Area and Energy-Efficient Crosstalk Avoidance Codes for On-Chip Buses

Srinivasa R. Sridhara, Arshad Ahmed, and Naresh R. Shanbhag
Coordinated Science Laboratory/ECE Department
University of Illinois at Urbana Champaign
1308 W Main St. Urbana IL 61801
{sridhara,ahmed4,shanbhag}@uiuc.edu

Abstract

Capacitive crosstalk between adjacent wires in long on-chip buses significantly increases propagation delay in the deep submicron regime. A high-speed bus can be designed by eliminating crosstalk delay through bus encoding. In this paper, we present an overview of the existing coding schemes and show that they require either a large wiring overhead or complex encoder-decoder circuits. We propose a family of codes referred to as ***overlapping codes*** *that reduce both overheads. We construct two codes from this family and demonstrate their superiority over existing schemes in terms of area and energy dissipation. Specifically, for a 1-cm 32-bit bus in 0.13-µm CMOS technology, we present a 48-wire solution that has 1.98× speed-up, 10% energy savings and requires 20% less area than shielding.*

1 Introduction

On-chip global buses are increasing in length with increasing die sizes, resulting in large propagation delays [1]-[3]. The delays of these buses can limit the system performance in many high-speed microprocessors [2, 3]. This trend is anticipated to worsen in the future due to the increasing gap between gate delay and interconnect delay brought about by shrinking feature sizes. In deep submicron (DSM) era, the coupling capacitance is significant compared to the bulk capacitance. Hence, the capacitive crosstalk due to the transitions on adjacent wires leads to a significant increase in the worst-case delay [4]-[7]. This increase in the delay is referred to as the crosstalk delay. Coding techniques [5]-[7] have been proposed to avoid crosstalk delay.

Coding is the process of mapping information bits or data words into codewords such that the codewords exhibit certain desired properties. In order to prevent crosstalk delay, any two codewords following one another on the bus should not have transitions that incur the crosstalk delay penalty. This can be achieved by either avoiding specific data patterns [6] or avoiding opposing transitions on adjacent wires [7]. However, for large buses, it is impractical to encode all bits at once due to the prohibitive complexity of the encoder-decoder (codec) circuits. Therefore, *partial coding* [6, 7] is employed in which the bus is broken into sub-buses of smaller width which are encoded into sub-channels. These sub-channels are then combined in such a way so as to avoid crosstalk delay at their boundaries. This recombination requires additional wires [6, 7] and additional codec delay [6].

In this paper, we present overlapping codes. In this partial coding technique, adjacent sub-channels are overlapped in order to obtain compact buses for a given data rate. While such a scheme reduces the wiring overhead and codec complexity of combining sub-channels, it places additional restrictions on the component partial codes. Depending on the partial codes used, we show that these restrictions can be satisfied either by reducing the code rate or by using memory to track the state of the partial code. We construct two codes using this technique and show that at a given throughput, the wiring and the computational overheads are reduced compared to existing schemes.

2 Bus Models

In this section, we review the analytical models for delay and energy dissipation in DSM buses. In this paper, we assume an n-bit parallel bus in a single metal layer. Further, we assume that rise time of the drivers and the loss in the interconnects are such that the inductance can be safely ignored [3]. Such DSM buses can be modeled as distributed RC networks with coupling capacitance between adjacent wires.

2.1 Delay Model

The delay of line l ($1 < l < n$) of the bus is given by [5]

$$T_l = \tau_0 \left[(1+2\lambda)\Delta_l^2 - \lambda\Delta_l(\Delta_{l-1}+\Delta_{l+1})\right], \qquad (1)$$

where τ_0 is the delay of a crosstalk-free line, λ is the ratio of the coupling capacitance to the bulk capacitance, and Δ_l is the transition occurring on line l. Δ_l is equal to 1 for 0-to-1 transition, -1 for 1-to-0 transition, and 0 for no transition.

