

# Delay and Peak Power Minimization for On-Chip Buses using Temporal Redundancy

K. Najeeb, Vishal Gupta, V. Kamakoti  
 Indian Institute of Technology Madras  
 Chennai - 600 036, India  
 kama@cs.iitm.ernet.in

Madhu Mutyam\*  
 International Institute of Information Technology  
 Gachibowli, Hyderabad - 500 032, India  
 madhu\_mutyam@iiit.ac.in

## ABSTRACT

In this paper, we propose a novel *temporal redundancy* based encoding technique for delay and peak power minimization. The proposed encoding scheme is tested with the SPEC2000 CINT benchmarks for 90nm and 65nm technologies. The experimental results show that our approach is very effective in reducing the peak power. From the delay perspective, our approach reduces the delay by at least 11% (4%) in the address (data) buses compared to the data transmission without encoding.

**Categories and Subject Descriptors:** B.4.3 Input/output and data communications: Interconnects (Subsystems)

**General Terms:** Design, Performance, Energy

**Keywords:** Crosstalk, Low-power, Coding

## 1. INTRODUCTION

As device geometries shrink and clock speeds get faster, interconnect delay and power consumption are becoming increasingly significant [6, 16, 18]. It has been shown that the delay and power consumption of a long on-chip bus are strongly dependent on the coupling capacitance between the wires [12, 13] and they become prominent when adjacent wires simultaneously make transitions in opposite directions. Several bus encoding schemes for reducing delay and power consumption have been proposed in the literature [7, 8, 9, 12, 13, 15, 17]. All these techniques employ *spatial redundancy*, i.e., they use extra bus wires for encoding.

Analytical model for estimating propagation delay in DSM buses has been proposed in [13]. Let  $d_t = (d_t^1, d_t^2, \dots, d_t^n)$  denote the  $t^{\text{th}}$   $n$ -bit data transmitted on the bus. The delay for transmitting the  $(t+1)^{\text{th}}$  data on the bus is given by the following formula [13]. Defining  $T_k(d_t, d_{t+1})$ ,  $1 \leq k \leq n$ , as follows:

\*Supported under BOYSCAST Fellowship, Department of Science and Technology (DST), India

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GLSVLSI'06, April 30–May 2, 2006, Philadelphia, PA, USA.  
 Copyright 2006 ACM 1-59593-347-6/06/0004 ...\$5.00.

Crosstalk Class	Relative Delay on the middle wire	Transition Patterns
1	0	---,--- ↑,↑ ---,--- ↓,↓ ---, ↑ - ↑,↑ - ↓,↓ - ↑,↓ - ↓
2	$C_L R_T$	↑↑↑,↓↓↓
3	$C_L R_T (1 + \lambda)$	- ↑↑,↑↑ -, - ↓↓,↓↓ -
4	$C_L R_T (1 + 2\lambda)$	- ↑ -, - ↓ -, ↓↓↑,↑↓↓, ↑↑↓,↓↑↑
5	$C_L R_T (1 + 3\lambda)$	- ↑↓, - ↓↑,↑↑ -, ↑↓ -
6	$C_L R_T (1 + 4\lambda)$	↓↑↓,↑↑↑

Table 1: Crosstalk classes (here  $\lambda = \frac{C_L}{C_I}$ )

$$\frac{T_k(d_t, d_{t+1})}{C_L \cdot R_T} = \begin{cases} ((1 + \lambda)\Delta_1 - \lambda\Delta_2)\Delta_1 & k = 1 \\ ((1 + 2\lambda)\Delta_k - \lambda(\Delta_{k-1} + \Delta_{k+1}))\Delta_k & 1 < k < n \\ ((1 + \lambda)\Delta_n - \lambda\Delta_{n-1})\Delta_n & k = n \end{cases}$$

where  $R_T$  is the total resistance,  $C_L$  is the ground capacitance,  $C_I$  is the interwire capacitance,  $\Delta_k = d_{t+1}^k - d_t^k$ , and  $\lambda = \frac{C_L}{C_I}$ . The propagation delay,  $T(d_t, d_{t+1})$ , for transmitting  $d_{t+1}$  after  $d_t$  is defined as follows:

$$T(d_t, d_{t+1}) = \max \{T_k(d_t, d_{t+1}) \mid 1 \leq k \leq n\}.$$

Table 1 classifies the crosstalk delay effect on a wire  $k$  into six classes, depending upon the state of  $k$  (the middle wire) and its adjacent wires [9, 13]. From this table, it is clear that the worst-case crosstalk delay for transmitting data items is  $C_L R_T (1 + 4\lambda)$  (Class 6 in Table 1). This paper proposes an encoding scheme that *completely eliminates Class 5 and Class 6 crosstalk on all bus wires* resulting in reduced delay and peak power when compared to normal data transmission (i.e., without any coding) over the bus.

