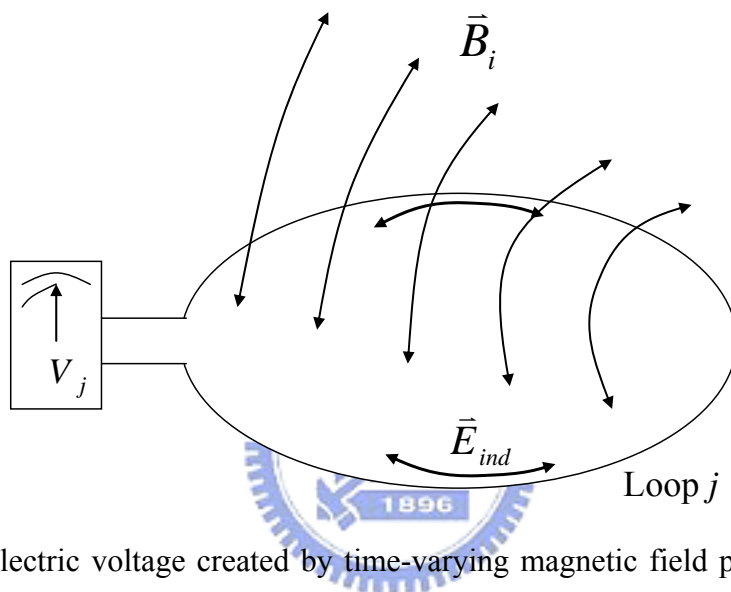


Figure 6: Electric voltage created by time-varying magnetic field passing through a conductor loop. The integral of the magnetic field over the loop area is referred to as the magnetic flux.

Ampere's Law gives us the first part of the inductive process: the creation of the magnetic field. Only if these magnetic fields vary with time do these fields create induced electric fields. Therefore, time-varying currents are required for induction, the relationship of which is *Faraday's Law*:

$$\oint_{\text{Loop } j} \vec{E}_{\text{ind}} \cdot d\vec{l} = V_j = -\frac{d\Phi_{ij}}{dt} \text{ with } \Phi_{ij} = \int_{S_j} \vec{B}_i \cdot d\vec{s}_j \quad (4)$$

where V_j is the electromotive force induced in circuit loop j due to the time-varying current I_i in circuit loop i (see **Figure 6**). Here, Φ_{ij} is the magnetic flux in loop j due to the current I_i in loop i , \vec{B}_i is the magnetic flux density arising from current I_i in loop i , and S_j represents the surface bounded by the loop j .

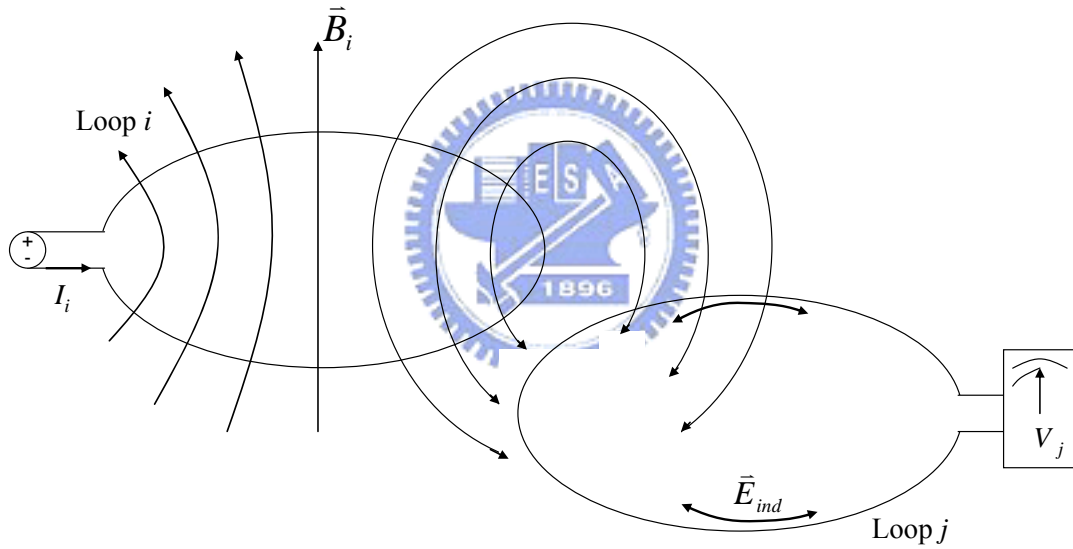


Figure 7: Magnetic field created by the time-varying current in loop i induces voltage in victim loop j , since some of the magnetic field passes through j .

The induced electric field can be integrated along the victim loop and results in an induced voltage which adds to the already existing voltage due to the resistance of the loop:

$$V_j = - \oint_{\text{Loop } j} \vec{E}_{\text{ind}} \cdot d\vec{l} . \quad (5)$$

Figure 7 summarizes the combination of these three effects which combine to generate a voltage drop in the victim loop j due to a time-varying current in loop i . All three relationship involved, Equation (3), (4), and (5), are linear. Therefore, the resulting combined relationship between time-derivatives of currents in the loops and the resulting induced voltage drops is linear as well:

$$V_j = L_{ij} \frac{dI_i}{dt} \quad \text{with } I_i = \int_{\text{crosssection of } i} \vec{j} \cdot d\vec{S} . \quad (6)$$

L_{ij} is the coupling (mutual) inductance of loop i upon loop j and I_i is the current flowing through loop i . If loop i and j are the same, the coefficient L_{ii} is then the self inductance of loop i .

1.1.2.2 Worst-case Inductive Coupling

From *Faraday's Law*, as shown in Equation (4), the electromotive force induced in a closed circuit is equal to the negative rate of increase of the magnetic flux linking the circuit.

$$V_j = -\frac{d\Phi_{ij}}{dt} \text{ with } \Phi_{ij} = \int_{S_j} \vec{B}_i \cdot d\vec{s}_j \quad (7)$$

where V_j is the electromotive force induced in circuit loop j due to the time-varying current I_i in circuit loop i . Here, Φ_{ij} is the magnetic flux in loop j due to the current I_i in loop i , \vec{B}_i is the magnetic flux density arising from current I_i in loop i , and S_j represents the surface bounded by the loop j . As shown in **Figure 8(a)**, the time-varying (increasing) current of the leftmost aggressor wire will induce a downward time-varying (increasing) magnetic field on the victim wire. For simplicity, the aggressor and the victim loops are also shown in **Figure 8(a)** (dotted loops).

Therefore, the mutual flux Φ is positive and also increases with time because of the same direction of the resulting magnetic flux density \vec{B} and the victim loop \vec{s} ; in other words, $\vec{B} \cdot d\vec{s}$ is also positive. In conclusion, from Equation (7), the induced voltage on victim loop is negative; that is, the induced current on the victim wire flows in the reverse direction of the victim current. Hence, while all neighboring wires simultaneously switch in the same direction as the victim wire does, they will all induce a current of the different direction on the victim wire as shown in **Figure 8(a)**.

Therefore, the charge current of the victim wire will be reduced. This implies that the charging time (delay) will increase due to the long-range coupling. We can conclude that as the inductance of wires becomes more significant than the capacitance, the

worst-case switching pattern with the maximum delay is when all wires simultaneously switch in the same direction. Meanwhile, these patterns will also result in the largest noise between each other.

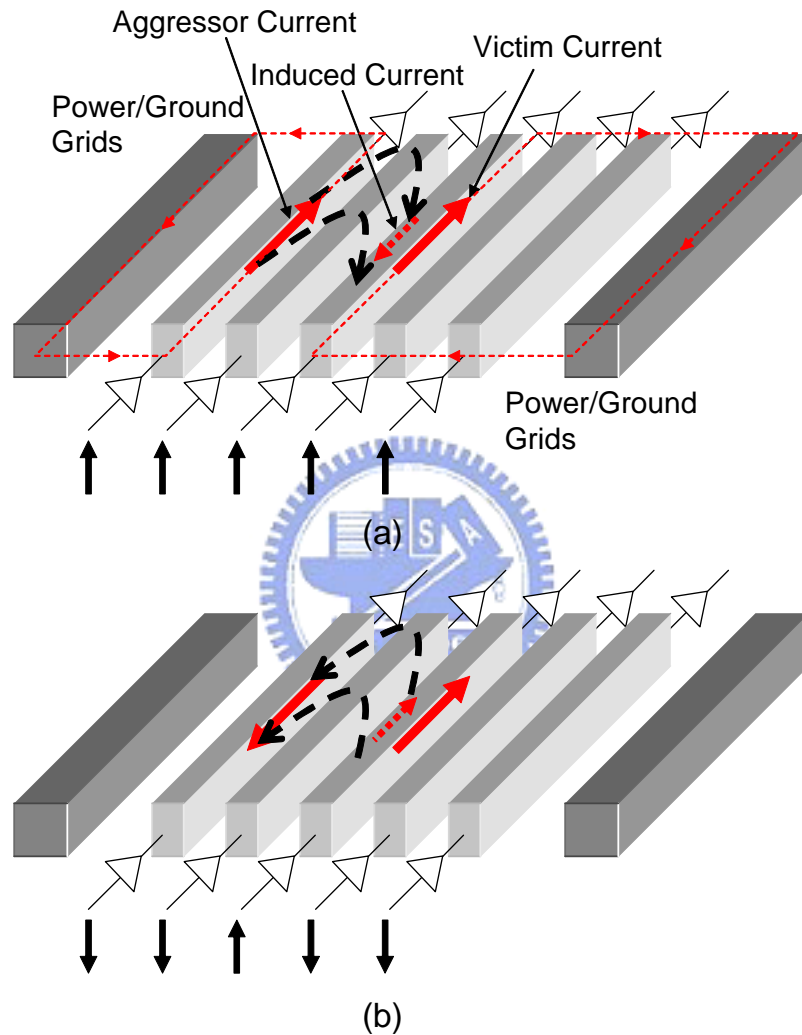
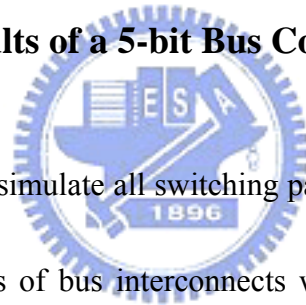


Figure 8: An *LC* cross-coupled 5-bit bus structure. (a) The switching pattern of the worst-case delay of the central wire in the *RL* model. (b) The switching pattern of the best-case delay of the central wire in the *RL* model. (\uparrow : switch from “0” to “1”. \downarrow : switch from “1” to “0”.)

Contrary, while all neighboring wires simultaneously switch in the opposite

direction of the victim wire, they will all induce a current of the same direction on the victim wire as shown in **Figure 8(b)**. Therefore, the charge current of the victim wire will be increased. This implies that the charging time (delay) will decrease due to the long-range coupling. We can conclude that as the inductance of wires becomes more significant than the capacitance, the best-case switching pattern with the minimum delay is when all aggressor wires simultaneously switch in the opposite direction of the victim wire.

1.1.2.3 Simulation Results of a 5-bit Bus Considering *RL* Effects



In this subsection, we simulate all switching patterns on the 5-bit bus structure considering the ***RL(C)*** effects of bus interconnects working at GHz range (i.e., the capacitance effects can be neglected). The simulation results of the extracted ***RL(C)*** circuit model for the 5-bit bus are shown in **Table 2**. From **Table 2**, we observe that the worst-case switching pattern changes from $\downarrow\downarrow\uparrow\downarrow\downarrow$ (in **Table 1**) to $\uparrow\uparrow\uparrow\uparrow\uparrow$ and the best-case switching pattern changes from $\uparrow\uparrow\uparrow\uparrow\uparrow$ (in **Table 1**) to $\downarrow\downarrow\uparrow\downarrow\downarrow$ as predicted in previous subsection. Therefore, the worst-case and best-case switching patterns are completely different considering ***RC*** and ***RL*** effects. Therefore, as the process technology keeps shrinking and the clock frequency continues increasing, it is very important to consider the inductance effect on the bus structure to derive encoding

schemes to reduce bus delay. Otherwise, the encoding schemes might not improve or even worsen the on-chip bus delay because of the redundant logics and wires. Further, we also observe that the largest overshoot noise occurs for the pattern $\uparrow\uparrow\uparrow\uparrow$, as shown in **Table 2**.

Table 2: Simulation results of a 5-bit bus considering *RLC* effects. ($V_{dd} = 1.2V$)

Switching pattern	50% delay of the central wire (ps)	Delay comparison with that of $00\uparrow 00$	Max swing (V)	Noise (% of V_{dd})
$\uparrow\uparrow\uparrow\uparrow$	97	51.56%	1.71	42.50%
$\uparrow\downarrow\uparrow\uparrow$	88	37.50%	1.47	22.50%
$\uparrow\downarrow\uparrow\downarrow$	63	-1.56%	1.31	9.20%
$\uparrow\downarrow\uparrow\downarrow$	75	17.19%	1.17	-2.50%
$\uparrow\downarrow\uparrow\downarrow$	51	-20.31%	1.04	-13.33%
$\downarrow\uparrow\uparrow\uparrow$	78	21.88%	1.56	30.00%
$\downarrow\uparrow\uparrow\downarrow$	55	-14.06%	1.4	16.67%
$\downarrow\uparrow\uparrow\downarrow$	45	-29.69%	1.13	-5.83%
$\downarrow\downarrow\uparrow\uparrow$	66	3.13%	1.33	10.83%
$\downarrow\downarrow\uparrow\downarrow$	37	-42.19%	0.903	-24.75%
$00\uparrow 00$	64	0.00%	1.31	9.17%

1.1.2.4 Simulation Results of a 5-bit Bus Considering *RLC* Effects

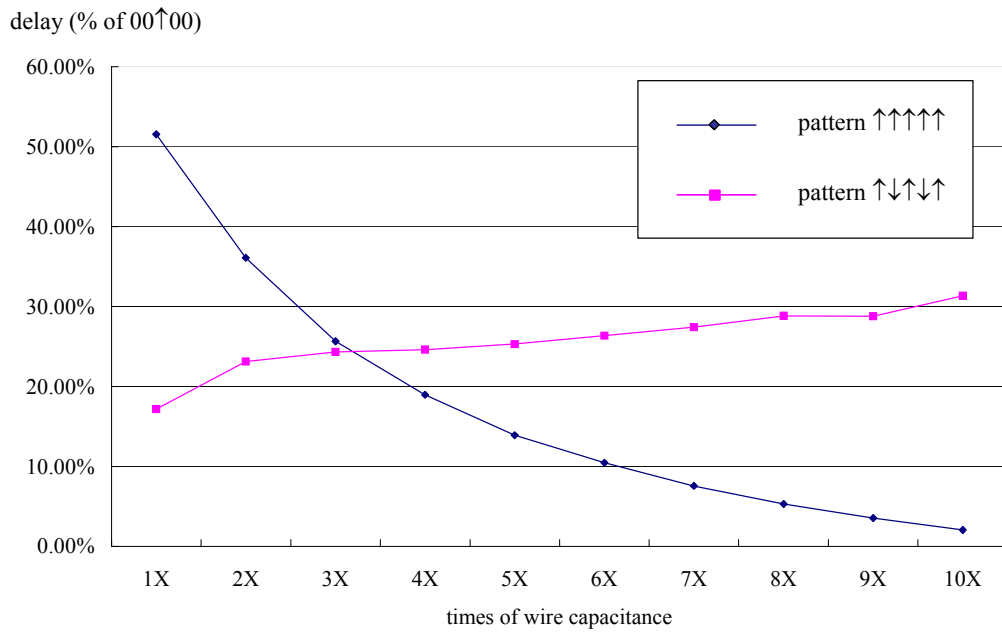


Figure 9: The delays (% of that of pattern 00↑00) of the worst-case switching pattern with various wire capacitances.

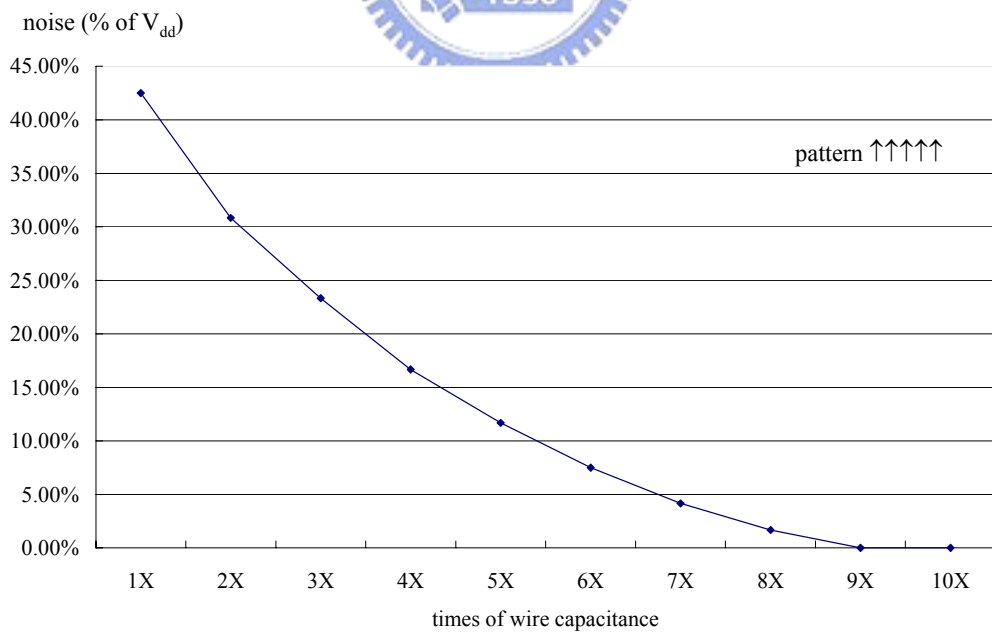


Figure 10: The inductive noise is less significant as wire capacitance increases.

Since Cao et al. [18] claimed that the worst-case switching pattern for a 5-bit bus should be $\uparrow\downarrow\uparrow\downarrow\uparrow$ considering capacitive and inductive coupling, we also conducted simulations to see whether the worst-case switching pattern will change or not when capacitance effects become dominant. We simulated with the extracted *RLC* circuit model of the 5-bit bus by increasing the wire capacitance step by step. The simulation results are shown in **Figures 9** and **10**, and the complete switching patterns when capacitance effects dominate (10X of wire capacitance) are listed in **Table 3**.

Table 3: Simulation results of the 5-bit bus when wire capacitance becomes dominant (10X wire capacitance).

Switching pattern	50% delay of the central wire (ps)	Delay comparison with that of 00 \uparrow 00
$\uparrow\uparrow\uparrow\uparrow\uparrow$	443	2.07%
$\uparrow\downarrow\uparrow\uparrow\uparrow$	480	10.60%
$\uparrow\downarrow\uparrow\uparrow\downarrow$	436	0.46%
$\uparrow\downarrow\uparrow\downarrow\uparrow$	570	31.34%
$\uparrow\downarrow\uparrow\downarrow\downarrow$	530	22.12%
$\downarrow\uparrow\uparrow\uparrow\uparrow$	405	-6.68%
$\downarrow\uparrow\uparrow\uparrow\downarrow$	347	-20.05%
$\downarrow\uparrow\uparrow\downarrow\downarrow$	348	-19.82%
$\downarrow\downarrow\uparrow\uparrow\uparrow$	433	-0.23%
$\downarrow\downarrow\uparrow\downarrow\downarrow$	358	-17.51%
00 \uparrow 00	434	0.00%

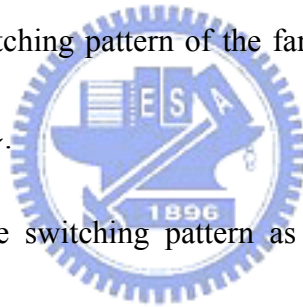
From **Figure 9** and **Table 3**, we observe that the worst-case switching pattern for the 5-bit bus changes from $\uparrow\uparrow\uparrow\uparrow\uparrow$ to $\uparrow\downarrow\uparrow\downarrow\uparrow$. From **Table 3**, while considering

the worst (best) case switching pattern as wire capacitance dominates, we should first consider the immediate neighbors for the worst (the best) case capacitive coupling and then consider farther neighbors for the worst (the best) inductive coupling. For example, if the central wire switches from “0” to “1”, the best-case switching pattern for the 5-bit bus as wire capacitance dominates is as follows:

(1). The best-case switching pattern of the immediate neighbors due to capacitive coupling is $*\uparrow\uparrow\uparrow*$, where $*$ denotes that the bus signal can switch from “0” to “1” or from “1” to “0”.

(2). The best-case switching pattern of the farther neighbors due to inductive coupling is $\downarrow*\uparrow*\downarrow$.

By (1) and (2), the best-case switching pattern as wire capacitance dominates is $\downarrow\uparrow\uparrow\uparrow\downarrow$.



To further investigate the change of the worst-case switching pattern when capacitance effects dominate, we also conducted simulations with varying signal rise times. As shown in **Figure 11**, the worst-case switching pattern for the 5-bit bus also changes from $\uparrow\uparrow\uparrow\uparrow\uparrow$ to $\uparrow\downarrow\uparrow\downarrow\uparrow$ when we increase the signal rise time (i.e., decrease the working frequency). This phenomenon also conforms to the trend when capacitance effects dominate since the impedance of wire capacitance will increase as the working frequency decreases. We should note that the frequency of interest here is

583.3 MHz as the rise time is set to 600 ps, for which the capacitance effects dominate. (See [19] for the formula to determine whether the inductance effects are significant.) Finally, from **Figure 10**, we can also observe that the inductive noise will be less significant as wire capacitance becomes dominant.

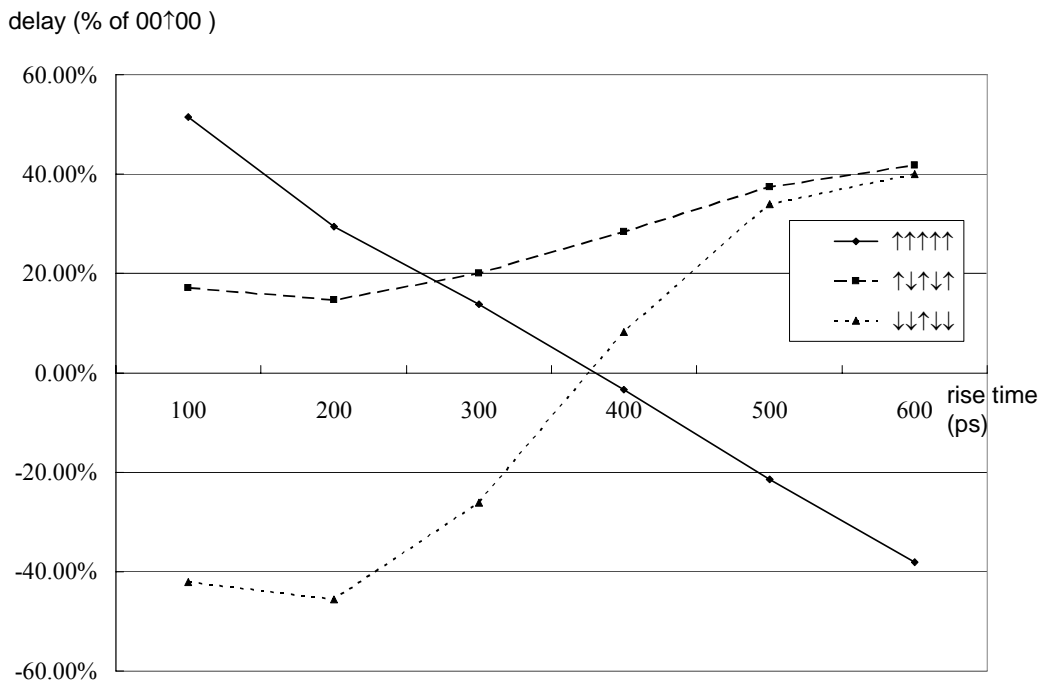


Figure 11: The delays (% of that of the pattern 00↑00) of the worst-case switching pattern with various signal rise times.

1.2 Bus Encoding

1.2.1 Basic Concept

In this dissertation, we consider the coplanar bus structure as shown in **Figure 3** to build our encoding scheme. In the coplanar bus structure, we assume that each driver (receiver) has a uniform size and the driver is also assumed to be symmetric such that the effective output resistance is the same for both rising and falling signal transitions. In the bus structure, each signal wire has a uniform width, pitch, length and height. Given the parameters of wires (length, width, height and pitch), delay constraint, working frequency and the number of data bits, we will generate a valid code set that map the data patterns. The valid code set is obtained at the cost of $m - n$ extra bus wires. The valid code set is defined as that any transition between codes within this set is guaranteed to meet the delay constraint. The overall bus structure is shown in **Figure 12**. The valid code set of the global bus contains only 2^n out of 2^m possible codes. The specific 2^n codes are selected to minimize the coupling effects when a transition occurs between any two of them.

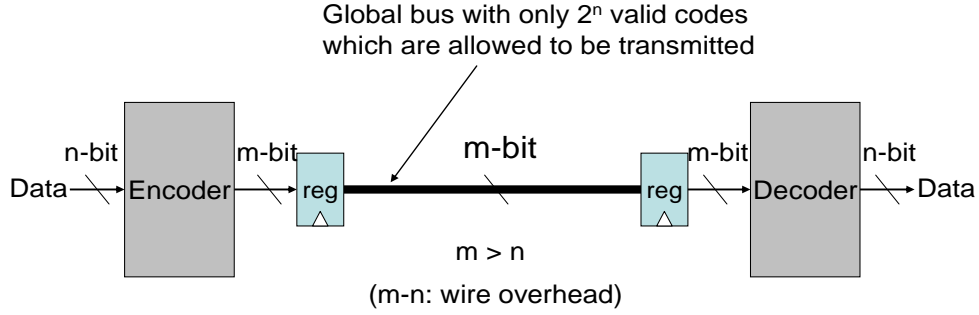


Figure 12: The overall bus structure including the encoder and decoder.

In this dissertation, ramp signal is given to the input of the driver and the transition time τ_r can be calculated from the working frequency f by Equation (2) [15].

An example is given here to demonstrate the delay reduction by using bus encoding. Given a 2-bit bus (4 data patterns: 00, 01, 10, and 11) with 1000 μm signal wire length as shown in **Figure 13**, all the transition delays obtained by HSPICE simulations are listed in **Table 4**. If the delay constraint of the bus is set to 30 ps, the constraint is obviously not met when the transition pattern $\uparrow\downarrow$ or $\downarrow\uparrow$ occurs on the bus. To reduce the worst-case delay of the bus, a bus encoding is introduced. The new bus structure with the encoding scheme which is generated by using our method is shown in **Figure 14**. Four codes (000, 001, 100, and 101) are chosen as a valid code set to represent the original 4 data patterns. All the transition delays of the encoded bus are listed in **Table 5**. From **Table 5**, it is observed that all transition delays of the encoded bus meet the delay constraint. Therefore, the worst-case delay of the original 2-bit bus is successfully reduced to meet the delay constraint by using the bus

encoding technique with one extra bus wire.



Figure 13: .A 2-bit bus example.

Table 4: The transition delays of the 2-bit bus shown in **Figure 13**. (↑: transit from “0” to “1”; ↓: transit from “1” to “0”; -: no transition)

Transition Pattern	Transition delay (ps)
- ↑, ↑ -, ↓ -, - ↓	27.21
↑ ↓, ↓ ↑	38.73
↑ ↑, ↓ ↓	11.97

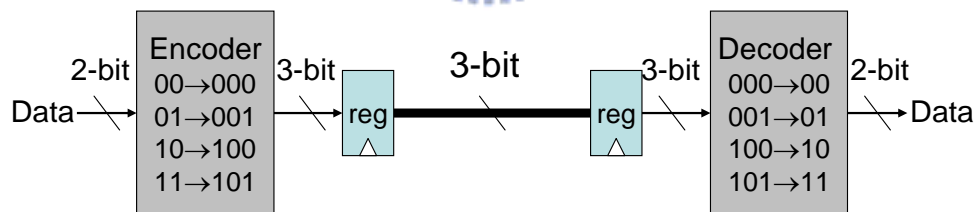


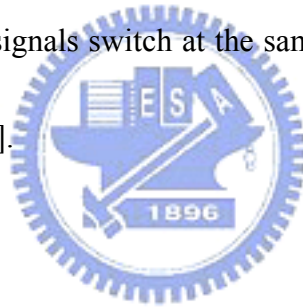
Figure 14: .The encoded bus after encoding the 2-bit bus in **Figure 13** to a 3-bit one.