2.2 Energy Model

The average dissipated energy per bus transfer depends on the statistical distribution of the data and is given by [8]

$$E = \mathrm{tr}\,(C_T \mathcal{A})\, V_{dd}^2, \qquad (2)$$

Δ_{l-1}	Δ_l	Δ_{l+1}	Relative Delay
–	–	–	0
↑	↑	↑	1
–	↑	↑	$1+\lambda$
–	↑	–	$1+2\lambda$
↑	↑	↓	$1+2\lambda$
↓	↑	–	$1+3\lambda$
↓	↑	↓	$1+4\lambda$

Table 1. Relative delay of line l.

where $\mathcal{A}$ is the transition activity matrix [8], $\mathrm{tr}(X)$ is the trace of the matrix X, V_{dd} is the supply voltage, and C_T is a $n \times n$ capacitance matrix given by

$$C_T = \begin{bmatrix} 1+\lambda & -\lambda & 0 & \cdots & 0 \\ -\lambda & 1+2\lambda & -\lambda & \cdots & 0 \\ 0 & -\lambda & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & 1+2\lambda & -\lambda \\ 0 & 0 & \cdots & -\lambda & 1+\lambda \end{bmatrix} \cdot C, \quad (3)$$

where the C is the total bulk capacitance of a line.

3 Crosstalk Avoidance Coding

In this section, we review existing coding schemes which avoid crosstalk delay. The delay model described in Section 2 indicates that the delay of a line depends on the transitions on the line and its adjacent lines. Table 1 lists the delay of line l of an n-bit bus, where $1 < l < n$, for certain combinations of transitions. In the table, ↑ indicates a 0-to-1 transition, ↓ indicates a 1-to-0 transition, and "–" indicates no transition on the line. From Table 1, we see that with large values of λ, the worst case delay can be significantly higher than the delay in the absence of coupling. Coding techniques [5]-[7] have been proposed in which the relative delay is limited to $1+2\lambda$.

A very simple method to reduce the relative delay from $1+4\lambda$ to $1+2\lambda$ is to insert a grounded wire between every data wire on the bus. This scheme, referred to as *shielding*, requires $n = 2k-1$ wires to transmit k-bits of data on the bus. Coding schemes also reduce the delay, but require fewer wires than shielding. In the succeeding subsections, we enumerate the properties desired in a code to avoid crosstalk delay and classify existing coding schemes based on these properties.

3.1 Crosstalk avoidance code properties

In a (n,k) memoryless code used in crosstalk delay avoidance, a k-bit data word is encoded as an n-bit codeword such that the 2^k n-bit codewords can be sent on the bus in any arbitrary sequence with relative delay less than or equal to $1+2\lambda$. This is possible if one of the following two conditions is satisfied.

- **Forbidden transition** [7]: A transition from one codeword to another codeword does not cause adjacent wires to transition in opposite directions.

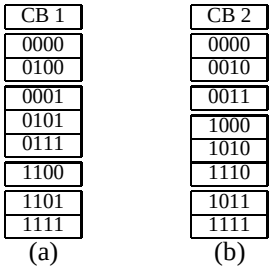


Figure 1. Forbidden transition codewords for $n=4$: (a) codebook with "0101" codeword and (b) codebook with "1010" codeword.

- **Forbidden pattern** [6]: None of the codewords has the bit pattern "101" or "010" appearing in it.

Note that the forbidden pattern condition allows opposing transitions to occur on adjacent wires as long as they do not cause delays greater than $1+2\lambda$; the forbidden transition condition allows forbidden patterns in its codewords as long as they do not cause opposing transitions.

3.2 Forbidden transition codes

The largest set of codewords satisfying the forbidden transition condition is the set of codewords that can transition to a *class 1 codeword* (a codeword with alternating 0 and 1 bits) without generating forbidden transitions [7]. For example, the two largest sets (codebooks) of 4-bit codewords that satisfy the forbidden transition condition are shown in Figure 1. The (4,3) code in CB1 can be used to implement a 3-bit bus at a code rate (defined as k/n) of 75%. The encoder and the decoder are simple combinational circuits requiring nine 2-input gates [7]. For forbidden transition codes, the code rate does not always increase with n. The (4,3) codes have the highest rate for $n < 10$.