In this paper, a novel *temporal redundancy* based approach to encode on-chip buses is proposed to reduce both delay and peak power.

## 2. RELATED WORK

Bus encoding techniques to eliminate Class 5 and Class 6 crosstalk have been proposed in [7, 17]. Though these techniques reduce the worst-case delay, they are difficult to implement and have large area overhead (for a 32-bit data, the technique in [17] requires 46 wires without memory and 40 wires with memory and the technique in [7] requires 52 wires). Similar to the ideas of [7, 17], a new coding technique is proposed in [15] to obtain nearly 50% reduction in delay along with 10% reduction in energy, but it requires 48 wires to encode 32-bit data.

A bus encoding technique to minimize power consumption and eliminate Class 5 and Class 6 crosstalk is proposed in

[10] which requires 55 wires to encode 32-bit data. Coupling-driven bus encoding technique [8] reduces power consumption by 30% on an average.

An encoding technique based on temporal redundancy for on-chip delay and energy minimization is proposed in [11]. Instead of using temporal redundancy for each data item to be transmitted, temporal redundancy is used only when the next data forms a Class 5 or Class 6 crosstalk with the data present on the bus.

### 3. BUS ENCODING USING TEMPORAL REDUNDANCY

In the proposed temporal redundancy based  $(n, m, k)$ -encoding scheme, the original  $n$ -bit data packet to be transmitted is encoded into two  $(\frac{n}{k}) * m$ -bit data packets, where  $m < k$ , and transmitted over two consecutive cycles. In other words, the  $n$ -bit data is split into  $(\frac{n}{k})$  number of  $k$ -bit data blocks. Each of these  $k$ -bit data blocks is encoded into two  $m$ -bit data blocks  $m_1$  and  $m_2$ . All the  $m_1$  blocks are transmitted first followed by the transmission of the  $m_2$  blocks in the next cycle.

Choosing appropriate values for  $n$ ,  $m$ , and  $k$  is crucial as it affects the performance of the bus. For obvious reasons,  $k$  is chosen to be a multiple of  $n$ . As  $k$  becomes large, the codec circuit becomes more complex resulting in increased delay and power consumption. So it is desired to have  $k$  as small as possible. To get a one-to-one mapping of a  $k$ -bit data into two  $m$ -bit data blocks  $m$  needs to be at least  $k/2$ . The proposed encoding scheme needs  $m$  to be greater than  $k/2$ . As  $m$  becomes large, the number of wires in the encoded bus increases resulting in increased delay and power consumption. For  $k = 2$ , there is no integer value of  $m$ . Hence,  $k$  is chosen to be 4 and  $m$  is chosen to be 3. Illustrating the above through an example, a 32-bit data is split into eight 4-bit blocks, encoded as sixteen 3-bit blocks (two each per 4-bit block), and transmitted as two 24-bit data blocks.

For the correct operation of buses the clock period  $T_c$  should be sufficiently large so that all the transitions in the bus have enough time to be completed [13]. For example, if a Class 6 pattern occurs over the bus, from Table 1 it is seen that  $T_c \geq C_L R_T (1 + 4\lambda)$ . Note that, in the proposed  $(n, 3, 4)$ -encoding scheme, the size of the bus reduces from  $n$  to  $(\frac{n}{k}) * m$ . By assuming that the new bus occupies the same interconnect area as that of the original bus, we increase the spacing between wires using the following formula:

$$nw + (n - 1)s = (\frac{n}{k})mw + ((\frac{n}{k})m - 1)s' \quad (1)$$

where  $w$  is the width of a wire,  $s$  is the original spacing, and  $s'$  is the new spacing between wires. Increasing the spacing between wires, decreases  $C_I$  and increases  $C_L$  [5], results in  $\lambda$  decrease. Also, note that the proposed scheme eliminates Class 5 and Class 6 crosstalk and two encoded transmissions are done for each one of the original transmission. From Table 1 we see that the proposed scheme shall result in a better performance when compared to the normal transmission if and only if

$$2C'_L R_T (1 + 2\lambda') + \delta < C_L R_T (1 + 4\lambda),$$

where  $C'_L$  and  $C'_I$  are the values of the ground and interwire capacitances, respectively, for the new bus,  $\lambda' = \frac{C'_I}{C'_L}$ , and  $\delta$

