PAPER

BAT: Performance-Driven Crosstalk Mitigation Based on Bus-grouping Asynchronous Transmission

Guihai YAN[†a)], ***Nonmember***, **Yinhe HAN**[†], ***Member***, **Xiaowei LI**[†], ***and*** **Hui LIU**[†], ***Nonmembers***

SUMMARY Crosstalk delay within an on-chip bus can induces a severe transmission performance penalty. Bus-grouping Asynchronous Transmission (BAT) scheme is proposed to mitigate the performance degradation. Furthermore, considering the distinct spatial locality of transition distribution on some types of buses, we use the locality to optimize the BAT. In terms of the implementation, we propose the Differential Counter Cluster (DCC) synchronous mechanism to synchronize the data transmission, and the Delay Active Shielding (DAS) to protect some critical signals from crosstalk and optimize the routing area overhead. The BAT is scalable with the variation of bus width with little extra implementation complexity. The effectiveness of BAT is evaluated by focusing on the on-chip buses of a superscalar microprocessor simulator using the SPEC CPU2000 benchmarks. When applied to a 64-bit on-chip instruction bus, the BAT scheme, compared with the conservative approach, Codec and Variable Cycle Transmission (DYN) approaches, improves performance by $55^{+}\%$, $10^{+}\%$, $30^{+}\%$, respectively, at the expense of 13% routing area overhead.

key words: *crosstalk delay, on-chip buses, bus-grouping transmission, asynchronous, shielding*

1. Introduction

With the technology being pushed to nano-meter geometries, the coupling capacitance of interconnect buses is growing to dominate the total capacitance [11], which causes severe propagation delay on on-chip buses [1] and thereby results in the performance bottleneck of modern digital system in many situations.

Many approaches have been proposed to tackle this problem. Those approaches can be classified into two major categories: *Shielding* [5] [10] and *Codec* [9] [3].

The *Shielding* approaches include "passive shield" and "active shield" [5]. Both types of approaches use the effective Miller capacitance to reduce the total effective capacitance between neighbor bus wires. In general, the *"active shield"* can achieve better shielding performance than "passive shield" at the cost of more complex layout design and power dissipation, and both impose considerable area overhead.

Although shielding schemes can be used to protect a few critical signal wires from crosstalk, it is inappropriate to adopt these schemes to masses of connect wires due to high area overhead in many situations.

[†]The authors are with the Institute of Computing Technology, Chinese Academy of Science, Beijing, 100080, China, and with Graduate School of Chinese Academy of Sciences, Beijing, 100039, China.

a) E-mail: yan_guihai@ict.ac.cn

The *Codec* approaches are based on the key observation: crosstalk delay depends on different pattern transitions—some Crosstalk-Sensitive (CS) transitions induce longer delay, while the other Crosstalk-Insensitive (CI) transitions induce shorter delay. Thus, the crosstalk delay can be reduced by encoding the CS patterns to the CI patterns [9].

To adopt the *Codec* approaches, two primary questions need to be answered: 1) How much overhead (the number of extra bits) do they impose to make the buses obtain immunity against depressing crosstalk delay? 2) How to implement the practical codec methods? To the first question, B.Victor et al, through comprehensive mathematic computing, have proved that the lower bound of overhead is about 44% [3]. This result implies significant area-inefficiency. To the second question, besides the searching-and-selecting approaches in full code space, there are few comprehensive mathematic-based construction methods. In other words, it is very hard to construct a practical codec strategy, especially for a large bus width (such as 64 or wider) where searching-and-selecting approaches are too time-consuming to put into practical use.

Besides the above conventional schemes, there are a few of new schemes:

Delay-line [7]: Considering that skewing the switch time of CS transitions can also reduce the effective Miller capacitance, thereby mitigating the crosstalk delay, M. Ghoneima et al. presented *Delay-line* scheme. However, it needs lots of complicated synchronizing and calibrating operations.

Variable Cycle Transmission(DYN) [6]: DYN handles the data transmission through *variable cycle assignment*—the CS transitions are assigned more transmission cycles, while the CI transitions are assigned less transmission cycles. If the bus width is relatively narrow, DYN, generally, can significantly improve performance.

Unfortunately, as the bus width increases to meet the high bandwidth transmission, the CS transitions tend to be more common, which makes the dynamic cycle assignment scheme less efficient. This phenomenon results from typical *"cask effect"*.

To sum up, the above approaches, according to the different means to deal with the Crosstalk-Sensitive transitions, either absolutely **eliminate** the emergency of CS transitions [9][3], or **mitigate** the negative ef-

fects of CS transitions [5][7][6]. The former is at the expense of significant area overhead (extra wire placement and codec logic) which is unacceptable due to the limited on-chip area and strict layout design in many situations, while the later is at the cost of complexity of transmission structure (of course some moderate area overhead). However, the complexity of the transmission structure might be reduce to an acceptable level through developing some dedicated schemes.

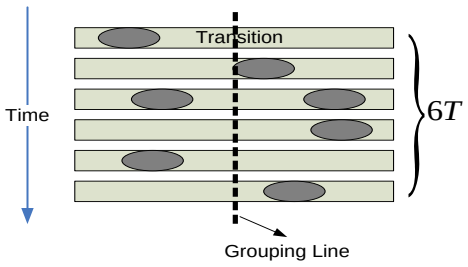
In this paper, we alleviate the transmission performance degradation by employing the "**mitigate**" strategy. We propose a new scheme by extending the *Variable Cycle Transmission* (DYN) scheme, called *Bus-grouping Asynchronous Transmission* (BAT).