It has been shown in [7] that the number of valid n-bit codewords $M_{FT}(n)$ satisfying the forbidden transition condition is

$$M_{FT}(n) = F_{n+2}, \quad (4)$$

where F_n is the Fibonacci sequence satisfying $F_n = F_{n-1} + F_{n-2}$ with initial conditions $F_1 = F_2 = 1$. The maximum number of data bits that can be encoded using n wires is

$$k = \lfloor \log_2(M_{FT}(n)) \rfloor. \quad (5)$$

If $M_{FT}(n)$ is greater than 2^k, then we can choose a subset of 2^k codewords in $\binom{M_{FT}(n)}{2^k}$ ways. Further, the 2^k data patterns can be mapped to the 2^k codewords in $2^k!$ ways. Thus, the total number of options for the codebook and the mapping is $\binom{M_{FT}(n)}{2^k} 2^k!$. For example, for $(n,k) = (6,4)$, an exhaustive search for a codebook and a mapping resulting in the optimal codec implementation requires evaluating 4.3×10^{17} options. The computational complexity of such a search is prohibitive. Hence, in order to estimate the codec complexity for forbidden transition codes, we use an arbitrary code book and mapping and obtain an implementation

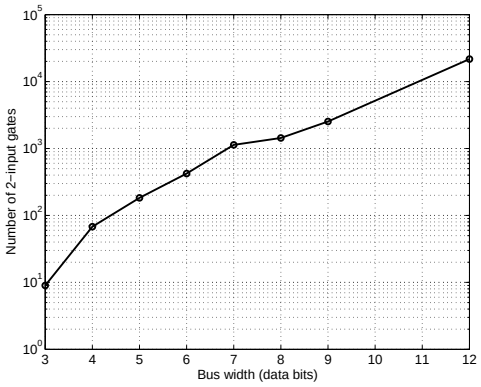


Figure 2. Hardware requirements for forbidden transition codes.

using the boolean optimization software SIS [9]. Figure 2 shows the number of 2-input gates required in the codec for data-width k from 3 to 12. From the figure, we see that the hardware complexity grows exponentially with k.

Thus, for large buses, it is impractical to encode all k bits at once due to the large complexity in the design and the implementation of the codec. Therefore, partial coding [7] is employed wherein the bus is broken into sub-buses of smaller width which are encoded into sub-channels. In [7], two adjacent sub-channels are combined using a dedicated shield wire. We denote such codes as FTC($\hat{n},\hat{k}$), where $\hat{n}$ and $\hat{k}$ are the number of code bits and data bits in the largest sub-channel, respectively. For example, a 32-bit bus implemented using FTC(4,3) requires 53 wires with 10 sub-channels and two shielded data wires [7]. This partial coding solution requires 90 gates in the encoder and the decoder. A 48-wire solution is also possible for the 32-bit bus using FTC(17,12), where two sub-channels each with FTC(17,12), one sub-channel with FTC(10,7), and a single wire are employed. However, this scheme requires 44,654 two-input gates in the encoder and the decoder.

Note that the lower bound on n for $k = 32$ using memoryless codes is 46. If we use codes with memory, the lower bound reduces to 40 [7]. However, the complexity is excessive. Also, partial coding using codes with memory is more complex than memoryless partial coding. For example, a 48-wire solution for a 32-bit bus using (8,6) codes with memory requires more than 200,000 2-input gates as compared to 44,654 gates required by FTC(17,12).