4-bit	3-bit	4-bit	3-bit	4-bit	3-bit	4-bit	3-bit
0000	001 001	0100	011 001	1000	100 100	1100	111 001
0001	001 011	0101	011 011	1001	100 110	1101	111 011
0010	001 101	0110	011 111	1010	110 000	1110	111 101
0011	001 111	0111	100 000	1011	110 110	1111	111 111

Table 2: TCS Encoding

001	001	001	001
011	011	011	011
110	111	000	000
000	111	110	111
		000	111

Table 3: Possible cross-talk at TP2

is the codec delay. In the subsequent sections of this paper it is shown that the above inequality is indeed true for DSM buses.

### 4. ENCODING SCHEME

This section proposes a  $(n, 3, 4)$ -encoding scheme, namely *Temporal Crosstalk Shielding* (TCS) encoding, that eliminates Class 5 and Class 6 crosstalk when transmitting data on an on-chip bus. Consider two  $n$ -bit data  $D_1$  and  $D_2$  such that  $D_2$  is transmitted after  $D_1$ . Let  $p = \frac{n}{4}$ . As per the encoding scheme, we split both  $D_1$  and  $D_2$  into  $(d_1, d_2, \dots, d_p)$  and  $(d'_1, d'_2, \dots, d'_p)$ , respectively, where each  $d_i$  (or  $d'_i$ ),  $1 \leq i \leq p$ , is a 4-bit data. Consider four 4-bit data blocks  $d_i, d_{i+1}, d'_i$ , and  $d'_{i+1}$ , for any  $1 \leq i < p$ . Each of these 4-bit data blocks gets encoded as two 3-bit data blocks. Let the encoding of  $d_i, d_{i+1}, d'_i$  and  $d'_{i+1}$  be

$$\begin{aligned} & ((A_{20}, A_{10}, A_{00}), (A_{21}, A_{11}, A_{10})), \\ & ((B_{20}, B_{10}, B_{00}), (B_{21}, B_{11}, B_{10})), \\ & ((C_{20}, C_{10}, C_{00}), (C_{21}, C_{11}, C_{10})), \text{ and} \\ & ((D_{20}, D_{10}, D_{00}), (D_{21}, D_{11}, D_{10})), \text{ respectively.} \end{aligned}$$

The transmissions of the encoded versions of  $D_1$  and  $D_2$  shall result in the following four transmissions (in order) on six adjacent wires:

Transmission	Data
1	$(A_{20}, A_{10}, A_{00}, B_{20}, B_{10}, B_{00})$
2	$(A_{21}, A_{11}, A_{10}, B_{21}, B_{11}, B_{10})$
3	$(C_{20}, C_{10}, C_{00}, D_{20}, D_{10}, D_{00})$
4	$(C_{21}, C_{11}, C_{10}, D_{21}, D_{11}, D_{10})$

It is straightforward to see that an encoding scheme that eliminates Class 5 and Class 6 crosstalk in the six wires during the above four transmissions, for every  $16^4$  possible combinations of values for  $d_i, d_{i+1}, d'_i$ , and  $d'_{i+1}$ , can eliminate Class 5 and Class 6 crosstalk on the entire bus. This is due to the fact that the above transmission considers arbitrary adjacent blocks and also consecutive transmissions. Hence, the problem reduces to finding a one-to-one mapping from 4-bit numbers onto 6-bit numbers satisfying the above-mentioned condition.

The crosstalk in the above four transmissions can occur at *three transition points*, namely, between transmissions 1 and 2 (TP1), 2 and 3 (TP2), or 3 and 4 (TP3). An exhaustive search on different 4-bit to 6-bit mappings was done and none of the mappings could eliminate Class 5 and Class

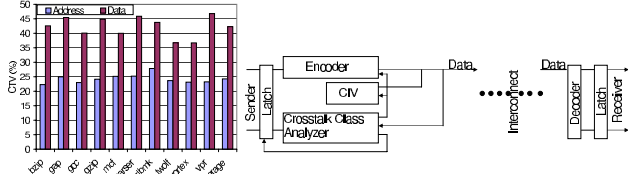


Figure 2: Data transmission mechanism used in the TCS

Figure 1: Benchmark-wise CTV coding technique for both address and data buses

6 crosstalk on all of the three above-mentioned transition points. However, the TCS encoding given in Table 2 satisfies the following theorem.

