Power Effective Bus Encoding Scheme with No Crosstalk and Minimized Bus Transitions

Young Chul Kim[1] and Youn Jin Lee[2]

[1]*Dept. Electronics and Computer Engineering, Chonnam National Univ.*
[2]*Dept. Communication, R&D Institute of LG Innotek Co.*

yckim@chonnam.ac.kr, yjlee@lginnotek.com

Abstract

In this paper, we propose a power effective bus encoding scheme which can minimize crosstalk problems as well as bus transitions effectively. In current submicron technology, minimizing propagation delay and power consumption on buses is one of the most important design objectives in the field of System-on-Chip (SoC) design. Crosstalk between adjacent wires on bus may create significant propagation delay. Elimination or minimization of such faults is crucial not only for the system performance, but also for the reliability of SoC designs. Experimental results show that our proposed encoding technique can save dynamic power by up to 23.3% for 4- bit buses, while completely removing crosstalk delay.

Keywords: *Bus-encoding, crosstalk, SoC, coupling capacitance, dynamic power*

1. Introduction

The optimization of crosstalk delay is critical in ultra-deep submicron technology [1-3]. Crosstalk occurs when a signal on a wire affects the signal on a neighboring wire due to the capacitive and inductive coupling between neighboring wires. Crosstalk may cause problems with by hastening or delaying signal transition or may cause logic failure by inducing a glitch or spurious signal transition on the victim wire. It can also cause increased energy consumption in on-chip communications, which is a function of coupling capacitances [4-6].

However, most previous work on bus encoding is targeted at either minimizing the bus transition or minimizing the crosstalk delay, but not both. M.R. Stan [7] and Osborne [8] proposed an encoding scheme which is efficient for off-chip buses where line capacitance and impedances are suitably adjusted to reduce crosstalk noise. However, for on-chip buses, the major issue in terms of power consumption is regarding the inter-wire coupled capacitance, minimizing this is key for reducing power consumption. Harmander's method [9] is well suited to reducing the worst cases of crosstalk. However, there is no guarantee that the method is also power-efficient. In contrast, Stanislaw [10] proposed a scheme that is power efficient, but can't avoid the worst cases of crosstalk. Khan [11] and S. K. Verma [12] proposed a scheme to reduce both crosstalk noise and power dissipation, but it can't reduce crosstalk when transition from "01" to "10" occurs. Hsieh and Chen [13] proposed a bus encoding scheme which exploit probabilistic information in the data. However, this scheme has limitations because of its computational complexity when the bus size becomes large.

In this paper, we propose an adaptive encoding method that targets the problem of both bus transition minimization and crosstalk reduction. We classify crosstalk type and solve the problem by applying an invert-block for minimizing bus transition and a

convert-block for eliminating crosstalk. It transforms the incoming data in such a way as to efficiently eliminate the worst cases of crosstalk.

The remainder of this paper is organized as follows: We present the related work in Section 2. We describe our proposed bus encoding schemes in Section 3. Section 4 shows the experimental results. In Section 6, we conclude.

2. Preliminaries

This section presents six types of crosstalk by considering four adjacent wires shown in Figure 1. The classification of crosstalk into types is done to emphasize two aspects of the encoding scheme. The first is the elimination/minimization of the worst crosstalk and the second is energy efficiency.

2.1. Bus Energy Dissipation

Muhammad Khellah [14] gave an approximate energy function for the self and coupled switching activity by considering a lumped model of the bus. The same lumped model shown in Figure 1 is considered here for the four adjacent wires. The model is used to provide an expression for the energy consumption when each type of crosstalk is considered alone and then derive a total energy expression when all types of crosstalk coupling occur together. All-pairs of adjacent lines are assumed to have the same coupling capacitance C_I, and all signal changes appear on the bus lines at the same time based on the energy dissipation model used [15]. According to this model, we classify crosstalk into six types based on position and number of crosstalk coupling.