Table 5: The transition delays of the 3-bit bus shown in **Figure 14**.

Transition Pattern	Transition delay (ps)
-- ↑, -- ↓	24.61
↑ --, ↓ --	24.58
↑ - ↓, ↓ - ↑	29.24
↑ - ↑, ↓ - ↓	19.69

Since transistors mainly operate in the linear region during signal transitions, we assume that all drivers' output resistances are linear throughout our simulation. Therefore, the drivers are modeled as simple linear resistances. In addition, the receivers are replaced by equivalent gate capacitances. Thus, the built circuit model for the coplanar bus structure contains only linear *RLC* elements. In other words, the built circuit is an *Linear Time Invariant (LTI)* system.

In our simulation, we assume that synchronous registers are located at the transmitter side. Thus all the signals switch at the same time on the bus, which is not an uncommon assumption [13].



1.2.2 Previous Works

As the process technology advances, the coupling capacitance between neighboring wires becomes the dominant component of the total wire capacitance due to the higher wire aspect ratio and smaller wire pitch. Moreover, when the clock frequency increases over the GHz range, the inductance effects of on-chip interconnect structures are becoming increasingly significant. Considering these

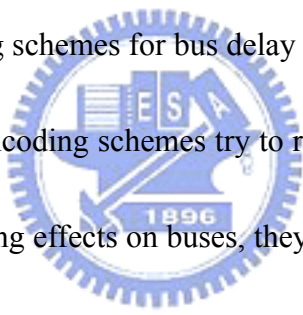
issues in DSM designs, recently many bus encoding schemes have been proposed to mitigate the effects. A large portion of the bus encoding schemes are proposed to reduce the bus power consumption [20 – 27]. Some of them focus on minimizing the bus coupling delay [9, 28 – 32].

Most proposed bus encoding schemes for reducing the coupling delay considers only capacitance effects on the bus structure. There is not much work in the literature considering inductance effects on the bus structure to develop encoding schemes to reduce bus delay. Considering only the capacitive coupling effect, Sotiriadis and Chandrakasan [28], Victor and Keutzer [29], Baek et al. [30], and Hirose and Yasurra [9] proposed their bus encoding techniques to eliminate the crosstalk delay. Besides, Cha et al. [31] also proposed a software-based bus encoding technique to reduce the capacitive coupling delay. They all try to avoid the simultaneous inverse transitions of neighboring wires.

1.2.3 Our Contribution and Proposed Bus Encoding Schemes

While considering the *RLC* circuit model for the bus structure, our works (as

described in subsection 1.1.2) and [32] point out that the worst-case switching pattern could be very different from that of only considering *RC* effects. First, when the inductance effect dominates, the worst-case switching pattern with the largest on-chip bus delay is when all wires simultaneously switch in the same direction. Furthermore, we indicate that while considering the *RLC* effects of interconnects, the worst-case switching pattern varies from different levels of interconnect and different working frequencies. Hence, as inductance cannot be neglected in today's high-speed circuit design, it is very important to consider the *RLC* effects with different bus parameters to develop the proper encoding schemes for bus delay reduction.



Since most previous encoding schemes try to reduce the worst-case delay only based on the capacitive coupling effects on buses, they might gain no improvement on the worst-case delay due to the significant inductive coupling [9, 28 – 31]. Although the bus encoding scheme proposed in work [32] consider the inductance effects to build their encoding schemes, they neglect the capacitive coupling and thus the applications of their bus encoding schemes are limited to the cases in which delay mainly depends on the inductance coupling effects. However, as pointed out before, the worst-case switching pattern varies with given design parameters considering the *RLC* effects of interconnects. Therefore, while developing bus encoding schemes, both capacitance and inductance should be considered.

Most existing works consider only **RC** effects (the worst-case switching pattern resulting from coupling capacitance) to develop their encoding schemes to reduce the bus delay and/or the bus power consumption. Therefore, their works may not be suitable for the very deep-submicron designs when the inductive coupling becomes significant.

In this dissertation, we first show that the worst-case switching pattern that incurs the longest bus delay while considering the **RLC** effect is quite different from that while considering **RC** effect alone. It implies that the existing encoding schemes based on the **RC** model may not improve or possibly worsen the delay when the inductance effects become dominant.

Then we propose a *bus-invert* method to reduce the worst-case on-chip bus delay with the dominance of the inductance coupling effect. Simulation results have shown that our encoding method can significantly reduce the worst coupling delay of the buses with strong inductive coupling.

From our simulation results, we observe that the worst-case switching pattern could vary with given design parameters considering the **RLC** effects of interconnects. However, the proposed *bus-invert* method can be utilized to reduce the bus coupling delay only when the inductance effects dominate. Therefore, we propose a flexible encoding scheme for on-chip buses that can consider the given parameters to reduce

the *LC* coupling delay. In addition, with some modification, this new encoding scheme can be utilized to predict and lengthen the signal propagation length of a bus under the given constraints. Simulation results are also given to support our claims.

To further reduce the coupling noise, we propose a joint shield insertion and bus encoding scheme for global bus design in nanometer technologies. With the user-given bus parameters, the working frequency, the scheme can effectively reduce the *LC* coupling effects and hence, minimize the bus coupling delay. Simulation results show that the proposed scheme can significantly reduce the coupling delay.

Recently, the power consumption of on-chip interconnect is another crucial issue in high performance circuit designs. Therefore, we also utilize the joint shield insertion and bus encoding method to minimize the power consumption of buses. With the user-given bus parameters, the working frequency, and the delay constraint, the scheme can minimize the bus power consumption subject to the delay constraint by effectively reducing the *LC* coupling effects. Simulation results show that the proposed scheme can significantly reduce the coupling delay and the power consumption of a bus according to the delay constraint.

1.3 Organization

The remainder of this dissertation is organized as follows. Chapter 2 describes our proposed bus-invert encoding scheme for reducing the bus coupling delay when the inductance effects dominate. In Chapter 3 we present a flexible encoding scheme for on-chip buses that can consider the given parameters to reduce the *LC* coupling delay, and we also modify this scheme to be capable of predicting and lengthening the signal propagation length of a bus under the given constraints. Chapter 4 proposes a joint shield insertion and bus encoding scheme for global bus design in nanometer technologies. In Chapter 5, we utilize the joint shield insertion and bus encoding method to minimize the power consumption of buses. Finally, Chapter 6 concludes our work.

Chapter 2

BUS-INVERT CODING SCHEME

2.1 Motivation



From the prediction and simulation results in Chapter 1, we can conclude that as the inductance of wires becomes more significant than the capacitance, the worst-case switching pattern with the maximum delay is when all wires simultaneously switch in the same direction. Therefore, if the worst-case switching pattern that all bus wires simultaneously transit in the same direction can be avoided by bus encoding, the performance (speed) of buses can be enhanced. In addition, the inductive noise can also be alleviated by removing the worst-case pattern.

2.2 The Bus-invert Scheme

Inspired by Stan's low-power *bus-invert* method [20] for reducing the transition activities to reduce the bus transition power, we propose a *bus-invert* method to reduce the on-chip bus delay due to coupling effects while inductance effects dominate. Our bus-invert method inverts the input data when the number of bits switching in the same direction is more than half of the number of signal bits. The remaining problem is how to implement the coding architecture with low complexity. For the implementation, we propose an encoder architecture shown in **Figure 15**.

There are three types of possible signal transitions: type I: \uparrow (switching from "0" to "1"), type II: \downarrow (switching from "1" to "0"), and type III: 0 (no switching). If we refer to $x_i(n)$ as an input signal and to $x_i(n-1)$ as its previous input signal, then type I is $(x_i(n), x_i(n-1)) = (1, 0)$, type II is $(x_i(n), x_i(n-1)) = (0, 1)$, and type III is $(x_i(n), x_i(n-1)) = (0, 0)$ or $(1, 1)$. With the input $x_i(n)$ and $x_i(n-1)$, the codeword generator generates $(q_L, q_H) = (0, 1)$ for type I, $(1, 0)$ for type II, and $(0, 0)$ for type III. Then all q_L 's are inputs to the majority voter (L) and all q_H 's to the majority voter (H). Finally, from the output of the majority voter L or H, we can detect if the number of type I or II transitions is more than half of the number of signal bits. If one of the majority voters' outputs is high, the input signal should be inverted. The majority voters can be

implemented by using either a tree of full-adders or resistors combined with a voltage comparator [19].

Since the additional *invert* line will contribute to transitions, it should also be considered. Let N be the total number of signal bits of a bus excluding the *invert* line. The output of the majority voter is asserted when $\lceil (N+1)/2 \rceil$ inputs are high. If N is odd, the example encoder architecture is just as that shown in **Figure 15(b)**. Hence, after encoding, the worst-case switching pattern occurs when $(N+1)/2$ signal bits switching in the same direction, where N is odd. If N is even, the encoder architecture is somewhat different as that shown in **Figure 15(a)**. The major differences are that we need an extra input $INV(n-1)$ for our encoder and $INV(n) = INV(n-1)'$ or $INV(n-1)$ depending on if INV_t is high or low. Hence, after encoding, the worst-case switching pattern is that $N/2$ signal bits switch in the same direction, where N is even.

The circuitry of the receiver is relatively simple because it only needs to conditionally invert the receiving data to get a correct data value. If N is odd, the receiving data need to be inverted only when the *invert* line is high. If N is even, the receiving data need to be inverted only when the *invert* line has a transition.

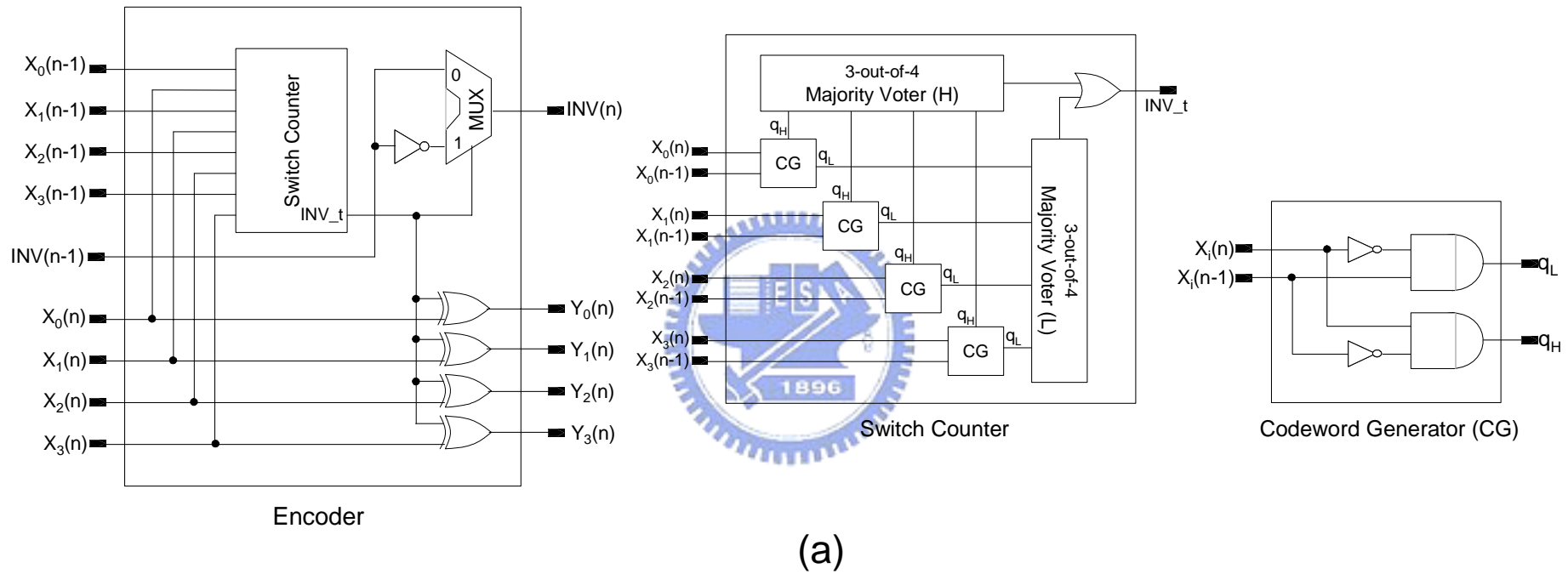


Figure 15: (a) A 4-bit bus encoder for the *bus-invert* scheme.

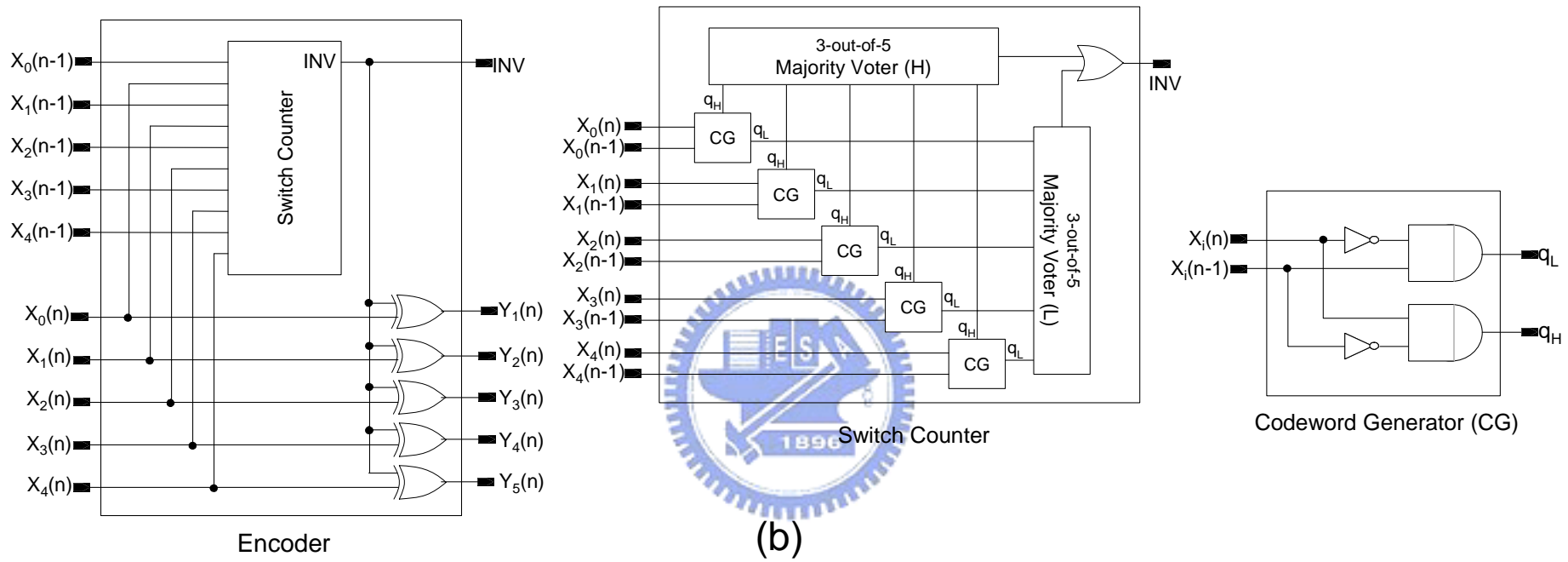


Figure 15: (b) A 5-bit bus encoder for the bus-invert scheme.

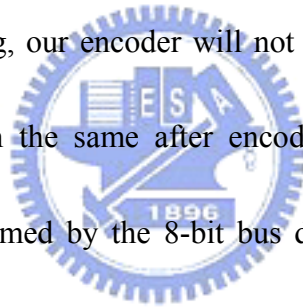
For today's high performance circuits, there are typically only 14 to 16 FO4 (fanout-of-4 inverter [33]) delays per clock [34]. Hence, the delay overhead introduced by our encoder should be minimized. Let d_{OR2} and d_{XOR2} be the delay of a two-input OR gate and that of a two-input XOR gate, respectively. For an N -bit bus, the critical path delay $D(N)$ of our N -bit bus encoder is given by

$$D(N) = d_{\text{Codeword Generator}} + d_{\lceil (N+1)/2 \rceil\text{-out-of-}N \text{ Majority Voter}} + d_{OR2} + d_{XOR2} \quad (8)$$

where $d_{\text{Codeword Generator}}$ equals the delay of an inverter, d_{INV} , plus d_{OR2} , (i.e., $d_{\text{Codeword Generator}} = d_{INV} + d_{OR2}$), and the delay $d_{\lceil (N+1)/2 \rceil\text{-out-of-}N \text{ Majority Voter}}$ is given by $\log_{3/2} N * d_{\text{full-adder}}$ (delay of a full adder) since we use a full-adder tree to implement the majority voter. Therefore, for a typical 8-bit bus encoder with optimized logic and the full-adder circuit implemented as a mirror-type adder, the critical path of the encoder has a delay of 10 FO4, which is about two-thirds of the clock cycle time. This delay overhead is similar to that of the low-power *bus-invert* method. Nevertheless, this delay overhead is the 'worst-case' scenario. Since the encoding logic could be fused with the logic of the IP block, this delay overhead could be reduced with the simultaneous optimization of encoding and the IP block logic.

Like the *bus-invert* method, our method can also reduce the bus transitions. The reduction of the bus transition count occurs when there are $\lceil (N+1)/2 \rceil$ bits transit

in the same direction. For this case, our encoder will invert the current data and the transition count will be reduced. Take an 8-bit bus as an example. For the transition pattern (00↑↑↑↑↑↑) before encoding, the transition count is 5. After encoding, the transition pattern changes to (↑↑000000(↑)) or (↓↓000000(↑)), and the transition count is reduced to 3 (the additional transition (↑) is due to the signal transit on the invert line). Therefore, our encoding scheme can also reduce the average power consumed by the bus in terms of the average transition count. However, the peak power dissipation after encoding will remain the same. For the transition pattern (↑↓↑↓↑↓↑↓) before encoding, our encoder will not invert the current data (i.e., the transition pattern will remain the same after encoding, and this transition pattern causes the peak power consumed by the 8-bit bus due to the coupling capacitance between wires). Since the oppositely switching signals are good for reducing the inductively coupling delay, our encoder will keep these transitions unless there are $\lceil (N+1)/2 \rceil$ bits transit in the same direction. However, the oppositely switching signals are the worst for the power consumption when considering the capacitive coupling effects. To analyze the peak power and the average power consumed by the buses together with the power consumed by the encoder circuit, we need more sophisticated analysis. However, this is beyond the scope of this dissertation. In this dissertation, we focus on the study of the worst-case pattern that causes the longest delay on the bus



due to the strongly inductive coupling and the development of a modified *bus-invert* method to reduce the worst-case delay.

2.3 Simulation Results

2.3.1 Bus Coupling Delay Reduction

With the parameters given in Chapter 1, we conducted our simulations by varying bus signal bits with or without using the proposed *bus-invert* method. The simulation results are shown in **Figures 16** and **17**.

From **Figure 16**, we observe that coupling inductance has greater impacts on bus delay as the number of bus bit lines increases. For a tight *LC* cross-coupled bus, as shown in **Figure 16**, the increase (%) of the worst-case switching delay grows about linearly with the number of bus bit lines. Hence, for a high-frequency, tight *LC* cross-coupled bus, the delay due to signals simultaneously switching in the same direction should be considered.

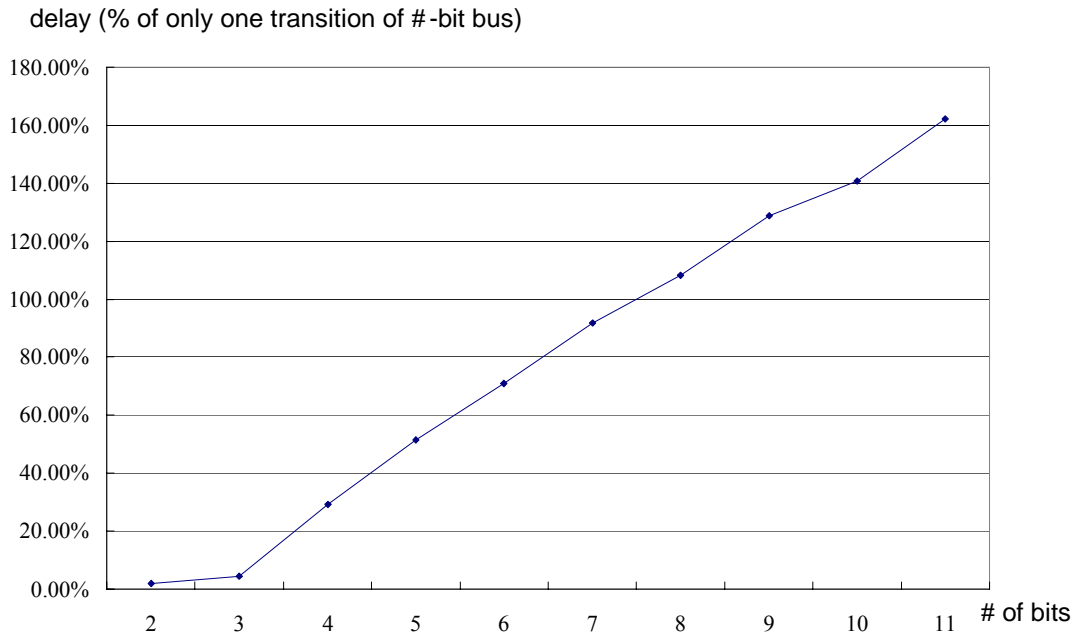


Figure 16: The worst-case #-bit bus delay (% of the delay of only one transition pattern of the #-bit bus) with # varying from 2 to 11.

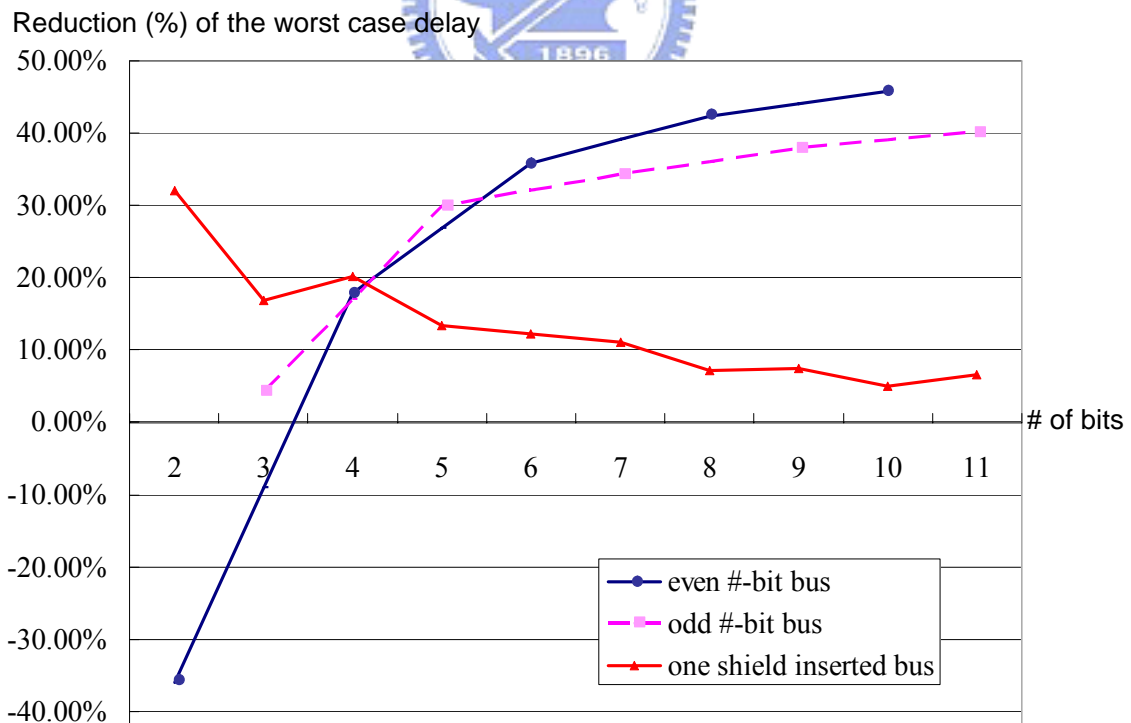
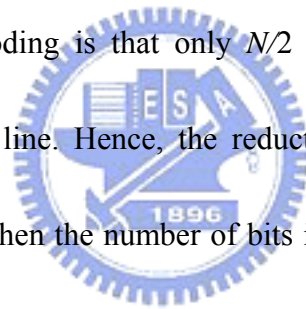


Figure 17: The reduction of worst-case delay for a #-bit bus by using the *bus-invert* method and the shield insertion technique with the # varying from 2 to 11.

As shown in **Figure 17**, our encoding method can significantly reduce the worst-case switching delay; in other words, the bus performance can be improved. Besides, our encoding method can obtain even better reduction rate as the number of a bus bit lines increases. However, since the encoder architectures for even-bit and odd-bit buses are slightly different, the delay reductions are also a little different. For an N -bit bus, if N is odd, the worst-case switching pattern after encoding is $(N+1)/2$ signal bits switching in the same direction including the *INV* line. For N is even, the worst-case pattern after encoding is that only $N/2$ signal bits switch in the same direction, including the *INV* line. Hence, the reduction curve of even-bit buses is above that of odd-bit buses when the number of bits is larger than 5 (see **Figure 17**).

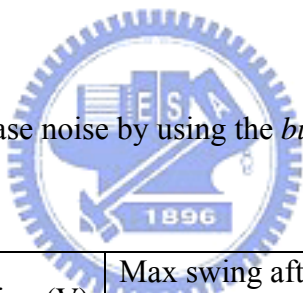
With one wire overhead, while the inductive coupling delay decreases with increasing the bus width by using our method, the delay of the one-shield-inserted bus increases with increasing the bus width. This is because the inductive coupling of the one-shield-inserted bus increases with increasing the bus width (increasing return path), but our method could effectively reduces the worst-case coupling when the bus width increases. We should also note that for the 2-bit bus, our encoding method will worsen the worst-case delay because the additional *INV* line will introduce large additional coupling to the victim line. In other words, the delay of the worst-case after



encoding for 2 bit lines plus one *INV* line will be larger than the worst-case for only 2 bit lines.

In addition to reducing the worst-case delay, our method has the side effects of decreasing the maximum ground-bounce and eliminating the maximum inductive noise. For example, as shown in **Table 6**, the average reduction of maximum inductive noise is about 17%. Since the ground-bounce and the inductive noise are also the worst when all signal wires switch in the same direction, our method can also reduce these effects.

Table 6: Reduction of worst-case noise by using the *bus-invert* method for bus widths ranging from 2 to 11.



#-bit bus	Max swing (V)	Max swing after encoding (V)	Noise reduction (%)
2	1.59	1.33	19.55%
3	1.63	1.44	13.19%
4	1.67	1.41	18.44%
5	1.71	1.47	16.33%
6	1.72	1.47	17.01%
7	1.74	1.49	16.78%
8	1.74	1.46	19.18%
9	1.76	1.51	16.56%
10	1.75	1.49	17.45%
11	1.76	1.52	15.79%
Average noise reduction			17.03%

Since previous works of encoding for bus delay reduction such as [9, 28 – 31] only consider coupling capacitance, the real worst-case pattern due to coupling LC might happen with their encoding. Besides, they might use more than one additional line in their encoding scheme when the number of bit lines is more than 8. Therefore, their encoding schemes might not be suitable for the high-performance bus applications when inductance effects become significant.

2.3.2 Delay Overhead of the Bus Encoder

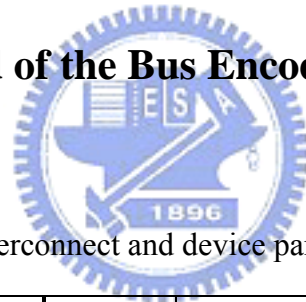


Table 7: Interconnect and device parameters used.

Technology (nm)	180	150	130	100	70
Across chip clock (MHz)	1200	1400	1600	2000	2500
Projected clock period (ps)	833	714	625	500	400
Area (mm ²)	340	385	430	520	620
Vdd (V)	1.65	1.35	1.35	1.05	0.75
Metal resistivity ρ ($\mu\Omega$ -cm)	2.2	2.2	2.2	2.2	1.8
Dielectric constant	2.75	2.25	1.75	1.75	1.5
4X min. wire width (nm)	720	600	520	400	280
4X min. wire spacing (nm)	960	840	680	560	400
Metal aspect ratio	1.8:1	2.0:1	2.1:1	2.4:1	2.7:1
Via aspect ratio	2.2:1	2.4:1	2.5:1	2.7:1	2.9:1
FO4 (ps)	51.1	48.7	45.8	39.2	21.9
Buffer input cap. (fF)	0.60	0.55	0.425	0.35	0.21
Buffer R_d (K Ω)	3.72	4.52	4.50	4.78	4.84

To investigate the delay introduced by our bus encoder and the coupling delay reduction by using our encoding method, we conduct the following simulations to show the delay reduction considering the delay overhead of the encoder for different technology nodes.