**Theorem 1:** The TCS encoding eliminates Class 5 and Class 6 crosstalk at TP1 and TP3.

**Proof:** Consider the TP1 as mentioned above. The  $A$  and  $B$  values are transmitted on six adjacent lines. From the TCS encoding table (Table 2), it is easy to show that there are no opposite transitions on adjacent lines carrying  $A$  and  $B$  values. From Table 2, it is clear that  $A_{00} = A_{10}$  for all the sixteen encodings. This ensures that there are no opposite transitions on adjacent lines in the boundary between  $A$  and  $B$ . This proves that there is no Class 5 or Class 6 crosstalk at the TP1. A similar argument proves the Theorem for the TP3.  $\square$

TCS encoding partially removes Class 5 and Class 6 crosstalk at TP2. An example of possible crosstalk at TP2 is illustrated in Table 3.

**Remark 1:** Out of the  $16^4$  combinations of codes as in the above case, the fraction of them leading to Class 5 or Class 6 crosstalk at TP2 is defined as the Crosstalk Value (CTV).

Note that each 4-bit data is transmitted as two 3-bit data one after the other. Denoting the two transmissions as the *first transmission* and *second transmission*, respectively, the above-mentioned Class 5 and Class 6 crosstalk occur only after the second transmission of one 4-bit data and before the first transmission of the next 4-bit data. To eliminate such a crosstalk at TP2, we employ *temporal shielding*. Unlike the *spatial shielding* [5], where shielding wires are inserted between adjacent wires, in temporal shielding, shielding data is inserted between two consecutive data items. We use a *Crosstalk Class Analyzer* to detect Class 5 and Class 6 crosstalk at TP2 and transmit a *Crosstalk Identification Vector* (CIV) before transmitting the next data only if there is such a crosstalk. In order not to create Class 5 or Class 6 crosstalk with the previously transmitted data, we consider a *zero vector* [00...00] as the CIV. The previous four transmissions in the above example shall now become five transmissions due to the transmission of the CIV as shown in Table 4.

For a given encoding scheme, the CTV (defined in Remark 1) is a measure of the number of CIV transmissions required. Figure 1 shows benchmark-wise CTV for both address and data buses. For the decoder to distinguish between a normal data transmission and a CIV, it is essential that [00...00] should never be transmitted during the *first transmission* of any normal data item. This is ensured in our encoding as none of the sixteen encodings shown in Table 2 have 000 as their first three bits. Figure 2 shows the

Node ( $nm$ )	90	65
Width ( $nm$ )	237	160
Spacing ( $nm$ )	237	160
Thickness ( $nm$ )	498	325
Height ( $nm$ )	498	325
$R$ ( $\Omega/mm$ )	187	423
$V_{DD}$ (V)	1	0.7
$f_{clk}$ (GHz)	3.99	6.73

Table 5: Technology parameters for global wires derived from [4]

Parameter	Without coding		With coding	
Node ( $nm$ )	90	65	90	65
Spacing ( $\mu m$ )	237	160	402	271
$C_L$ (fF/mm)	27.260	22.127	36.287	29.330
$C_I$ (fF/mm)	91.943	70.159	55.909	42.711