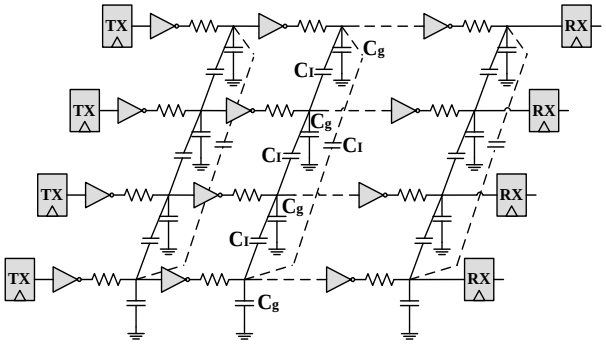

Figure 1. Four-line cyclic bus

2.2. Crosstalk Model

Six Types of crosstalk for all possible transition cases of the 4-bit bus line are shown in Table 1. Here α_c is the relative transition activity coefficient and change_info represents transition from previous data to current data.

1) Type 1: A transition or transitions in the same direction for adjacent wires cause type 1 crosstalk. Note that the real bus is often more than 4 bits, so α_c is same for both the transition from “1000” to “0000” and the transition from “0100” to “0000”.

2) Type 2: Transitions for two non-adjacent wires cause type 2 crosstalk. Type 2 crosstalk can also arise in the case of transitions in the same direction in one of the two nonadjacent wires. For example, the transition from “0010” to “1111”.

3) Type 3: Type 3 crosstalk occurs if two adjacent wires undergo opposite state transitions. Meanwhile, this crosstalk will also occur in the case where one or both adjacent wires have transitions in the same direction as one of the two wires mentioned above. For example, the transition from "0111" to "1000".

4) Type 4: Transitions for two adjacent wires in the opposite direction as a non-adjacent wire at the same time. For example the transition from "0101" to "1000".

5) Type 5: Transitions for three adjacent wires in opposite directions or three wires in the opposite direction as a wire in the same direction cause type 5 crosstalk. For example, the transition from "0100" to "1011".

6) Type 6: Transitions for four wires in the opposite direction cause type 6 crosstalk. For example, the transition from "0101" to "1010".

The dynamic energy dissipation per cycle of a bus line due to self- transition can be written as follows:

$$E_{ig} = \alpha_g \cdot C_g \cdot V^2 \tag{1}$$

where C_g is the wire-to-substrate capacitance and α_g is the self- transition energy coefficient of the line of interest, which takes the value of 0 when the line is quiet at 0 or at V, and the value of ½ when the line has rising or falling transition activity.

Therefore, the net transition activity can be given by

$$E/E_{ig} = N+2\cdot\lambda\cdot (7\cdot N_7+5\cdot N_5+4\cdot N_4+3\cdot N_3+2\cdot N_2+1\cdot N_1) \tag{2}$$

where λ is the ratio of inter-wire capacitance(C_I) to the wire-to-ground capacitance(C_g), N and N_i are the total self and type i transition activity, respectively. Equation (3) then gives the energy saving.

$$Energy\ saving = (1 - N_C/N_U) \cdot 100\% \tag{3}$$

where N_U and N_C are net transition activity in the non-encoded and corresponding encoded data, respectively.

3. Proposed Algorithm

The proposed encoding scheme is based on an intrinsic property exhibited by 4-bit binary sequences. From the energy equation given in (2), it becomes evident that crosstalk is the major source of energy consumption in on-chip communication. Crosstalk also causes considerable and unpredictable delays in signal transitions and can result in logic errors that can cause partial or complete system failure. Therefore, our proposed algorithm aims to eliminate typical crosstalk from type 1 to type 6 in series.

The proposed bus encoding scheme is shown in Figure 2. The scheme consists of an invert-block, a convert-block, and a detect-block. Depending on bus data transitions, the detect-block chooses one of two blocks: the invert-block or convert-block. The input bus data $x(n)$ on 4-bit bus is XORed with the previous bus state $y(n-1)$. The output $z(n)$ is divided into two parts.