The parameters used are adopted from the 1997 National Technology Roadmap for Semiconductors (NTRS'97) [1] and the simulation results in [35]. These parameters are shown in **Table 7**. We consider a typical 8-bit bus with the total routing length of half perimeter of a chip and 4 times of the minimum wire width and spacing. The length of each wire segment between two buffers is 3000 μm . In addition, the buffers are sized to maintain equal input and output transition times, which is a classical design criterion for a buffer sizing. The simulation results are listed in **Table 8**. Column 2 shows the half perimeter of a chip according to the chip area reported in **Table 7**, assuming that chips are of the square shape and thus the half perimeter of a chip is $2\sqrt{Area}$. Column 3 lists the number of required wire segments for signals passing through the half perimeter of a chip (i.e., the half perimeter of a chip $2\sqrt{Area}$ (mm) divided by the length of a wire segment which is 3 mm). Column 4 shows the delay overhead induced from our 8-bit bus encoder. The delay gains (the worst-case delay of the bus before encoding minus that after encoding) of

the signals passing through one wire segment (3000 μm) and through half perimeter of a chip are shown in Columns 5 and 6, respectively. The overall delay gains $((\text{Column 6} - \text{Column 4}) / \text{Column 4} * 100\%)$ are given in Column 7. Finally, the noise reduction is shown in Column 8.

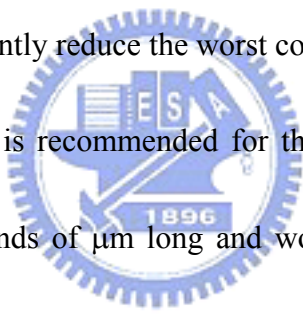
Columns 2 and 3 in **Table 8** reveal the increase of the chip size as the technology advances. Since the intrinsic gate delay decreases as the feature size shrinks, the delay overhead of our encoder decreases as well (see Column 4). We report the worst-case delay overhead derived in previous section (see Equation (8)) in our simulations. From Columns 5-7, we observe that the overall delay gain tends to increase as the technology advances although the delay gain of each wire segment decreases. For example, as shown in our simulations, the delay gain for signals passing through the half perimeter of a chip is only about 20% for the 0.18- μm process while this gain increases to about 167% for the 0.07- μm process. The reasons are two-fold: (1) the decrease of the intrinsic gate delay and (2) the increase of the chip size. To further improve the overall delay gain by reducing the delay overhead, designers can also use dynamic logic to implement the encoding circuit. In addition to the coupling delay reduction, our method can also reduce the maximum inductive coupling noise for long interconnects by about 30%. The simulation results are shown in Column 8.

Table 8: The simulation results of the coupling delay reduction by using our encoding method, and the delay overhead of the encoder for different technology nodes.

	Half perimeter (mm)	# of req. wire segments	Delay overhead (10FO4) (ps)	Delay gain – 1 wire seg. (ps)	Delay gain – full chip (ps)	Overall delay gain ((full chip-overhead)/overhead) (%)	Noise reduction (%)
180nm	18.44 X 2	12	511	51.324	615.888	20.53	32.57
150nm	19.62 X 2	13	487	43.594	566.722	16.37	31.97
130nm	20.74 X 2	14	458	34.464	482.496	5.35	31.68
100nm	22.80 X 2	15	392	38.333	574.995	46.68	31.19
70nm	24.90 X 2	17	219	34.442	585.514	167.36	30.34

2.4 Summary

In DSM, the inductance effect has changed the worst-case switching pattern with the maximum bus delay. For a 5-bit bus structure, the worst-case switching pattern is $*\downarrow\uparrow\downarrow*$ or $*\uparrow\downarrow\uparrow*$ considering **RC** effects, but the worst-case pattern changes to $\uparrow\uparrow\uparrow\uparrow$ or $\downarrow\downarrow\downarrow\downarrow$ considering **RLC** effects. Hence, in this chapter, we have also proposed a *bus-invert* method to reduce the worst-case on-chip bus delay with the dominance of the inductance coupling effect. Simulation results have shown that our encoding method can significantly reduce the worst coupling delay of a bus.

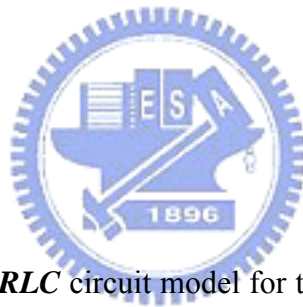


Our encoding scheme is recommended for the cases when buses or parallel signal wires are about thousands of μm long and work above GHz frequencies. At such working frequencies, the gate delay overhead of our encoder should be small enough. If we choose the full-adder tree to implement the majority voter, the delay of majority voter is $O(\log_{1.5} N) * (\text{full-adder delay})$, where N is the total number of signal bits of a bus. In other words, if N is very large, our encoder may cause timing violations. To solve this problem, we can divide the original bus into sub-buses by inserting ground wires between sub-buses. Hence, the overall problem is a gate delay (and thus process) dependent optimization problem. Therefore, we shall solve this problem in our future work.

Chapter 3

FLEXIBLE BUS ENCODING SCHEME

3.1 Motivation



While considering the *RLC* circuit model for the bus structure, as discussed in Chapter 1, the worst-case switching pattern could be very different from that of only considering *RC* effects. First, when the inductance effect dominates, the worst-case switching pattern with the largest on-chip bus delay is when all wires simultaneously switch in the same direction. Furthermore, while considering the *RLC* effects of interconnects, the worst-case switching pattern varies from different levels of interconnect and different working frequencies. Hence, as inductance cannot be neglected in today's high-speed circuit design, it is very important to consider the *RLC* effects with different bus parameters to develop the proper encoding schemes for

bus delay reduction.

Since most previous encoding schemes [9, 28 – 31] try to reduce the worst-case delay only based on the capacitive coupling effects on buses, they might gain no improvement on the worst-case delay due to the significant inductive coupling. Although our proposed bus-invert encoding scheme and [32] consider the inductance effects to build encoding schemes, the capacitive coupling is neglected and thus the applications of their bus encoding schemes are limited to the cases in which delay mainly depends on the inductance coupling effects. However, as discussed in Chapter 1, the worst-case switching pattern varies with given design parameters considering the *RLC* effects of interconnects. Based on this fact, in this chapter, we propose a flexible encoding scheme for on-chip buses considering the given parameters to reduce the *LC* coupling delay. By extending the conventional one to the new problem which is how to increase the signal propagation length on a bus by using bus encoding methods considering *RLC* effects under given parameters and constraints. In this chapter, we also propose a flexible signal propagation length increasing flow which incorporates the flexible encoding scheme to solve the extended problem.

3.2 Proposed Flexible Encoding Scheme

3.2.1 Flexible Bus Encoding Flow

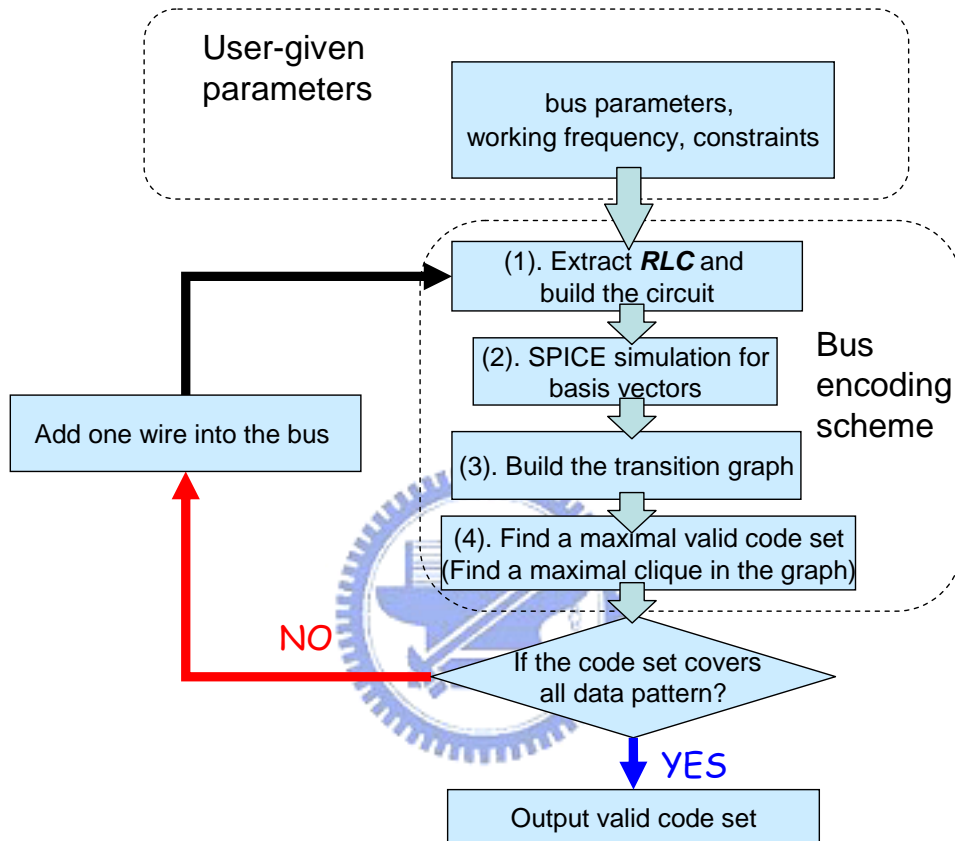


Figure 18: Our flexible bus encoding flow.

Figure 18 illustrates our overall encoding flow. At first, users should give the bus parameters (bus width n , wire length, wire height, wire width, wire pitch, Power/Ground wire width, Power/Ground-to-signal pitch), the working frequency, and the delay constraint. With the given parameters, we build the 3D bus structure and then extract the resistances, capacitances, and inductances of bus wires. After

extraction, the equivalent *RLC* circuit will be built. Next, the built circuit will be simulated by using HSPICE with the *basis vectors* which will be defined later. By applying superposition theorem [36] of linear circuits, we can establish the transition graph efficiently. From the transition graph, we apply a greedy algorithm to find a valid code set in which every transition between a code pair meets the delay constraint. Next, we will check whether the code set covers all data patterns. If so, the code set will be output to map to the data patterns. Otherwise, we add one more bit line to the bus structure and redo from the Step (1). The details of each step will be described in the following.



3.2.1.1 Extract *RLC* from the Bus

In Step (1) of our bus encoding scheme, with the given parameters, we use FastCap [11] and FastHenry [12] to extract the *RLC* parameters of the given bus structure and construct the SPICE model. The detailed flow of Step (1) is shown in **Figure 19**. FastCap can extract the self and coupling capacitances of wires, while FastHenry is developed to extract the resistances, self inductances, and coupling inductances of wires. With these extracted *RLC* parameters, the equivalent *RLC* circuit model can be built. The circuit model is constructed as π -segments using series resistances, inductances and shunt capacitances. Finally, the equivalent circuit model

is specified in SPICE format.

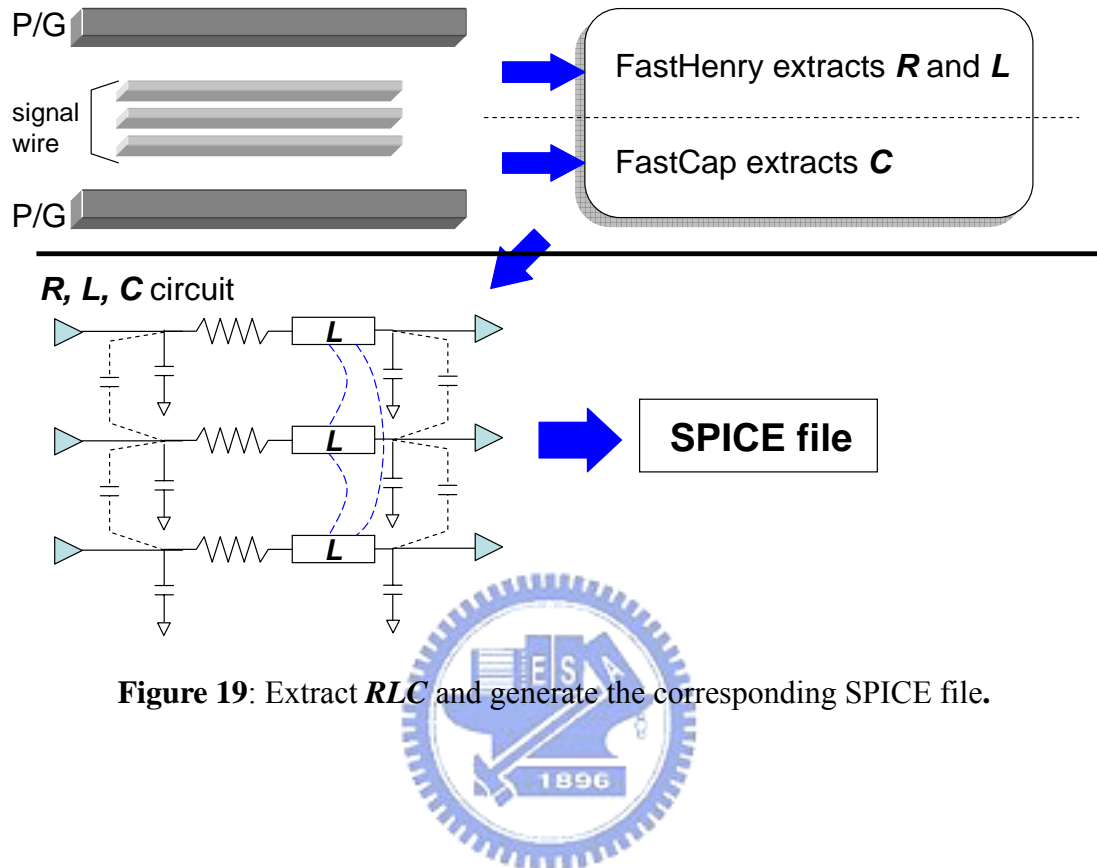


Figure 19: Extract *RLC* and generate the corresponding SPICE file.

3.2.1.2 Simulate the *Basis Vectors* by HSPICE

After building the equivalent *RLC* circuit model of the bus, one can easily obtain the transition delay for a specific input pattern pair by simply conducting HSPICE simulation. However, for an n -bit bus, there are 2^n input patterns and 4^n possible transition patterns in total. It is extremely time-consuming to simulate all possible transition patterns by HSPICE when n goes higher. The complexity of HSPICE simulation time is $4^n \cdot (\text{HSPICE simulation time for a transition pattern})$. Therefore, we develop another method based on the superposition theorem [36] to

significantly reduce the simulation time. The superposition theorem states that, for an *LTI* circuit, the resulting effects (current and voltage differences) of the independent current (or voltage) sources in the circuit can be considered separately, and then summed up to obtain the overall results. Based on this idea, we first simulate the *basis vectors* which are independent sources to the built *RLC* circuit. Then we can obtain the real delay of each transition pattern by superposing the simulation results of the *basis vectors*.

What are the *basis vectors* for a bus? We define them as all independent transitions of bus inputs. For example, the six *basis vectors* of a 3-bit bus are -- \uparrow , - \uparrow -, \uparrow --, -- \downarrow , - \downarrow - and \downarrow --. Throughout this paper, “-” represents a stable input (stable at low or high), “ \uparrow ” represents an input changing from low to high, and “ \downarrow ” represents an input changing from high to low. Therefore, for an *n*-bit bus consisting of input signals $b_0, b_1, b_2, \dots, b_{n-1}$, the *basis vectors* can be expressed as:

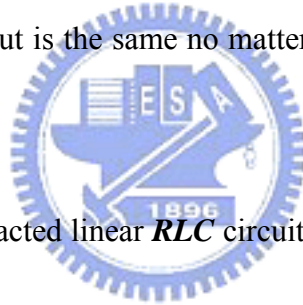
basis vectors of an *n*-bit bus =

$$\begin{aligned} & \{(b_0 \ b_1 \ b_2 \ \dots \ b_{n-1}) \mid b_0 \in \{\uparrow, \downarrow\} \text{ and } b_1, b_2, \dots, b_{n-1} \in \{-\}\} \cup \\ & \{(b_0 \ b_1 \ b_2 \ \dots \ b_{n-1}) \mid b_1 \in \{\uparrow, \downarrow\} \text{ and } b_0, b_2, \dots, b_{n-1} \in \{-\}\} \cup \\ & \quad \bullet \\ & \quad \bullet \\ & \quad \bullet \\ & \{(b_0 \ b_1 \ b_2 \ \dots \ b_{n-1}) \mid b_{n-1} \in \{\uparrow, \downarrow\} \text{ and } b_0, b_1, \dots, b_{n-2} \in \{-\}\} \end{aligned} \quad (9)$$

The followings are some properties of the *basis vectors*.

Property 1: In an extracted linear *RLC* circuit of a bus, given a *basis vector*, such as $(- - \uparrow)$, the transition delay due to the switching input is the same no matter other stable inputs are at '0' or '1'.

Proof: Since coupling effects are due to the voltage change ($I_{induce} = C_{couple} \times (dV/dt)$) and the current change ($V_{induce} = L_{couple} \times (dI/dt)$), stable inputs contribute no noise to neighboring wires. Therefore, the transition delay due to the switching input is the same no matter other stable inputs are at '0' or '1'. \square



Property 2: In an extracted linear *RLC* circuit of a bus, given an integer k and $0 \leq k \leq n-1$, $\{(b_0 \ b_1 \ b_2 \dots b_{n-1}) \mid b_k \in \{\uparrow\} \text{ and } b_0, b_1, \dots, b_{n-1} \in \{-\}\}$ and $\{(b_0 \ b_1 \ b_2 \dots b_{n-1}) \mid b_k \in \{\downarrow\} \text{ and } b_0, b_1, \dots, b_{n-1} \in \{-\}\}$ are called a *dual basis vector pair*. For example, $(- - \uparrow, - - \downarrow)$, $(- \uparrow -, - \downarrow -)$, and $(\downarrow - -, \uparrow - -)$ are *dual basis vector pairs*. The voltage waveforms resulting from a *dual basis vector pair* are equal in magnitude but opposite in direction.

Proof: Since the extracted *RLC* circuit is all composed by linear elements, the built circuit is an *LTI* system. Therefore, if the input signals are with the same magnitude but opposite direction, then the output waveforms should

be equal in magnitude but opposite in direction. \square

Here we also define a *minimum basis vector set* as that all transition patterns can be obtained by superposing the *basis vectors* within this set:

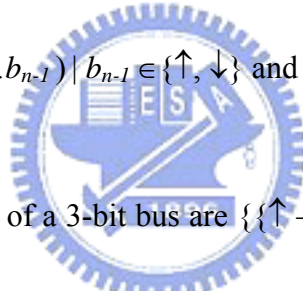
Minimum basis vector sets of an n -bit bus =

$$\{ \{ (b_0 \ b_1 \ b_2 \dots b_{n-1}) \mid b_0 \in \{\uparrow, \downarrow\} \text{ and } b_1, b_2, \dots, b_{n-1} \in \{-\}\} \},$$

$$\{ (b_0 \ b_1 \ b_2 \dots b_{n-1}) \mid b_1 \in \{\uparrow, \downarrow\} \text{ and } b_0, b_2, \dots, b_{n-1} \in \{-\}\} \},$$

-
-
-

$$\{ (b_0 \ b_1 \ b_2 \dots b_{n-1}) \mid b_{n-1} \in \{\uparrow, \downarrow\} \text{ and } b_0, b_1, \dots, b_{n-2} \in \{-\}\} \} \quad (10)$$



Note that the *basis vector sets* of a 3-bit bus are $\{ \{\uparrow \ --, \downarrow \ --\}, \{- \ \uparrow \ -, \ - \ \downarrow \ -\}, \{- \ \uparrow \ -, \ - \ - \ \downarrow \ -\} \}$. Therefore, based on **Property 1** and **2**, there are eight choices of the *minimum basis vector set* for a 3-bit bus (i.e., $(- \ - \ \uparrow, \ - \ \uparrow \ -, \ \uparrow \ --)$, $(- \ - \ \uparrow, \ - \ \uparrow \ -, \ \downarrow \ --)$, $(- \ - \ \uparrow, \ - \ \downarrow \ -, \ \uparrow \ --)$, $(- \ - \ \uparrow, \ - \ \downarrow \ -, \ \downarrow \ --)$, $(- \ - \ \downarrow, \ - \ \uparrow \ -, \ \uparrow \ --)$, $(- \ - \ \downarrow, \ - \ \uparrow \ -, \ \downarrow \ --)$, $(- \ - \ \downarrow, \ - \ \downarrow \ -, \ \uparrow \ --)$ and $(- \ - \ \downarrow, \ - \ \downarrow \ -, \ \downarrow \ --)$). The *minimum basis vector set* for a 3-bit bus can be arbitrary one of the above eight choices. In general, the *minimum basis vector set* of an n -bit bus has n elements and each element can be one of the dual basis vector pairs as shown in Equation (10). Hence, we only need to simulate the basis vectors of the chosen *minimum basis vector set*. Then we can use

the simulation results to obtain the delays of all possible transition patterns by applying the superposition theorem. The details and examples of Step (2) are shown in **Figure 20**. First, we apply one *basis vector* at a time as the input transition pattern on the bus. Second, we perform SPICE simulation and record the voltage waveforms of all signal wires. Then, we repeat this procedure for every *basis vector* until all *basis vectors* of the chosen *minimum basis vector set* are simulated.

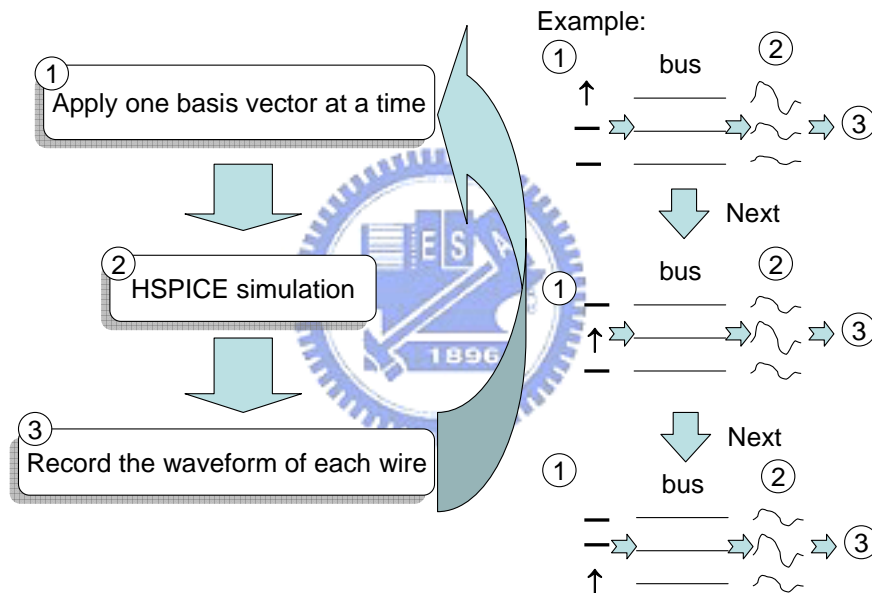


Figure 20: HSPICE simulations with the *basis vectors* of the *minimum basis vector set*.

3.2.1.3 Build the Transition Graph

In Step (3), we apply the superposition theorem to obtain the real delay of each transition pattern by using the simulation results of the *basis vectors* recorded in Step

(2). **Figure 21** illustrates how to obtain the real delay of an arbitrary transition pattern by superposing the simulation results of the *basis vectors*. First, we decompose the given transition pattern into certain *basis vectors*. Next, by looking up the simulation results of the *basis vectors* that have been obtained in Step (2) and superposing them, we can obtain the overall voltage waveform of each wire. Then the real delay of this transition pattern can be calculated. Our simulation results show that the results obtained from superposition exactly comply with those obtained from the real HSPICE simulation. To demonstrate the equivalence of the HSPICE simulation results and the superposition results, we show the results obtained by both methods of a 3-bit bus with input transition pattern $(-\uparrow\uparrow)$ for an example. The results are shown in **Figure 22**. From **Figure 22**, we can observe that the voltage waveforms obtained from these two methods are exactly the same.

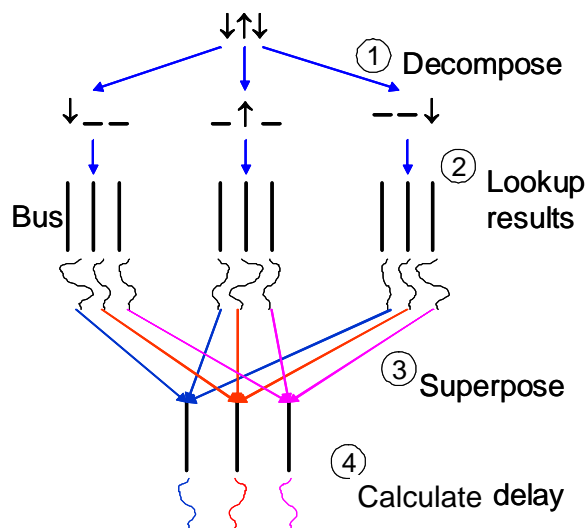


Figure 21: An example of superposition.

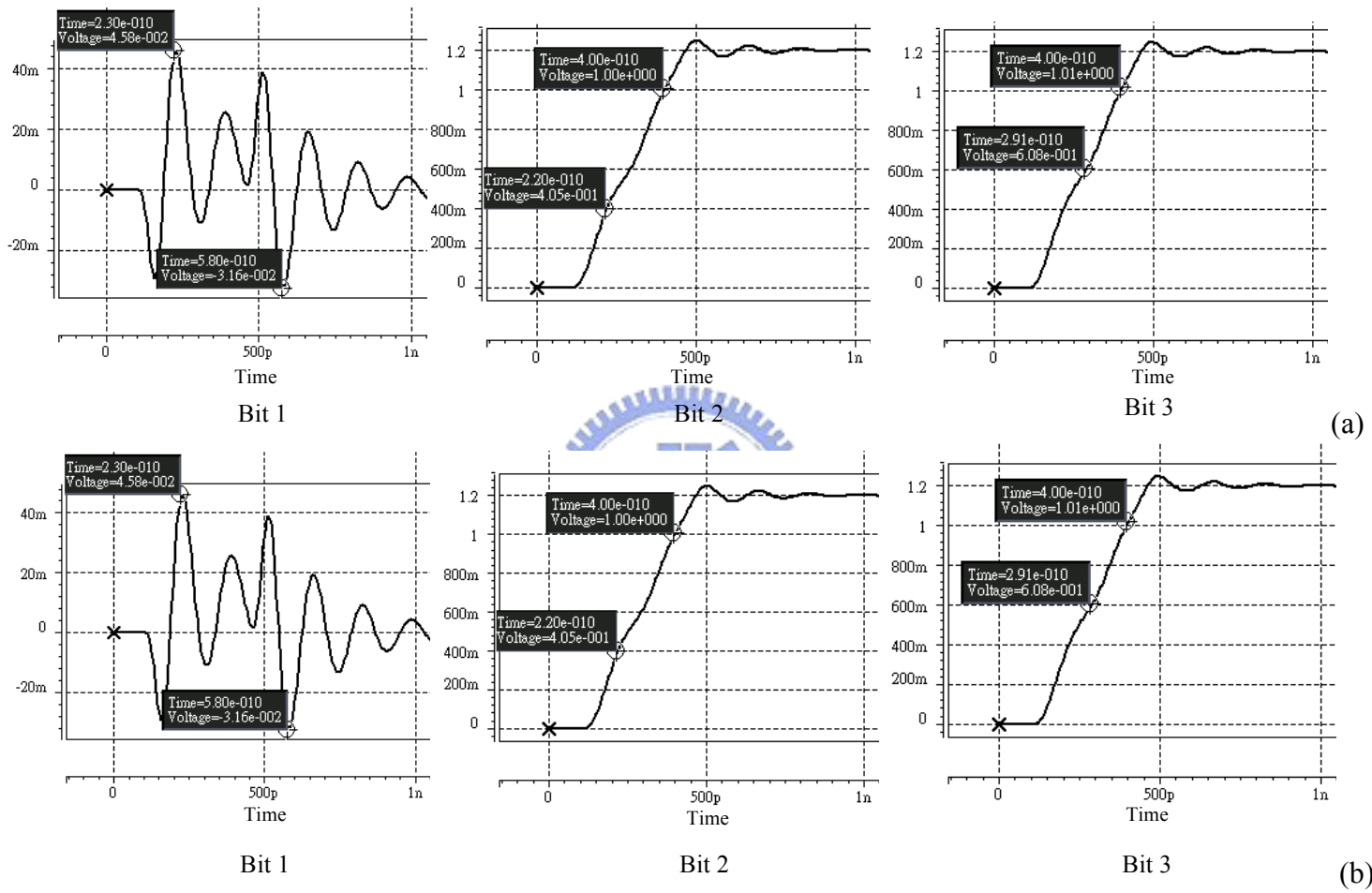


Figure 22: The voltage waveforms of the signal wires of a 3-bit bus obtained (a) by directly conducting HSPICE simulation with $(-\uparrow\uparrow)$ and (b) by using superposing the results of $(--\uparrow)$ and $(-\uparrow-)$.