Table 6: Capacitance values for different spacing between wires

data transmission mechanism used in our approach. Whenever new data is received, we simultaneously feed it to the *Encoder* and *Crosstalk Class Analyzer* (this way we can reduce the delay associated with the codec). The Encoder encodes the data using Table 2. The Crosstalk Class Analyzer contains two units in which one checks whether or not the middle bit in a 3-bit encoded data forms a Class 5 or Class 6 crosstalk with its adjacent bits (let us call it as *middle bit crosstalk unit*) and the other one checks whether or not the boundary bits of two different 3-bit encoded data forms a Class 5 or Class 6 crosstalk (let us say *boundary bit crosstalk unit*). For middle bit crosstalk unit, we maintain a  $16 \times 16$  table whose rows and columns are indexed by the sixteen 4-bit data. The table stores either 1 or 0 in each entry based on whether or not the lower 3-bit encoded data of the column index forms a Class 5 or Class 6 crosstalk with the upper 3-bit encoded data of the row index, respectively. The middle bit crosstalk unit always stores the previous 32-bit data (i.e., the original data) and whenever it receives a new data, it indexes into the table using the previous data and new data to know whether there is any Class 5 or Class 6 crosstalk. After checking for crosstalk, the middle bit crosstalk unit updates the previous data with the new data for future use. For boundary bit crosstalk unit, we categorize each 4-bit data (From Table 2) into one of the three classes from the set {01, 10, 11}. We know that each 4-bit data is encoded into two 3-bit data. We use the MSB and LSB pair of the first 3-bit encoded data for the categorization. For example, consider a 4-bit data 0001. It is encoded as 001 and 011 (From Table 2). Since the MSB and LSB pair of the first 3-bit encoded data is 01, 0001 belongs to 01-class. In this way, we categorize all the sixteen 4-bit data. Hence, 01-class = {0000, 0001, 0010, 0011, 0100, 0101, 0110}, 10-class = {0111, 1000, 1001, 1010, 1011}, and 11-class = {1100, 1101, 1110, 1111}. Whenever the boundary bit crosstalk unit receives a data, it checks which class the data belongs and considers the corresponding value. That is, if the data belongs to 10-class, the boundary bit crosstalk unit considers 10 as the value for the data. The boundary bit crosstalk unit compares this data with the data obtained by taking the MSB and LSB pairs of each of the 3-bit encoded data from the bus. Note that both crosstalk units work simultaneously to reduce the effective codec delay. If either of these units finds that there is a Class 5 or Class 6 crosstalk

Circuit	Power		Delay ( $ns$ )	Area ( $\mu m^2$ )
	Dynamic ( $mW$ )	Leakage ( $nW$ )		
Encoder	2.6899	59.7973	0.4	958.81
Decoder	2.6899	59.7973	0.4	958.81
Crosstalk Class Analyzer	0.2886	13.0563	0.4	209.664

Table 7: Delay and power overhead of the Encoder, Crosstalk Class Analyzer, and Decoder

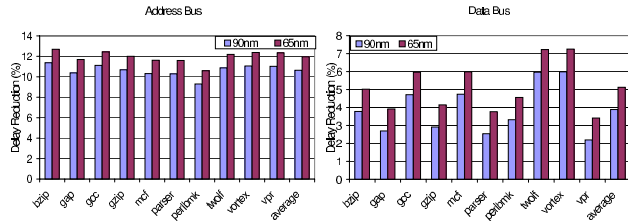


Figure 3: Delay reduction for both address and data buses

between present data and next data, the Crosstalk Class Analyzer sends a signal to the Encoder, CIV, and the latch at the sender side, so that the CIV transmits a zero vector, Encoder stops sending the encoded data for one cycle, and data (if any) at the sender side will be latched for two cycles. If there is no Class 5 or Class 6 crosstalk between present data and next data, the Encoder sends the first 3-bit encoded data, CIV stops sending a zero vector, and data (if any) at the sender side will be latched for one cycle. Note that we use  $C_L R_T(1 + 2\lambda)$  as the cycle period. At the receiver end, the Decoder maintains a decode bit which is set whenever the Decoder receives a data other than a zero vector and reset whenever the Decoder receives another data for decoding to obtain the original data. With the decode bit, we can differentiate whether the data received is a zero vector or an useful data. If the received data is a zero vector, the Decoder discards the data. Otherwise, it stores the data for one cycle and obtains the original data using the stored data and the data on the bus in the next cycle.