Table 1. Classified type of crosstalk

Type (α_c)	Previous data	Present data	Conversion Info.	Bit Info.
Type1 (α_c=1)	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ b_2\ b_1\ b_0$	1 0 0 0	
	$b_3\ b_2\ b_1\ b_0$	$b_3\ \overline{b_2}\ b_1\ b_0$	0 1 0 0	
	$b_3\ b_2\ b_1\ b_0$	$b_3\ b_2\ \overline{b_1}\ b_0$	0 0 1 0	
	$b_3\ b_2\ b_1\ b_0$	$b_3\ b_2\ b_1\ \overline{b_0}$	0 0 0 1	
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ b_1\ b_0$	1 1 0 0	$b_3 = b_2$
	$b_3\ b_2\ b_1\ b_0$	$b_3\ \overline{b_2}\ \overline{b_1}\ b_0$	0 1 1 0	$b_2 = b_1$
	$b_3\ b_2\ b_1\ b_0$	$b_3\ b_2\ \overline{b_1}\ \overline{b_0}$	0 0 1 1	$b_1 = b_0$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ b_0$	1 1 1 0	$b_3 = b_2 = b_1$
	$b_3\ b_2\ b_1\ b_0$	$b_3\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	0 1 1 1	$b_2 = b_1 = b_0$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	1 1 1 1	$b_3 = b_2 = b_1 = b_0$
Type2 (α_c=2)	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ b_2\ \overline{b_1}\ b_0$	1 0 1 0	
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ b_2\ b_1\ \overline{b_0}$	1 0 0 1	
	$b_3\ b_2\ b_1\ b_0$	$b_3\ \overline{b_2}\ b_1\ \overline{b_0}$	0 1 0 1	
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ b_1\ \overline{b_0}$	1 1 0 1	$b_3 = b_2$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ b_2\ \overline{b_1}\ \overline{b_0}$	1 0 1 1	$b_1 = b_0$
Type3 (α_c=3)	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ b_2\ b_1\ b_0$	1 0 0 0	$b_3 \neq b_2$
	$b_3\ b_2\ b_1\ b_0$	$b_3\ \overline{b_2}\ b_1\ b_0$	0 1 0 0	$b_2 \neq b_1$
	$b_3\ b_2\ b_1\ b_0$	$b_3\ b_2\ \overline{b_1}\ \overline{b_0}$	0 0 1 1	$b_1 \neq b_0$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ b_0$	1 1 1 0	$b_3 \neq b_2 = b_1$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ b_0$	1 1 1 0	$b_3 = b_2 \neq b_1$
	$b_3\ b_2\ b_1\ b_0$	$b_3\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	0 1 1 1	$b_2 \neq b_1 = b_0$
	$b_3\ b_2\ b_1\ b_0$	$b_3\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	0 1 1 1	$b_2 = b_1 \neq b_0$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	1 1 1 1	$b_3 \neq b_2 = b_1 = b_0$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	1 1 1 1	$b_3 = b_2 \neq b_1 = b_0$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	1 1 1 1	$b_3 = b_2 = b_1 \neq b_0$
Type4 (α_c=4)	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ b_1\ \overline{b_0}$	1 1 0 1	$b_3 \neq b_2$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ b_2\ \overline{b_1}\ \overline{b_0}$	1 0 1 1	$b_1 \neq b_0$
Type5 (α_c=5)	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ b_0$	1 1 1 0	$b_3 \neq b_2 \neq b_1$
	$b_3\ b_2\ b_1\ b_0$	$b_3\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	0 1 1 1	$b_2 \neq b_1 \neq b_0$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	1 1 1 1	$b_3 \neq b_2 \neq b_1 = b_0$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	1 1 1 1	$b_3 \neq b_2 = b_1 \neq b_0$
	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	1 1 1 1	$b_3 = b_2 \neq b_1 \neq b_0$
Type6 (α_c=7)	$b_3\ b_2\ b_1\ b_0$	$\overline{b_3}\ \overline{b_2}\ \overline{b_1}\ \overline{b_0}$	1 1 1 1	$b_3 \neq b_2 \neq b_1 \neq b_0$

If z(n) is “1111”, “1110”, “1101”, “1011”, “0111”, “1010”, “0101” or” 0110”, x(n) is inverted and de_info (decode information) is set to ‘0’. Otherwise x(n) is logically converted and de_info is set to ‘1’.

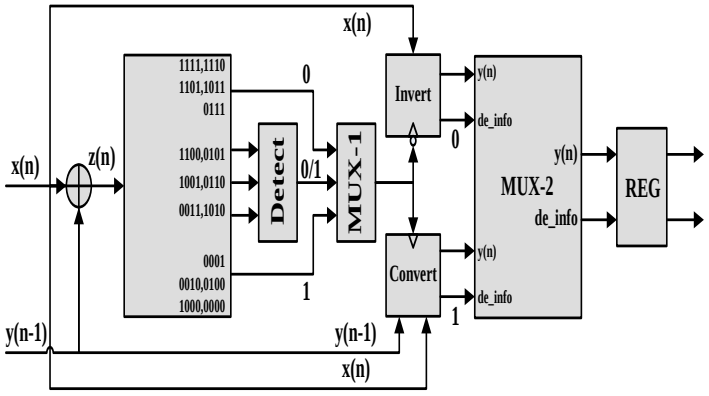

Figure 2. . Diagram of the proposed bus encoding scheme

3.1. Inverting Function

Let's denote the data value as the piece of information that has to be transmitted over the bus in a given time-slot. Then the bus value denotes the coded value. Typically, a code needs extra control bits. The Invert method proposed here uses one extra control bit called de_info.