The key idea of BAT can be illustrated in Fig.1. Every row denotes a pattern transition, the shaded ovals denote the Crosstalk Sensitive (CS) parts in these transitions. In this example there are six pattern transitions, and every transition encounters at least one CS part (denoted as shaded ovals). The transmission time is determined by the CS parts.

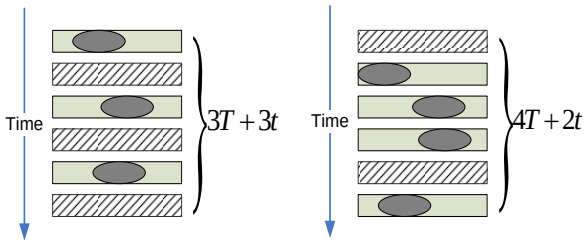
Applying the DYN approach, we will encounter the CS transitions in all transitions, which make this approach fail to improve the transmission performance. Assume that the CS-free transitions consume t period of time and the CS-contained transitions consume T period of time, where $t < T$. In the example, we need $6T$ to complete these patterns transmission. However, if we divide the bus into two subgroups, then it can be seen not all sub-transitions encounter the CS transitions (denoted as the striped transitions in Fig.1(b)). Then, we employe the DYN approach on the two group of sub-transitions, respectively. In the way of transmission, the more essential change is that the two group sub-transitions are independently transmitted on the bus (**asynchronous**), and assembled in the receiver. In the example, assuming there is sufficient buffer space at the receiver, we just require $\{4T + 2t\}$ $(\max\{3T + 3t, 4T + 2t\})$ to complete the original six transitions. The transmission performance is improved.

In particular, the main contributions of this work are as follows:

- We propose the Bus-grouping Asynchronous Transmission (BAT) scheme. Through dividing the original bus into proper-grain sub-buses by placing some elaborate shielding wires and applying the variable cycle transmission (DYN) scheme [6] on them, the potential performance can be exploited as much as possible.
- We use Crosstalk Factor (CF) to guide the bus-grouping scheme. We find that some types of buses, such as instruction buses, have distinct spatial locality of crosstalk effects. To evaluate the impact of crosstalk effects, we present the metric of Crosstalk Factor (CF). Through analyzing the bus CF distribution, we can optimize the BAT scheme.



(a) Transitions before Bus-grouping



(b) Transitions after Bus-grouping

Fig. 1 BAT Scheme. Each transition without CS part consumes time t, and with CS part consumes time T, where, $t < T$.

- We present the Differential Counter Cluster (DCC) synchronous mechanism and the Delay Active Shielding (DAS) scheme in terms of the implementation of the BAT scheme. The DAS scheme employs the *Active Shield* scheme [5] and *Delayed Line* scheme [7]. Both DCC and DAS are efficient and easy-implemented.

The rest of this paper is organized as follows: Section 2 analyzes the DSM bus transmission performance characteristics and presents the proposed BAT scheme. Section 3 presents the BAT implementation. The evaluation results are shown in section 4. Finally, Section 5 concludes this paper.

2. Proposed BAT Scheme

2.1 Bus Model

The DSM bus model is illustrated in Fig.2. The C_I denotes the capacitance between two adjacent wires and the C_L denotes the capacitance between a wire and the substrate. The aspect ratio (λ) is defined as $\lambda = C_I/C_L$.

Sotiriadis et al. have presented a comprehensive analysis of the delay estimation for coupled lines and multiple drivers based on Elmore delay [9][8]. The essential conclusions are depicted in Table 1 (Note that the last column is not included in Sotiriadis analysis results). All types of transitions between two (for boundary lines) or three (for intermediate lines) adjacent lines

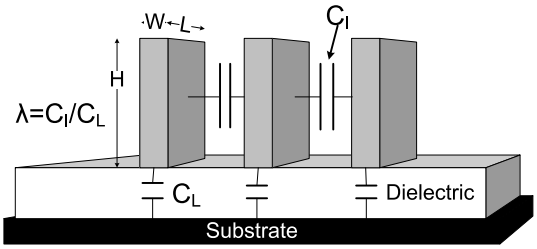


Fig. 2 Bus Model

are divided into six groups by the different delay factors, and then classified to four classes according to the level of significant delay vector. As Li et al. adopted in [6], we assign transmission cycles to different type of transitions by the different classes . For instance, if a transition is included in Class-II, then this transmission will consume 2 cycles.

Table 1 Delay factor of the victim lines and assigned transmission cycles (The symbols $-$, $\uparrow$ and $\downarrow$ stand for no, positive and negative transitions, respectively.)

