Temporal Redundancy Based Encoding Technique for Peak Power and Delay Reduction of On-Chip Buses

K. Najeeb[1], Vishal Gupta[1], V. Kamakoti[1*] and Madhu Mutyam[2]

[1]Dept. of Computer Science & Engg., Indian Institute of Technology Madras, India, 600 036, kama@cs.iitm.ernet.in

[2]International Institute of Information Technology, Hyderabad, India, madhu_mutyam@iiit.net

* Corresponding author: V. Kamakoti

Address:

Department of Computer Science & Engg

Indian Institute of Technology Madras,

Chennai. India – 600 036.

Office : (091) 44-2257 4368

Fax : (091) 44-2257 4352

Email : kama@cs.iitm.ernet.in

Date of Receiving: **to be completed by the Editor**

Date of Acceptance: **to be completed by the Editor**

Temporal Redundancy Based Encoding Technique for Peak Power and Delay Reduction of On-Chip Buses

K. Najeeb, Vishal Gupta, V. Kamakoti and Madhu Mutyam

Abstract *—Power consumption and delay are two of the most important constraints in current-day on-chip bus design. The two major sources of dynamic power dissipation on a bus are the* ***self capacitance*** *and the* ***coupling capacitance****. As technology scales, the interconnect resistance increases due to shrinking wire-width. At the same time, spacing between the interconnects decreases resulting in an increase in the coupling capacitance. This, in turn, leads to stronger* ***crosstalk*** *effects between the interconnects. In Deep Sub-Micron technology the coupling capacitance exceeds the self capacitance, which, in turn, cause more power consumption and delay on the bus. Recently, the interest has also shifted to minimizing* ***peak power*** *dissipation. The reason being that higher peak power leads to an undesired increase in switching noise, metal electromigration problems and operation-induced variations due to non-uniform temperature on the die. Thus, minimizing power consumption and delay are the most important design objectives for on-chip buses.*

Several bus encoding schemes have been proposed in the literature for reducing crosstalk. Most of these encoding techniques use ***spatial redundancy*** *that requires additional transmission wires on the bus. In this paper, a new* ***temporal encoding*** *scheme is proposed, which uses* ***self-shielding memory-less*** *codes to completely eliminate worst-case crosstalk effects and hence significantly minimizes power consumption and delay of the bus. A major advantage of the proposed temporal redundancy based encoding scheme is the reduction in the number of wires of the on-chip bus. This reduction facilitates extra spacing between the bus wires, when compared with the normal bus, for a given area. This, in turn, leads to reduced crosstalk effects between the wires. The proposed encoding scheme is tested with the SPEC2000 CINT benchmarks. The experimental results, when compared to the transmission over a normal bus, show that on an average the proposed technique leads to a reduction in the peak-power consumption by 51% (28%), 51% (29%) and 52% (30%) in the data (address) bus for 90nm, 65nm and 45nm technologies, respectively. For a bus length of 10mm the proposed technique also achieves 17%, 31% and 37% reduction in the bus delay for 90nm, 65nm and 45nm technologies, respectively, when compared to what is incurred by the data transmission on a normal bus.*

Keywords **—Bus Encoding, Crosstalk, Coupling Capacitance, Delay, Interconnect, Peak-Power, Self-shielding codes, Temporal Redundancy**.

1 INTRODUCTION

Low-power and high-performance are the most desired features for all the components in Deep Sub-Micron (DSM) microprocessors and System-On-Chip (SoC) designs. Global communication in such systems is typically achieved with buses. Such buses can be **on-chip** (between different functional blocks), **off-chip** (between different IC's on a PCB) or at the **system level** (as a back-plane bus connecting several boards together). In a microprocessor, the interconnection networks consume over 50% of the total system power [1]. This is especially true for global on-chip buses, which suffer from higher power consumption and larger propagation delay with every new generation of the shrinking technology. In addition, peak power or instantaneous power minimization for **on-chip** buses is essential due to the following reasons: 1) to maintain supply voltage levels; 2) to avoid the metal electromigration problem and thereby increase the reliability; and 3) to decrease the environmental variation due to non-uniform temperature on the die. Because of these reasons, the reduction in power consumption in a bus has become a serious concern for low power VLSI design.

Two major sources of dynamic power dissipation on a bus are the **coupling** or **inter-wire capacitance** (C_I) and the **self** or **ground capacitance** (C_L) (Refer Figure 1). The *coupling capacitance* of an interconnect is the capacitance with respect to adjacent interconnects running in the same metal plane, while the *self capacitance* is the capacitance with respect to metal layers above and below the interconnect. As technology shrinks, the **coupling capacitance** exceeds the **self capacitance**, which, in turn, causes increased delay and power dissipation on the interconnect lines. To sum up, with shrinking of feature sizes, increasing die sizes, scaling of supply voltage, increased interconnect density and faster clock rates, the on-chip buses suffer from higher power consumption and large propagation delay due to capacitive crosstalk [2]-[7]. Since, both power consumed and delay incurred by a system bus, increase with increase in the coupling and the self capacitances, minimizing these capacitances is a major challenge in modern *DSM* designs.