If *de_info* has a value of '1', the bus value is the same as the data value. When the *de_info* value is '0', the bus value will be the inverted data value. Peak power dissipation can be halved by using our inverting method. The process as follows:

1) Compute the Hamming distance between the current bus value and the previous bus value.

2) If the Hamming distance is larger than 2, set de_info to '0' and make the current bus value be the same as the inverted current data value.

3) Otherwise let de_info be '1' and let the current bus value be the same as the current data value.

4) The contents of the bus at the receiver's side must be conditionally inverted according to the de_info bit.

Transitions for two adjacent wires in opposite directions will cause large energy dissipation. For example, transitions from "01" to "10", "010" to "101", "0101" to "1010", and so on. In other words, when *z(n)* is "1111", "1110", and "0111", we make the current bus value to the same as the inverted current data value to eliminate the worst case of crosstalk.

However, the probability of transitions of two bits in 4-bit bus data is nearly 40%. Transitions for two adjacent wires in opposite direction will cause large energy dissipation. For example, transition from "0100" to "1000", "0010" to "0100", "0001" to "0010", and so on. In this case the convert-block will be used to solve the problem.

3.2. Converting Function

Stan and Burleson [7] proved that the Bus-Invert is optimal in the sense that given the same redundancy no other coding can achieve a better reduction in the number of transitions. This can be seen by considering all possible subsequent values on the bus.

If the next value differs by only one position and there will be $C^1_{n+1} = n+1$ possible values. Similarly there will be C^2_{n+1} values that generate two transition, C^3_{n+1} values that generate three transitions, up to $C^{n/2}_{n+1}$ values that will generate n/2 transitions (we can assume n to be even without losing generality). Thus,

$$C^0_{n+1} + C^1_{n+1} + C^2_{n+1} + \ldots + C^{n/2}_{n+1} = 2^n. \quad (4)$$

All 2n possible subsequent values are taken into account. The average number of transitions per bus cycle will be

$$(0 \cdot C^0_{n+1} + 1 \cdot C^1_{n+1} + 2 \cdot C^2_{n+1} + \ldots + n/2 \cdot C^{n/2}_{n+1})/2^n. \quad (5)$$

It can be seen that the bus-invert method uses all patterns up to, at most, n/2 transitions. Any other code with 2n code words can either use a permutation of these same patterns and will exhibit the same average bus activity or use patterns but with more than n/2 transitions and generates more bus activity.

This means the self-transition shown as "¯" in Table 1 can't be further eliminated, but the crosstalk caused by transitions for two adjacent wires in opposite directions can be eliminated by changing the position of "¯". The proposed convert-block is shown in Figure 3, where *x(n)* is the 4-bit input data, *y(n)* is the current 4-bit output data, *y(n-1)* is the 4-bit previous output data, and *z(n)* is *x(n)* XOR *y(n-1)*. The function of the convert-block is to simply change the position of "¯" from b_2 to b_1 or from b_1 to b_2. For example, if *z(n)* is "1100", there will be large energy dissipation caused by transitions of b_3 and b_2 in opposite directions, the convert-block can change *z(n)* into "1010", then the worst case of crosstalk in the coupling capacitance between b_2 and b_1 will be efficiently eliminated.

Next, we look in to whether the worst case of crosstalk can be eliminated completely as well as keeping self-transitions to a minimum. Note that for 1-bit decoding information, we can only use two kinds of logic operation, meanwhile either an invert or convert must be performed.

•If z(n) is one of the cases "1111", "1110", "1101", "1011" and "0111", the invert-block will change z(n) into "0000", "0001", "0010", "0100" and "1000", respectively.

•If z(n) is one of the cases "0000", "0001", "0010", "0100" and "1000", the convert-block will change z(n) into "0000", "0001", "0100", "0010" and "1000", respectively.