Group	*D.Factor*	*Transition Type*	*Class*	*Cycles*
The victim lines belong to intermediate lines				
1	0	$(-,-,-)(\uparrow,-,\downarrow)$ $(\downarrow,-,\uparrow)(\uparrow,-,\uparrow)$ $(\downarrow,-,\downarrow)(-,-,\uparrow)$ $(-,-,\downarrow)(\uparrow,-,-,)$ $(\downarrow,-,-)$	I	1
2	1	$(\uparrow,\uparrow,\uparrow)(\downarrow,\downarrow,\downarrow)$		
3	$1+\lambda$	$(-,\uparrow,\uparrow)(-,\downarrow,\downarrow)$ $(\uparrow,\uparrow,-)(\downarrow,\downarrow,-)$		
4	$1+2\lambda$	$(-,\uparrow,-)(-,\downarrow,-)$ $(\uparrow,\uparrow,\downarrow)(\uparrow,\downarrow,\downarrow)$ $(\downarrow,\uparrow,\uparrow)(\downarrow,\downarrow,\uparrow)$	II	2
5	$1+3\lambda$	$(-,\uparrow,\downarrow)(\uparrow,\downarrow,-)$ $(\downarrow,\uparrow,-)(-,\downarrow,\uparrow)$	III	3
6	$1+4\lambda$	$(\downarrow,\uparrow,\downarrow)(\uparrow,\downarrow,\uparrow)$	IV	4
The victim lines belong to boundary lines (the left line is the victim line.)				
1	0	$(-,-)(-,\uparrow)(-,\downarrow)$		
2	1	$(\uparrow,\uparrow)(\downarrow,\downarrow)$	I	1
3	$1+\lambda$	$(\uparrow,-)(\downarrow,-)$		
4	$1+2\lambda$	$(\downarrow,\uparrow)(\uparrow,\downarrow)$	II	2

2.2 "Cask Effect" and Bus-grouping

Consider a n-bit bus and a sequence of patterns: $P_1, P_2, \ldots, P_k$, where $P_i = \{v_i^1, v_i^2, \ldots, v_i^n\}$ $(0<i<k)$. The transition between the two consecutive patterns P_i, P_{i+1} is denoted as $P_i \rightarrow P_{i+1}$. As indicated in Table 1, this transition is evaluated as one of the four transitions classes: Class-I, Class-II, Class-III and Class-IV (listed in the 4^{th} column of Table 1).

The corresponding delay vector of the transition $P_i \rightarrow P_{i+1}$ is defined as $D_i = \{d_i^1, d_i^2, \ldots, d_i^n\}$, where d_i^j $(1<j<n)$ denotes the theoretical required number of cycles to transmit the j^{th} bit of P_{i+1}. Clearly, the required cycles to transmit the P_{i+1} succeeding P_i is $\max\{d_i^1, d_i^2, \ldots, d_i^n\}$. So, we define the *Required Transmission Cycles (RTC)* of P_{i+1} as

$$RTC^i = \max D_i = \max\{d_i^1, d_i^2, \ldots, d_i^n\}. \quad (1)$$

For example, there is a transition between the two patterns: $\{01001001\} \rightarrow \{00110100\}$. The corresponding transition is $\{-\downarrow\uparrow\uparrow\downarrow\uparrow-\downarrow\}$, and the delay vector D is $\{13224311\}$. According to Table 1, this transition belongs to the Class-IV, so we need four cycles to complete this transmission.

"Cask Effect": From (1) we find that if any one element of the D_i is equal to the delay factor of "worst" transition, the whole pattern transmission has to endure the most conservative situation in spite of the fact that the transitions on other lines might be very crosstalk insensitive.

In order to mitigate the "cask effect", we propose the *Bus-grouping Asynchronous Transmission (BAT)* scheme: Divide every pattern into sub-patterns at the transmitter, asynchronously transmit these sub-patterns, and then assemble these sub-patterns to original pattern at the receiver.

To evaluate transmission performance, we develop a metric T_{total} which is the total time to transmit the k patterns $(P_1, P_2, \ldots, P_k)$, where

$$T_{total} = \sum_{i=0}^{k-1} RTC^i. \quad (2)$$

Assuming a bus is divided into two sub-buses at the s^{th} bit[†], the corresponding two sub-patterns are $\{P_i^1, P_i^2, \ldots, P_i^s\}$ and $\{P_i^{s+1}, P_i^{s+2}, \ldots, P_i^n\}$, where $i = 1, 2, \ldots, k$. The required total time of transmitting the two sets of sub-patterns is evaluated respectively as follows:

$$T_{total}^1 = \sum_{i=0}^{k-1} RTC_1^i, \quad T_{total}^2 = \sum_{i=0}^{k-1} RTC_2^i.$$

If there are sufficient buffers at the receiver, we just need T'_{total} to transmit the k patterns. Where

$$T'_{total} = \max\{T_{total}^1,\ T_{total}^2\}. \quad (3)$$

The condition that $T'_{total} < T_{total}$ is always satisfied in practical situations; thus, the performance can be improved by adopting BAT scheme. Furthermore, it can be inferred *if we divide the original bus into more fine-grain sub-buses, we can obtain better transmission performance*, at the expense of more area overhead. So there is an intrinsic tradeoff between performance and area overhead.

2.3 Optimizing BAT

It is a reasonable grouping strategy to equally divide

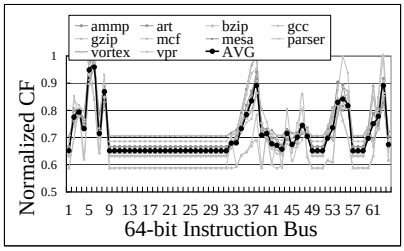
[†]The d_i^s and d_i^{s+1} may be altered since we place a shielding line between s^{th} and $s+1^{th}$ lines, so we need to reevaluate the d_i^s and d_i^{s+1}.

a bus into identical sub-buses if transmitted data is according roughly uniform distribution. However, if the data distribution exhibits some kinds of spatial locality, this equally grouping strategy is not efficient enough. Taking advantage of the locality, we could achieve higher performance by dividing the original bus into several sub-buses with different width.

