Coding for System-on-Chip Networks: A Unified Framework

Srinivasa R. Sridhara and Naresh R. Shanbhag
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
1308 W Main St., Urbana IL 61801
[sridhara,shanbhag]@uiuc.edu

ABSTRACT

In this paper, we present a coding framework derived from a communication-theoretic view of a DSM bus to jointly address power, delay, and reliability. In this framework, the data is first passed through a nonlinear source coder that reduces self and coupling transition activity and imposes a constraint on the peak coupling transitions on the bus. Next, a linear error control coder adds redundancy to enable error detection and correction. The framework is employed to efficiently combine existing codes and to derive novel codes that span a wide range of trade-offs between bus delay, codec latency, power, area, and reliability. Simulation results, for a 1-cm 32-bit bus in a 0.18-μm CMOS technology, show that 31% reduction in energy and 62% reduction in energy-delay product are achievable.

Categories and Subject Descriptors: B.4.3 [Input/output and data communications]: Interconnections (Subsystems)

General Terms: Design, Performance, Reliability

Keywords: Bus coding, crosstalk avoidance, low-power, low-swing, error-correcting codes

1. INTRODUCTION

With shrinking of feature sizes, increasing die sizes, scaling of supply voltage, increasing interconnect density, and faster clock rates, global system-on-chip buses are suffering from large propagation delay due to capacitive crosstalk [3, 10, 12, 13], high power consumption due to both parasitic and coupling capacitance [5, 10, 15] and increased susceptibility to errors due to DSM noise [4, 9]. Coding schemes have been proposed to alleviate these problems.

In on-chip buses, low-power codes (LPC) were first employed to reduce transition activity resulting in low-power buses [8, 11]. However, these schemes ignored coupling capacitances that are significant in DSM buses. Codes that reduce both self and coupling transitions were then proposed [5, 10, 15]. The increased coupling capacitance also leads to increase in the delay due to capacitive crosstalk. Crosstalk avoidance codes (CAC) that reduce the delay by forbidding certain transitions causing crosstalk were recently proposed [3, 10, 12]. Though crosstalk between adjacent wires is addressed by CAC, other forms of DSM noise such as power grid fluctuations, crosstalk from non-bus wires, electromagnetic interference makes buses susceptible to errors. Further, the use of low-swing signaling aggravates the reliability problem. Error control coding (ECC) was proposed in [4] as a way to achieve energy efficiency in I/O signaling in the presence of DSM noise. This idea was extended to on-chip buses in [1].

DAC'04, June 7–11, 2004, San Diego, California, USA.

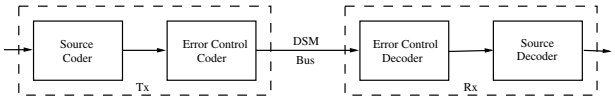

Figure 1: Generic coding system for an on-chip DSM bus.

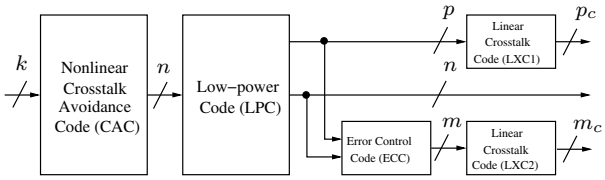

Figure 2: A unified coding framework.

Though, solutions not based on coding do exist for crosstalk prevention [13] and power reduction [10, 14], they are usually technology- and implementation-dependent. Coding provides an elegant alternative that is technology-independent. Further, coding can provide a common framework for jointly optimizing bus design for energy efficiency, speed, and reliability.

In this paper, we derive such a framework by viewing the DSM bus as a noisy communication channel. A generic coding system for DSM buses is shown in Figure 1. In [8], a source coder is employed to reduce self transition activity in buses. For closely coupled DSM buses, we can envisage the use of a source coder to not only reduce transition activity but also reduce average and/or peak coupling transitions. The reduction of average coupling transitions reduces power dissipation and reduction of peak coupling transitions reduces crosstalk delay. Further, we can employ a error control scheme to combat errors that arise due to DSM noise.