The problem of reducing capacitive crosstalk effects on buses depends on the transitions (toggle in the value) on the bus lines. The bus lines are classified as *aggressor lines* and victim *lines* depending on the transition activity of the signal they carry. The effect of an aggressor on a victim depends on a number of factors, and not every aggressor will inject an appreciable amount of noise into a victim. In addition, the crosstalk effect may cause a switching (toggling) wire to inject a noise bump on an adjacent silent (non-toggling) wire leading to functional defects. The *Miller effect* suggests that the crosstalk capacitance varies with the switching behavior of a victim wire and its neighbors [8].

For example, consider three wires that run in parallel for a significant distance with minimum space. If the middle wire switches from low to high, while its neighbors are simultaneously switching from high to low, the effective capacitance of the middle wire becomes doubled compared to the case when the neighbors are quiet. On the other hand, if all the three wires are simultaneously switching in the same direction, the coupling capacitance of the middle wire becomes zero.

It has been shown that the delay and power for a long bus is strongly a function of the coupling capacitance between the wires [5]. Figure 2 shows the effect of coupling for a bus of width three and varying lengths, implemented in *65nm* technology. The bus model was simulated using the *Eldo* tool of Mentor Graphics. The *65nm* technology parameters for the above simulation were derived from [9] and are listed in the Table X.

In this paper, a new **temporal encoding** scheme is proposed, which uses **self-shielding memory-less** codes to completely eliminate Class 5 and Class 6 crosstalk effects [5] and hence significantly minimizes power consumption and delay of the bus.

2 ANALYTICAL MODELS FOR DELAY AND ENERGY CONSUMPTION

Analytical models for estimating crosstalk related delay and energy consumption in DSM buses have been proposed in [5], [10], [11]. These models are extensions of the standard Elmore delay model that account for arbitrary number of lines driven by independent sources and a distributed coupling component. The effect of transition patterns is also taken into account enabling the estimation of the delay on a sample by sample basis instead of considering a single worst case scenario. In this paper, the above mentioned analytical models are used to perform a qualitative analysis of the proposed approach and compare the same with the normal bus transmission. Figure 3 shows a model for the drivers and buses in DSM technologies. Consider a bus with n lines. Let $d_t = (d_t^1, d_t^2,...d_t^n)$ denote the t^{th} n-bit data transmitted on the bus. The delay for transmitting the $(t+1)^{th}$ data on the bus is given by the following formula [5]. Defining $T_k(d_t,d_{t+1})$, $1 \le k \le n$, as follows:

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

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

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

Similarly, the total energy, $E(d_t,d_{t+1})$, consumed during the transmission of d_{t+1} is given by [11], [12].

$$E(d_t, d_{t+1}) = \sum_{k=1}^{n} E_k(d_t, d_{t+1}), \quad where$$

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

For example, given n = 3, d_t = 010 and d_{t+1} = 101, then, $T(d_t,d_{t+1}) = C_L.R_T(1+4\lambda)$ and $E(d_t,d_{t+1}) = C_L(2+4\lambda)$. On the other hand, if d_t = 000 and d_{t+1} = 111, then, $T(d_t,d_{t+1}) = C_L.R_T$ and $E(d_t,d_{t+1}) = 3C_L$. During a transmission, a wire *k* in the bus is in one of the following states, namely, a *rising* (0 → 1) state, a *falling* (1→0) state or a *constant* (0→0, 1→1) state, depending on the values transmitted on the wire *k* during the previous and current transmissions. The rising, falling and constant states are denoted by (↑), (↓) and (-), respectively. From the above formula, it is seen that, the delay and energy consumed by a wire *k* during a transmission, depends not only on the state of the wire *k* but also on the states of the lines adjacent to wire *k*. Table I classifies the crosstalk delay effect on a wire *k* into *six* classes, depending upon the state of wire *k* (the middle wire) and its adjacent wires [5], [12]. 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 I), followed by Class *5*. Exactly similar classification can be made for energy consumption on a wire *k*. Here too, the Class *6* crosstalk on a wire results in maximum energy consumption, followed by Class *5*. Table II shows the number of coupling transitions for different transition patterns. Here again, the Class *5* and Class *6* transition patterns yield maximum number of coupling transitions. This suggests that elimination of Class *5* and Class *6* crosstalk patterns shall reduce both the delay and energy consumption of the bus. This is precisely what is attempted by the technique proposed in this paper.