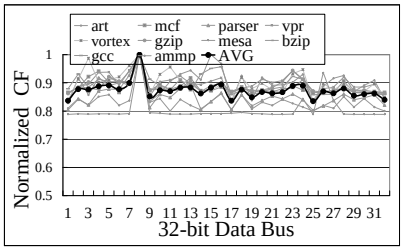
We define *Crosstalk Factor (CF)* to describe the spatial locality. If some lines have higher *CF*, it tends to suffer severer crosstalk delay. The *CF* can be reflected in the delay matrix which is composed of D_i. The *CF* of the j^{th} line is denoted as

$$CF_j = \sum_{i=1}^{k} d_i^j.$$

In our observation, through 10 SPEC CPU2000 benchmarks simulating, we find that the *CF* distribution of different benchmarks is similar in trends with each other on the same bus, and the CF distribution on different buses shows distinct difference between each other (such as between instruction bus and data bus in the experiment). Furthermore, in contract the *CF* locality on data bus, the *CF* locality on instruction bus is more distinct, as shown in Fig.3.



(a) CF distribution on 64-bit instruction bus

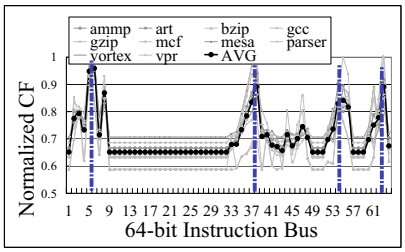


(b) CF distribution on 32-bit data bus

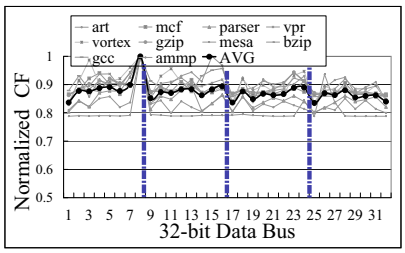
Fig. 3 CF distribution on instruction bus and data bus

There are four obvious *CF* peaks in Fig.3(a), which results from the adopted instruction format. In the experiment, we adopts 3-operand instruction format. There are 4 "activity fields" in this format: 3 operands fields and 1 opcode field, which is popular in modern superscalar architecture.

We can use the locality to implement more efficient grouping strategy than the identical grouping strategy for this type of buses on which transmitted data exhibits distinct locality. Group the bus into sub-buses at these points where the distribution of CF presents "peak"s. As Fig.4 shown, we divide the instruction bus into sub-buses with different width and the data bus into sub-buses with identical width.



(a) CF distribution on 64-bit instruction bus



(b) CF distribution on 32-bit data bus

Fig. 4 Bus-grouping on instruction bus and data bus

The bus-grouping is realized by placing several delicate shielding lines into the original bus. **Although the shielding lines placement seems application-specific in Fig.4, the basic idea can be extend to general situations**. Because in practical design processes, the *CF* distribution can be obtained by running a high-level system simulator(such as a C/C++ simulator). This work can be done in early design phases of chip design process, and the locality characteristic can be used to guide high performance bus fabrication.

3. Implementation of BAT

The transmission structure is illustrated in Fig.5. It is composed of a Crosstalk Delay Analysis (CDA) module, a data Valid-indicating Signal Generator (VSG), a Differential Counter Cluster (DCC), a FIFO (First In First Out) Buffer Pool (BP) and a sub-pattern Assembler.

The DCC component is responsible for synchronizing the sub-buses transmission. The CDA analyzes the data transition, and conveys the crosstalk information to the VSG module. The VSG module, combining with the synchronization information generated by the DCC module, can synthesize a set of data valid-indicating signals which indicate the receiver to sample the sub-patterns. The work of the VSG is delayed one cycle

after the work of the CDA. At the receiver, the Assembler can "assemble" these sub-patterns taken apart at the transmitter to original patterns as long as there are no empty buffers.

The following shows more details about the DCC synchronous mechanism. Additionally, in view of the importance of protecting data valid-indicating signals from other signals influence by crosstalk, we will detail the proposed shielding scheme—Delay Active Shielding(DAS).

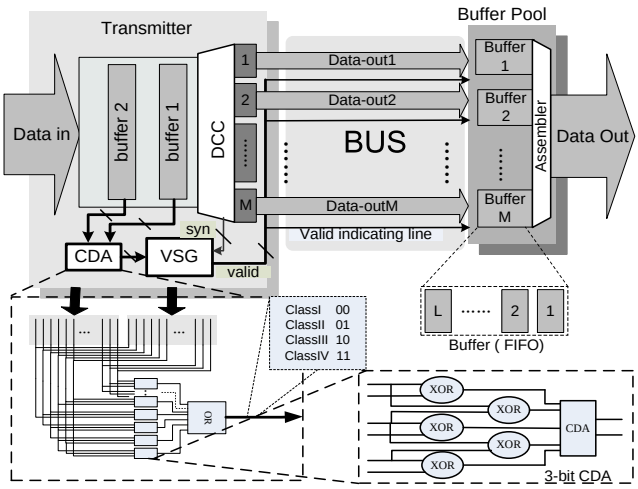


Fig. 5 BAT Structure

3.1 DCC Synchronous Mechanism

These receiver buffers have to be guarantied not overflow. This work can be done at the transmitter since we known the differential numbers of transmitted patterns on every pair of sub-buses.