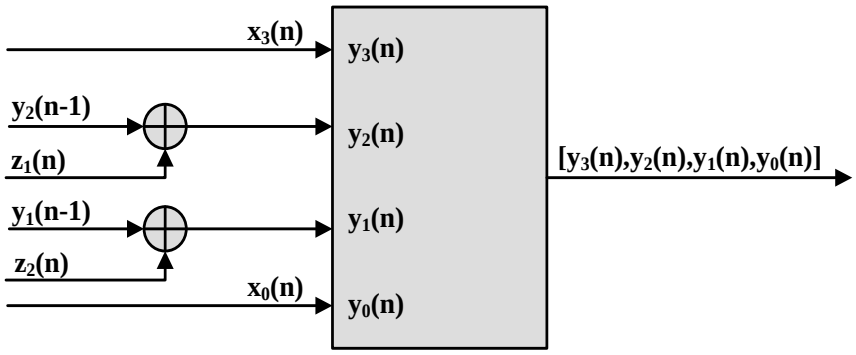

Figure 3. Proposed convert-block

In Khan's algorithm [11], the 4-bit input bus data performs XOR on "0101" and "1010". Then, it calculates the worst case of crosstalk in the coupling capacitance

between two adjacent wires in the opposite direction. Finally, based on the coupling capacitance, it decides on the operation to be performed. However, in the case where *z(n)* is equal to "0000", the possible results of these operations are "0101" and "1010". As such, the self-transition will increase if *z(n)* becomes "1001" or "0110". Both kinds of operation may cause the worst case of crosstalk simultaneously. Meanwhile, our proposed algorithm never increases the self-transition activity, which is one of its key advantages over other schemes.

3.3. Detecting Function

By using the invert-block transitions in four or three wires can be eliminated completely. However, to deal with the transitions in two wires with a 4-bit bus data is a challenging task.

For two adjacent wires, if the transitions are in the same direction, there will be no worst case of crosstalk in the coupling capacitance. That is, α_c is 0. If transitions occur in the opposite direction, it will cause the worst case of crosstalk. That is, α_c is 2. For any line *i* and *i+1* with $i<n-1$, we can create a four-state Markov chain. The states "00", "01", "10", and "11" represent the logic values of the two bus lines having the same binary values. The transition probabilities from one state to another are $P_{00} = P_{01} = P_{10} = P_{11} = 0.25$. Hence, each state's probability in a steady state is 0.25. Therefore, $P(\alpha_c = 0) = 0.375$, $P(\alpha_c = ½) = 0.5$, $P(\alpha_c = 2) = 0.125$. Consequently, the probability of transitions for two adjacent wires in the same direction is 50%, while the probability of transitions for two adjacent wires in the opposite direction is also 50%. The proposed converting function can accomplish either changing transitions for two adjacent wires into transitions in two nonadjacent wires or changing transitions for two nonadjacent wires into transitions for two adjacent wires. The detect-block is designed to detect the crosstalk of coupling capacitance in two adjacent wires.

In summary, our proposed algorithm implemented with three functional blocks can minimize self-transition activity, eliminate the worst cases of crosstalk, and lead to the overall crosstalk being efficiently reduced to a minimal level.

4. Simulation Result

The proposed algorithm was implemented in Matlab and executed on an Intel Core 2 dual 1.8 GHz computer. We used a sets of random data for 4-bit, 8-bit, 16-bit, 32-bit and 64-bit buses with various λ.

Table 2 shows the power saving results for Traditional Bus-Invert [7], Harmander's algorithm [9], Khan's scheme [11], Gray code, Convolution code, and our proposed scheme.

When λ = 0 it means that crosstalk does not exist and only self-transition activities are considered. When λ = 0, Harmander's algorithm is effective at eliminating the worst cases of crosstalk delay but at the cost of increasing self-switching activity, this means that the efficiency of Harmander's bus encoding scheme is restricted by frequent crosstalk. Khan's algorithm is similar to Harmander's algorithm, but there is no cost associated with self-switching activity. Gray code is meaningless for the data bus. Convolution code is effective for self-switching elimination but requires a larger overhead. The proposed algorithm is nearly the same as Bus-Invert for minimizing self -switching activity while same area overhead is required.