3.3 Forbidden pattern codes

Forbidding bit patterns "101" and "010" in codewords results in more codewords for the same n than forbidding opposing transitions. The number of codewords is given by [6]

$$M_{FP}(n) = 2F_{n+1}. \quad (6)$$

However, this increase in the number of codewords translates into at most one additional data bit that can be encoded for the same n and complexity still grows exponentially with k. Further, combining sub-channels using grounded wires is not possible as the forbidden patterns can appear at the boundary. Instead in [6], two sub-channels or groups are placed adjacent to each other and if the last bit of the first group differs from the first bit of the second group, then the bits of the second group are inverted. A group complement bit is transmitted to enable the decoder to correctly decode the group. We denote this family of forbidden pattern codes as FPC($\hat{n},\hat{k}$), where $\hat{n}$ and $\hat{k}$ are the number of code-bits and data-bits in a sub-channel, respectively. For large bus width, this scheme suffers from the rippling of data in the group complement logic. For example, a 52-wire solution for a 32-bit bus requires 310 two-input gates and has a logic depth of 36 two-input gates [6].

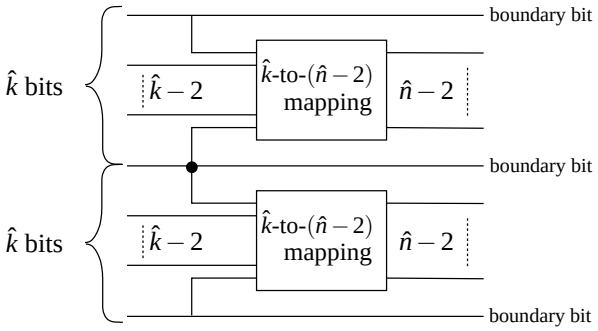
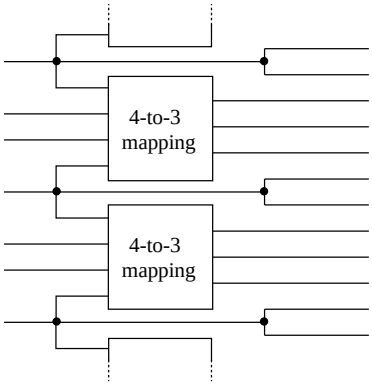


Figure 3. Overlapping codes.

Data	Code
0000	00000
0001	00001
0010	00110
0011	00011
0100	01100
0101	00111
0110	01110
0111	01111
1000	10000
1001	10001
1010	11000
1011	10011
1100	11100
1101	11001
1110	11110
1111	11111

(a)



(b)

Figure 4. Forbidden pattern overlapping codes (FPOC): (a) forbidden pattern codes for $n = 5$, (b) overlapping code for $(\hat{n},\hat{k}) = (5,4)$.

4 Overlapping Codes

In the previous section, we concluded that encoding a wide bus is non-trivial and that existing partial coding techniques involve either large wiring or logic gate overhead. Therefore, we need a technique to place sub-channels next to each other without requiring shielded wires as in [7] or complement logic and additional wires as in [6]. We propose to employ overlapping codes for this purpose. In overlapping codes, two adjacent sub-channels are overlapped at their boundary as shown in Figure 3. If $\hat{n}$ and $\hat{k}$ are the number of code-bits and data-bits in the sub-channel, then $\hat{k}$ data

Data	$C_{past}=0$						$C_{past}=1$					
	$S_{past}=0$			$S_{past}=1$			$S_{past}=0$			$S_{past}=1$		
	Code	S	C	Code	S	C	Code	S	C	Code	S	C
000	0000	d	0	0000	d	0	0000	d	1	0000	d	1
010	0100	1	0	0100	1	0	0010	1	1	0010	1	1
001	0001	0	0	0001	0	0	0011	d	1	0011	d	1
011	0111	1	0	0111	1	0	0001	0	0	0111	1	0
100	1100	d	0	1100	d	0	1000	0	1	1000	0	1
110	1000	0	1	1110	1	1	1110	1	1	1110	1	1
101	1101	0	0	1101	0	0	1011	0	1	1011	0	1
111	1111	d	0	1111	d	0	1111	d	1	1111	d	1

(a)