We can employ this generic system to design optimum codes that achieve the best possible performance for a given amount of redundancy (additional wires in our case) following either graph-theoretic [7, 12] or information-theoretic approaches [4, 10]. However, such approaches provide us with bounds on the achievable results and are not, in general, useful for designing practical schemes. The focus of this paper is to build a framework through which practical codes can be derived. Therefore, we construct a unified

framework based on the generic system but using the component codes that are known to be implementable. Such a framework is shown in Figure 2. We employ the framework to derive a wide variety of practical joint codes that represent a whole range of trade-offs between delay, power, area, and reliability.

2. BACKGROUND

In this section, we present an overview of the existing coding schemes and define our notation and terminology.

2.1 Low-Power Coding (LPC)

We refer to codes that reduce the average transition activity as low-power codes. A simple but effective scheme for general purpose data buses is bus-invert coding [11]. The effectiveness of bus-invert coding decreases with increase in the bus width. Therefore, for wide buses, the bus is partitioned into several sub-buses each with its own invert bit. In this paper, we denote bus-invert codes as BI(i), where i is the number of invert bits.

Codes that reduce coupling transition activity have been proposed for DSM buses with significant coupling capacitance [5, 10, 15]. However, these codes are complex and require significant overhead.

Note that bus-invert coding is nonlinear. It has been shown in [10] that linear codes do not reduce transition activity.

2.2 Crosstalk Avoidance Coding (CAC)

The delay of a line in the bus depends on the transitions on the line and lines adjacent to it. The worst-case delay of a line is $(1+4\lambda)\tau_l$, where τ_l is the delay of the line without any coupling and λ is the ratio of coupling capacitance to bulk capacitance [10]. The purpose of the crosstalk avoidance coding is to limit the worst-case delay to $(1 + 2\lambda)\tau_l$.

Crosstalk avoidance codes proposed in [12] reduce the worst-case delay by ensuring that a transition from one codeword to another codeword does not cause adjacent wires to transition in opposite directions. We refer to this condition as forbidden transition condition. Shielding the wires of a bus by inserting grounded wires between adjacent wires is the simplest way to satisfy this condition. A forbidden transition code (FTC) that requires less wires that shielding has been proposed in [12]. It can be shown that there is no linear code that satisfies the forbidden transition condition while requiring less wires than shielding.

The worst-case delay can also be reduced by avoiding bit patterns "010" and "101" from every codeword [3]. We refer to this condition as forbidden pattern condition. The simplest method to satisfy the forbidden pattern condition is to duplicate every data wire. Further, it can be shown that there is no linear forbidden pattern code (FPC) that satisfies the forbidden pattern condition while requiring less wires than duplication.

2.3 Error Control Coding (ECC)

Error control is possible if the Hamming distance between any two codewords in the codebook is greater than one [2]. If the minimum Hamming distance between any two code words is two, then all single errors appearing on the bus can be detected. If the minimum Hamming distance is three, then all single errors can be corrected. Error detection is simpler to implement than error correction but requires retransmission of the data when an error occurs.

In this paper, we focus on linear and systematic error correcting codes. In systematic codes, a few redundant bits are added to the input bits, which are unchanged, to generate the codeword. Hamming codes [2] are an example of linear systematic error correcting codes.

3. A UNIFIED CODING FRAMEWORK

The schemes LPC, CAC, and ECC can be combined into a system as shown in Figure 1 if the following conditions, derived based on the properties of the codes described in Section 2, are satisfied.

1. CAC needs to be the outermost code as, in general, it involves nonlinear and disruptive mapping from data to codeword.
2. LPC can follow CAC as long as LPC does not destroy the peak coupling transition constraint of CAC.
3. The additional information bits generated by LPC need to encoded through a linear CAC to ensure that they do not suffer from crosstalk delay.
4. ECC needs to be systematic to ensure that the reduction in transition activity and the peak coupling transition constraint are maintained.
5. The additional parity bits generated by ECC need to be encoded through a linear CAC to ensure that they do not suffer from crosstalk delay.