With the use of the superposition, we can calculate the transition delay between any two codes very fast without performing a real HSPICE simulation run. Then we build a transition graph to indicate if the transition delay between arbitrary two codes meets the delay constraint or not. **Figure 22** illustrates an example of the transition graph of a 3-bit bus where a vertex represents a code and an edge indicates that the transition delay between two corresponding codes meets the delay constraint.

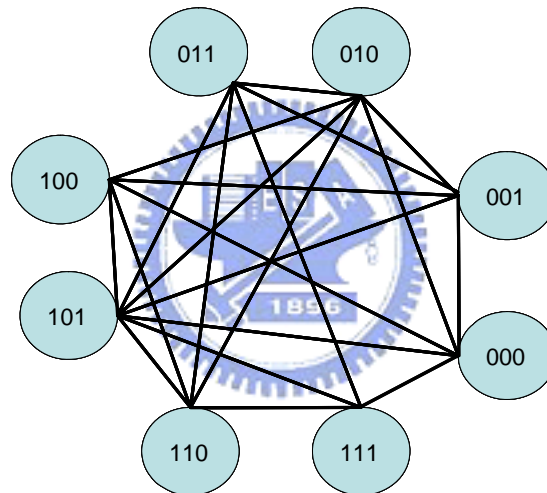


Figure 23: An example of the transition graph of a 3-bit bus.

With the use of the superposition theorem on an n -bit bus, the complexity to obtain all transition patterns' delays can be reduced from $(4^n) \cdot (\text{HSPICE simulation time for a transition pattern})$ to $n \cdot (\text{HSPICE simulation time for a basis vector}) + (4^n/2) \cdot (\text{superposition time})$. The first term, $n \cdot (\text{HSPICE simulation time for a basis vector})$, is due to simulating the *minimum basis vector set*, and the second term,

$(4^n/2)$ *(superposition time), is to superpose the simulation results of the *basis vectors*.

3.2.1.4 Find a Maximum Clique

After building the transition graph, we want to find a maximum clique in the graph. The reason to find a clique in the transition graph is to find a valid code set within which any transition between two codes is guaranteed to meet the delay constraint. Besides, the reason to find a maximum clique is because the number of found codes (vertices) should be greater or equal to the number of required data patterns (e.g., 2^n in **Figure 12**) with a certain wire overhead (e.g., $(m - n)$ in **Figure 12**). If the number of found codes is greater than that of required data patterns, it indicates that we could find a valid code set covering all data patterns with a smaller wire overhead. Therefore, we are always eager to maximize the number of found codes (i.e., minimize m). However, since finding a maximum clique is an NP-complete problem, we use a greedy algorithm to solve this problem within reasonable time. The pseudo code of the greedy algorithm is shown in **Figure 24**.

```

INPUT: A transition graph
OUTPUT: A clique
IF the graph is a clique, output this graph and exit
REPEAT
    Find a vertex  $v$  with the minimum degree in the graph
    Remove  $v$  and update the graph
UNTIL All remaining vertices in the graph form a clique
All vertices in the clique form a set CLIQUE
All vertices that deleted from the graph form a set DELETE
FOR Each  $v \in$  DELETE
    IF  $v$  and CLIQUE can form a larger clique
        THEN Update CLIQUE to the larger one
Output CLIQUE

```

Figure 24: Our greedy algorithm to find a maximum clique.

In the first loop, we first check if this graph is a clique. If it is a clique, we output this graph directly. Otherwise, we delete the vertex which has the fewest edges and update the graph. We repeat the above step until the remaining vertices form a clique. Take **Figure 22** as an example, the edge degree of each vertex in **Figure 22** is $\{5, 5, 6, 5, 5, 7, 5, 4\}$ from vertex (000) to vertex (111) counterclockwise. Obviously, this graph is not a clique. Hence, in the first loop, we first remove (111) since it has smallest edge degree and then update the graph. Next, we pick (000) as the next vertex to be removed. Then (001) and (011) are selected to be removed subsequently. Finally, the remaining vertices (010, 100, 101, 110) form a clique as shown in **Figure 25**.

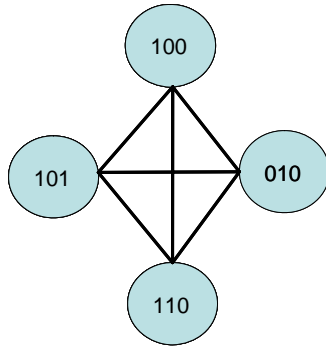


Figure 25: The greedy search result of **Figure 22**.

Since our greedy method is a simple heuristic algorithm, it could not guarantee to find an optimal solution. In order to further improve the result, we add an extra post processing step to improve the outcome. The second loop tries to add vertices removed previously back into the identified clique. Each attempt is checked if the newly added vertex can form a larger clique. If it is the case, the clique is updated to a larger one. By using this method, we could find a larger clique without consuming significant computing power. From our experimental results, the second loop does really enlarge the clique found in the first loop occasionally. To demonstrate the effectiveness of the second loop, an example is shown in **Figure 26**. From **Figure 26**, the extended clique sizes increased by second loop tend to increase with the clique sizes.

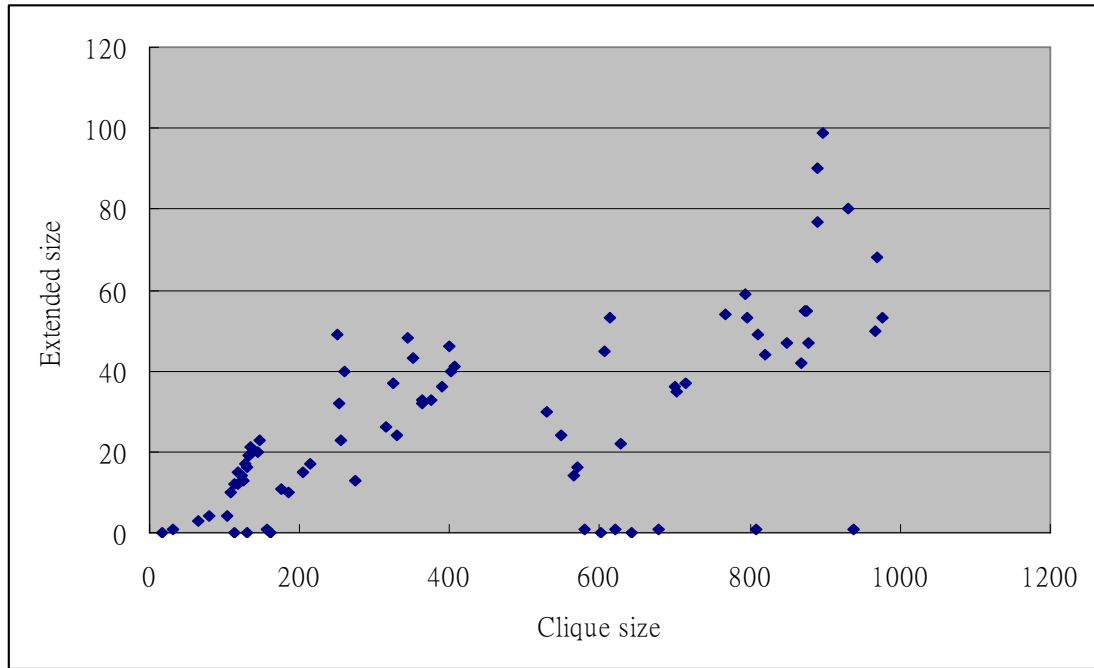


Figure 26: The number of extended clique size by using the second loop in **Figure 24**.



3.2.2 Simulation Results

3.2.2.1 Simulation Time Reduction by Superposition

Here, we compare the computation time of building the transition graph by using our superposition method and pure HSPICE simulations. Our simulation environment is described as follows. The length, width, height and pitch of signal wires are 2000 μ m, 2 μ m, 2 μ m and 4 μ m, respectively. The bus working frequency is 1GHz and the supply voltage is 1.2V. The bit number of the bus width varies from 3

to 18. The simulation results are shown in **Figure 27** and **Table 9**. The speedup ratio is calculated as $((\text{Pure HSPICE}) / (\text{Our Method})) - 1$. Obviously, our approach is much more effective than pure HSPICE simulation especially for the wider bus. **Table 9** illustrates that our method provides a dramatic speedup against pure HSPICE simulation. Moreover, the transition graphs obtained by our method and pure HSPICE simulations are exactly the same. In other words, as a result of the superposition theorem, the transition delays that obtained by our method and pure HSPICE simulation are exactly identical.

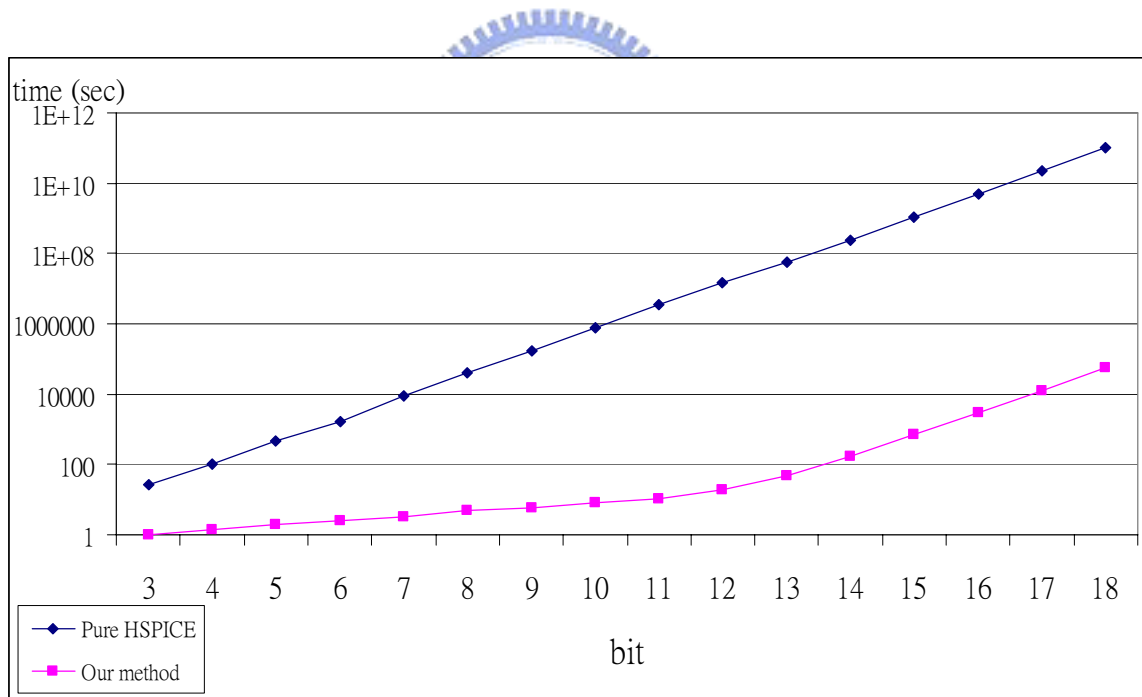


Figure 27: The runtime of our method and pure HSPICE simulation.

Table 9: The runtime improvement of our method.

Bit	Pure HSPICE (sec)	Our Method (sec)	Speedup
3	27	1	27
4	104	1	104
5	464	2	242
6	1729	3	576
7	8700	3	2900
8	39977	5	7995
9	176161	6	29360
10	802161	8	100270
11	3405775	11	309616
12	14931722	19	785880
13	57646516	48	1200969
14	247497488	176	1406236
15	1124207744	718	1565749
16	4964982272	3003	1653341
17	22282291200	12537	1777322
18	98749890560	57362	1721521

3.2.2.2 Crosstalk Delay Reduction by Using Our Encoding Scheme

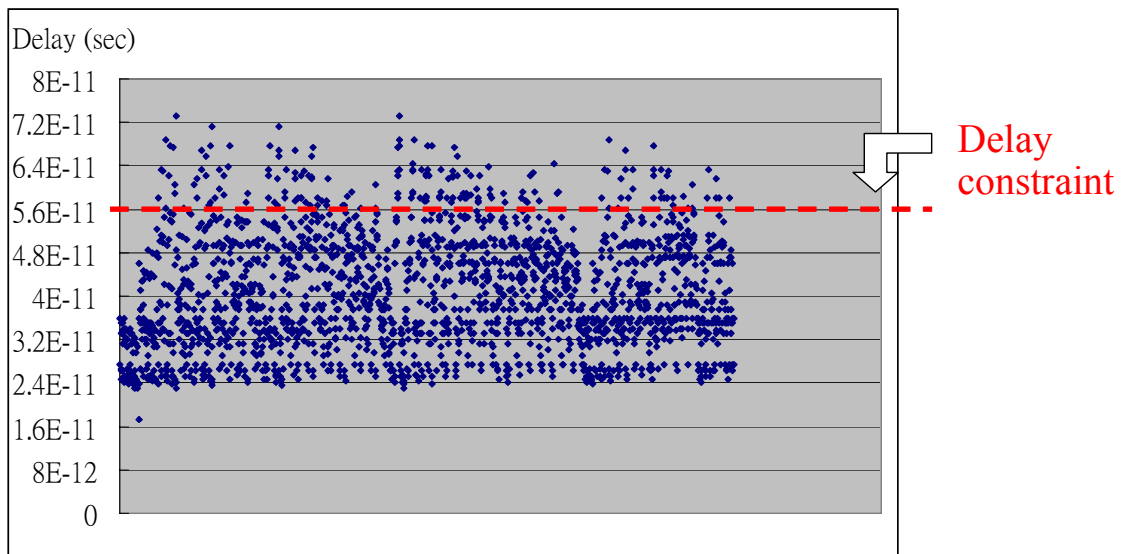


Figure 28: Transition delay of the bus (6 bit) before encoding.

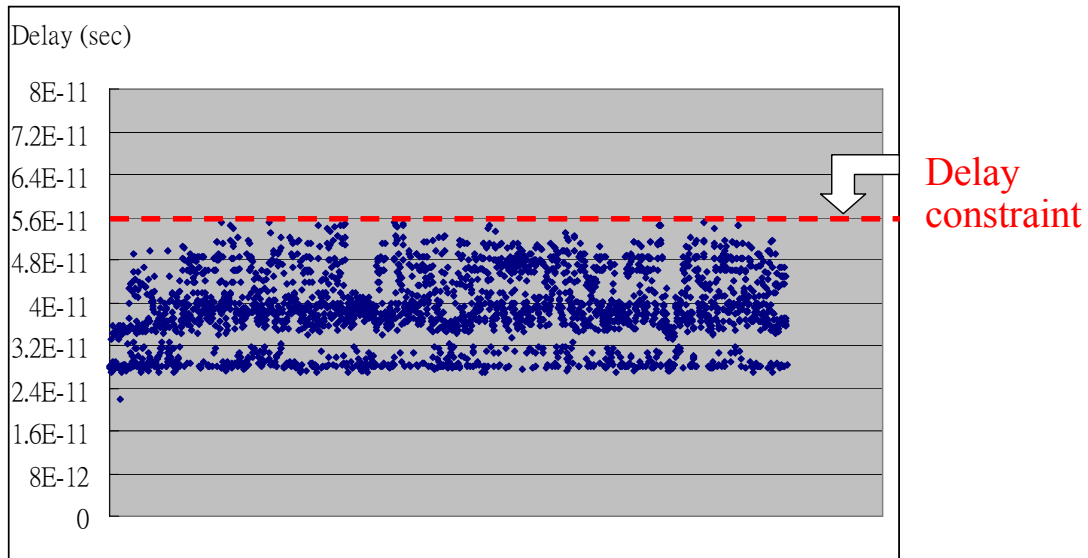


Figure 29: Transition delay of the bus (7 bit) after encoding.

In the following, we give an example to demonstrate the delay reduction using our encoding flow. Given a 6-bit bus (i.e. 64 data patterns) with the wire parameters the same as that given in previous subsection, and the delay constraint is set to 56 picoseconds. The transition delays of all transition patterns on the 6-bit bus are shown in **Figure 28**. Each point on the graph represents a transition delay. Obviously, some transitions violate the delay constraint. By using our encoding flow for the 6-bit bus, the size of the found maximum valid code set is 38, which is smaller than 64 (less than the number of data patterns). Thus, we will add one more bit to the 6-bit bus and redo our encoding flow. After using our encoding flow for the 7-bit bus, a maximum valid code set of size 73 is found. All transition delays between any two patterns in this valid code set meet the delay constraint as shown in **Figure 29**. Therefore, we

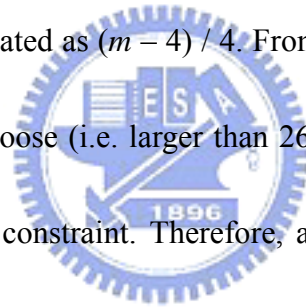
output the valid code set with the 7-bit bus as the final result.

Table 10: The wire overheads versus different delay constraints for a 4-bit data bus.

Delay constraint (ps)	m-bit bus	Wire overhead
30	4	0.00
29	4	0.00
28	4	0.00
27	4	0.00
26	4	0.00
25	5	0.25
24	5	0.25
23	5	0.25
22	5	0.25
21	6	0.50
20	6	0.50
19	6	0.50
18	7	0.75
17	9	1.25

The delay constraint directly decides that the types of the transitions which are allowed to transmit on the bus (i.e. meet the delay constraint). When the delay constraint is tight, few types of transitions meet the delay constraint. Therefore, we need more extra wires to encode the data patterns. On the other hand, if the delay constraint is loose, many types of transitions are allowed to be transmitted on the bus.

Hence, only few extra wires are needed to encode the data patterns. We give an example to demonstrate this phenomenon by using our encoding flow with different delay constraints. Give a bus parameter which the length, width, height and pitch of signal wires are $1000\mu\text{m}$, $2\mu\text{m}$, $2\mu\text{m}$ and $4\mu\text{m}$, respectively. The supply voltage is 1.2V. The number of data bits is 4 (16 data patterns). The simulation results of needed extra wires with different delay constraints are listed in **Table 10**. The second column in **Table 10** (i.e. m -bit bus) represents the number of the bus wires needed to generate the valid code set using our encoding flow. The last column represents the wire overhead ratio which is calculated as $(m - 4) / 4$. From **Table 10**, we can observe that when the delay constraint is loose (i.e. larger than 26 picoseconds), all transitions of the 4-bit bus meet the delay constraint. Therefore, all transitions are allowed to be transmitted on the bus. Thus, no additional wires are needed to encode the data patterns and the wire overhead ratio is 0. However, when the delay constraint is tight (e.g. delay constraint = 25 picoseconds), some transitions of 4-bit bus are invalid. Hence, an additional wire is needed to find a valid code set that can map the 16 data patterns. In other word, we need an additional wire to enlarge the size of generated valid code set that is large or equal to the number of the data patterns. As the delay constraint is tighter than 25 picoseconds, there are only some types of transitions allowed to be transmitted on the bus. Thus, we need more extra wires to encode the



data patterns than the loose delay constraint case. In other words, the wire overhead ratio becomes very large when the delay constraint is very tight. From **Table 10**, we can observe that when the delay constraint is set to 17 picoseconds, 5 extra wires are needed to meet the delay constraint by using our encoding method.

3.2.2.3 Comparisons with Shield Insertion Method

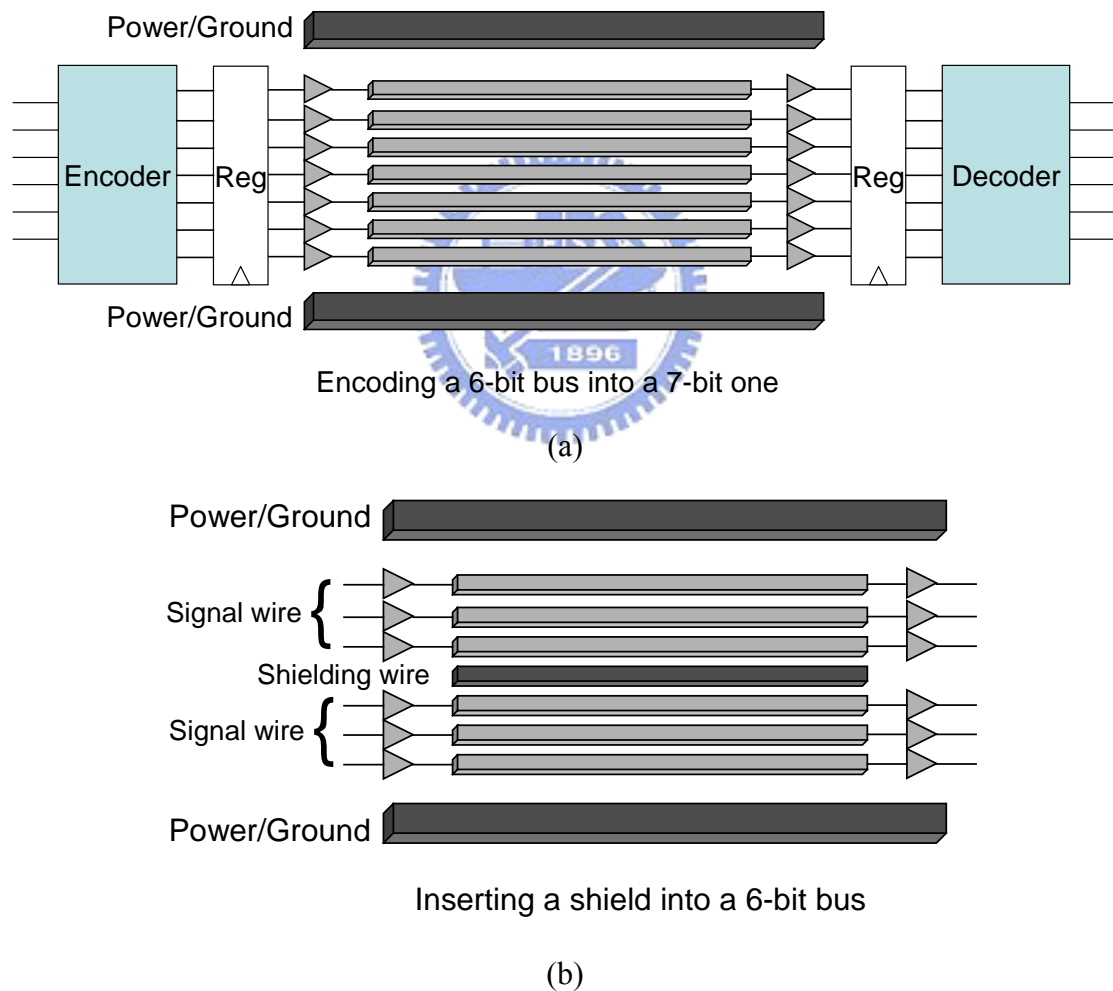


Figure 30: The bus structures after (a) using our encoding method and (b) applying the shield insertion technique.

Here, we conduct some simulations to compare the improvement of the worst-case delay by using our method with that by using the conventional shield insertion technique. Given a bus structure, the number of data bits is 6 (64 data patterns), and the wire overhead is one (only one additional wire is allowed). The bus length, width, height and pitch of signal wires are $2000\mu\text{m}$, $0.8\mu\text{m}$, $2\mu\text{m}$ and $2\mu\text{m}$, respectively. The supply voltage is 1.2V. **Figure 30(a)** and **(b)** illustrates the buses after using our flow and inserting one shield.

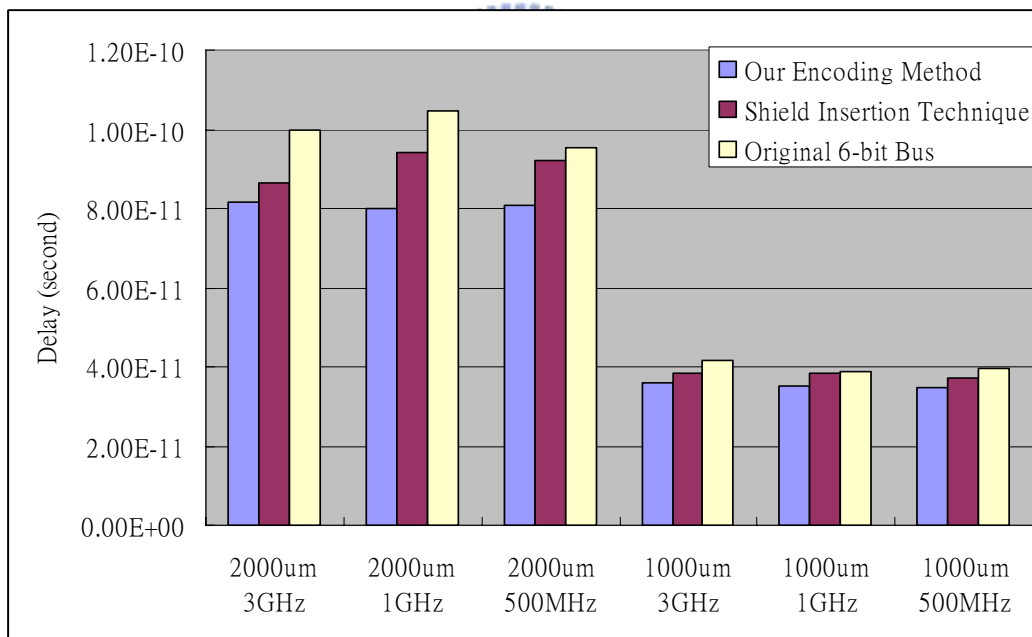


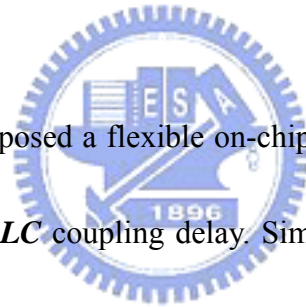
Figure 31: The worst-case transition delay by using our method and the shield insertion technique.

The simulation results under different working frequencies and bus lengths are conducted and the worst-case transition delays are shown in **Figure 31**. Besides, the

worst-case transition delay of the original 6-bit bus is also shown in **Figure 31**.

Although both our flow and the shield insertion technique can reduce the worst-case transition delay of buses, our flow always outperforms the shield insertion technique under different working frequencies and bus lengths as shown in **Figure 31**. Therefore, our flow is effective in reducing the coupling delay under different working frequencies and bus lengths comparing to the conventional shield insertion technique.

3.2.2 Summary of Proposed Flexible Encoding Scheme



In this section, we proposed a flexible on-chip bus encoding flow considering bus parameters to reduce the *LC* coupling delay. Simulation results have shown that our encoding method can significantly reduce the coupling delay of a bus with given delay constraints and parameters. Comparing with the conventional shield insertion technique, our encoding scheme always outperforms the shield insertion technique under different working frequencies and bus lengths.

Based on the superposition theorem, we also proposed a method to speed up our encoding flow. Simulation results show that our method can significantly reduce the run time. In addition, as a result of the superposition theorem, the transition delays that obtained by our method and pure HSPICE simulation are exactly identical.

Moreover, since both the capacitive coupling and inductive coupling effects are considered in our encoding scheme, both capacitance- and inductance-dominated cases of on-chip buses can be also handled very well by using our method.

Although the codec circuit will introduce some delay overheads, these delay overheads can be hidden in a fully pipelined circuit. In addition, since the codec logic could be fused with the logic of the IP block, the delay and area overhead due to the codec could be reduced with the simultaneous optimization of encoding and the IP block logic. Moreover, as processes scale, the circuit codec overheads will be reduced further.