Table 2. Power saving comparison of different encoding method

	Power saving(%)	Bus width(bit)				
		4	8	16	32	64
	Original	0	0	0	0	0
	[4]	12.574	12.496	13.214	13.665	12.26
	[6]	-9.687	-8.668	-8.891	-9.508	-9.575
	[8]	1.735	1.803	2.119	1.942	1.282
	[14]	0.45	-0.566	0.543	0.0149	0.0601
	[15]	18.368	18.704	19.296	19.199	18.571
	Ours	12.554	12.115	13.054	13.279	12.27
	Original	0	0	0	0	0
	[4]	10.016	19.47	22.914	21.968	22.787
	[6]	9.383	17.564	21.089	22.611	23.424
	[8]	12.165	20.864	24.248	25.045	25.832
	[14]	1.217	1.599	1.530	1.226	1.2132
	[15]	13.311	21.163	24.534	25.514	26.296
	Ours	23.33	31.175	34.118	34.487	35.175
	Original	0	0	0	0	0
	[4]	10.016	23.972	27.902	27.132	28.052
	[6]	9.383	23.522	27.475	29.18	30.074
	[8]	12.165	25.029	28.904	29.899	30.784
	[14]	1.366	1.771	1.679	1.390	1.373
	[15]	13.311	21.593	25.645	26.936	27.858
	Ours	23.33	35.112	38.466	39.024	39.793
	Original	0	0	0	0	0
	[4]	10.016	27.058	31.267	30.595	31.569
	[6]	9.383	27.606	31.783	33.585	34.517
	[8]	12.165	27.884	32.046	33.154	34.092
	[14]	1.471	1.471	1.780	1.500	1.479
	[15]	13.311	21.887	26.395	27.89	28.902
	Ours	23.33	37.811	41.4	42.066	42.879
	Original	0	0	0	0	0
	[4]	10.016	31.012	35.517	34.645	35.973
	[6]	9.383	32.839	37.225	39.118	40.08
	[8]	12.165	31.543	36.014	37.242	45.887
	[14]	1.613	2.040	1.907	1.639	1.613
	[15]	13.311	22.265	27.342	29.088	30.209
	Ours	16.13	30.209	40.08	35.973	46.74

The power savings for 4-bit bus values are obviously less than the others when $\lambda = 1, 2, 3, 4$. This is because the ratio in 4-bit bus between the coupling capacitance and line-to-ground capacitance is far less than in 8-bit bus, however, the ratio of C_I to C_g in 32-bit and 64-bit is not obvious. From the simulation results, it is noted that the bigger λ is, the more power saving is possible.

For the number of extra wires inserted, Table 3 shows the wire overhead required to implement different algorithms. The area-overhead of the proposed algorithm is the as

in the traditional Bus-Invert scheme. Also as the bus width gets wider, the effectiveness of our approach improves.

Table 3. Number of extra wires

Bus width (bit)	[4]	[6]	[8]	[14]	[15]	Our's
4	6	6	6	4	7	6
8	13	13	13	9	15	13
16	27	26	27	19	31	27
32	55	53	55	39	63	55
64	111	109	111	87	127	111

5. Conclusions

In this paper, we presented an adaptive low power bus encoding technique that addresses energy loss and delay problems faced by today's tightly coupled on-chip buses implemented in ultra-deep submicron SoC system. Unlike previous encoding methods which either addressed minimizing the power consumption or eliminating crosstalk delay exclusively, we tackled both these problems together. Specifically, we classified crosstalk type based on position and the number of crosstalk couplings, then solved the problem by using an inverting function to minimize bus transitions and a converting function to eliminate crosstalk. Experimental results showed that, as VLSI technology advances further, the proposed algorithm becomes better suited for bus encoding.

Acknowledgements

This research was financially supported by the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea (NRF) through the Human Resource Training Project for Regional Innovation (No. 2012-04A0301912010100).

References

[1] S. Shrivastava and H. Parameswaran, "Statistical Crosstalk Noise Analysis Using First Order Parameterized Approach for Aggressor Grouping", Proceeding of the 9th International Symposium on Quality Electronic Design, **(2008)** March, pp. 445-449, San Jose, CA, USA.

[2] S. Chede, K. Kulat and R. Thakare, "A significance of VLSI Techniques for Low Power Real Time Systems", J. International Journal of Computer Applications, vol. 1, no. 22. **(2010)**, pp. 86-91.