A framework satisfying the above conditions is shown in Figure 2. LXC1 and LXC2 are linear crosstalk avoidance codes based on either shielding or duplication. Nonlinear CACs can not be used because error correction has to be done prior to any other decoding at the receiver. In Figure 2, a k-bit input is coded using CAC to get an n-bit codeword. The n-bit codeword is encoded to reduce the average transitions through LPC resulting in p additional low-power information bits. ECC generates m parity bits for the $n+p$ code bits. The m parity bits and p low-power bits are further encoded for crosstalk avoidance to obtain m_c and p_c bits, respectively, that are sent over the bus along with n code bits.

In the remainder of this section, we develop a variety of codes based on the unified framework that allow for trade-off between delay, power, area, and reliability. The codes and their components are listed in Table 1. Some of the known codes are also listed for comparison. The new codes are shown in bold in the remainder of the paper.

3.1 Joint LPC and CAC

Combining LPC and CAC codes is a hard problem as both are nonlinear codes and, even when such a combination is possible, the resulting code is inefficient. For example, it not possible to combine bus-invert coding with FTC as inverting an FTC codeword destroys its crosstalk avoidance property. However, we show that FTC reduces the average coupling power dissipation as it avoids the high power-consuming opposing transitions on adjacent lines. Thus, FTC codes can independently be used for crosstalk avoidance and low-power.

3.2 Joint LPC and ECC

A joint low-power and error-correcting code can be obtained by adding parity information to the low-power coded data and low-power information bits. While it is possible to combine any low-power code with an error correcting code according to the framework, the total coding delay will be equal to the sum of the individual coding delays resulting a large delay or, in case of pipelined systems, latency. Here, we propose a way of reducing this delay for the important class of bus-invert based LPC and parity based ECC.

In bus-invert based LPC, the data bits are conditionally inverted based on a metric. Therefore, the inputs to ECC in Figure 2 are either the original data bits or their complement. In parity based ECC schemes, parity bits are generated through XORing the input bits. We use the following property of XOR operation to reduce the total delay of the joint code.

Category	Coding Scheme	CAC	LPC	ECC	LXC1	LXC2
LPC	BI(1)	–	BI(1)	–	–	–
	BI(4)	–	BI(4)	–	–	–
	BI(8)	–	BI(8)	–	–	–
CAC	Shielding	Shielding	–	–	–	–
	FTC	FTC	–	–	–	–
ECC	Hamming	–	–	Hamming	–	
	HammingX	–	–	Hamming	–	Half Shielding
LPC+ECC	**BIH**	–	BI(1)	Hamming	–	–
CAC+ECC	**FTC+HC**	FTC	–	Hamming	–	Shielding
	DAP	Duplication	–	Parity	–	–
	DAPX	Duplication	–	Parity	–	Duplication
All	**DAPBI**	Duplication	BI(1)	Parity	Duplication	–

Table 1: Codes based on the framework. New codes derived from the framework are shown in bold.

Property of XOR: If an odd (even) number of the inputs of an XOR gate are inverted, then the output is inverted (unchanged).

In the proposed scheme, we determine the parity bits of the ECC using the original data bits, instead of waiting for invert bits of the LPC to be computed. Once the invert bits are computed, the parity bits resulting from odd number of input bits are conditionally inverted using the invert bits. Thus, parity generation and invert bit computation can occur in parallel reducing the total delay to the maximum of the two. Though decoding still occurs serially, the decoding delay of the joint code is not significantly higher as bus-invert decoding involves just conditionally inverting the received bits using the invert bits. The joint code that results from such a combination of bus-invert code BI(1) and Hamming code is referred to as bus-invert Hamming (**BIH**) code as listed in Table 1.

3.3 Joint CAC and ECC

A joint crosstalk avoidance and error-correction code can be obtained by combining a crosstalk avoidance code with an error-correcting code. However, the coding overhead will be significant as the joint code is a concatenation of the two individual codes. Here, we propose a code that has significantly lower overhead.