Assuming these buffers length is L, a set of bidirectional counters, ranging from $-L$ to L, are required. These counters, initialed to zero, record the differential numbers of patterns transmitted on every pair of sub-buses. For instance, $C(i,j)$ is such a counter monitoring the transmission state of the i^{th} group and the j^{th} group. If a valid sub-pattern transmission is completed through the i^{th} group sub-bus, $C(i,j)$ is increased by one for $j = 1, 2, \ldots, i-1, i+1, \ldots, n$; and, if a valid sub-pattern transmission is completed through the j^{th} group sub-bus, $C(i,j)$ is decreased by one for $i = 1, 2, \ldots, j-1, j+1, \ldots, n$. When $C(i,j)$ overflow, stop the i^{th} group sub-bus transmission and hold its state. When $C(i,j)$ underflow, stop the j^{th} group sub-bus and hold its state. We employ a counter "tree" to implement this counter cluster. The synchronous logic can be explained as follows (*'OF' short for 'OverFlow', 'UF' short for 'UnderFlow', '+' means logical OR*):

- Hold i^{th} sub-bus, if and only if $\{\mathrm{OF}(C(i,1)) + \mathrm{OF}(C(i,2)) + \cdots + \mathrm{OF}(C(i,i-1)) + \mathrm{OF}(C(i,i+1)) + \cdots + \mathrm{OF}(C(i,n))\}$ is true;
- Hold j^{th} sub-bus, if and only if $\{\mathrm{UF}(C(1,j)) + \mathrm{UF}(C(2,j)) + \cdots + \mathrm{UF}(C(j-1,j)) + \mathrm{UF}(C(j+1,j)) + \cdots + \mathrm{UF}(C(n,j))\}$ is true.

The number of required counters for a bus divided into g sub-buses is C_g^2. For instance, 6 (C_4^2) counters are required for a bus grouped into 4 sub-buses.

3.2 DAS Scheme

Our goal is developing a not only robust valid-indicating signal protection scheme, but also area-efficient shielding scheme. Here, "area" mainly imply the area occupied by data Valid-indicating wires and grouping wires.

The Active-Shield scheme [5] can be illustrated in Fig.6. The shielded line—Valid-indicating line—is in the middle. The shielding lines are driven by the same signal with different strength (because the shielding wires are narrower and have smaller load capacity than the shielded wire.) So the transition types of Valid-indicating signal belong to the Group-2 whose delay factor is just 1 (shown in Table 1) . The valid-indicating signal can be shielded very well.

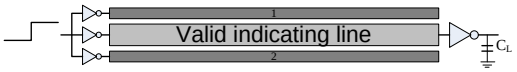


Fig. 6 Active-shield Scheme

Furthermore, can these active-shielding wires be reused as grouping wires to reduce the extra wires overhead? The answer is **positive**, as long as we make some alteration to the original Active-Shield scheme. Otherwise, the signal transitions on data lines will be influenced by valid-indicating signals transitions, which might lead the receiver to capture unstable data. Inspired by the Delay-line scheme [7], we address this problem by skewing the switch of valid-indicating transmission off the data transmission and capturing. The