$C_{past}=0$			$C_{past}=1$		
Code	Data	C	Code	Data	C
0000	000	0	0000	000	1
0001	001	0	0011	001	1
0100	010	0	0010	010	1
0111	011	0	0001	011	0
1100	100	0	0111	011	0
1101	101	0	1000	100	1
1000	110	1	1011	101	1
1110	110	1	1110	110	1
1111	111	0	1111	111	1

(b)

Figure 5. Mappings for FTOC(4,3): (a) encoder and (b) decoder.

bits are mapped to the central $\hat{n}-2$ bits of the codeword, and the boundary bits of the data word form the boundary bits of the codeword. This coding technique will avoid crosstalk delay if the following two conditions are satisfied simultaneously,

1. Overlapping does not cause crosstalk delay in the boundary bits.
2. In the sub-channels, a mapping with unchanged boundary bits exists from data words to codewords.

It is easy to show that the forbidden pattern codes do not satisfy the first condition and the forbidden transition codes do not satisfy the second condition. In this section, we construct codes that satisfy both the conditions.

4.1 Forbidden pattern overlapping codes

We denote this family of codes as FPOC($\hat{n}$,$\hat{k}$). An example of forbidden pattern code with $(\hat{n},\hat{k})=(5,4)$ that satisfies condition 2 is shown in Figure 4(a). However, the forbidden patterns "101" and "010" will appear if two codes are overlapped. To solve this problem, we duplicate the boundary bits as shown in Figure 4(b). Now, forbidden patterns cannot occur at the boundary. Further, forbidden patterns cannot occur within the sub-channels as the sub-channels transmit valid codewords. Thus, the relative delay is reduced to $1+2\lambda$. However, the code rate has dropped to $(\hat{k}-1)/\hat{n}=3/5$. Forbidden pattern overlapping codes eliminate the large ripple delay that occurs in the group complement logic of the existing forbidden pattern codes. However, the wiring overhead remains the same as in [6].

4.2 Forbidden transition overlapping codes

We denote this family of codes as FTOC($\hat{n}$,$\hat{k}$). For the sub-channel code that satisfies the forbidden transition condition, we use the $(4,3)$ code (Figure 1) as it has a high code rate and can be manipulated to satisfy both the required conditions listed above. For any $(\hat{n},\hat{k})$ sub-code, we need $2^{\hat{k}-2}$ codewords for each of the 4 possible combinations of the boundary bits. For the $(4,3)$ code, we need 2 codewords for each of the 4 boundary conditions. The codewords with the same boundary bits are grouped together in Figure 1. As can be seen, neither of the two codebooks satisfies condition 2 listed above. Codebook CB1 has only one codeword with boundary bits "1", "0" and codebook CB2 has only one codeword with boundary bits "0", "1". However, codebooks CB1 and CB2 have 3 codewords with boundary bits "0", "1" and "1", "0" respectively. We satisfy condition 2 by transitioning between the two codebooks.

The mappings for the code, referred to as FTOC(4,3), is shown in Figure 5. When CB1 ($C_{past}=0$) is being used and the current input data word is "110", we transition to CB2 ($C=1$) as shown in Figure 5(a) and encode "110" as either "1000" or "1110" depending on the previous codeword on the sub-channel. This is because the codeword "1000" cannot follow the codewords in the set {"0100", "0101", "0111"} ($S=1$) of CB1 without causing opposing transitions and "1110" cannot follow {"0001", "0101", "1101"} ($S=0$). Since neither "1000" nor "1110" can follow "0101", we remove it from CB1. Similarly, the codeword "1010" is removed from CB2 and a transition from CB2 to CB1 occurs for data word "011". The mappings when CB2 is the previous codebook ($C_{past}=1$) are also shown in Figure 5(a). The encoder needs memory to keep track of the current codebook index C and the set index S of the previous codeword. The decoder also tracks the current codebook C. The mappings for the decoder are shown in Figure 5(b).

This code achieves a code rate of 66.6% and requires 48 wires to implement a 32-bit bus, which compares favorably to the lower bound of 46 wires for memoryless codes. Note that it is not guaranteed for all $\hat{n}$ that condition 2 can be satisfied by combining the two codebooks.