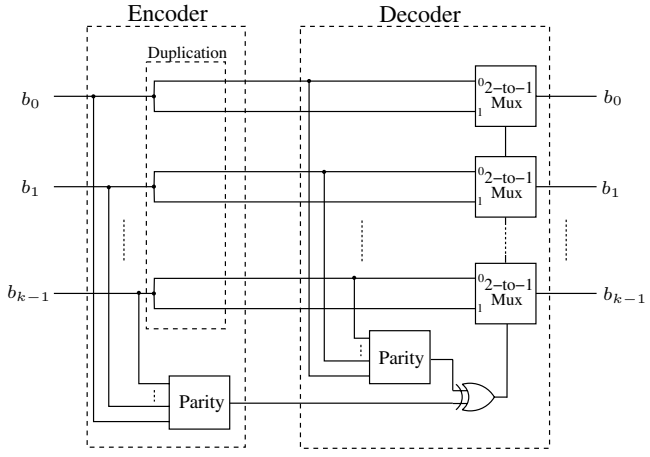

Figure 3: CAC+ECC: Duplicate-add-parity (DAP)

Consider the duplication scheme for avoiding crosstalk delay. This code has a Hamming distance of two as any two distinct codewords differ in at least two bits. We can increase the Hamming distance to three by appending a single parity bit. This code referred to as duplicate-add-parity (**DAP**) is shown in Figure 3.

To decode, we recreate the parity bit by using one set of the received data bits and compare that with received parity bit. If the two match, the set of bits used to recreate the parity bit is chosen as the output, else the other set is chosen as shown in Figure 3. Since a single error will at most affect one of the sets or the parity bit, it is correctable. Note that the **DAP** code is similar to boundary shift code (BSC) [7], which is based on forbidden transition condition, but has better performance as shown in Section 4.

3.4 Joint LPC, CAC, and ECC

We can combine all three component codes to arrive at a joint code that has low-power, crosstalk avoidance and error correction properties. However, only FPC based CAC and bus-invert based LPC can be used. Further, the overhead due to the combination will significant. Here, we consider the combination of the **DAP** code with bus-invert based LPC to create the joint code referred to as duplicate-add-parity bus-invert (**DAPBI**) code. As this code uses bus-invert based LPC and parity based ECC, we employ the technique described in Section 3.2 to reduce the encoder delay. Further, the invert bit is duplicated (LXC1) to ensure error-correction and crosstalk avoidance for the bit.

3.5 Encoder Delay in Systematic Codes

Based on the unified framework, we describe a technique to eliminate encoder delay of systematic codes at the cost of additional wires. In systematic codes, the data bits are transmitted without any modification and a few additional bits (parity bits in our case) are also transmitted. Therefore, the encoder delay only slows down the parity bits but still has an impact on the overall delay of the bus. We can eliminate the encoder delay from slowing down the bus by using linear crosstalk avoidance codes (LXC2) to speed-up the parity bits. Since, parity bits are few in number, we can employ a wide variety LXC2 codes such as half shielding, shielding, duplication, and triplication. In this paper, we consider **HammingX** code, which is a Hamming code that employs half shielding as LXC2, and **DAPX** code, which is a **DAP** code that employs duplication of its parity bit.

4. SIMULATION RESULTS

We consider a metal 4 bus in a 0.18-μm standard CMOS technology. The wires have a length of 1-cm, minimum width of 0.27-μm and minimum spacing of 0.36-μm. The worst-case bus delay with a 50$\times$ minimum driver is obtained using HSPICE. The worst-case delay is also obtained under forbidden transition and forbidden pattern conditions.