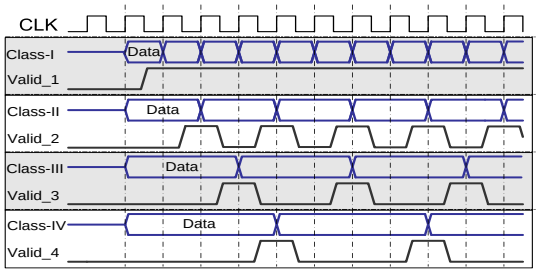


Fig. 7 DAS Timing sequence

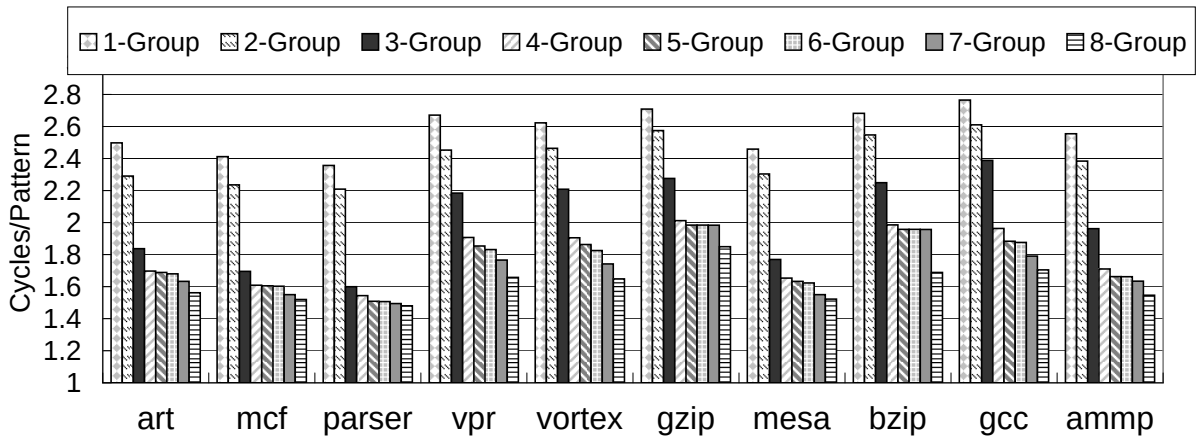


Fig. 8 BAT applied to instruction bus

implementation is using the positive clock edge to trigger the data transmitting at the sender and the data capturing at the receiver, and using the negative edge to trigger the transmitting of data valid-indicating signals. The reason why this implementation is feasible is that the crosstalk factor of valid-indicating line is just 1 (as shown in Table 1), which is less than a half of the crosstalk factor of Class-II transitions. Therefor, a half cycle is enough to setup the valid-indicating signal at the receiver. Through these operations, the valid-indicating signal transmission is delayed half cycle after the data transmission, which will greatly mitigate the crosstalk effect between bus wires and valid-indicating signal wires which are reused as grouping wires. We call the modified shielding scheme as Delay Active-Shield (DAS) scheme. The timing sequence is illustrated in Fig.7.

Finally, it can be seen that the BAT scheme is **scalable** with the bus width with little extra implementation complexity, which makes it easy to be adopted in the situations where the width of bus is large (such as 64 or wider) and transmission performance is critical.

4. Evaluation

The performance of BAT scheme is evaluated using Simplescalar3.0 tool set [4] on 10 SPEC2000 CPU benchmarks. Firstly, since the instruction fetch delay is the performance bottleneck of modern superscalar CPU [2], we study the data flow between Level-1 instruction cache (L1-icache) and instruction buffer unit. Similarly to [2], a Harvard architecture is adopted. Secondly, we study the data transmission on data bus connecting Level-1 Data cache (L1-dcache) and datapath. The width of instruction bus (I-bus) and data bus (D-bus) are 64-bit and 32-bit, respectively.

4.1 Performance Comparisons

We use *Mean Required Transmission Cycle* ($\overline{RTC}$) to evaluate transmission performance. We compare BAT scheme against several other representative crosstalk mitigation schemes: Codec (CDC) [3], Passive-Shied (PSD) [10], Active-Shied (ASD) [10], Variable Cycle Transmission (DYN) [6] and the original conservative (ORI) approach.

The CDC approach can transform the Class-IV and Class-III transitions into Class-I or Class-II transitions; therefor, two-cycle period is required to complete a patten transmission. The PSD approach interleaves the original bus wires with shielding wires, so any original transition will be transformed into Class-II. Therefore, both $\overline{RTC}$s of CDC and PSD approaches are 2 cycles. The ASD approach can transform all original transition into Class-I, which implies the best performance: $\overline{RTC}$ is 1 cycle, but imposes the worst area overhead in all of the mentioned approaches.

The DYN approach employs variable cycle transmission [6], so the RTC is variable from 1 cycle to 4 cycles.

In addition, the $\overline{RTC}$ of ORI approach is 4 cycles. (Notice that the transmission performance of ORI approach, in reality, maybe achieves 1 cycle per pattern, but here the time period of 1 cycle is equal to the time period of the mentioned 4 cycles.)

4.1.1 BAT Applied to Instruction Bus

Firstly, we simulate the transmission process on instruction-bus (I-bus). The CF distribution is shown in Fig.3(a). The buffer size is configured to 8 words (4×64-bit). Fig.8 shows the $\overline{RTC}$ variation with different number of sub-groups, The average $\overline{RTC}$ of the 10 benchmarks can be reduced to 2.57 using DYN approach (which is equivalent to 1-Group configuration in our approach). We can further reduce it to 1.79 with 4-Group configuration and even fall to 1.62 with 8-Group configuration.

Fig.8 also indicates that the performance marginal profit is positive with the variation of groups from 2

to 8. However, when the number of groups exceeds 4, the performance marginal utility is unattractive. So 4-Group configuration is an optimum trade-off between performance speed-up and area overhead.

Compared with ORI, DYN [6] and Codec [3] approaches, the average performance improvement using BAT scheme with 4-Group configuration is 55.3%, 30.4% and 10.5% respectively.

4.1.2 BAT Applied to Data Bus

We applied BAT to data-bus (D-bus) with identical group configuration. The buffer size is configured to 4 words (4×32-bit). The result is shown in Fig.9. From this Figure, we find the intrinsic crosstalk effect within D-bus is more mitigate than that within I-bus, so the D-bus $\overline{RTC}$ is less than instruction bus (I-bus) on average. Although the improvement of BAT applied to D-bus is not as significant as BAT applied to I-bus, compared with DYN scheme, we still gain 12.5% performance improvement on average with 4-Group configuration.

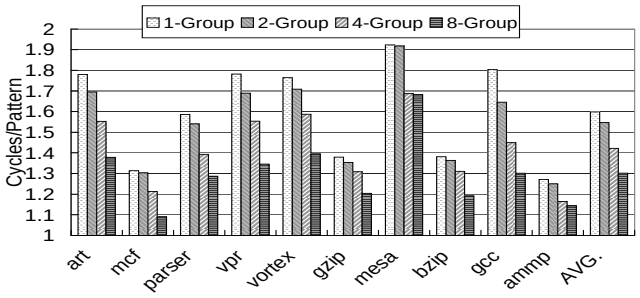


Fig. 9 BAT applied to data bus

Furthermore, from I-bus and D-bus experimental results, we can infer that adopting BAT scheme is more suitable in large bus width and crosstalk intensive situations.