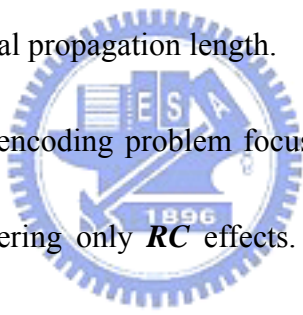


3.3 Flexible Encoding Scheme for Increasing Bus Propagation Length

3.3.1 Motivation

With the increase of chip size and clock frequency, the global interconnect delay is likely to be larger than one clock period. Hence, on-chip signals can no longer reach the entire die in one clock cycle [37]. Recently, due to strong noise coupling

effects in DSM, the one-cycle signal propagation length is further decreased especially for global interconnects. Therefore, there are many works that have been proposed to reduce crosstalk effects and the interconnect delay. Layout techniques such as the shield insertion [8, 38 – 40], the wire sizing [38, 41, 42], the gate sizing [38, 42], the repeater insertion [43 – 45], and the net reordering [8] have been proposed to solve the related problems. In addition, circuit-level solutions like the booster [46], the differential signaling [47, 48], and the bus encoding [9, 28 – 31] have been discussed as well. In this section, we focus on improving the bus encoding techniques to increase the signal propagation length.



The conventional bus encoding problem focuses on how to reduce the delay on a fixed length bus considering only **RC** effects. In this section, we extend the conventional one to a new problem, i.e., how to increase the signal propagation length on a bus by using bus encoding methods considering **RLC** effects under given parameters and constraints. Therefore, the conventional bus encoding problem becomes a sub-problem of our extended problem. Hence, in this section, we propose a flexible signal propagation length increasing flow to solve the extended problem. The proposed flow combines a new bus encoding scheme and a curve-fitting method. The new bus encoding scheme can effectively reduce the **LC** coupling effects on on-chip buses, and hence, improve the worst-case switching delay. To improve the efficiency

of our flow, we adopt the curve-fitting method to reduce the runtime of our flow. Since this work can also estimate the propagation length of a given bus structure, we believe that this work is feasible to be integrated into a floorplan or a routing tool.

3.3.2 Signal Propagation Length Increasing Flow

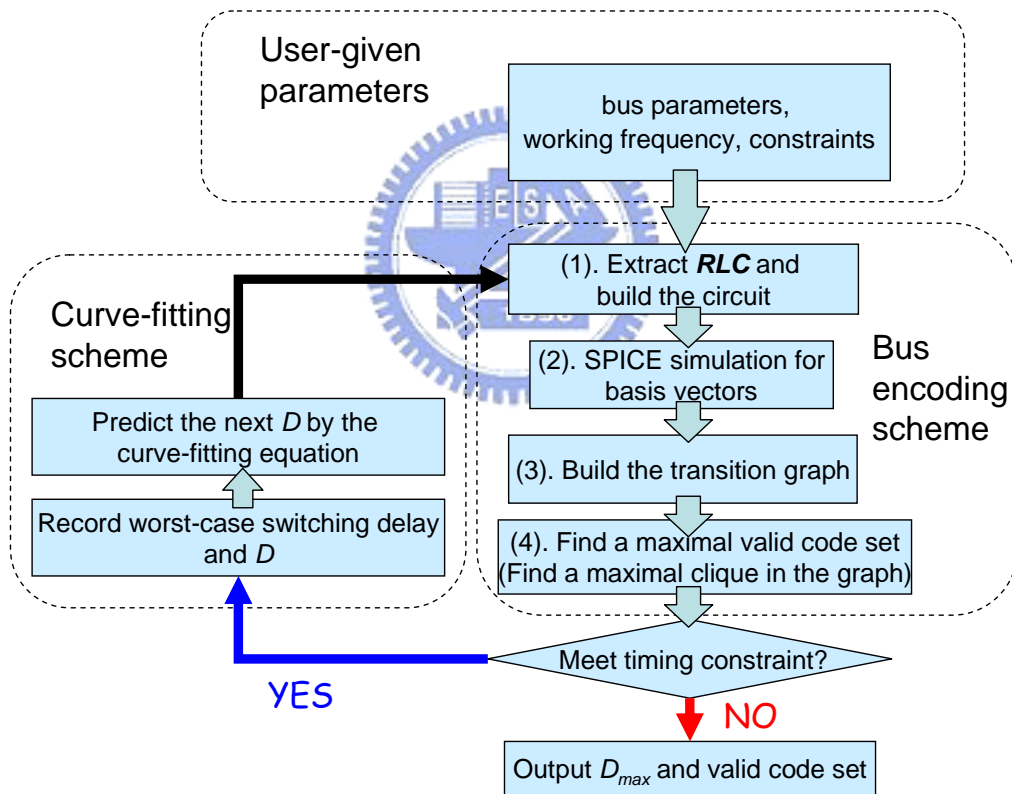


Figure 32: The signal propagation length increasing flow. (D : bus propagation length; D_{max} : maximum bus propagation length)

Our signal propagation length increasing flow is shown in **Figure 32**. It

mainly comprises two parts – the bus encoding scheme and the curve-fitting method. At first, users should give the bus parameters (bus width n , initial wire length D , wire height, wire width, wire pitch, Power/Ground wire width, Power/Ground-to-signal pitch), the working frequency, the delay constraint, and the wire overhead constraint $m - n$. Then with these parameters, we run our encoding scheme and check if the delay constraint can still be met under the wire overhead constraint. If the delay constraint can be met, the bus length is predicted and increased by the curve-fitting method for the next iteration. This process is not terminated until the delay constraint is not met. When this is the case, the last recorded length which still meets the delay constraint is reported as the maximum propagation length D_{max} , and a valid code set is generated at the same time. The details of the bus encoding method are as described in the previous section (see Section 3.2). The details of the curve-fitting method are discussed in the following.

3.3.2.1 Curve-Fitting Method

With a given D , our flow will iteratively increase D when the timing constraint is still met. In order to minimize the runtime, the iterations of our flow should be kept as few as possible. Hence, the maximum propagation length should be successfully predicted within few iterations instead of just incrementally increasing a fixed amount

of predicted length for each iteration. Therefore, we adopt a curve-fitting method to fast predict the maximum propagation length and thus reduce the number of required iterations.

To use the curve-fitting method, we have to find a suitable fitting equation of the interconnect delay with coupling effects. A closed form delay equation is given for a gate driving an *RLC* wire segment with a gate capacitance load [49]. In [49], two extreme cases need to be considered. For one extreme case where $L \rightarrow 0$, the delay reduces to $0.37RCl^2$ where R is the unit length wire resistance, C is the unit length wire capacitance, and l is the wire length. In this case, the wire delay is squarely dependent on the wire length. For the other extreme case where $R \rightarrow 0$, the delay reduces to \sqrt{LC} where L is the unit length wire inductance. In this case, the wire delay is linearly dependent on the wire length. Therefore, it is desirable to use a quadratic equation to fit the delay of a single wire when the *RLC* effects of the wire are considered. Furthermore, we can also use a quadratic equation to fit the worst-case switching delay of an n -bit parallel coupled bus because the switching aggressors only change the effective wire capacitance and inductance of a victim wire. We also use various curve-fitting methods to fit the worst-case delay curve with respect to the wire length. The simulation results of a 7-bit bus are shown in **Figure 33**. From **Figure 33**, we can observe that the quadratic fitting equation best matches the wire delay

comparing to other fitting equations. Therefore, by using the quadratic curve-fitting, the maximum propagation length can be predicted efficiently within few iterations in our flow.

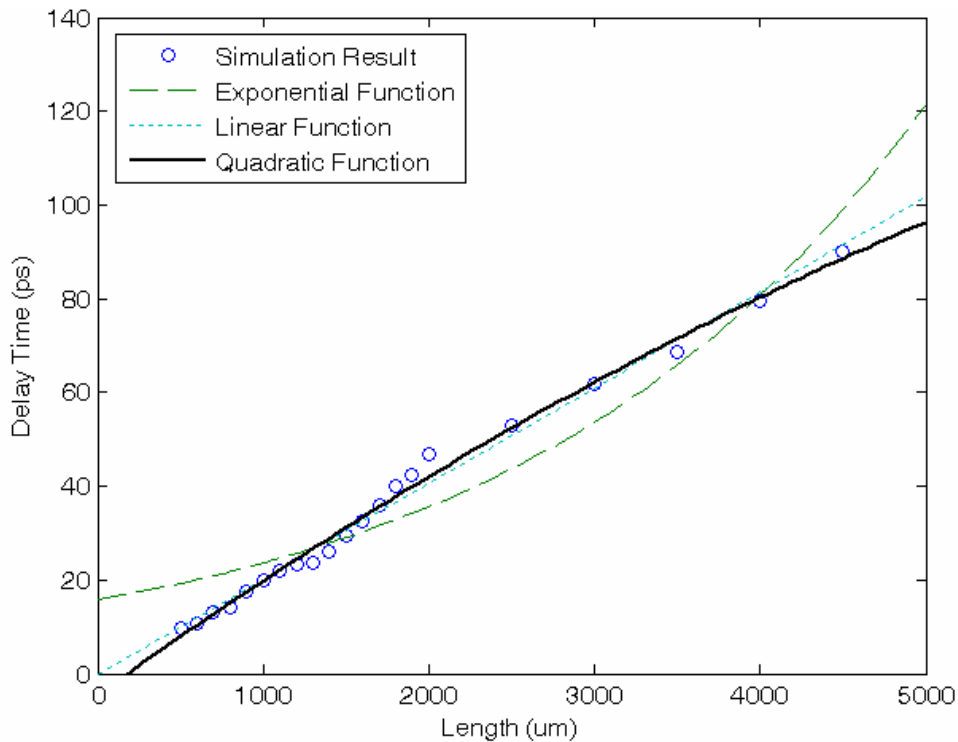


Figure 33: Curve-fitting with different functions.

In the following, we details our curve-fitting method adopted in our flow. First, with the given bus structure, we give two lengths and measure the worst-case switching delay. With the measured delays with respect to the wire lengths, we can obtain a curve-fitting equation. Next, by using the fitting equation and the given delay constraint, the propagation length can be predicted. Then, the ‘real’ worst-case delay

with respect to the predicted length is measured. With the new measured data, the fitting equation will be modified and used to predict the next length with the delay constraint. The procedures of the measuring delay, the modifying equation, and the predicting length will be repeated iteratively until two conditions are satisfied: (i). the measured delay of the j -th (j is integer) predicted length is less than or equal to the delay constraint (ii). the difference between the j -th and the $j+1$ -th predicted length is less than a predefined threshold. Finally, the j -th predicted length that satisfies the above conditions will be outputted as the maximum propagation length D_{max} .



3.3.3 Simulation Results

3.3.3.1 Signal Propagation Increase

With the user-given parameters, the following simulation results show that our flow can increase the signal propagation length of buses by reducing coupling effects. For example, if the delay constraint is set to 20ps, and the width, height and pitch of signal wires are given as $2\mu\text{m}$, $2\mu\text{m}$ and $4\mu\text{m}$, respectively. The bus working frequency is 1GHz and the supply voltage is 1.2V. For a 4-bit data bus (16 data patterns), the simulation results of the signal propagation length with different

numbers of wires are shown in **Figure 34**. From **Figure 34**, we can observe that the improvement of the maximum propagation length is generally better when adding more wires into the bus (more wire overhead). Similar simulation results can be obtained in **Figure 35** for a 6-bit data bus (64 data patterns). To verify whether the estimated maximum propagation length and the valid code set are correct, SPICE simulation is conducted. It is confirmed that all transitions meet the given delay constraint.

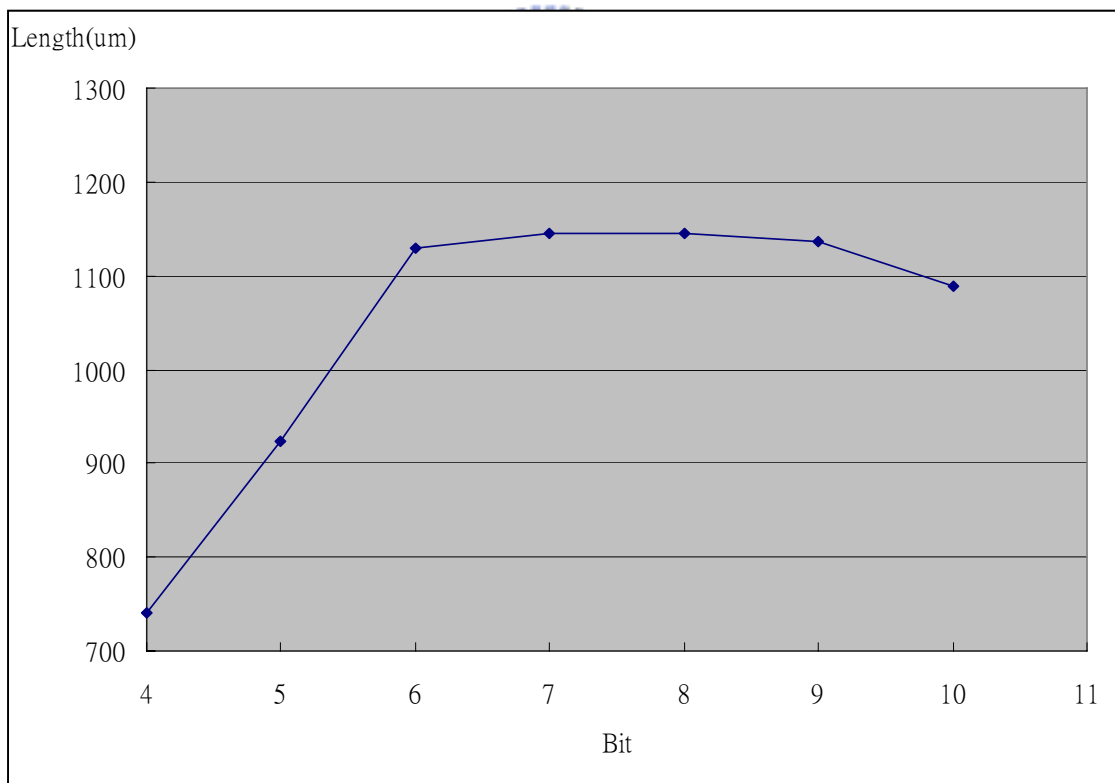


Figure 34: The maximum propagation length vs. different wire overheads for a 4-bit data bus.

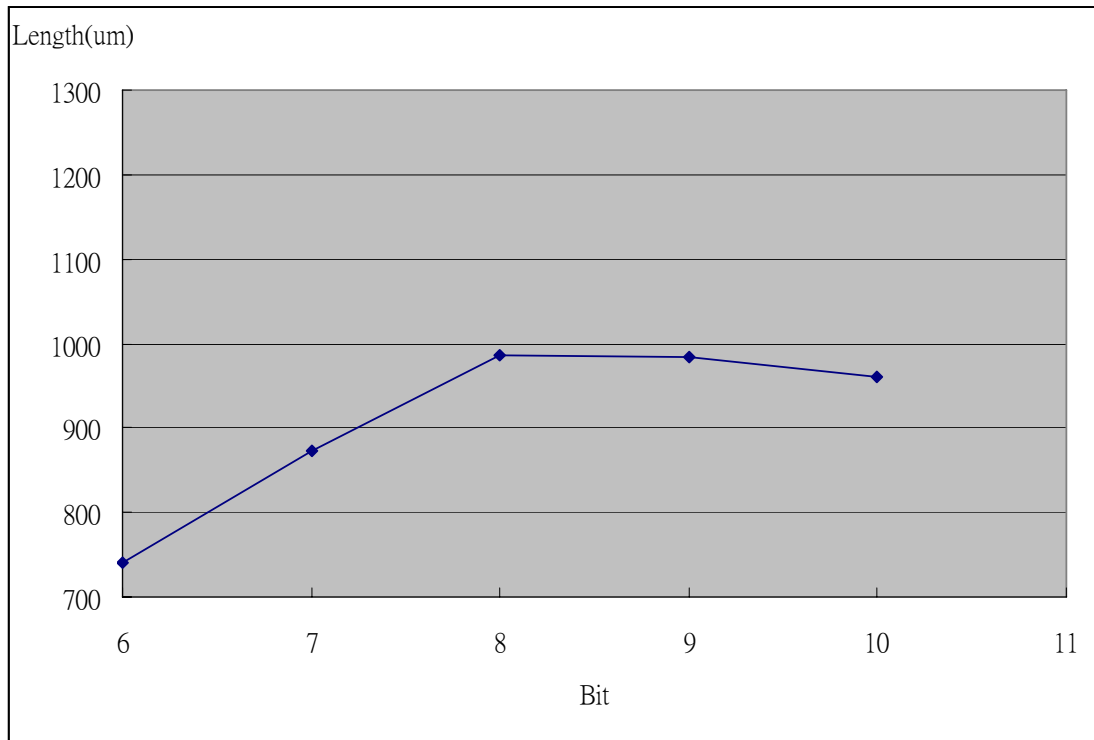
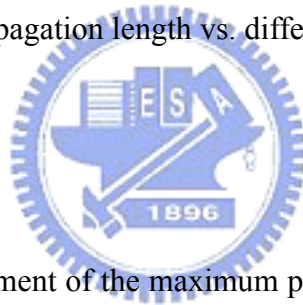


Figure 35: The maximum propagation length vs. different wire overheads for a 6-bit data bus.



However, the improvement of the maximum propagation length saturates after adding a certain number of wires into the bus as shown in **Figures 34** and **35**. After adding a certain number of wires, more added wires may even decrease the improvement of the propagation length. This is because the additional wires increase the bus width (increase the current loop width of each wire). Therefore, the inductance effects on the buses increase with every additional wire as well. Hence, there is an optimal number of extra wires for a given bus structure and working frequency. By using our flow, designers can easily obtain the improvement curve of the maximum propagation length regarding to a different number of additional wires. With this

information, designers can decide the optimal number of additional wires for a bus with a target propagation length.

3.3.3.2 Comparisons with Shield Insertion Method

We also conduct some simulations to compare the improvement of our flow with the conventional shield insertion technique. Given a bus structure, the number of data bit is 6 (64 data patterns), the delay constraint is set to 20 ps, and the wire overhead is one (only one additional wire is allowed). The width, height and pitch of signal wires are given as 2 μ m, 2 μ m and 4 μ m, respectively. The supply voltage is 1.2V.

Figure 30 illustrates the buses after using our flow and inserting one shield. The simulation results under different working frequencies are shown in **Figure 36**.

Although both our flow and the shield insertion technique can increase the propagation length of buses, our method always outperforms the shield insertion technique under several different working frequencies as shown in **Figure 36**.

Therefore, our flow is more effective in increasing the signal propagation length under different working frequencies comparing to the conventional shield insertion technique. Moreover, as the working frequency increases, the increase of propagation length by using our method is more obvious than that by inserting one shield. Since the inductive coupling is a long-range effect and is getting worse as frequency

increases, the signal propagation length could be greatly reduced when a chip works at high frequency due to the strong inductive coupling. By effectively reducing the inductive coupling effect, our method provides better improvement than simply inserting a shielding wire.

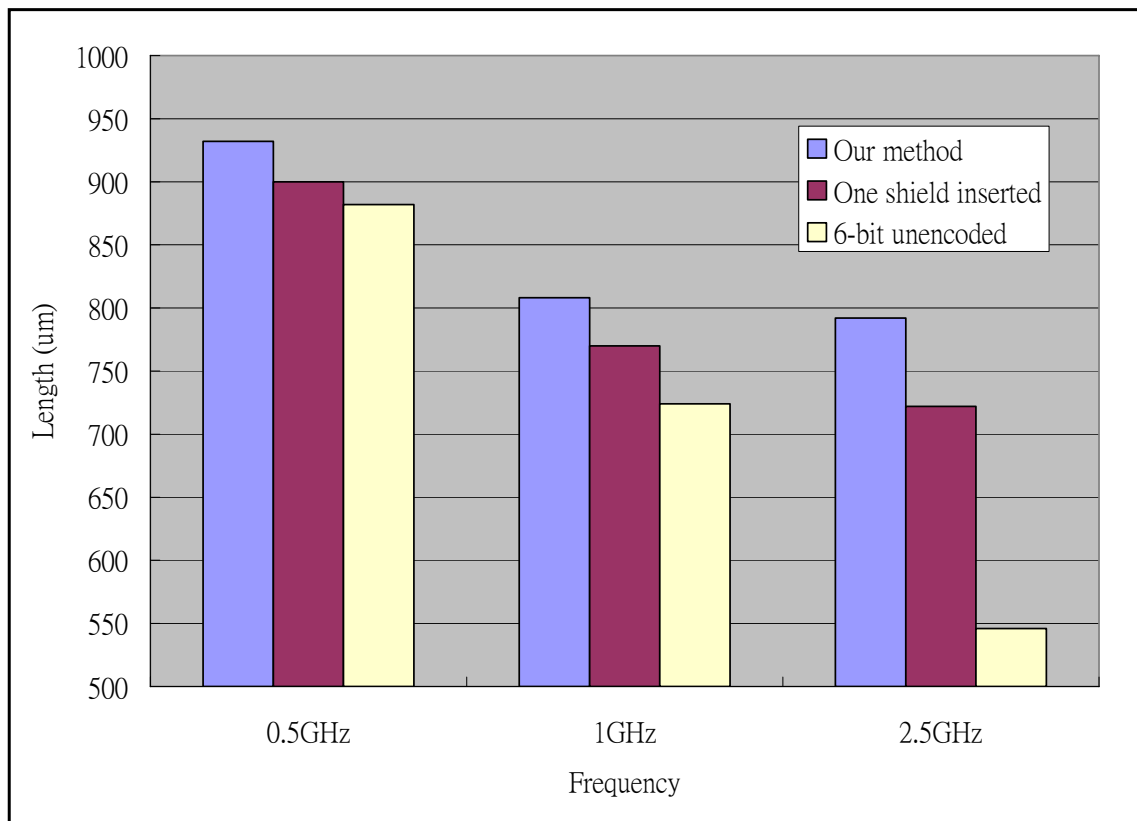


Figure 36: The maximum propagation length by using our flow and the shield insertion technique under different frequencies.

Next, with the same wire dimension and delay constraint, we also compare our work with the shield insertion technique by varying the bus width from 3-bit to 8-bit. The bus working frequency is 1GHz and the wire overhead is one. The simulation

results with different bus width are shown in **Figure 37**. Again, although both our flow and the shield insertion technique can increase the propagation length of buses, our method always outperforms the shield insertion technique for different bus width as shown in **Figure 37**. Therefore, our flow is also more effective in increasing the signal propagation length for different bus width comparing to the conventional shield insertion technique.

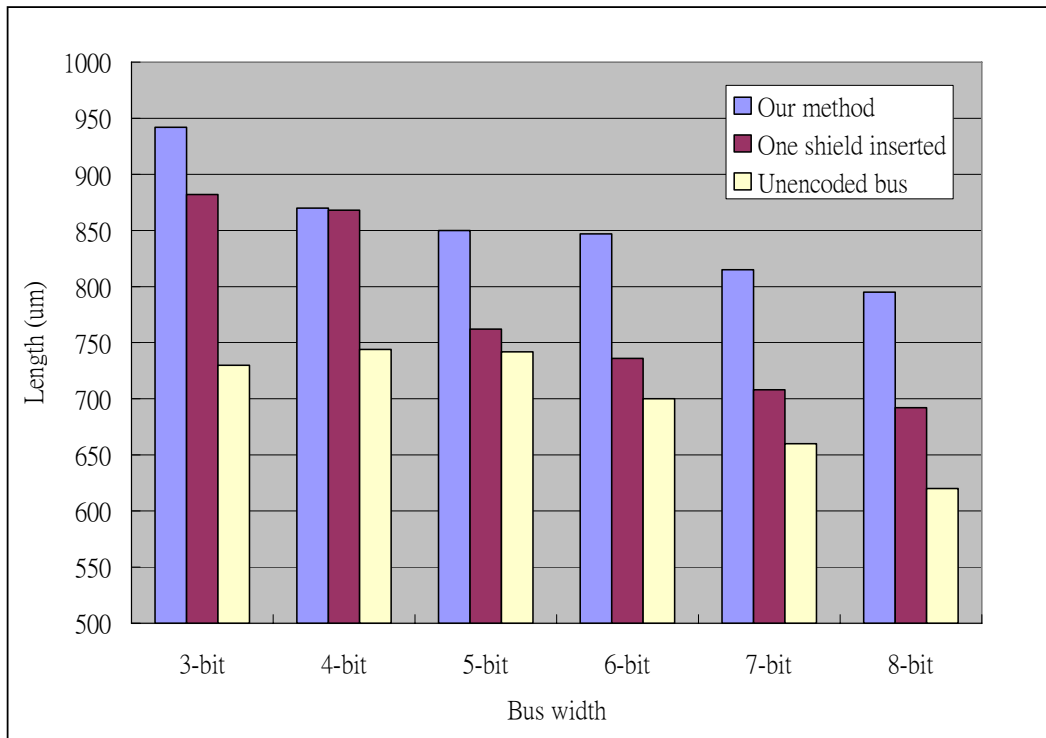


Figure 37: The maximum propagation length by using our flow and the shield insertion technique for different bus width.

3.3.3.3 Codec Overhead

In this subsection, codec overheads are estimated from the synthesized gate-level netlist obtained using TSMC 180-nm CMOS standard cell library. The bus working frequency is 1GHz. The ITRS scaling trend is employed to estimate the delay and area of the codec overheads for other technology node. The estimated results are listed in **Tables 11** and **12**. The Encoder (k to l) in **Tables 11** and **12** represents that the encoder takes a k -bit input and encodes it as an l -bit code, and the Decoder (l to k) represents that the decoder takes an l -bit code and decodes it to a k -bit output. From **Tables 11** and **12**, we observe that both delay and area overhead of a codec significantly grow with the input size k . For example, for the case of “Encoder (3 to 4) + Decoder (4 to 3)” at 180-nm technology node, it only needs about a one-fourth pipeline stage to encode and decode data. However, at the same technology node, the “Encoder (6 to 8) + Decoder (8 to 6)” case needs about a full pipeline stage to process data. Therefore, to avoid the large overhead induced by the codec, designers should choose small k to implement the codec. Hence, for a wide bus, we suggest that, first, divide the wide bus to several sub-groups by inserting shield wires between sub-groups, and then encode/decode each sub-group independently.

Table 11: Delay overheads of codecs.

Delay unit: ps	180-nm	150-nm	130-nm	100-nm	70-nm	50-nm
Encoder (3 to 4) + Decoder (4 to 3)	250	214	188	150	120	100
Encoder (4 to 5) + Decoder (5 to 4)	400	343	300	240	192	160
Encoder (4 to 6) + Decoder (6 to 4)	460	394	345	276	221	184
Encoder (5 to 6) + Decoder (6 to 5)	530	454	398	318	255	212
Encoder (5 to 7) + Decoder (7 to 5)	650	557	488	390	312	260
Encoder (6 to 7) + Decoder (7 to 6)	710	609	533	426	341	284
Encoder (6 to 8) + Decoder (8 to 6)	820	703	615	492	394	328

Although, the codec introduces some amount of delay and area overhead in the system, as shown in **Tables 11** and **12**, the overhead can be mitigated when technology advances. In addition, the fact that the reported delay and area overhead are under the ‘worst-case’ scenario. Since the encoding/decoding logic could be fused with the logic of the IP block, the delay and area overhead could be reduced if the encoder/decoder is optimized along with the IP block simultaneously. In addition, once the input has been encoded to a code, the code can be used repeatedly between repeater stages and pipeline stages for a long propagation length. Hence, we believe

that the induced overhead of the codec is acceptable and our proposed flow will be practical as technology advances.

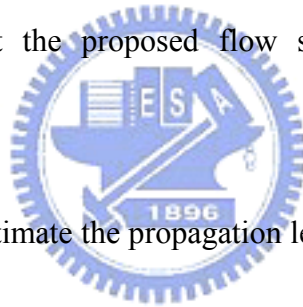
Table 12: Area overheads of codecs.