[3] J. Singh, K. Saha and G. L. Pahuja, "Adaptive Method for Minimization of Power Consumption in Sequential Circuits Through DVS and Error Prediction", J. International Journal of Advanced Science and Technology, vol. 46, **(2012)** September, pp. 17-24.

[4] V. C. Nguyen, V. T. Pham and B. K. Moon, "A New Energy Saving Mechanism in IEEE 802.16e/m", J. International Journal of Energy, Information and Communications, vol. 2, issue 4, **(2011)** November, pp. 157-168.

[5] H. G. Ryu and D. H. Kim, "Design of Wireless System with Minimum Eb/No and Optimization of Power Consumption", J. International Journal of Energy, Information and Communications, vol. 3, issue2, **(2012)** May, pp. 35-44.

[6] T. Zhang and S. S. Sapatnekar, "Simultaneous shield and buffer insertion for crosstalk noise reduction in global routing", J. IEEE Trans. on Very Large Scale Integration (VLSI) system, vol. 15, no. 6, **(2007)**, pp. 624-636.

[7] M. R. Stan and W. P. Burleson, "Bus-Invert Coding for Low Power I/O", J. IEEE Trans. on Very Large Scale Integration (VLSI) system, vol. 3, no. 1, **(1995)** March, pp. 49-58.

[8] S. Osborne, A. T. Erdogan, T. Arslan and D. Robinson, "Bus encoding architecture for low-power implementation of an AMBA based SoC platform", Proceeding of the IEE Computers and Digital Techniques, vol. 149, no. 4, **(2002)** July, pp. 152–156.

[9] H. Singh, R. Rao, K. Agarwal, D. Sylvester and R. Brown, "Dynamically Pulsed MTCMOS with Bus Encoding for Reduction of Total Power and Crosstalk Noise", J. IEEE Trans. On Very Large Scale Intergration (VLSI) system, vol. 18, no. 1, **(2010)** January, pp. 166-170.

[10] S. J. Piestrak, O. Sentieys, "Designing Efficient Codecs for Bus-Invert Berger Code for Fully Asymmetric Communication", J. IEEE Trans. On Circuits and System-II: Express Briefs, vol. 57, no. 10, **(2010)** October, pp. 777–781.

[11] Z. Khan, T. Arslan and A. T. Erdogan, "Low power system on chip bus encoding scheme with crosstalk noise reduction capability" Proceeding of the IEEE Computer and Digital Techniques, vol. 153, no. 2, **(2006)** March, pp. 101-108.

[12] S. K. Verma and B. K. Kaushik, "Crosstalk and Power Reduction Using Bus Encoding in RC Coupled VLSI Interconnects", Proceeding of IEEE Third International Conference on Emerging Trends in Engineering and Technology, **(2010)**, pp. 735-740.

[13] W. W. Hsieh, P. Y. Chen, C. Y. Wang and T. T. Hwang, "A Bus-Encoding Scheme for Crosstalk Elimination in High-Performance Processor Design", J. IEEE Trans. On Computer-Aided Design of Integrated Circuits Syst., vol. 26, no. 12, **(2007)** December, pp. 2222-2227.

[14] M. Khellah and M. Ghoneima, "A skewed repeater bus architecture for on-chip energy reduction in microprocessors", Proceeding of the IEEE International Conference on Computer Design, **(2005)** October, pp. 253-257.

[15] R. -B. Lin, "Inter-Wire Coupling Reduction Analysis of Bus-Invert Coding", J. IEEE Trans. Circuit and Systems, vol. 55, no. 7, **(2008)** August, pp. 1911-1920.

Authors

Young Chul Kim

Young Chul Kim received his PhD from Michigan State University, USA, the MS from the University of Detroit, USA, and BS in electronics engineering from Hanyang University, Korea. In 1993, he joined the Department of Electronics Engineering at Chonnam National University (CNU) where he is currently a professor. From 2000 to 2004, he was a director of IDEC at CNU. Since 2004, he has become the chief of the LG Innotek R&D center at CNU. His research interests are SoC design and development using IPs and low power design.

Youn Jin Lee

Youn Jin Lee received her BS in 2008 and MS in 2012, respectively, in electronics engineering from Chonnam National University, Korea, She has worked in the LG Innotek since 2012. Her current research interests are low power system design and embedded system design.