4.2 Overhead Analysis

The overhead consists of two parts: 1) **time overhead**, caused by extra circuit logic on the transmission path, and 2) **area overhead**, not only caused by the extra circuit logic, but also some extra shielding wires. The **time overhead** is insignificant because only the DCC module which just increases a delay of a "transmission" gate is on the transmission critical path. The **area overhead** mostly results from a) extra shielding wires, b) extra buffers and c) the transmission logic. From experimental results, we find that b) and c) is not so substantial to modern VLSI. The original on-chip buffers, moreover, can be reused as the required buffers with minor modifications in many cases. Of the three parts, a) is the most important one which must be dealt with, especially when the bus is routed using scare top-level metal resources.

The bus routing area consists not only the metal wire area, but also the space between the neighbor wires. Empirically, as Li et al. assumed in [6], the wire is set equal to the space in width when we compute the area overhead. In addition, since we just concern about the relative overhead, the length of the bus is insubstantial.

For the ASD approach, the active shielding wire is a little "fatter" than the original data wire for the sake of manufacture considerations. Generally, the active shielding wire is twice to triple as wide as the ordinary wire [5], and the area overhead is about 13% in the most conservative situation (we adopt the most conservative value–3). Although we use the most conservative value to evaluate the normalized area, the routing area overhead is still far more efficient than CPC, PSD and ASD approaches do. Furthermore, except the ASD approach which occupies the most routing area, our BAT provides the best performance compared to the other approaches. The variation of area overhead among different approaches is shown in Table 2. The performance parameter $\overline{RTC}$ is also listed in this table for comparison.

Table 2 Wire routing overhead and $\overline{\mathbf{RTC}}$

Approach	Normalized Area	$\overline{RTC}$ (Avg. cycles/pattern)
ORI	100	4
DYN [6]	103	2.57
CPC [3]	145	2
PSD	199	2
ASD [5]	201	1
BAT	**113**	**1.79**

Finally, we study the impact of buffer size on transmission performance. Our experimental results show that the performance improvement is disproportional to the buffer size, as indicated by Fig.10. Too large buffers do not lead to significant improvement in performance (but impose chip area), which implies that a set of small buffers is enough to synchronize the data receiving without sacrificing the BAT performance. Fig.10 shows that 4–8 words buffer is an optimum choice.

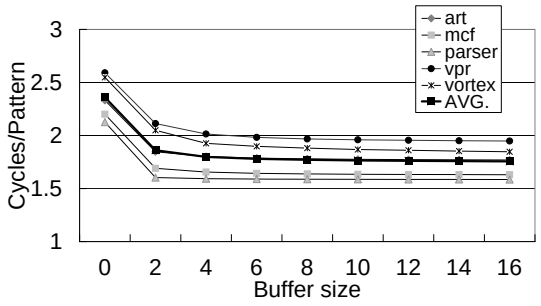


Fig. 10 The $\overline{\mathbf{RTC}}$ VS. Buffer size

We implement BAT structure for a 64-bit bus in Verilog-HDL and synthesize it using the Synopsys Design Compiler with a target UMC 0.18um technology. The bus is configured to 4 groups, and receiver buffers are configured to 4 words (4×32-bit). The **total overhead** (the sum of combinational and non-combinational circuit area of **CDA, VSG, DCC and Assembler**) is about 81540 μm^2, and this overhead is acceptable.

5. Conclusions

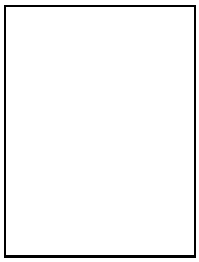
This paper presents a new on-chip bus transmission scheme: Bus-grouping Asynchronous Transmission (BAT). BAT can significantly mitigate the delay effects of crosstalk sensitive transition and thereby accelerates data transmission. Furthermore, from our experimental observation, we find that Crosstalk Factor distribution on some types of buses are of distinct spatial locality, such as on instruction buses. This characteristic can be used to optimize the BAT performance. In terms of BAT implementation, two efficient techniques are presented: Difference Counter Cluster (DCC) synchronous mechanism and Delay Active Shielding (DAS). We evaluate the effectiveness of the BAT scheme focusing on the on-chip buses of a modern microprocessor and using the SPEC CPU2000 benchmarks. When applied to on-chip instruction bus, the proposed scheme improves performance by 55.3% compared to the original pessimistic approach that always assumes the worst case at the expense of 13% routing area overhead.

Acknowledgments

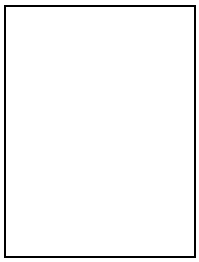
The authors gratefully thank Dr. Tong Liu for his reviewing this paper and valuable comments and also thanks Dr. Lei Zhang, Dr, Shaohua Lei, Dr, Binzhang Fu for lots of helpful discussions.