Area unit: μm^2	180-nm	150-nm	130-nm	100-nm	70-nm	50-nm
Encoder (3 to 4) + Decoder (4 to 3)	302	210	158	93	46	23
Encoder (4 to 5) + Decoder (5 to 4)	1437	998	750	444	217	111
Encoder (4 to 6) + Decoder (6 to 4)	1590	1104	829	491	240	133
Encoder (5 to 6) + Decoder (6 to 5)	2399	1666	1251	740	363	185
Encoder (5 to 7) + Decoder (7 to 5)	3499	2430	1825	1080	529	270
Encoder (6 to 7) + Decoder (7 to 6)	6130	4257	3197	1892	927	473
Encoder (6 to 8) + Decoder (8 to 6)	9517	6609	4964	2937	1439	734

3.3.4 Summary of Signal Propagation Length Increasing Flow

In this section, we propose a flexible maximum signal propagation length

increasing flow that can estimate and increase the one cycle signal propagation length on buses with the user-given constraints and bus parameters. Our flow combines a new bus encoding scheme and a curve-fitting method. The new bus encoding scheme can effectively reduce the *LC* coupling effects on on-chip buses, and hence, improve the worst-case switching delay. In addition, the signal propagation length can be efficiently predicted by the proposed curve-fitting method. Therefore, our flow can produce a valid code set of a bus structure that achieves the maximum signal propagation length under the given constraints with reasonable runtime. Finally, simulation results show that the proposed flow significantly increases the bus propagation length.



Since this work can estimate the propagation length of a bus, we believe that it is feasible to be incorporated into a floorplan tool or a routing tool. In the future, we intend to integrate our work with some layout techniques such as the shield insertion or the buffer insertion. Therefore, the maximum signal propagation length of a bus could possibly be increased further by a proper mixture of these techniques.

Chapter 4

JOINT SHIELD INSERTION AND BUS ENCODING SCHEME

4.1 Motivation

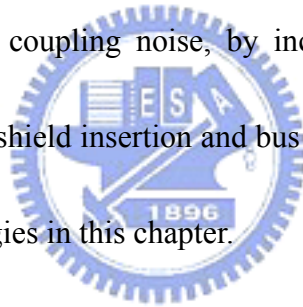


Coupling inductance noise occurs when signal switching causes transient current to flow through the loop formed by the signal wire and current return path, thereby creating a changing magnetic field (see **Figure 7**). This induces a voltage on a victim wire, which is in or near this loop. This induced noise voltage can destroy the logical information carried by that line. Even when noise does not cause functional failure, it has an impact on delay and slew [50].

Since the inductance effects have significant impact on performance and signal integrity, several techniques have been proposed to deal with these effects. By

simultaneously using the net ordering and shield insertion technique [8, 51], the capacitive and inductive coupling noise can be reduced. Zhong et al. developed the *twisted-bundle layout structure* for minimizing inductive coupling noise [52]. Another technique makes use of the active shielding [53, 54] to reduce the inductive and capacitive coupling effects. The most common of these techniques are: shielding [55] where signal lines are interdigitated with Vdd or ground alternatively in order to provide isolation of signal lines from their neighboring signals, and buffer insertion [56] where buffers are inserted in long lines to reduce crosstalk noise and delay.

To further reduce the coupling noise, by incorporating the shield insertion technique, we propose a joint shield insertion and bus encoding scheme for global bus design in nanometer technologies in this chapter.



4.2 Optimized Shield Insertion

In this section, we propose the optimized shield insertion method which is incorporated in the joint shield insertion and bus encoding scheme.

To find the optimized shield insertion method which resulting in minimum

inductive coupling, we exhaustively insert the shields in different positions of a given bus with a given number of shields and extract the self and coupling inductance of all combinations of the resulting bus structure. Then the bus structure with minimum inductive coupling effects will be reported as the optimized bus structure. **Figure 38** demonstrates the all combinations of two-shield-inserted bus structures for a 6-bit bus.

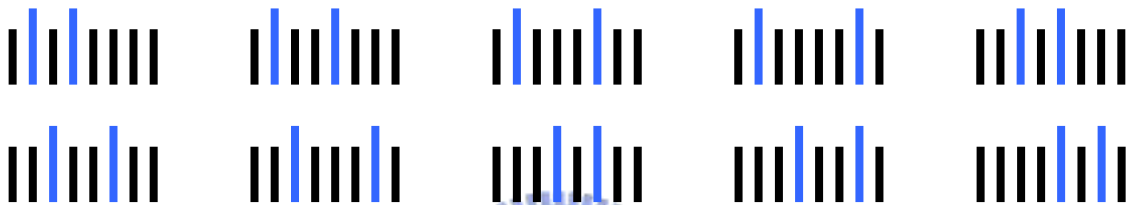


Figure 38: All combinations of two-shield-inserted bus structures for a 6-bit bus. In this figure, the black lines denote the signal wire and the grey and longer lines denote the inserted power/ground wires.



Figure 39: The optimized bus structures with different number of shield inserted in a 6-bit bus. In this figure, the black lines denote the signal wire and the grey and longer lines denote the inserted power/ground wires.

For a 6-bit bus, the optimized bus structures with different number of shield inserted are shown in **Figure 39**. From the simulation results, we can conclude that the optimized shield insertion of a bus is uniformly inserting the shielding wires

between the signal wires. This is due to the fact that by uniformly inserting the shielding wires between the signal wires of a bus, the worst-case current return path can be effectively minimized. Therefore, from Equation (7), we know that with the same magnetic field, if the area of the current loop can be minimized (shorter current return path), then the coupling inductance can be minimized. Take **Figure 40** for an example. The worst-case current return path *A* of the one-shield inserted bus with bad strategy is significantly longer than the path *B* of the optimized one-shield insertion bus. From, our simulation results, the worst-case coupling inductance of Bus *A* is also larger than that of Bus *B*. Therefore, we can conclude that by uniformly inserting the shielding wires between the signal wires of a bus (i.e., optimized shield insertion), the worst-case inductive coupling of the bus can be minimized.

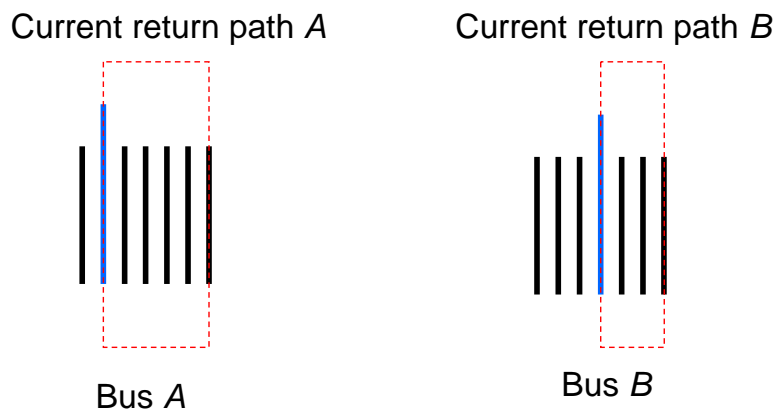


Figure 40: The worst-case current return paths of a 6-bit bus with one shield inserting in different position.

4.3 Proposed Joint Shield Insertion and Bus Encoding Scheme

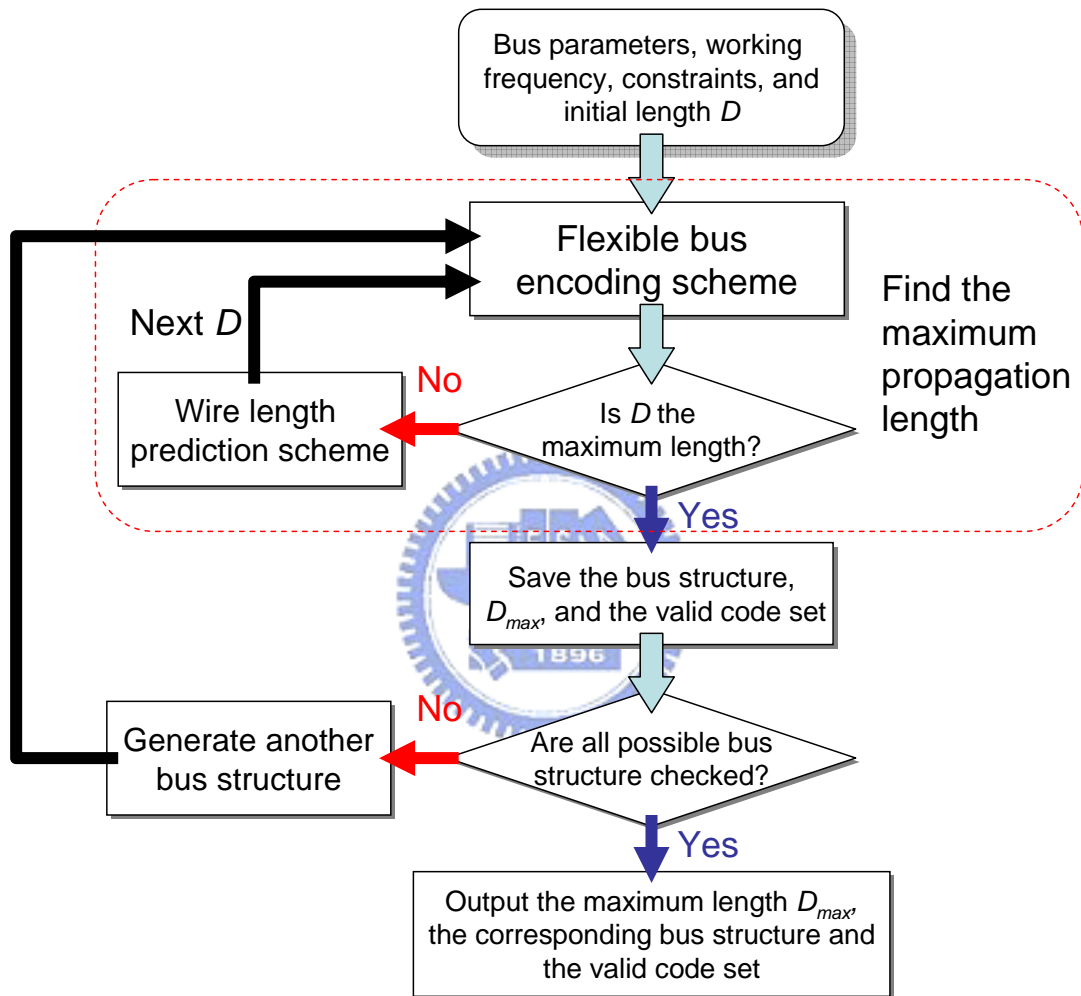


Figure 41: The signal propagation length increasing flow with joint shield insertion and bus encoding scheme.

To compare the efficiency of the joint shield insertion and bus encoding scheme with that of the proposed scheme in Chapter 3, we also utilize the new scheme for the lengthening signal propagation length problem as formulated in Section 3.3.

The details are described in the following.

4.3.1 Signal Propagation Length Increasing Flow with Joint Shield Insertion and Bus Encoding Scheme

Our signal propagation length increasing flow with joint shield insertion and bus coding scheme is shown in **Figure 41**. It mainly comprises two parts – the finding maximum propagation length flow with the flexible encoding scheme and the bus structure generation scheme (shield insertion scheme). In first part, users should give the bus parameters (bus width n , initial wire length D , wire height, wire width, wire pitch, Power/Ground wire width, Power/Ground-to-signal pitch), the working frequency, the delay constraint, and the wire overhead constraint $m - n$. Then with these parameters, we run our encoding scheme and check if the delay constraint can still be met under the wire overhead constraint. If the delay constraint can be met, the bus length is predicted and increased by the curve-fitting method for the next iteration. This process is not terminated until the delay constraint is not met. When this is the case, the last recorded length which still meets the delay constraint is reported as the maximum propagation length D_{max} , and a valid code set is generated at the same time.

The details of the first part are as described in Section 3.3. In the second part, we generate all combinations of bus structures by substituting the signal wires with the shielding wires, and then we can find out the best structure that is suitable for coding to increase the bus signal propagation length. The details of the bus structure generation scheme are discussed in the following.

The bus structure generation scheme generates all combinations of bus structures by substituting the signal wires with the shielding wires. This scheme starts from the original flexible bus encoding scheme with the wire overhead constraint m , i.e., no shielding wire inserted in this step. Next, one signal wire is substituted by one shielding wire which is inserted in the bus by the optimized shield insertion technique, and hence, a new bus structure is generated. Following, two signal wires will be substituted by two inserted shields, and another bus structure will be generated. This procedure is repeated until no substitution can be performed. In other words, all $m - n$ signal wires are substituted by shielding wires. An example is given here to demonstrate the bus structures generated by our bus structure generation flow (see **Figure 42**). For a 6-bit bus with $m = 6$, 4 bus structures will be generated by our scheme as shown in **Figure 42**. Therefore, for an n -bit bus with a given wire overhead constraint m , our scheme will generate total $n - m + 1$ bus structures. These bus structures are then used by the finding maximum propagation length flow to

individually find the maximum propagation length D_{max} .

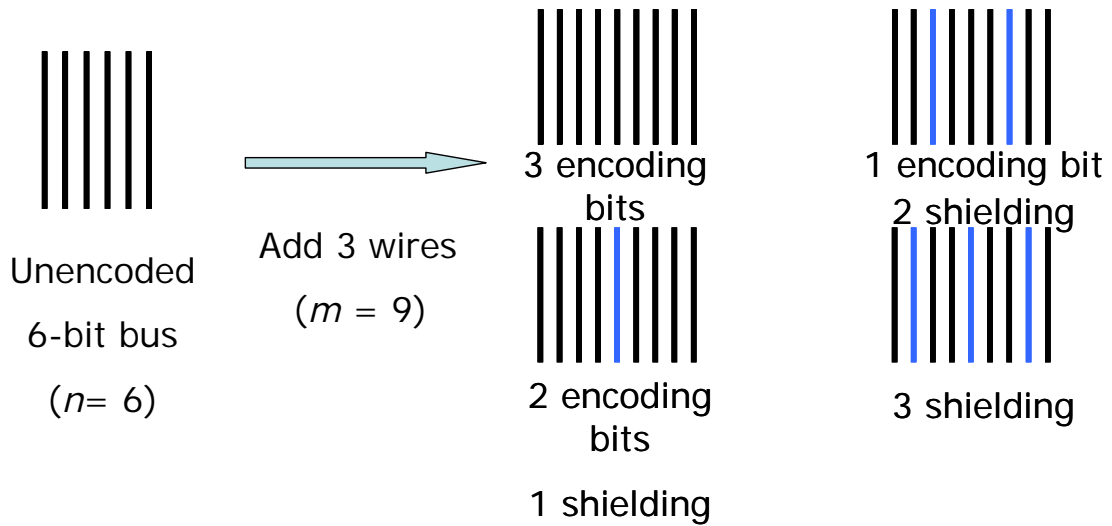


Figure 42: The bus structures generated by our bus structure generation scheme.



4.3.2 Simulation Results

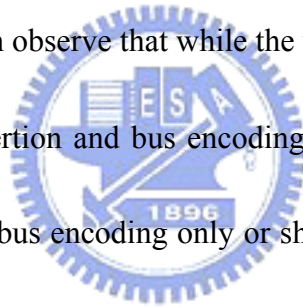
With the user-given parameters, the following simulation results show that our flow can effectively increase the signal propagation length of buses by utilizing the joint shield insertion and bus encoding scheme. For example, if the delay constraint is set to 200ps, and the width, height and pitch of signal wires are given as $2\mu\text{m}$, $2\mu\text{m}$ and $4\mu\text{m}$, respectively. The bus working frequency is 1GHz and the supply voltage is 1.2V. For a 6-bit data bus, we vary the wire overhead constraint ($m - n$) from 1 to 5 and list the simulation results of all bus structures in **Table 13**.

Table 13: Simulation results of all bus structures for a 6-bit bus with m varying from 6 to 11. (E# denotes # of additional signal wires added into the original bus; S# denotes # of shielding wires inserted into the original bus)

Bus structure		D_{max} (μm)	ΔD (μm)	Improvement (%)
$m = n = 6$	Original	4552	0	0.0%
$n = 6$ $m = 7$	E1	5236	684	15.0%
	S1	4944	392	8.6%
$n = 6$ $m = 8$	E1 S1	5520	968	21.3%
	E2	5589	1037	22.8%
	S2	5140	588	12.9%
$n = 6$ $m = 9$	E2 S1	5854	1302	28.6%
	E1 S2	5841	1289	28.3%
	E3	5773	1221	26.8%
	S3	5841	1289	28.3%
$n = 6$ $m = 10$	E3 S1	6044	1492	32.8%
	E2 S2	6449	1897	41.7%
	E1 S3	5933	1381	30.3%
	E4	5951	1399	30.7%
	S4	5979	1427	31.3%
$n = 6$ $m = 11$	E4 S1	6150	1598	35.1%
	E3 S2	6456	1904	41.8%
	E2 S3	6508	1956	43.0%
	E1 S4	6599	2047	45.0%
	E5	6091	1539	33.8%
	S5	6552	2000	43.9%

Columns 1 and 2 in **Table 13** reveal the bus structures used in the new signal propagation length increasing flow. We report the maximum signal propagation length D_{max} outputted by our flow in Column 3. The increasing length ΔD ($=D_{max_original\ bus} - D_{max_current\ bus\ structure}$) is listed in Column 4. Column 5 shows the improvement ($\Delta D / D_{max_original\ bus} * 100\%$). The simulation results of E# is obtained by only using bus encoding technique as described in Section 3.3. On the other hand, the simulation results of S# is obtained by only using shield insertion technique. The simulation results of the joint shield insertion and bus encoding scheme are reported as E#S#.

From **Table 13**, we can observe that while the wire overhead is more than 2 ($m - n > 2$), the joint shield insertion and bus encoding scheme always obtain the best improvement than that of the bus encoding only or shield insertion only (see **Figures 43** and **44**). This proves the efficiency of the joint shield insertion and bus encoding scheme for reducing the *LC* coupling effects. However, while the wire overhead is less than or equal 2 ($m - n \leq 2$), bus encoding only can obtain the best results than those of the other two techniques.



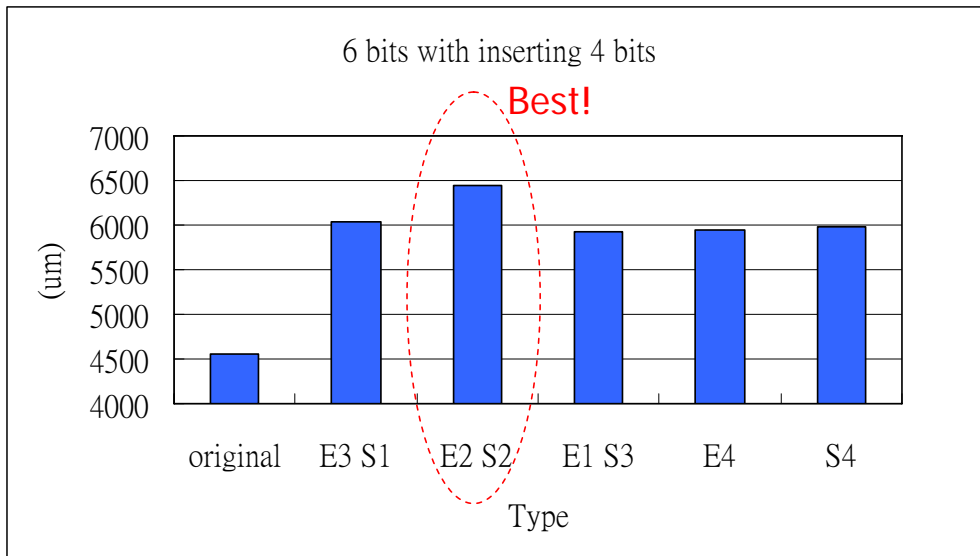


Figure 43: Increasing length of different bus structures with $m - n = 4$.

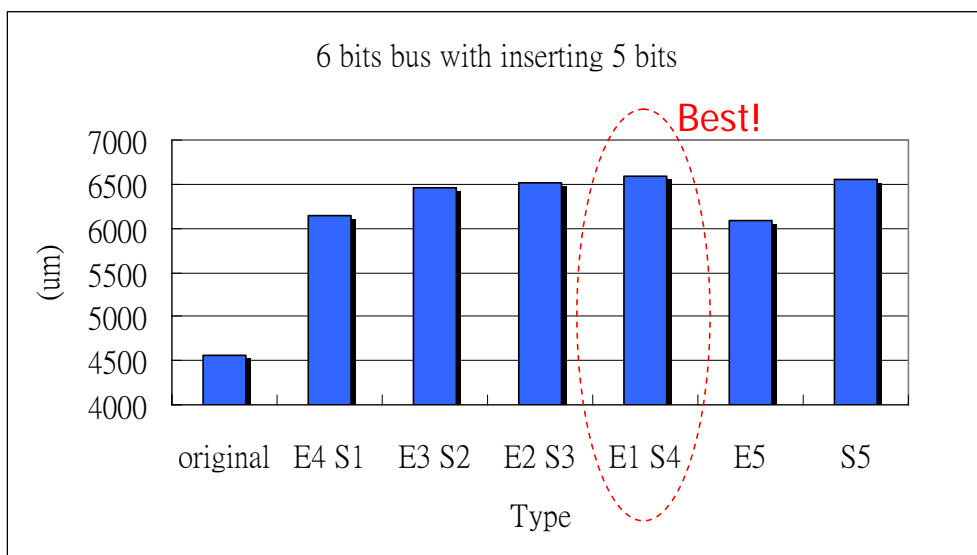


Figure 44: Increasing length of different bus structures with $m - n = 5$.

Figure 45 shows the increasing length a 6-bit bus with m varying from 6 to 9 by using only bus encoding, only shield insertion, and joint shield insertion and bus encoding method. From **Figure 45**, we can also observe that while the wire overhead is more than 2, the joint shield insertion and bus encoding scheme always obtain the

best improvement than that of the bus encoding only or shield insertion only. However, while the wire overhead is less than or equal 2, bus encoding only can obtain the best results than those of the other two techniques. Besides, we observe that with large wire overhead, the improvement of the shield insertion technique is better than the flexible bus encoding scheme.

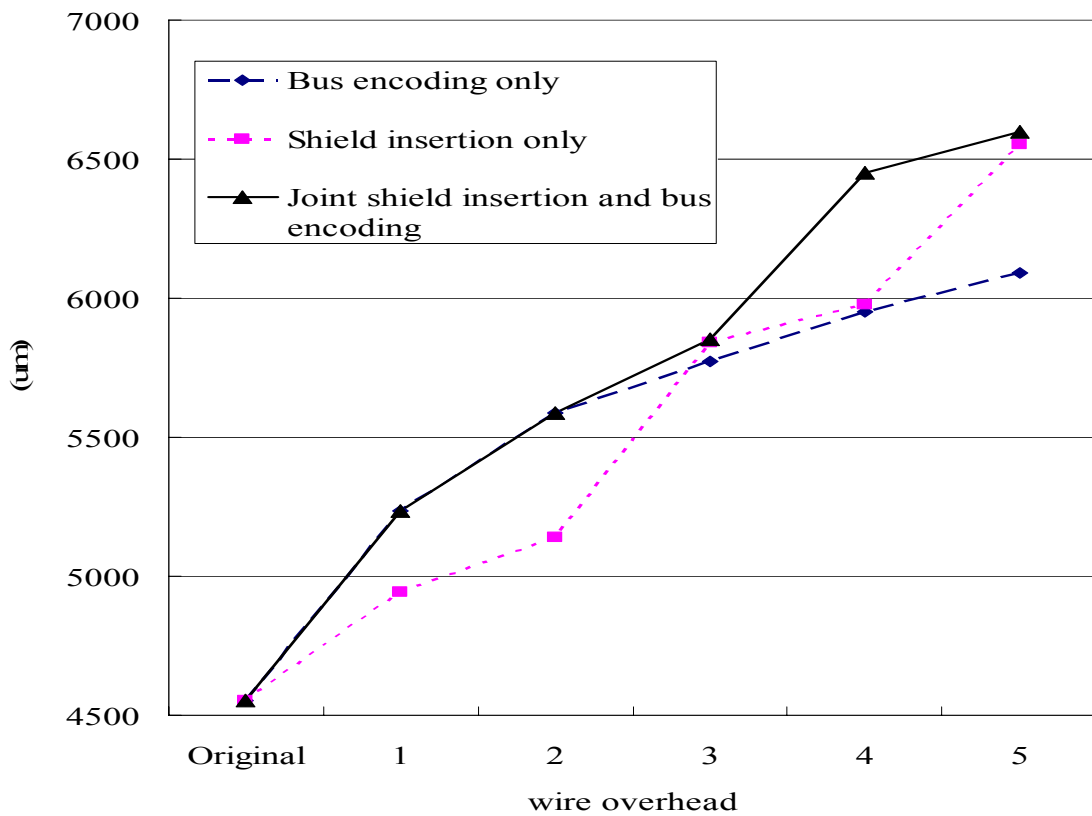


Figure 45: Increasing length of a 6-bit bus with m varying from 6 to 9 by using only bus encoding, only shield insertion, and joint shield insertion and bus encoding.

4.4 Summary

In this chapter, we proposed the optimized shield insertion technique which can minimize the coupling inductance effects. Then by incorporating the optimized shield insertion technique with our flexible bus encoding scheme, we proposed the joint shield insertion and bus encoding scheme and utilized this scheme in the new signal propagation length increasing flow. Simulation results show that the new scheme can gain more improvement of the increase of the bus signal propagation length than that gained by using the shield insertion only or the flexible bus encoding only. Therefore, we can conclude that the joint shield insertion and bus encoding scheme can reduce the coupling noise more effectively than the shield insertion technique and the flexible bus encoding scheme.

Chapter 5

FLEXIBLE ON-CHIP BUS ENCODING FOR POWER MINIMIZATION UNDER DELAY CONSTRAINT



5.1 Motivation

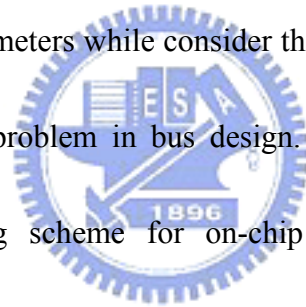
Low power and high performance operation are necessary for all components in microprocessors and system-on-chip (SOC) designs. This is especially true for global buses, whose delay and power dissipation show an increasing trend with technology scaling. According to International Technology Roadmap for Semiconductors (ITRS) [2] and National Technology Roadmap for Semiconductors

(NTRS) [1] gate delay reduces with scaling, while global wire delay increases. Therefore, the delay of global buses will act as the performance bottleneck in many high performance system-on-chip (SOC) designs. Further, interconnect networks consume 20%-36% of total system power in many large SOCs [57]. Therefore, it is crucial to reduce the delay and power consumption of the on-chip bus to improve the circuit performance in DSM designs.

Due to the higher wire aspect ratio and smaller wire pitch, the coupling capacitance between neighboring wires becomes the dominant component of the total wire capacitance. Moreover, when the clock frequency increases over GHz, the inductance effects on on-chip interconnects are becoming increasingly significant [1, 2]. However, most existing works focus on reducing the effects resulting from coupling capacitance on the bus structure. There is not much work in the literature considering inductance effects on the bus structure to develop encoding schemes for reducing the bus delay and power consumption. Considering only the capacitive coupling effect, Sridhara et al. [58], Sotiriadis and Chandrakasan [28], Victor and Keutzer [29], Baek et al. [30], and Hirose and Yasurra [9] proposed their bus encoding techniques to eliminate the crosstalk delay. They try to avoid the simultaneous inverse transitions of neighboring wires. Based on eliminating the same worst capacitive coupling patterns, Baek et al. [30], Sridhara et al. [58], Subrahmanya et al. [59],

Lindkvist et al. [24], and Sotiriadis et al. [60] develop their own bus encoding scheme to reduce the bus power consumption.

Since most previous encoding schemes [9, 24, 28 – 30, 58 – 60] optimize the bus delay and/or power consumption considering only the capacitive coupling, they might not be suitable for today's high-performance on-chip bus design due to the significant inductive coupling. Although the work [32] considers the inductance effects to build their encoding schemes, they only reduce the bus delay. However, as discussed in Chapter 1, the worst-case pattern causing the largest transition delay varies with given design parameters while consider the **RLC** effects on buses. Besides, the power is also a crucial problem in bus design. Based on these facts, we first propose a flexible encoding scheme for on-chip buses considering the given parameters to minimize the bus power consumption subject to the delay constraint by effectively reducing the **LC** coupling effects. Then we also propose a joint shield insertion and bus encoding scheme to solve this problem.



5.2 Flexible Bus Encoding Scheme for Power Minimization Under Delay Constraint

With given parameters of wires (length, width, height and pitch), delay constraint, working frequency (or slew rate) and the number of data bit, our proposed flexible bus encoding scheme will generate a valid code set that has **the minimal total transition power to map the data patterns**. The valid code set is obtained at the cost of $m - n$ extra bus wires. Any valid code set must satisfy the property that any transition between codes within this set is guaranteed to meet the delay constraint.