5 Simulation Results and Discussions

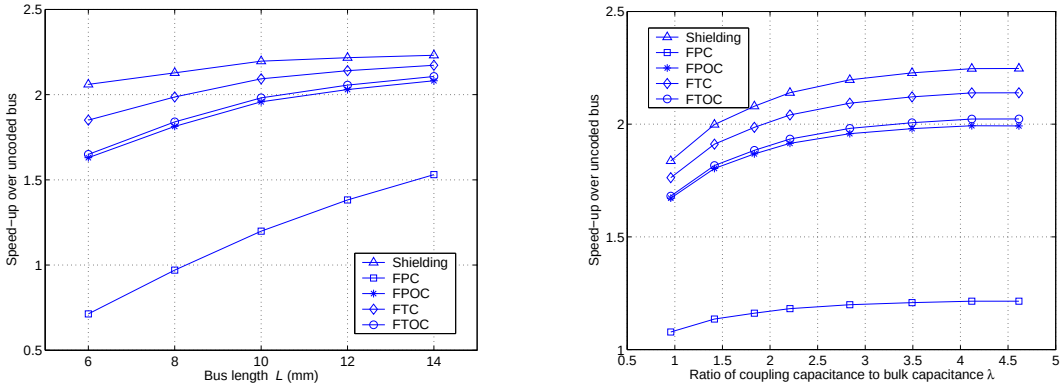
In this section, we compare the proposed codes with the existing schemes by designing a global 32-bit bus in a 0.13-μm CMOS technology. Table 2 lists the number of wires, the delay, and the average energy dissipation of the 32-bit bus employing the various codes. Note that the average bus energy per bus transition is a function of the statistical distribution of the data, which depends on the application. Here, we assume that the data is spatially and temporally uncorrelated, with "0" and "1" being equiprobable.

Compared to the uncoded bus, all the codes improve the bus delay from $(1+4\lambda)\tau_0$ to $(1+2\lambda)\tau_0$ but differ in terms of the number of wires required. The proposed FTOC(4,3) code requires the least number of wires. It requires 48 wires, which corresponds to 24% reduction compared to shielding.

The codes also reduce the coupling component of energy dissipation of the bus by avoiding some of the high-energy coupling transitions while satisfying forbidden transition and forbidden pattern conditions. However, this comes at the cost of increased self transitions as seen in Table 2. For

Table 2. Code comparison for a 32-bit bus.

Coding Scheme	Bus			Codec overheads		
	No. of wires	Delay ($\times\tau_0$)	Average energy ($\times CV_{dd}^2$)	Area (μm^2)	Delay (ps)	Average energy (pJ)
Uncoded	32	$1+4\lambda$	$8.0+15.5\lambda$	0	0	0
Shielding	63	$1+2\lambda$	$8.0+15.5\lambda$	0	0	0
FPC(5,4)	52	$1+2\lambda$	$13.0+13.0\lambda$	9503	1784	9.10
FPOC(5,4)	52	$1+2\lambda$	$13.0+13.0\lambda$	6290	250	5.26
FTC(4,3)	53	$1+2\lambda$	$8.9+12.4\lambda$	1830	107	1.61
FTOC(4,3)	48	$1+2\lambda$	$10.4+11.1\lambda$	9760	235	5.78



(a) (b)

Figure 6. Speed-up over uncoded 32-bit bus: (a) as a function of bus length L at $\lambda = 2.8$ and (b) as a function of λ at $L = 10$ mm.

example, FTOC(4,3) reduces the coupling transition component from 15.5 to 11.1, but increases the self-transition component from 8 to 10.4. Therefore, the value of λ plays an important role in determining reduction in both delay and energy.

Except shielding, all other codes have additional codec overheads that reduce the effectiveness of the codes. Table 2 lists area, delay, and average energy dissipation of the codecs. The overheads are estimated from synthesized gate-level netlists obtained using a 0.13-μm CMOS standard cell library.