We assume that, due to DSM noise, voltage scaling below the nominal supply voltage $V_{dd} = 1.8$ V is not possible without lowering the reliability requirement. Further, we assume that noise is Gaussian distributed [1, 4]. Then, the probability of bit error is given by

$$\epsilon = Q\left(\frac{V_{dd}}{2\sigma_N}\right), \tag{1}$$

where $Q(\cdot)$ is the Gaussian probability of error function and σ_N^2 is variance of the additive noise. For a k-bit uncoded

Category	Coding Scheme	Area			Worst-case Delay (ps)					Average Energy (pJ)					Energy × Delay
		# of wires	Codec μm^2	Overhead(%)	Enc	Dec	Bus	Total	Speed-up	Codec	Bus V_{dd}	Bus $\hat{V}_{dd}$	Total	Savings (%)	
Uncoded	None	32	0	0	0	0	12006	12006	1.00	0	274.9	274.9	274.9	0	1.00
LPC	BI(1)	33	10885	8.7	1757	106	12006	13689	0.88	41.6	244.0	244.0	285.6	-3.9	1.20
	BI(4)	36	10084	17.8	1019	106	12006	13131	0.91	37.4	225.4	225.4	262.8	4.4	1.05
	BI(8)	40	13088	32.1	479	106	12006	12591	0.95	40.5	215.9	215.9	256.4	6.7	0.98
CAC	Shielding	63	0	98.6	0	0	5589	5589	2.15	0	274.9	274.9	274.9	0	0.47
	FTC	53	2790	68.2	70	91	5589	5750	2.09	4.5	234.7	234.7	239.2	13.0	0.42
ECC	Hamming	38	10385	24.3	533	1022	12006	13561	0.88	40.9	327.8	177.9	218.8	20.4	0.90
	HammingX	41	10385	33.9	0	1022	12006	13028	0.92	40.9	327.8	177.9	218.8	20.4	0.86
LPC+ECC	**BIH**	39	21439	33.1	1863	1128	12006	14997	0.80	83.1	296.9	161.4	244.5	11.0	1.11
CAC+ECC	**FTC+HC**	65	19356	114.8	760	1222	5589	7571	1.59	63.3	287.6	160.3	224.4	18.4	0.52
	BSC	65	8293	109.2	761	1107	5589	7457	1.61	23.2	330.8	170.2	193.4	29.6	0.44
	DAP	65	5627	107.8	687	943	5475	7105	1.69	19.6	330.8	170.2	189.8	31.0	0.41
	DAPX	66	5627	111.0	0	943	5475	6418	1.87	19.6	332.2	171.0	190.6	30.7	0.38
All	**DAPBI**	67	16568	119.7	1863	1049	5475	8387	1.43	61.4	293.3	151.1	212.5	22.7	0.54

Table 2: Comparison of codes for a 32-bit bus (0.18-μm CMOS, Metal 4, L=1 cm, W=0.27 μm, S=0.36 μm, 50$\times$ minimum driver).

bus, the probability of word error is $P_{unc}(\epsilon) = k\epsilon$. If the residual probability of word error with ECC is $P_{ecc}(\epsilon)$, then $P_{ecc}(\epsilon) < P_{unc}(\epsilon)$. For a given reliability requirement, we can reduce the supply voltage to

$$\hat{V}_{dd} = V_{dd}\frac{Q^{-1}(\hat{\epsilon})}{Q^{-1}(\epsilon)} \tag{2}$$

such that $P_{ecc}(\hat{\epsilon}) = P_{unc}(\epsilon)$. P_{ecc} for Hamming and **DAP** codes are given by

$$P_{ham}(\epsilon) = \binom{k+m}{2}\epsilon^2, \quad P_{dap} = \frac{3k(k+1)}{2}\epsilon^2. \tag{3}$$

In this paper, we assume a word-error rate requirement of 10^{-20}.

The average energy per bus transfer is computed assuming that the data is spatially and temporally uncorrelated and "0" and "1" are equally likely to appear. In case of ECC, the bus energy is also computed at the reduced supply voltage $\hat{V}_{dd}$.

The coding schemes are synthesized using the 0.18-μm CMOS library and optimized for speed. The overhead required in terms of area, delay, and energy for coding is obtained from synthesized gate level netlists. The codecs use nominal supply voltage in order to ensure reliable coding and decoding operations.

Table 2 compares the coding schemes derived from the framework to existing codes in terms of area, delay and energy dissipation for a 32-bit bus. We make the following observations based on the table.

ECC codes Hamming and **HammingX** provide significantly higher power savings than LPC codes based on bus-invert coding. Both sets of codes have similar delay and area overhead.