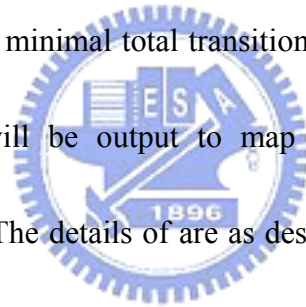
The details of our proposed scheme are described in the following.



5.2.1 Encoding Flow for Power Minimization

Figure 46 illustrates our overall encoding flow. At first, users should give the bus parameters (bus width n , wire dimensions, wire pitch, Power/Ground grid dimensions, Power/Ground-to-signal pitch), the working frequency, and the delay constraint. With the given parameters, we extract the resistances, capacitances, and inductances of bus wires. After extraction, the equivalent **RLC** circuit will be built.

Next, the built circuit will be simulated by using HSPICE with the *basis vectors* which will be defined later. By applying superposition theorem of linear circuits, we can establish the transition delay graph efficiently. From the transition delay graph, we apply a greedy algorithm to find a valid code set in which every transition between a code pair meets the delay constraint. Next, we will check whether the code set covers all data patterns. If not, we add one more bit line to the bus structure and redo from the Step (1). Otherwise, the code set will be output to the next step to build the transition power graph. After building the transition power graph, we apply a heuristic search to find a code set with minimal total transition power from the outputted code set. Finally, the code set will be output to map to the data patterns with the corresponding bus structure. The details of are as described in Section 3.2. However, the Properties 1 and 2 used in Section 3.2 should be modified to Properties 3 and 4 as shown below.



Property 3: In an linear *RLC* circuit of a bus, given a *basis vector*, such as

$(- \dots \uparrow)$, the transition delay and power due to the switching input is the same no matter other stable inputs are at '0' or '1'.

Property 4: In an linear *RLC* circuit of a bus, given an integer k and

$0 \leq k \leq n-1, \{(z_0 \ z_1 \ z_2 \ \dots \ z_{n-1}) \mid z_k \in \{\uparrow\} \text{ and } z_0, z_1, \dots, z_{n-1} \in \{-\}\} \text{ and } \{(z_0$

$z_1 \ z_2 \ \dots \ z_{n-1} \) \mid z_k \in \{\downarrow\}$ and $z_0, z_1, \dots, z_{n-1} \in \{-\}$ are called a *dual basis vector pair*. The voltage and current waveforms resulting from a *dual basis vector pair* are equal in magnitude but opposite in direction.

The details of Steps (5) and (6) are described in the following.

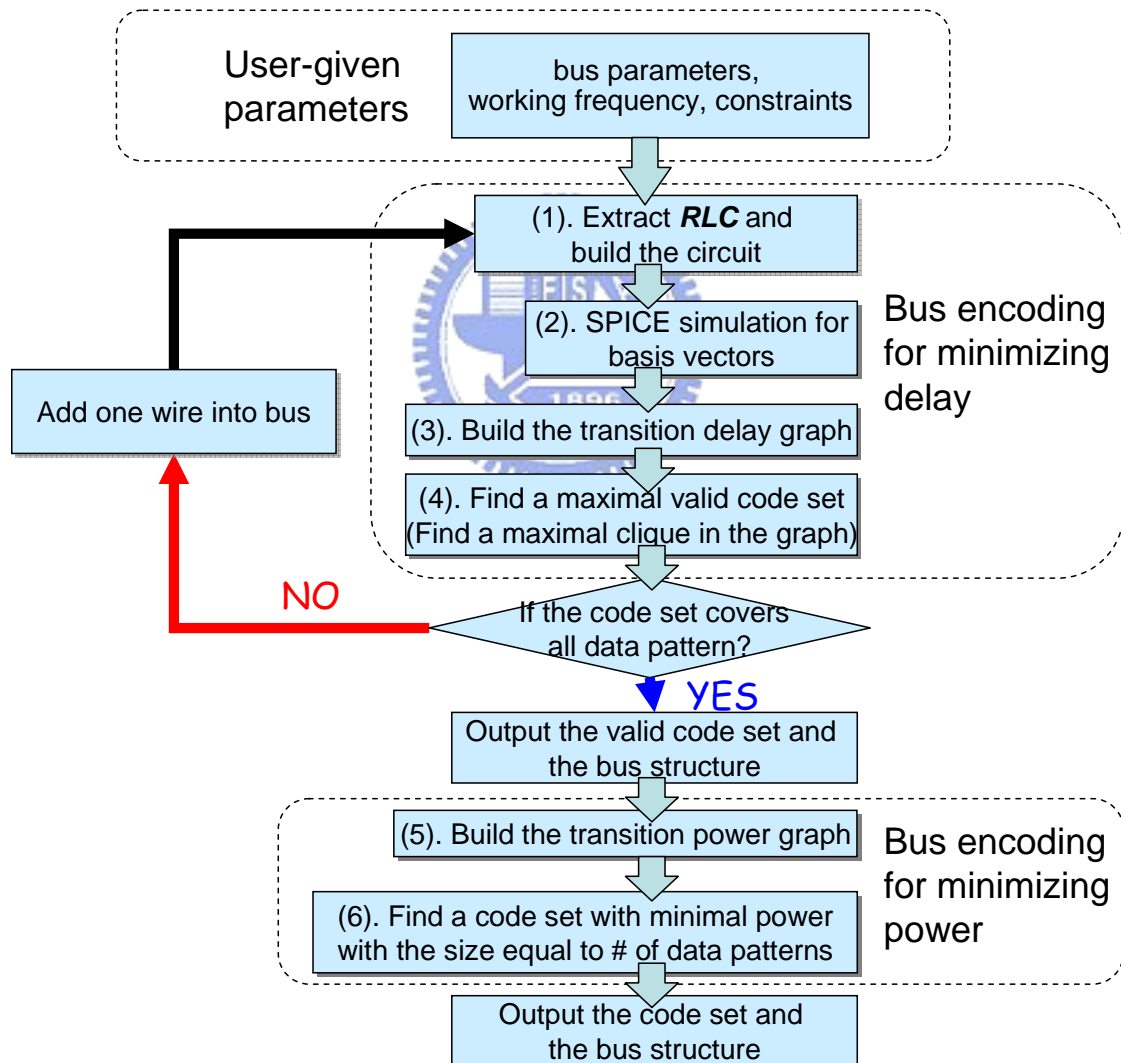


Figure 46: The proposed flexible encoding flow for bus power minimization.

5.2.1.1 Build the Transition Power Graph

With the valid code set generated in Step (4), we then build the transition power graph to reveal that the transition power between any two codes in this set. Since the current in each wire can be obtained by using the superposition, the transition power between any two codes can be obtained from the following equation:

$$\text{Transition power } P = \sum_{i=1}^n V_{dd} \cdot I_{i \text{ avg}} \quad (11)$$

where $I_{i \text{ avg}}$ is the average current in wire i . Hence, in the transition power graph, a vertex represents a code and an undirected weighted edge indicates that the transition between two corresponding codes with an edge weight equal to the transition power.

Again, in this step, no HSPICE simulation is needed to be conducted.

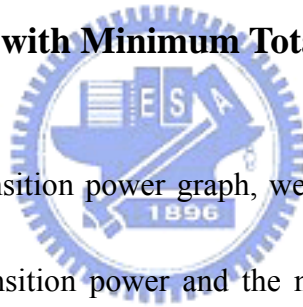
Therefore, a significant amount of simulation time can be saved in Step (5).

Table 14: Transition currents obtained form HSPICE simulations and the superposition method.

Transition	bit1	bit2	bit3	bit4
↑ ↑ ↑ ↑ (0->15)	1	2	3	4
superposition	0.000200724	0.0002256	0.00022552	0.000200728
spice	0.000200724	0.0002256	0.00022552	0.000200728
↓ ↑ ↓ ↑ (10->5)	1	2	3	4
superposition	0.000375124	-0.0005636	0.00056352	-0.00037513
spice	0.000375124	-0.0005636	0.00056352	-0.00037513
↓ ↓ ↓ ↑ (14->1)	1	2	3	4
superposition	0.000375124	-0.0004	-0.00022552	-0.00020073
spice	0.000375124	-0.0004	-0.00022552	-0.00020073

To demonstrate the equivalence of the transition current obtained from HSPICE simulation results and the current obtained from superposition results, we show the results obtained by both methods of a 4-bit bus with some input transition patterns for an example. The results are shown in **Table 14**. From **Table 14**, we can observe that the transition currents obtained from these two methods are exactly the same.

5.2.1.2 Find a Code Set with Minimum Total Edge Weight



After building the transition power graph, we want to find a code set in this graph with minimal total transition power and the number of vertices equal to the number of the data patterns. The reason to find this code set in the transition power graph is that we want to minimize both the average power and peak power consumption of the bus by mapping the data patterns to the code set. In Step (6), we use a heuristic algorithm to solve this problem. The pseudo code of the algorithm is shown in **Figure 47**.

Before entering the loop, we first check if the number of vertices in the graph is equal to the number of the data patterns. If so, we output this graph directly. Otherwise, we delete the vertex which has the maximum total edge weight of all its

edges and update the graph. We repeat the above steps until the remaining number of vertices equal to 2^n .

```
INPUT: An undirected weighted transition graph and the bus width  $n$ ;  
OUTPUT: A vertex set with minimal total edge weight of all edges;  
IF the number of the vertices in the graph is equal to  $2^n$ , output this  
graph and exit;  
REPEAT  
    Find a vertex  $v$  which has maximum total edge weight of all its edges;  
    Remove  $v$  and update the graph;  
UNTIL The number of the vertices in the graph is equal to  $2^n$ ;  
Output the graph;
```

Figure 47: Pseudo code of finding a code set with minimal edge weight.



5.2.2 Simulation Results

5.2.2.1 Delay and Power Reduction

Here, we present the simulation results of reducing the power consumption of the on-chip bus subject to the delay constraint by using our encoding scheme.

In the following, the length, width, height, pitch of signal wires, and Power/Ground-to-signal pitch are $2000\mu\text{m}$, $2\mu\text{m}$, $2\mu\text{m}$, $4\mu\text{m}$, and $13\mu\text{m}$, respectively.

The bus working frequency is 1GHz and the supply voltage is 1.2V. The delay constraint is set to 300ps and the coupling to self capacitance ratio is 5. **Table 15**

shows the simulation results of a 5-bit bus with wire overheads ($m - n$) varying from 1 to 2, while **Table 16** lists the results of a 6-bit bus. In **Tables 15** and **16**, column 2 shows the simulation results of the original bus without encoding. Columns 3 and 5 list the simulation results of the encoded buses which are only encoded to meet the delay constraint without power minimization (denoted as “No PM”). The simulation results of the encoded buses with power minimization (denoted as “With PM”) are shown in columns 4 and 6. D_{worst} denotes the worst-case switching delay of the bus, while ΔP_{peak} and ΔP_{avg} denote the peak and average power saving, respectively. Pat_{worst_D} and Pat_{worst_P} represent the worst-case switching patterns for the bus delay and power consumption, respectively.

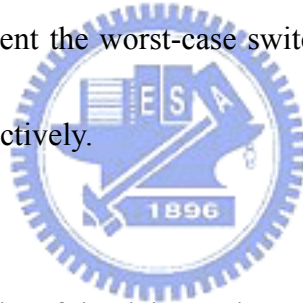


Table 15: The simulation results of the delay and power minimization for $n = 5$ and $m = 6\sim 7$. (\uparrow : switching from “0” to “1”; \downarrow : switching from “1” to “0”; $_$: no switching)

Scheme	Original $n = 5$	Encoded $m = 6$		Encoded $m = 7$	
		No PM	With PM	No PM	With PM
D_{worst} (ps)	329	270	263	267	256
Pat_{worst_D}	$\uparrow\downarrow\uparrow\downarrow\uparrow$	$\uparrow\uparrow\downarrow\uparrow\downarrow$	$\downarrow\downarrow\uparrow\downarrow$	$\downarrow\uparrow\uparrow\downarrow\uparrow$	$\uparrow\uparrow\uparrow\uparrow\downarrow$
ΔP_{avg} (%)	0	-0.24	16.98	-14.42	19.83
ΔP_{peak} (%)	0	25.68	43.84	15.80	53.84
Pat_{worst_P}	$\uparrow\downarrow\uparrow\downarrow\uparrow$	$\downarrow\uparrow\uparrow\downarrow\downarrow\uparrow$	$\downarrow\uparrow\uparrow\downarrow\downarrow\downarrow$	$\downarrow\uparrow\uparrow\uparrow\downarrow\downarrow\uparrow$	$\uparrow\uparrow\uparrow\uparrow\downarrow$

Table 16: The simulation results of the delay and power minimization for $n = 6$ and $m = 7\sim 8$.

Scheme	Original $n = 6$	Encoded $m = 7$		Encoded $m = 8$	
		No PM	With PM	No PM	With PM
D_{worst} (ps)	338	266	267	273	264
$Pat_{worst D}$	↑↑↓↑↓↑	↓↑ ↓↑↑↓	↓↓↓↑ ↓↑	↓↑ ↓↓↓↓	↓↓↓↓↓↑
ΔP_{avg} (%)	0	4.63	15.68	-4.93	21.54
ΔP_{peak} (%)	0	25.72	44.36	21.15	52.38
$Pat_{worst P}$	↓↑↓↑↓↑	↑↓ ↑↓↑↑	↓↑ ↑↓	↑↓↑↑ ↑↓	↓↓↓↑↑↓

As shown in **Tables 15** and **16**, the worst-case switching delays of the original buses do not satisfy the delay constraint (300ps). However, after applying our bus encoding scheme, all the worst-case switching delays of the encoded buses are less than the delay constraint. As shown in Row 2, since the switching patterns resulting in the worst-case delay are eliminated by our encoding scheme, the worst-case bus delay can be minimized to meet the delay constraint. From **Tables 15** and **16**, we observe that the average power consumptions of the encoded buses without power minimization could be worse than those of the original buses, although the peak power saving can be up to 25%. However, when the buses encoded both for delay and power minimization, the average and peak power saving can be up to 16% and 44%, respectively, with one wire overhead ($m - n = 1$). For two wire overheads, the average and peak power saving can be up to 21% and 53%, respectively. In these simulations,

we can also observe that there is only a slight or even no difference between the worst-case delay and power pattern as shown in **Tables 15** and **16**. This is because the capacitance effect is the dominant factor in these simulations.

Next, with the same wire parameters and supply voltage, we vary the bus working frequency from 100MHz to 5GHz and list the simulation results in **Table 17**.

The coupling to self capacitance ratio is set to 2.5. From **Table 17**, we can observe that the worst-case delay pattern changes from the capacitance-dominated one to the inductance-dominated one as the working frequency increases. Thus the efficiency of

the power optimization of our encoding scheme tends to decrease as the frequency increases. This is because when the inductance effect becomes more significant than the capacitance effect, the worst-case delay pattern changes to the

inductance-dominated one, but the worst-case power pattern is still the capacitance-dominated one. Therefore, when the delay minimization is performed by

using our encoding scheme at high working frequency, the worst-case delay pattern which is the inductance-dominated one will be eliminated, but the worst-case power

pattern will be left. Hence, after the delay optimization, the remaining valid codes are mainly bad for the power minimization.

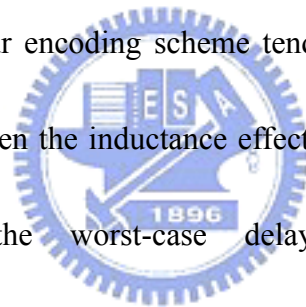


Table 17: The simulation results of the delay and power minimization for $n = 5$ and $m = 7$ with the working frequency varying from 100MHz to 5GHz.

Scheme	100MHz		500MHz		1GHz		3GHz		5GHz	
	$n = 5$	$m = 7$	$n = 5$	$m = 7$	$n = 5$	$m = 7$	$n = 5$	$m = 7$	$n = 5$	$m = 7$
D_{worst} (ps)	222	222	222	152	194	208	188	260	109	102
$Pat_{worst D}$	↑↓↑↓↑	↑↓↑↑↑↑↑	↓↓↓↑↓	↑↓↓↓↓	↓↑↓↑↓	↓↓↓↑↓	↓↑↓↑↓	↓↓↓↑↓	↑↑↑↑↑	↓↑↑↑↑
ΔP_{avg} (%)	0	14.23	0	19.48	0	16.61	0	13.38	0	9.99
ΔP_{peak} (%)	0	38.03	0	54.00	0	41.88	0	39.15	0	44.12
$Pat_{worst P}$	↑↓↑↓↑	↑↓↑↑↑↑↑	↑↓↑↓↑	↑ ↓↓↑↑	↑↓↑↓↑	↑↑↑↓↑↑↑	↑↓↑↓↑	↑↑↑↑↓↑↑	↑↓↑↓↑	↑ ↓↑↑ ↓

5.2.2.2 Power Minimization Only

We utilize our encoding scheme to optimize the bus power consumption only by loosening the delay constraint. The wire parameters and supply voltage are the same with the previous subsection. The working frequency is set to 1GHz and the coupling to self capacitance ratio is 5. The simulation results are listed in **Table 18**. As shown in **Table 18**, when optimizing the bus power consumption only, the average and peak power saving can be up to 17% and 47%, respectively, with one wire overhead. For two wire overheads, the average and peak power saving can be up to 23% and 51%, respectively. In addition, the worst-case delay can also gain a little improvement.



Table 18: The simulation results of the power minimization only for $n = 5\sim 6$ and $m = 7\sim 8$.

Scheme	$n = 5$			$n = 6$		
	$m = 5$	$m = 6$	$m = 7$	$m = 6$	$m = 7$	$m = 8$
D_{worst} (ps)	329	328	315	338	330	335
ΔP_{avg} (%)	0	17.78	22.10	0	16.41	23.33
ΔP_{peak} (%)	0	43.84	45.90	0	47.48	51.09

With the same wire parameters and working frequency, we vary wire overheads from 1 to 5 with $n = 5$ and the simulation results are shown in **Figure 48**.

From **Figure 48**, we observe that both the peak and average power saving tend to increase as the wire overhead increase. This is due to the fact that as the wire overheads increase, there are more codes could be used to optimize the power consumption than that of few wire overheads.

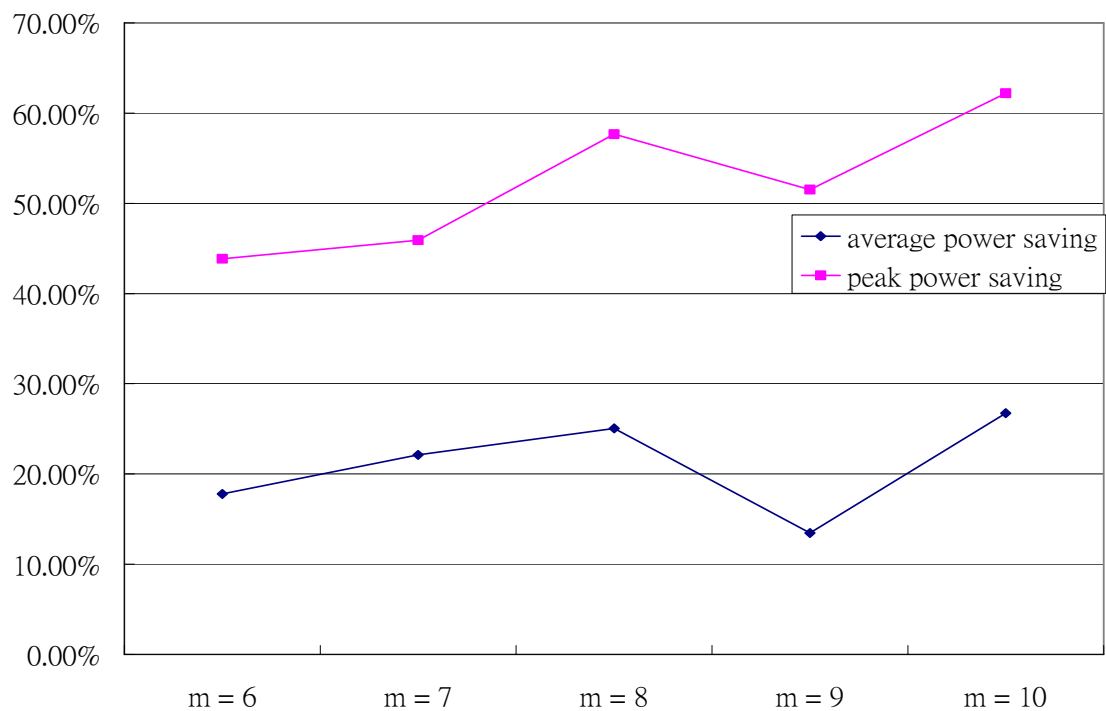


Figure 48: Power reduction of $n= 5$ and m varying from 6 to 10.

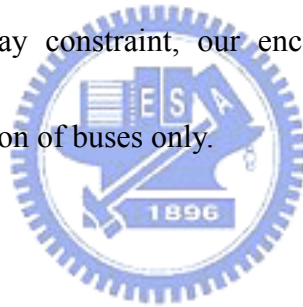
5.2.3 Summary of Flexible Bus Encoding Scheme for Power Minimization Under Delay Constraint

In this section, we propose a flexible on-chip bus encoding flow considering

bus parameters to minimize the bus power consumption subject to the delay constraint by effectively reducing the *LC* coupling effects. Simulation results have shown that our encoding method can significantly reduce the coupling delay of a bus with given delay constraints and parameters. Meanwhile, the power consumption can be reduced.

Based on the superposition theorem, the transition delays and currents obtained by our method and pure HSPICE simulation are exactly identical. In addition, many HSPICE simulations can be reduced and thus a great amount of the simulation time can be saved.

By loosening the delay constraint, our encoding scheme can be used to optimize the power consumption of buses only.



5.3 Joint Shield Insertion and Bus Encoding Scheme for Power Minimization Under Delay Constraint

5.3.1 Motivation

By inserting shields into buses, the inductive coupling can be effectively reduced. Therefore, by incorporating the optimized shield insertion technique, our

flexible encoding scheme could find a better valid code set for the bus power reduction. This is because that the inductive coupling is minimized by the shielding, and hence, our encoding scheme could optimize for the bus power reduction. Hence, to improve the performance of our scheme, we also propose a joint shield insertion and bus encoding scheme for the global bus global bus power minimization under a given delay constraint in this section.

5.3.2 Joint Shield Insertion and Bus Encoding Flow for Power Minimization



Figure 49 illustrates our overall joint shield insertion and bus encoding flow for the bus power minimization. There are four steps in our proposed flow. At Step (1), with the initial bus structure (i.e., inserted $m - n$ extra signal wires to the original bus) and the given bus parameters, we utilize the flexible bus encoding scheme to find the valid code set with minimum total transition power (see Section 3.2 for details). Then, we will check whether the code set covers all data patterns. If not, we conduct Step (3) to generate another bus structure (see Section 4.3 for details) and then redo from the Step (1). Otherwise, the code set will be output to Step (2) (see Section 5.2 for details) to build the transition power graph, and then we apply a heuristic search to find a code

set with minimal total transition power from the outputted code set. Following, the found minimum power valid code set will be recorded together with the corresponding bus structure, and then we will check if all bus structures are tried (i.e., all $m - n$ extra signal wires substituted by shielding wires). If not, we conduct Step (3) to generate another bus structure and then redo from the Step (1). Otherwise, the best valid code set (with minimum total transition power) will be output to map to the data patterns with the corresponding bus structure.

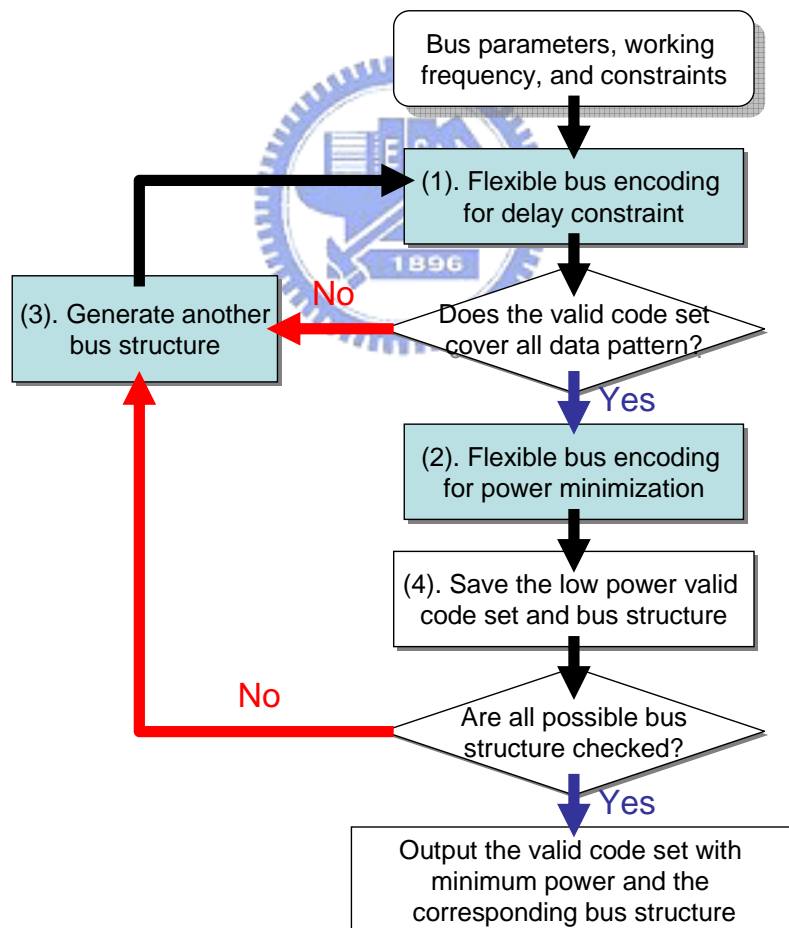


Figure 49: The proposed joint shield insertion and bus encoding flow for bus power minimization.

5.3.3 Simulation Results

Here, we present the simulation results of reducing the power consumption of the on-chip bus subject to the delay constraint by using the proposed joint shield insertion and bus encoding scheme.

In the following, the length, width, height, pitch of signal wires, and Power/Ground-to-signal pitch are $2000\mu\text{m}$, $2\mu\text{m}$, $2\mu\text{m}$, $4\mu\text{m}$, and $13\mu\text{m}$, respectively. The supply voltage is 1.2V. We vary the bus working frequency from 0.5 to 5 GHz, and the simulation results of a 6-bit bus with wire overheads ($m - n$) varying from 1 to 5 are shown in **Figures 50, 51, and 52**.

From **Figures 50, 51, and 52**, we observe that when the wire overhead $m - n$ is larger than 3, the average power reduction of the joint shield insertion and bus encoding scheme could be more than that of the flexible bus encoding scheme. This is because a wider bus has more significant inductance effects than a narrower one, and the inductance coupling can be effectively minimized by the optimized shield insertion technique. Therefore, for tight wire overhead constraint (≤ 3 wires), we could obtain very well average power minimization by using the flexible bus encoding scheme. However, for the wire overhead constraint larger than 3, the joint shield insertion and bus encoding scheme could obtain better average power reduction

than that of the flexible bus encoding scheme.

In addition, from **Figures 50, 51, and 52**, we observe that as the working frequency increasing, our proposed joint shield insertion and bus encoding scheme could also obtain better average power reduction than that of the flexible bus encoding scheme. This is due to the significant inductance effects at high working frequency. Hence, by applying the optimized shield insertion technique first for the inductance-dominated bus, then we could obtain a better valid code set for the bus power reduction by flexible bus encoding. Therefore, our proposed scheme is very suitable for minimizing the power consumption of the inductance-dominated bus.