## 5. EXPERIMENTAL VALIDATION AND CONCLUSION

The methodology described in Section 4 is used to design a 10mm global bus in the top metal layer for 90nm and 65nm technology nodes. The ITRS technology parameters are shown in Table 5. Table 6 shows the capacitance values for the original spacing (as shown in Table 5) and the new spacing, which is calculated using Equation 1. The Predictive Technology Model [1] is used for calculating the capacitance values. We designed the Encoder, Crosstalk Class Analyzer, and Decoder in Verilog and synthesized it using the Synopsys Design Compiler with 90nm TSMC technology library. Delay and power overhead of the Encoder, Crosstalk Class Analyzer, and Decoder are shown in Table 7. We used SimpleScalar 3.0 [2] and the SPEC2000 CINT [3] benchmark suite to simulate the performance of different on-chip buses between the processor datapath and L1 I-cache/D-cache. For each benchmark, we fast-forwarded the first 100 million instructions and then simulated the next 100 million instructions. Figure 3 shows the delay reduction percentages

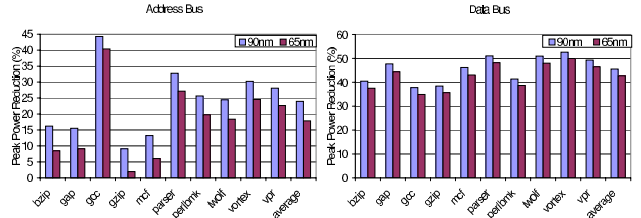


Figure 4: Peak power reduction for both address and data buses

for both address and data buses compared to the base case (without any coding). Note that the results in Figure 3 include the codec delay. The delay reduction is high in the case of address bus because of less CTV. Overall, our technique achieves 11% (4%) and 12% (5%) reduction in the address (data) bus for 90nm and 65nm technologies, respectively. Figure 4 shows the peak power reduction of our technique compared to the base case for both address and data bus. We include the codec power overhead in the results. On an average, our technique reduces the peak power consumption by 46% (24%) and 43% (17%) in the data (address) bus for 90nm and 65nm technologies, respectively. Our technique consumes less peak power because of three reasons: 1) less number of wires 2) reduced coupling capacitance, and 3) no Class 5 and Class 6 crosstalk transitions.

## 6. REFERENCES

- [1] Berkeley predictive technology model. <http://www-device.eecs.berkeley.edu/~ptm/interconnect.html>
- [2] SimpleScalar Toolset. <http://www.simplescalar.com>.
- [3] SPEC CPU2000 Benchmark. <http://www.spec.org>
- [4] International Technology Roadmap for Semiconductors, 2001. <http://public.itrs.net>
- [5] R. Arunachalam et al. Optimal shielding/spacing metrics for low power design. In *Proceedings of IEEE Computer Society Annual Symposium on VLSI*, pages 167-172, 2003.
- [6] F. Caignet et al. "The Challenge of Signal Integrity in Deep-submicrometer CMOS Technology". *IEEE*, 89(4), 2001, pp. 556-573.
- [7] C. Duan et al. "Analysis and Avoidance of Crosstalk in On-chip Buses". In *Hot Interconnects*, 2001, pp. 133-138.
- [8] K. Kim et al. "Coupling-driven Signal Encoding Scheme for Low-power Interface Design". In *ICCAD*, 2000, pp. 318-321.
- [9] L. Li et al. "A Crosstalk Aware Interconnect with Variable Cycle Transmission". In *DATE*, 2004, pp. 102-107.
- [10] C.-G. Lyuh et al. "Low Power Bus Encoding with Crosstalk Delay Elimination". In *ASIC/SOC*, 2002, pp. 389-393.
- [11] M. Mutyam et al. "Delay and Energy Efficient Data Transmission for On-Chip Buses". In *ISVLSI*, 2006.
- [12] P. Sotiriadis et al. "Low Power Bus Coding Techniques Considering Inter-wire Capacitances". In *CICC*, 2000, pp. 507-510.
- [13] P. Sotiriadis et al. "Reducing Bus Delay in Sub-micron Technology using Coding". In *ASPAC*, 2001, pp. 109-114.
- [14] P. Sotiriadis et al. "A Bus Energy Model for Deep Submicron Technology". *TVLSI*, 10(3), 2002, pp. 341-350.
- [15] S. R. Sridhara et al. "Area and Energy-efficient Crosstalk Avoidance Codes for On-chip Buses". In *ICCD*, 2004, pp. 12-17.
- [16] D. Sylvester et al. "Analytical Modeling and Characterization of Deep-submicrometer Interconnect". *IEEE*, 89(5), 2001, pp. 634-664.
- [17] B. Victor et al. "Bus Encoding to Prevent Crosstalk Delay". In *ICCAD*, 2001, pp. 57-63.
- [18] J. Yim et al. "Reducing Cross-coupling among Interconnect Wires in Deep-submicron Datapath Design". In *DAC*, 1999, pp. 485-490.