References

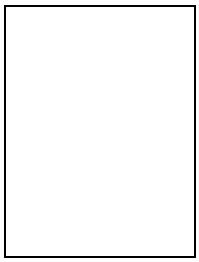
[1] International Technology Roadmap For Semiconductors 2005 Edition, **Interconnect**, Avaiable:`http://www.itrs.net`.
[2] A. Falcon, A.Ramirez, and V. Valero, "A low-complexity, high-performance fetch unit for simultaneous multithreading processors", In Proceedings of 10th International Symposium on High Performance Computer Architecture, pp.244–253, Feb. 14–18, 2004.
[3] B. Victor, K. Keutzer, "Bus encoding to prevent crosstalk delay", IEEE/ACM International Conference on Computer Aided Design, pp.57–63, Nov. 4–8, 2001.
[4] D. Burger and T. M. Austin, "The simplescalar tool set, version2.0", Computer Arch. News, 1997.
[5] H.Kaul, D.Sylvester, and D.Blaauw, "Active Shields: A New Approach to Shielding Global Wires", In Proceeding of Great Lakes Symposium on VLSI, pp.112–117, April, 2002.
[6] L. Li, N. Vijaykrishnan, M. Kandemir, and M.J. Irwin, "A crosstalk aware interconnect with variable cycle transmission", In Proceedings of Design Automation and Test in Europe Conference and Exhibition, Volume 1, pp.102–107, Feb. 16–20, 2004.
[7] M. Ghoneima, Y. Ismail, "Delayed line bus scheme: a low-power bus scheme for coupled on-chip buses", In Proceedings of the International Symposium on Low Power Electronics and Design, pp.66–69, Aug. 9–11, 2004.
[8] P.P. Sotiriadis, "Interconnect Modeling and Optimization in Deep Sub-Micron Technologies", PhD Dissertation, at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY, pp.181–203, 2002.
[9] P.P. Sotiriadis, and A. Chandrakasan, "Reducing bus delay in submicron technology using coding", In Proceedings of Asia and South Pacific-Design Automation Conference,pp.109–114, 2001.
[10] R.Arunachalam, E.Acar, and S.R.Nassif, "Optimal shielding/spacing metrics for low power design", In Proceedings of IEEE Computer Society Annual Symposium on VLSI, pp.167–172, 2003.
[11] S. Youngsoo, and T. Sakurai, "Coupling-driven bus design for low-power application-specific systems", In Proceeding of Design Automation Conference, pp.750–753, 2001.
[12] SPEC CPU2000 Benchmark. Avaiable:`http://www.spec.org/`.
[13] J.M. Philippe, S. Pillement, and O. Sentieys, "Area efficient temporal coding schemes for reducing crosstalk effects", In Proceedings of 7th International Symposium on Quality Electronic Design, pp.334–339, Mar. 27–29, 2006.
[14] W. Kuo; Y. Chiang, T. Hwang, and A. Wu, "Performance-Driven Crosstalk Elimination at Postcompiler LevelThe Case of Low-Crosstalk Op-Code Assignment", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume 26, Issue 3, pp.564-573, Mar. 2007.
[15] J.M. Rabaey, A.Chandrakasan, and B.Nikolic, "Digital Integrated Circuits, A design perspective", Second Edition, PEARSON EDUCATION ASIA LIMITED and TSINGHUA UNIVERSITY PRESS, 2004



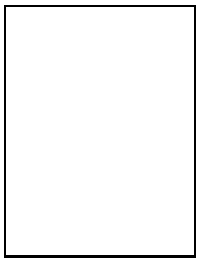
Guihai Yan received his B. Eng. from Peking University in 2005. Mr Yan now is PhD candidate in computer science at Institute of Computing Technology, Chinese Academy of Sciences. His research interests include ASIC design, design for reliability and embedded system.



Yinhe Han received his B. Eng. from Nanjing University of Aeronautics and Astronautics (China) in 1997, and his Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences in 2006. His research interests include VLSI/Soc design, design for testability. He is a member of IEEE. He received the IEEE Test Technology Technical Council Best Paper Award of Asian Test Symposium 2003.



Xiaowei Li received his B.Eng. and M.Eng. degrees in computer science from Hefei University of Technology (China) in 1985 and 1988 respectively, and his Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences in 1991. Dr. Li joined Peking University as a Postdoctoral Research Associate in 1991, and was promoted to Associate Professor in 1993, all with the Department of Computer Science and Technology. From 1997 to 1998, he was a Visiting Research Fellow in the Department of Electrical and Electronic Engineering at the University of Hong Kong. In 1999 and 2000, he was a Visiting Professor in the Graduate School of Information Science, Nara Institute of Science and Technology, Japan. He Joined the Institute of Computing Technology, Chinese Academy of Sciences as a professor in 2000. His research interests include VLSI/Soc design verification and test generation, design for testability, low-power design, dependable computing. Dr. Li received the Natural Science Award from the Chinese Academy of Sciences in 1992, the Certificate of Appreciation from IEEE Computer Society in 2001. He is a senior member of IEEE. He is an area editor of the Journal of Computer Science and Technology and an associate editor-in-chief of the Journal of Computer-Aided Design & Computer Graphics (in Chinese).



Hui Liu received his B. Eng. from Shandong University in 2005. She is now PhD candidate in computer science at Institute of Computing Technology, Chinese Academy of Sciences. Her research interests include design for testability, overtesting and delay test.