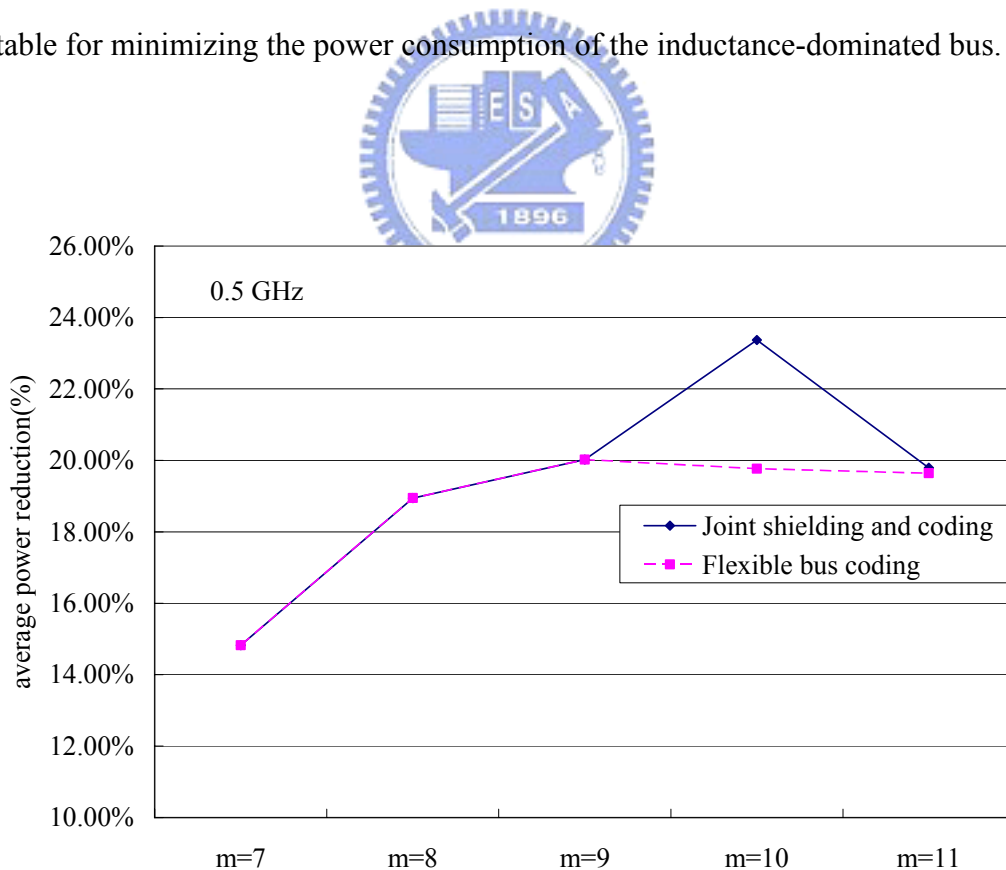


Figure 50: Average power reduction of the flexible bus encoding scheme and the joint shield insertion and bus coding scheme for 0.5 GHz working frequency.

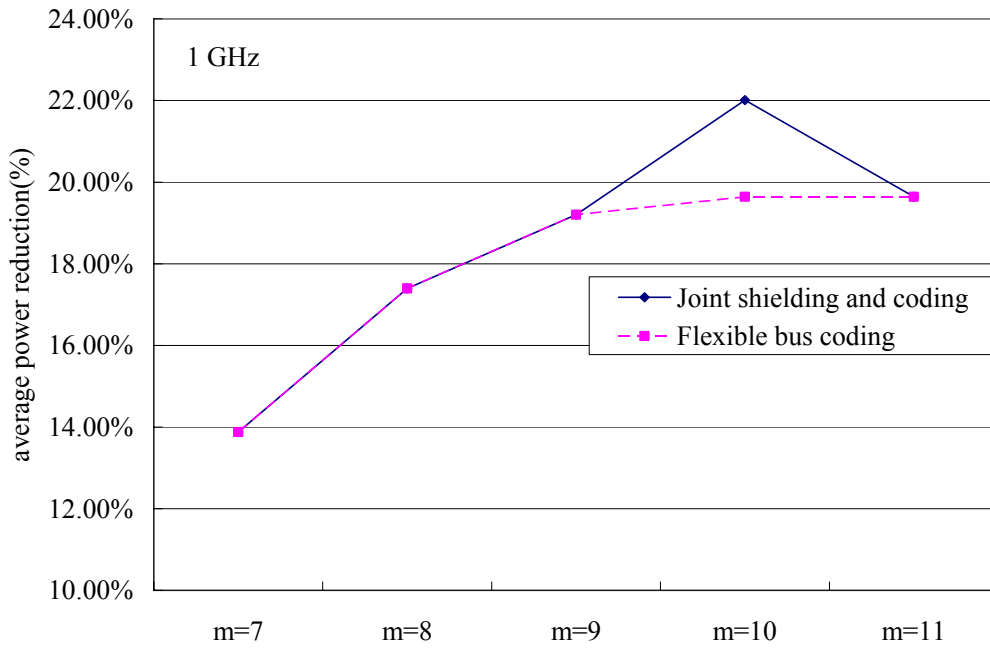


Figure 51: Average power reduction of the flexible bus encoding scheme and the joint shield insertion and bus coding scheme for 1 GHz working frequency.

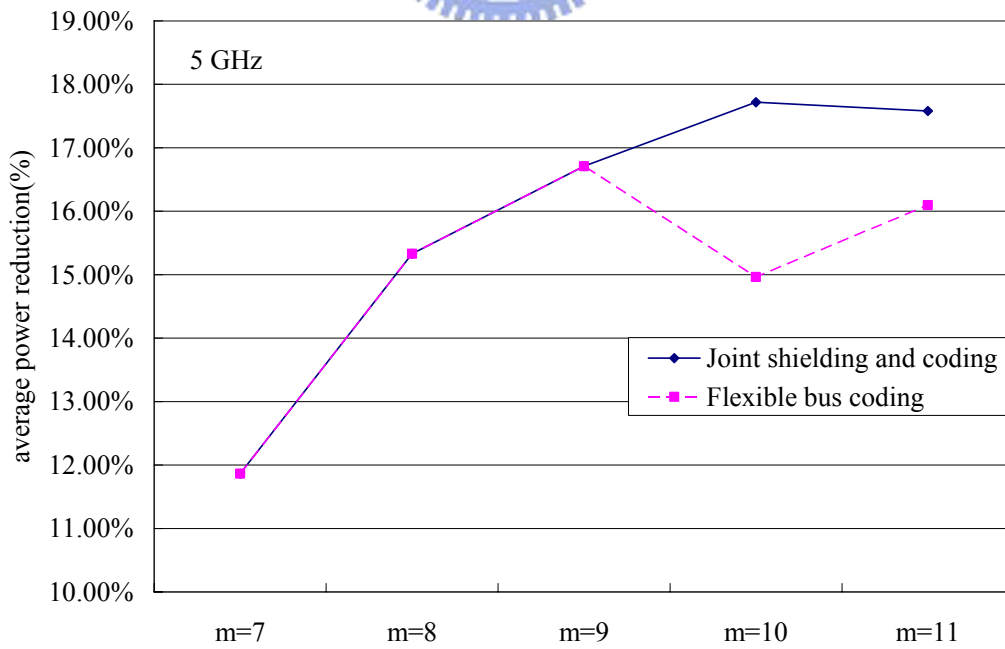
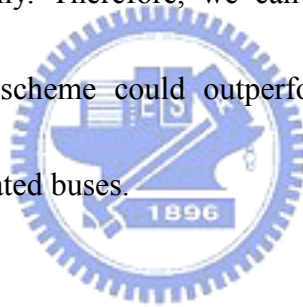


Figure 52: Average power reduction of the flexible bus encoding scheme and the joint shield insertion and bus coding scheme for 5 GHz working frequency.

5.3.4 Summary

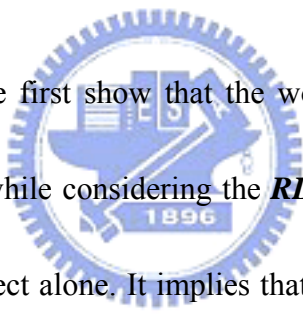
In this section, by incorporating the optimized shield insertion technique with our flexible bus encoding scheme, we proposed the joint shield insertion and bus encoding scheme for bus power minimization under a given delay constraint. For loose wire overhead constraints and high working frequency, simulation results show that the new scheme can gain more average power reduction than that gained by using the flexible bus encoding only. Therefore, we can conclude that the joint shield insertion and bus encoding scheme could outperform the flexible bus encoding scheme for inductance-dominated buses.



Chapter 6

CONCLUSIONS & FUTURE

WORKS



In this dissertation, we first show that the worst-case switching pattern that incurs the longest bus delay while considering the *RLC* effect is quite different from that while considering *RC* effect alone. It implies that the existing encoding schemes based on the *RC* model may not improve or possibly worsen the delay when the inductance effects become dominant. Then, we proposed several bus encoding schemes to minimize the bus coupling delay and the power dissipation. Simulation results are also given to show the performance of our proposed schemes.

In Chapter 1, while considering the *RLC* circuit model for the bus structure, our finding show that the worst-case switching pattern could be very different from that of only considering *RC* effects. First, when the inductance effect dominates, the worst-case switching pattern with the largest on-chip bus delay is when all wires

simultaneously switch in the same direction. Furthermore, we indicate that while considering the **RLC** effects of interconnects, the worst-case switching pattern varies from different levels of interconnect and different working frequencies. Hence, as inductance cannot be neglected in today's high-speed circuit design, it is very important to consider the **RLC** effects with different bus parameters to develop the proper encoding schemes for bus delay reduction.

In Chapter 2, we proposed a *bus-invert* method to reduce the worst-case on-chip bus delay with the dominance of the inductance coupling effect. Simulation results have shown that our encoding method can significantly reduce the worst coupling delay of the buses with strong inductive coupling.

From our simulation results, we observe that the worst-case switching pattern could vary with given design parameters considering the **RLC** effects of interconnects. However, the proposed *bus-invert* method can be utilized to reduce the bus coupling delay only when the inductance effects dominate. Therefore, in Chapter 3, we proposed a flexible encoding scheme for on-chip buses that can consider the given parameters to reduce the **LC** coupling delay. In addition, with some modification, this new encoding scheme can be utilized to predict and lengthen the signal propagation length of a bus under the given constraints. Simulation results are also given to support our claims.

In Chapter 4, to further reduce the coupling noise, we proposed a joint shield insertion and bus encoding scheme for global bus design in nanometer technologies. With the user-given bus parameters, the working frequency, the scheme can effectively reduce the *LC* coupling effects and hence, minimize the bus coupling delay. Simulation results show that the proposed scheme can significantly reduce the coupling delay.

Recently, the power consumption of on-chip interconnect is another crucial issue in high performance circuit designs. In Chapter 5, we first proposed a flexible encoding scheme to minimize the power consumption of buses under given delay constraints. To further improve the performance of the encoding scheme (further reduce the power consumption), we also utilized the joint shield insertion and bus encoding method to solve the problem. With the user-given bus parameters, the working frequency, and the delay constraint, both schemes can minimize the bus power consumption subject to the delay constraint by effectively reducing the *LC* coupling effects. Simulation results show that the proposed schemes can significantly reduce the coupling delay and the power consumption of a bus according to the delay constraint. In addition, the joint shield insertion and bus encoding method can gain more power reduction than that gained by only using the flexible bus encoding scheme.

In the future, we intend to incorporate the repeater insertion technique in our joint shield insertion and bus encoding scheme to further reduce the bus delay and power consumption. In addition, the ground bounce effect can also be considered and minimized in our future work. Consider the switching probabilities of the bus wires, we can also try to generate a specific code set for the bus used in ASIC to optimize the bus power and delay. If those issues can be considered, our encoding scheme will be more powerful.



Bibliography

- [1] Semiconductor Industry Association, *National Technology Roadmap for Semiconductors*, 1997.
- [2] Semiconductor Industry Association, *International Technology Roadmap for Semiconductors*, 2003.
- [3] L. Trevillyan, D. Kung, R. Puri, L. N. Reddy, and M. A. Kazda, "An Integrated Environment for Technology Closure of Deep-Submicron IC Designs," *IEEE Design and Test of Computers*, vol. 21, issue 1, pp. 14–22, Jan. –Feb., 2004.
- [4] P. P. Sotiriadis and A. P. Chandrakasan, "Bus Energy Reduction by Transition Pattern Coding Using a Detailed Deep Submicrometer Bus Model," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, issue 10, pp. 1280–1295, Oct., 2003.
- [5] M. A. Elgamel and M. A. Bayoumi, "Interconnect Noise Analysis and Optimization in Deep Submicron Technology," *IEEE Circuits and Systems Magazine*, vol. 3, issue 4, pp. 6–17, 2003.
- [6] M. H. Chowdhury, Y. I. Ismail, C. V. Kashyap, and B. L. Krauter, "Performance Analysis of Deep Sub micron VLSI Circuits in the Presence of Self and Mutual Inductance," *IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 197–200, 2002.
- [7] Y. I. Ismail, "On-Chip Inductance Cons and Pros," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 10, issue 6, pp. 685–694, Dec. 2002.
- [8] L. He and K. M. Lepak, "Simultaneous Shield Insertion and Net ordering for Capacitive and Inductive Coupling Minimization," *International Symposium on Physical Design*, pp. 55–60, 2000.
- [9] K. Hirose and H. Yasuura, "A bus delay reduction technique considering crosstalk," *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, pp. 441–445, March 2000.
- [10] R. Escovar and R. Suaya, "Optimal Design of Clock Trees for Multigigahertz Applications," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, issue 3, pp. 329–345, March, 2004.
- [11] K. Nabors and J. White, "FastCap: A Multipole Accelerated 3-D capacitance extraction program," *IEEE Trans. Computer-Aided Design*, vol. 10, No. 11, pp. 1447–1459, Nov. 1991.
- [12] M. Kamon, M. J. Tsuk, and J. K. White, "FastHenry: a Multipole-accelerated 3D Inductance Extraction Program," *IEEE Trans. Computer-Aided Design*, pp. 1750–1758, Sept. 1994.

- [13] H. Kriplani, F. N. Najm, and I. N. Hajj, "Pattern Independent Maximum Current Estimation in Power and Ground Buses of CMOS VLSI Circuits: Algorithms, Signal Correlations, and Their Resolution," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, issue 8, pp. 998–1012, Aug. 1995.
- [14] J. Chen and L. He, "Determination of Worst-Case Crosstalk Noise for Non-Switching Victims in GHz+ Interconnects," *Proceedings of Asia and South Pacific Design Automation Conference*, pp. 162–167, Jan., 2003.
- [15] C. K. Cheng, J. Lillis, S. Lin, and N. Chang, *Interconnect Analysis and Synthesis*, John Wiley & Sons, Inc., 2000.
- [16] D. K. Cheng, *Filed and Wave Electromagnetics*, 2nd Ed., Addison-Wesley, 1989.
- [17] M. W. Beattie and L. T. Pileggi, "Inductance 101: modeling and extraction," *Proceedings of Design Automation Conference*, pp. 323–328, 2001.
- [18] Y. Cao, X. Huang, N. H. Chang, S. Lin, O. S. Nakagawa, W. Xie, D. Sylvester, and C. Hu, "Effective On-Chip Inductance Modeling for Multiple Signal Lines and Application to Repeater Insertion," *International Symposium on Quality Electronic Design*, pp. 185–190, Mar. 2001.
- [19] Y. I. Ismail, E. G. Friedman, and J. L. Neves, "Figures of Merit to Characterize the Importance of On-Chip Inductance," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 7, issue 4, pp. 442–449, Dec. 1999.
- [20] M. Stan and W. Burleson, "Bus-invert coding for low-power I/O," *IEEE Transactions on Very Large Scale Integrated Systems*, vol. 3, issue 2, pp. 49–58, Mar. 1995.
- [21] M. Stan and W. Burleson, "Low-power encodings for global communication in CMOS VLSI," *IEEE Transactions on Very Large Scale Integrated Systems*, vol. 5, issue 12, pp. 444–455, Dec. 1997.
- [22] Y. Shin, S. I. Chae, and K. Choi, "Partial bus-invert coding for power optimization of system level bus," *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 127–129, 1998.
- [23] Y. Zhang, J. Lach, K. Skadron, M. Stan, "Odd/even bus invert with two-phase transfer for buses with coupling," *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 80–83, 2002.
- [24] T. Lindkvist, J. Lofvenberg, and O. Gustafsson, "Deep submicron bus invert coding," *Proceedings of the Nordic Signal Processing Symposium*, pp. 133–136, June 2004.
- [25] H. Kaul, D. Sylvester, M. A. Anders, and R. K. Krishnamurthy, "Design and analysis of spatial encoding circuits for peak power reduction in on-chip buses," *IEEE Transactions on Very Large Scale Integrated Systems*, vol. 13, issue 11, pp.

- 1225–1238, Nov. 2005.
- [26] H. S. Deogun, R. R. Rao, D. Sylvester, and D. Blaauw, “Leakage-and crosstalk-aware bus encoding for total power reduction,” *IEEE/ACM Design Automation Conference*, pp. 779–782, June, 2004.
- [27] S. R. Sridhara and N. R. Shanbhag, “A low power bus design using joint repeater insertion and coding,” *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 99–102, 2005.
- [28] P. P. Sotiriadis and A. P. Chandrakasan, “Reducing bus delay in submicron technology using coding,” *Proceedings of the Asia and South Pacific Design Automation Conference*, pp. 109–114, Feb. 2001.
- [29] B. Victor and K. Keutzer, “Bus encoding to prevent crosstalk delay,” *International Conference on Computer Aided Design*, pp. 57-63, Nov. 2001.
- [30] K. H. Baek, K. W. Kim, and S. M. Kang, “A low energy encoding technique for reduction of coupling effects in SOC interconnects,” *Proceedings of the 43rd IEEE Midwest Symposium on Circuits and Systems*, pp. 80–83, Aug. 2000.
- [31] M. Cha, C. G. Lyuh, and T. Kim, “Resource-constrained low-power bus encoding with crosstalk delay elimination,” *Proceedings of the Asia and South Pacific Design Automation Conference*, pp. 835–838, Jan. 2004.
- [32] M. Lampropoulos, B. M. Al-Hashimi, and P. Rosinger, “Minimization of crosstalk noise, delay and power using a modified bus invert technique,” *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, pp. 1372–1373, Feb. 2004.
- [33] D. Chinnery and K. Keutzer, *Closing the Gap Between ASIC & Custom - Tools and Techniques for High-Performance ASIC Design*. Kluwer, 2002.
- [34] R. Ho, K.W. Mai, and M. A. Horowitz, “The Future of Wire,” *Proceedings of the IEEE*, vol. 89, issue 4, pp. 490–504, April 2001.
- [35] J. Cong, “An Interconnect-Centric Design Flow for Nanometer Technologies,” *Proceedings of the IEEE*, vol. 89, issue 4, pp. 505–528, April 2001.
- [36] L. O. Chua, C. A. Desoer, and E. S. Kuh, *Linear and nonlinear circuits*, McGraw-Hill Inc, 1987.
- [37] D. Matzke, “Will physical scalability sabotage performance gain?” *IEEE Computer*, pp. 37–39, Sept. 1997.
- [38] Y. Cao, C. Hu, X. Huang, A.B. Kahng, S. Muddu, D. Stroobandt, and D. Sylvester, “Effects of global interconnect optimizations on performance estimation of deep submicron design,” *IEEE/ACM International Conference on Computer Aided Design*, pp. 56 – 61, Nov. 2000.
- [39] M. A. Elgamel, A. Kumar, and M. A. Bayoumi, “Efficient shield insertion for inductive noise reduction in nanometer technologies,” *IEEE Transaction on*

- Very Large Scale Integration Systems*, vol. 13, issue 3, pp. 401–405, Mar. 2005.
- [40] H. KuaI, D. Sylvester, and D. Blaauw, “Clock net optimization using active shielding,” *Proceedings of the 29th European Solid-State Conference*, pp. 265–268, Sept. 2003.
- [41] J. Cong, L. He, C. K. Koh, and Z. Pan, “Interconnecting sizing and spacing with consideration of coupling capacitance,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp.1164-1169, Sept. 2001.
- [42] M. A. El-Moursy and E. G. Friedman, “Optimum wire sizing of RLC interconnect with repeaters,” *Proceedings of the IEEE Great Lakes Symposium on VLSI*, pp. 27–32, April 2003.
- [43] 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, issue 5, pp. 471–481, May 2001.
- [44] L. P. P. van Ginneken, “Buffer Placement in distributed RC-tree networks for minimal Elmore delay,” *Proceedings of the International Symposium on Circuits and Systems*, pp. 865–868, 1990.
- [45] C. J. Alpert, A. Devgan, and S. T. Quay, “Buffer insertion for noise and delay optimization,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp.1633–1645, Nov. 1999.
- [46] A. Nalamalpu, S. Srinivasan, and W. Burleson, “Boosters for driving long on-chip interconnects: design issues, interconnect synthesis and comparison with repeaters,” *IEEE Transactions on Computer-Aided Design*, vol. 21, issue 1, pp. 50–62, Jan. 2002.
- [47] H. Zhang, V. George, and J. Rabaey, “Low-swing on-chip signaling techniques: effectiveness and robustness,” *IEEE Transactions on Very Large Scale Integration Systems*, vol. 8, issue 3, pp. 264–272, June 2000.
- [48] Y. Massoud, J. Kawa, D. MacMillen, and J. White, “Modeling and analysis of differential signaling for minimizing inductive crosstalk,” *Proceedings of the Design Automation Conference*, pp. 804–809, 2001.
- [49] Y. I. Ismail and E.G. Friedman, “Effects of inductance on the propagation delay and repeater insertion in VLSI circuits: a summary,” *IEEE Circuits and Systems Magazine*, vol. 3, issue 1, pp. 24–28, 2003.
- [50] M. H. Chowdhury, Y. I. Ismail, C. V. Kashyap, and B. L. Krauter, “Performance analysis of deep sub micron VLSI circuits in the presence of self and mutual inductance,” *IEEE Internstional Symposium on Circuits and Systems*, vol. 4, pp. IV-197–IV-200, May 2002.
- [51] T. Xue, E. Kuh, and D. Wang, “Post global routing crosstalk synthesis,” *IEEE*

- Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, pp. 1418–1430, Dec. 1997.
- [52] G. Zhong, C. K. Koh, and K. Roy, “A Twisted-Bundle Layout Structure for minimizing Inductive Coupling Noise,” *IEEE International Conference on Computer Aided Design*, pp. 406–411, 2000.
- [53] H. Kual, D. Sylvester, and D. Blaauw, “Performance optimization of critical nets through active shielding,” *IEEE Transactions on Circuits and Systems*, vol. 51, issue 12, pp. 2417–2435, 2004.
- [54] H. Kual, D. Sylvester, and D. Blaauw, “Active shielding of RLC global interconnects,” *IEEE European Solid-State Circuits Conference*, pp. 265–268, 2003.
- [55] D. Deutsch, G. Kopcsay, P. Restle, H. Smith, G. Katopis, W. Becker, P. Coteus, C. Surovic, B. Rubin, R. Dunne, T. Gallo, K. Jenkins, L. Terman, R. Dennard, G. Sai-Halasz, B. Krauter, and D. Knebel, “When are transmission-line effects important for on-chip interconnects?,” *IEEE Transactions on Microwave Theory Technique*, vol. 45, pp. 1836–1846, Oct. 1997.
- [56] L. Pileggi, “Coping with RC(L) interconnect design headaches,” *Proceedings of International Conference on Computer Aided Design*, Nov. 1995.
- [57] V. Soteriou and L. S. Peh, “Design-space exploration of power-aware on/off interconnection networks,” *Proceedings of the IEEE International Conference on Computer Design*, pp. 510–517, 2004.
- [58] S. R. Sridhara, A. Ahmed, and N. R. Shanbhag, “Area and energy-efficient crosstalk avoidance codes for on-chip buses,” *Proceedings of the IEEE International Conference on Computer Design*, pp. 12–17, 2004.
- [59] P. Subrahmanya, R. Manimegalai, V. Kamakoti, and Madhu Mutyam, “A bus encoding technique for power and cross-talk minimization,” *Proceedings of the 17th International Conference VLSI Design*, pp. 443–448, 2004.
- [60] P. P. Sotiriadis and A. Chandrakasan, “Low power bus coding techniques considering inter-wire capacitance,” *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 507–510, 2000.

作者簡介

About the Author

姓名	中文：涂尚璋	英文：Shang-Wei Tu
基本資料	生日：1977/04/10	籍貫：台灣省台南市
學歷	國立交通大學電子研究所博士班	2001/09 ~ 2006/09
	國立交通大學電子研究所碩士班	1999/09 ~ 2001/06
	國立交通大學電子工程學系	1995/09 ~ 1999/06
經歷	數位電路與系統助教	2006
	九十四年度交通大學電子所年度論文獎佳作	2005
	計算機輔助設計特論助教	2003 ~ 2005
	交通大學電子研究所博士班甄試第一名	2001
	程式語言助教	2000
	研究所學期成績優異獎	1999
	凌陽科技公司暑期實習	1999
	教育部大學校院積體電路電腦輔助設計軟體製作競賽 佳作	1999
	教育部大學校院積體電路電腦輔助設計軟體製作競賽 入圍	1999

研究領域

Circuit Modeling

Computer-Aided Design

Algorithms

專 長

EDA Algorithms

Hardware Description Language: Verilog

Programming Language: C/C++



著作目錄

涂尚瑋

Publication List

Shang-Wei Tu

依新法記點

國外期刊(共 4 點)

- (2 點, 長文) 1. **S. W. Tu**, W. Z. Shen, Y. W. Chang, T. C. Chen, and J. Y. Jou, "Inductance modeling for on-chip interconnects," *Journal of Analog Integrated Circuits and Signal Processing*, pp. 65-78, vol. 35, No. 1, April, 2003. (Invited paper for ISCAS'2002 special issue on analog and mixed signal circuits.)
- (2 點, 短文) 2. **Shang-Wei Tu**, Yao-Wen Chang, and Jing-Yang Jou, "RLC coupling-aware simulation and on-chip bus encoding for delay reduction," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2005. (Accepted)

國際會議(至多採計一篇)

- (1 點) 1. **S. W. Tu**, W. Z. Shen, Y. W. Chang, and T. C. Chen, "Inductance modeling for on-chip interconnects," *Proceedings of IEEE International Symposium on Circuits and System*, Vol. 3, pp. 787-790, 2002.
- (1 點) 2. **Shang-Wei Tu**, Jing-Yang Jou, and Yao-Wen Chang, "Layout techniques for on-chip interconnect inductance reduction," *Proceedings of the Asia South Pacific Design Automation Conference*, pp. 269-273, January, 2004.
- (1 點) 3. **Shang-Wei Tu**, Jing-Yang Jou, and Yao-Wen Chang, "RLC effects on worst-case switching pattern for on-chip buses," *Proceedings of IEEE International Symposium on Circuits and System*, pp. II 945-948, 2004.
- (1 點) 4. **Shang-Wei Tu**, Jing-Yang Jou, and Yao-Wen Chang, "RLC coupling-aware simulation for on-chip buses and their encoding for delay reduction," *Proceedings of IEEE International Symposium on Circuits and System*, 2005.
5. Jiun-Sheng Huang, **Shang-Wei Tu**, and Jing-Yang Jou, "On-chip bus encoding for LC cross-talk reduction," *IEEE International Symposium on VLSI Design, Automation and Test*, 2005.

待審論文

1. **Shang-Wei Tu**, Jiun-Sheng Huang, and Jing-Yang Jou, “Flexible on-chip bus encoding for LC crosstalk reduction,” *IEEE Transactions on Very Large Scale Integration Systems*, 2006. (Submitted)
2. **Shang-Wei Tu**, Jiun-Sheng Huang, Juinn-Dar Huang, and Jing-Yang Jou, “Increasing bus propagation length in deep submicron technology using coding,” *ACM Transactions on Large Design Automation Electronic Systems*, 2006. (Submitted)
3. Tzu-Wei Lin, **Shang-Wei Tu**, and Jing-Yang Jou, “On-chip bus encoding for power minimization under delay constraint,” *Proceedings of the Asia South Pacific Design Automation Conference*, 2007. (Submitted)

國內會議

1. **Shang-Wei Tu**, Wen-Zen Shen, Yao-Wen Chang, and Tai-Chen Chen, “On-chip inductance modeling for the coplanar interconnect structure,” *Proceedings of the 12th VLSI Design/CAD Symposium*, section A1, A1-6, August, 2001.
2. **Shang-Wei Tu**, Jing-Yang Jou, and Yao-Wen Chang, “Layout techniques for minimizing on-chip interconnect inductance,” *Proceedings of the 14th VLSI Design/CAD Symposium*, section A1, A1-8, August, 2003.
3. Tzu-Wei Lin, **Shang-Wei Tu**, and Jing-Yang Jou, “Flexible on-chip bus encoding for power minimization under delay constraints,” *Proceedings of the 17th VLSI Design/CAD Symposium*, 2006. (Accepted)