HammingX reduces the codec latency of Hamming code by 34% by eliminating the encoder delay at the cost of three additional wires. Compared to the uncoded bus, **HammingX** provides 20.4% power savings with 9% increase in delay and an area overhead of 33%.

The **BIH** code is able to reduce the bus energy by 41% through joint activity reduction and supply scaling but has significant codec energy in 0.18-μm technology resulting in an effective power savings of only 11%.

The crosstalk avoidance codes, shielding and FTC, provide greater than 2$\times$ speed-up. Though FTC was proposed as a CAC, we see that it also provides 13% energy savings as it avoids high energy-consuming opposing transitions. Though, FTC cannot be combined with bus-invert based schemes, it can independently be used as a joint LPC and CAC code.

CAC+ECC codes, by combining the properties of crosstalk delay avoidance, coupling activity reduction, and supply scaling, provide significant speed-up and energy savings with robustness to crosstalk and DSM noise. The **DAP** based codes outperform other codes because of the simplicity of the **DAP** codec. **DAPX** has 1.87$\times$ speed-up, 30.7% energy savings, 62% reduction in energy-delay product while requiring 111% area overhead.

The **DAPBI** code has the least bus energy and the lowest bus delay among all codes but has high codec overhead in the current technology to be competitive.

The codes trade-off delay and power dissipation in the bus with delay and power dissipation in the codec. This trade-off will be increasingly favorable in future technologies due to the increasing gap between gate delay and interconnect delay brought about by shrinking feature sizes and due to the longer bus lengths brought about by bigger die sizes. Therefore, coding schemes that result in low bus delay and energy such as **BIH**, **DAPBI** and **FTC+HC** will become more effective in the future.

5. ACKNOWLEDGMENTS

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

6. REFERENCES

[1] D. Bertozzi, L. Benini, and G. De Micheli, "Low power error resilient encoding for on-chip data buses," in *Proc. DATE*, 2002, pp. 102-109.

[2] R. E. Blahut, *Algebraic Codes for Data Transmission*. Cambridge, UK: Cambridge University Press, 2002.

[3] 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.

[4] R. Hegde and N. R. Shanbhag, "Toward achieving energy efficiency in the presence of deep submicron noise," *IEEE Trans. VLSI Syst.*, vol. 8, pp. 379–391, Aug. 2000.

[5] K. Kim, K. Baek, N. Shanbhag, C. Liu, and S. Kang, "Coupling-driven signal encoding scheme for low-power interface design," in *Proc. ICCAD*, pp. 318–321.

[6] 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.

[7] K. Patel and I. Markov, "Error-correction and crosstalk avoidance in DSM busses," in *Proc. SLIP*, 2003.

[8] S. Ramprasad, N. R. Shanbhag, and I. N. Hajj, "A coding framework for low-power address and data busses," *IEEE Trans. VLSI Syst.*, vol. 7, pp. 212–221, June 1999.

[9] K. L. Shepard and V. Narayanan, "Noise in deep submicron digital design," in *Proc. ICCAD*, 1996, pp. 147–151.

[10] P. P. Sotiriadis, *Interconnect Modeling and Optimization in Deep Sub-Micron Technologies*, Ph. D. dissertation, Massachusetts Institute of Technology, 2002.

[11] M. R. Stan and W. P. Burleson, "Bus-invert coding for low-power I/O ," *IEEE Trans. VLSI Syst.*, vol. 3, no. 1, pp. 49–58, March 1995.

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

[13] J. Yim and C. Kung, "Reducing cross-coupling among interconnect wires in deep-submicron datapath design," in *Proc. DAC*, 1999, pp. 485–490.

[14] H. Zhang, V. George, and J. Rabaey, "Low-swing on-chip signaling techniques: effectiveness and robustness," *IEEE Trans. on VLSI Syst.*, vol. 8, pp. 264–272, June 2000.

[15] Y. Zhang, J. Lach, K. Skadron, and M. R. Stan, "Odd/even bus invert with two-phase transfer for buses with coupling," in *Proc. ISLPED*, 2002, pp. 80–83.