In order to determine the impact of codec overheads, we consider a metal 4 bus with minimum width of 0.2 μm and minimum spacing of 0.2 μm. The value of λ depends on the metal coverage in upper and lower metal layers [1, 2]. We vary λ between the following two extreme scenarios. First, 100% metal coverage is assumed in metal layers 3 and 5, resulting in $\lambda = 0.95$. Second, all the bulk capacitance is assumed to be from metal 4 to the substrate, resulting in $\lambda = 4.6$. The bus length L is varied between 6 and 14 mm. We assume $50\times$ minimum-sized drivers and obtain bus delay and energy using HSPICE [10].

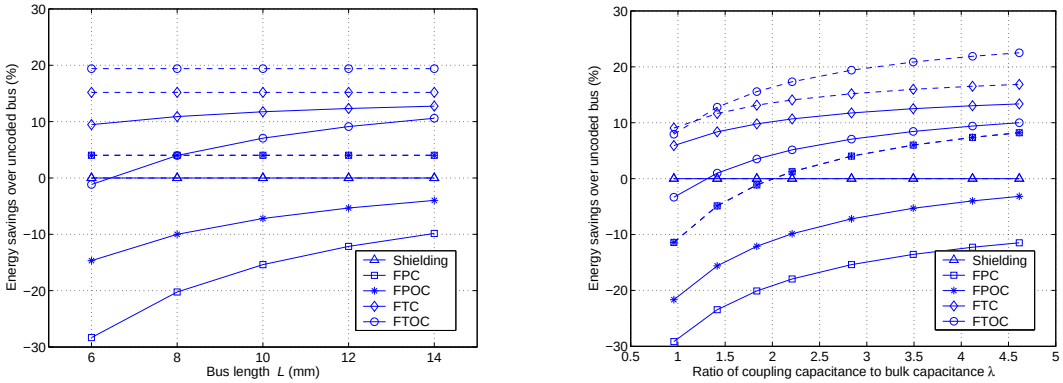
Figure 6 plots the speed-up achieved by codes. Speed-up is defined as the ratio of the uncoded bus delay to the total (bus+codec) delay of the coded bus. Thus, the additional latency due to coding is accounted for in the comparison. The speed-up achieved increases with L as shown in Figure 6(a). The codec delay has significant impact on the speed-up at small L. For large L, FTOC(4,3) achieves speed-up greater than $2.0\times$ over the uncoded bus. Comparing the two codes satisfying forbidden pattern condition, we see that the proposed FPOC(5,4) has $2\times$ speed-up compared to $1.2\times$ speed-up achieved by FPC(5,4).

Figure 6(b) plots the speed-up achieved for a 10-mm 32-bit bus as function of λ. Larger λ values result in higher speed-ups for the codes. Therefore, crosstalk avoidance codes will become more effective in future as it is estimated that λ will be as high as 10 in future technologies [6].

Note that shielding achieves the maximum possible speed-up at each L and λ as it has no coding latency. With technology scaling, speed-up due to the codes will approach the speed-up due to shielding because of the increasing gap between speeds of logic and interconnect. However, even in the current technology, the proposed codes have an advantage over shielding in terms of area and energy efficiency as shown next.

Figure 7(a) plots the energy savings compared to the uncoded bus as a function of L at $\lambda = 2.8$. The dashed curves indicate the bus energy savings achieved when the codec energy overhead is ignored. Note that shielding does not provide any energy savings. Energy savings for the codes increase with L. Even though bus energy savings for FTOC(4,3) is higher than FTC(4,3), the codec energy overhead is significant in the current technology and FTOC(4,3) approaches FTC(4,3) only for long buses. As discussed above, the codec overhead will decrease in future and total energy savings will approach the bus energy savings shown in the dashed curves.

Figure 7(b) plots the energy savings as a function of λ for a 10-mm bus. As expected, the energy savings of all codes except shielding increase with increase in λ. The achievable energy savings for FTOC(4,3) is 22% for $\lambda = 4.6$. It reduces



(a) (b)

Figure 7. Energy savings over uncoded 32-bit bus : (a) as a function of bus length L at $\lambda = 2.8$ and (b) as a function of λ at $L = 10$ mm. The dashed curves indicate the corresponding energy savings without coding overhead.

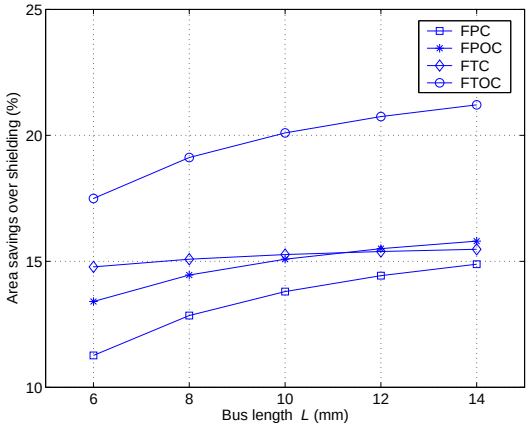


Figure 8. Area savings over shielding for a 32-bit bus.

to 10% when coding overhead is included. FPC(4,3) and FPOC(4,3) do not provide energy savings as the reduction in coupling transition component is canceled by the increase in self-transition component. Therefore, codes based on forbidden transition condition, FTC(4,3) and FTOC(4,3), are more suitable for crosstalk avoidance than codes based on forbidden pattern condition.

Finally, Figure 8 plots the area savings over shielding. As discussed in Section 3, reduction in area is the main motivation for employing codes instead of shielding. Note that the area savings are computed after including the codec area overhead. The proposed FTOC(4,3) provides 20% area savings over shielding for $L = 10$ mm and the savings increase with L. Similar to energy and delay, the codecs will occupy less area with technology scaling and area savings will improve in future.

6 Acknowledgments

This work was supported by the MARCO-sponsored Gigascale Systems Research Center and Intel Corporation.

References

[1] F. Caignet, S. Delmas-Bendhia, and E. Sicard, "The challenge of signal integrity in deep-submicrometer CMOS technology," *Proc. IEEE*, vol. 89, pp. 490–504, Apr. 2001.

[2] D. Sylvester and C. Hu, "Analytical modeling and characterization of deep-submicrometer interconnect," *Proc. IEEE*, vol. 89, pp. 634–664, May. 2001.

[3] D. Pamunuwa, L.-R. Zheng, and H. Tenhunen, "Maximizing throughput over parallel wire structures in the deep submicrometer regime," *IEEE Trans. VLSI Syst.*, vol. 11, pp. 224–243, Apr. 2003.

[4] H. Kawaguchi and T. Sakurai, "Delay and noise formulas for capacitively coupled distributed RC lines," in *Proc. ASP-DAC*, 1998, pp. 35–43.

[5] P. P. Sotiriadis and A. Chandrakasan, "Reducing bus delay in submicron technology using coding,", in *Proc. ASP-DAC*, 2001, pp. 109–114.

[6] C. Duan, A. Tirumala, and S. P. Khatri, "Analysis and avoidance of cross-talk in on-chip buses," in *Proc. Hot Interconnects*, 2001, pp. 133–138.

[7] B. Victor and K. Keutzer, "Bus encoding to prevent crosstalk delay," in *Proc. ICCAD*, 2001, pp. 57–63.

[8] P. P. Sotiriadis and A. Chandrakasan, "A bus energy model for deep submicron technology," *IEEE Trans. VLSI Syst.*, vol. 10, pp. 341–350, June 2002.

[9] E. M. Sentovich, *et. al.*, *SIS: A System for Sequential Circuit Synthesis*, Memorandum No. UCB/ERL M92/41, Electronics Research Laboratory, University of California, Berkeley, May 1992.

[10] *HSPICE Simulation and Analysis Manual*, Release U-2003.03-PA, Synopsys, March 2003.