3 PREVIOUS WORK

Many encoding methods have been presented to reduce the power dissipation on buses. The Bus invert method proposed in [13] achieves a 50% reduction in the maximum number (peak value) of self transitions and coupling transitions when compared to a normal transmission over an unencoded bus. However, this method fails to control the *average* coupling transition activity leading to less average power reduction. The failure is due to the ignoring of the coupling capacitance between interconnects that result in a significant power penalty on *DSM* global buses. A bus encoding technique to simultaneously minimize power consumption and eliminate crosstalk delay is proposed in [14]. This technique requires large number of *extra* interconnects. For example, it encodes a 32-bit data to a 55-bit data (extra of 23 bits). This leads to a heavy overhead in terms of routing congestion, Encoder-Decoder (CODEC) complexity and wire area penalty. In [15], a bus encoding technique is proposed to minimize coupling transitions by considering minimization of power consumption as the main objective. This technique does not eliminate classes 4, 5 and 6 crosstalk patterns. Hence, it is not much advantageous from the delay perspective.

Another bus encoding technique to minimize both energy and delay is proposed in [16], which can eliminate only the crosstalk classes 4 and 6. Here, the worst case delay is still due to the class 5 transitions, which is high. Moreover, like [14], this technique requires large number of extra interconnects. For example, like [14] it encodes a 32-bit data to a 55-bit one. The Odd/Even bus invert technique proposed in [17] is designed to target minimization of coupling energy and hence is not very effective from the delay perspective.

The simplest and most effective technique for preventing crosstalk delay is *shielding*. This involves providing a power supply wire, either ground or VDD, between every two adjacent wires on the bus, as shown in Figure 4. These constant-voltage wires act as electro-magnetic shields, and prevent activity on one signal wire from significantly coupling over to another signal wire. Though this naive scheme is effective to remove the crosstalk delay, the routing congestion and wire area penalty are significant due to the almost doubled number of wires.

Victor et al. proposed the self-shielding bus encoding technique [6] to prevent crosstalk delay. Two versions of the coding technique were proposed, namely, the *memory-based self-shielding code* and the *memory-less self-shielding code*. In the memory-based self-shielding technique, the encoder and the decoder require the previous transmitted data on the bus for encoding/decoding the current data. Thus, the previous transmitted data have to be *remembered* by storing the same in a memory. The encoder and decoder of a memory-less self-shielding code depend only on the current data for encoding/decoding the same. The technique in [6] was proved theoretically far better than just placing shielding wire between every adjacent wire. A thorough theoretical study of self-shielding codes is presented in [6]. Theoretical bounds on performance in terms of required channel width versus data bits are derived. For example, it is shown in [6] that, for a 32-bit data D, any memory-based self-shielding shall require *at least* 40-bits (extra of 8) to encode D, such that, the resulting transmission avoids class 5 and class 6 crosstalk patterns. It is also shown that, a memory-less shielding shall require *at least* 46 (extra of 14) bits to achieve the same.

Tiehan et al. [18] proposed a dictionary based encoding scheme for data buses. This approach uses an adaptive dictionary encoding scheme for minimization of power. The power reduction using this technique crucially depends upon the patterns in the transmitted data. In [19], an address bus encoding scheme is proposed. This scheme crucially depends upon the high degree of temporal correlation in the data transmitted over the address bus to reduce the power dissipation. In the absence of such correlation between consecutively transmitted data, this scheme partitions the data to be transmitted into two parts and transmit the same in two consecutive cycles, hence resulting in temporal redundancy.

In [20], a bus encoding technique using a variant of binary Fibonacci representation of integers is proposed and a recursive procedure to generate crosstalk delay free binary Fibonacci codewords is given. The generated Fibonacci codewords are similar to that of the memory-less self-shielding codes given in [6]. It has been shown in [6], [20] that *m*-bit crosstalk delay free binary Fibonacci codewords can be used to encode $\log_2(F_{m+2})$ bits, where F_{m+2} is the $(m+2)^{th}$ Fibonacci number. So, a 32-bit bus can be encoded using 46-bit crosstalk delay free binary Fibonacci codewords.

In [21], a bus encoding is proposed to obtain 10% energy reduction alone with delay reduction of nearly 50%. Bus encoding techniques to reduce the worst-case crosstalk delay by nearly 50% are proposed in [2], [14]. However, these techniques require large number of extra wires. For example, the techniques proposed in [2], [14] and [21] encode a 32-bit data into 52, 55 and 48 bits, respectively.

All the bus encoding schemes mentioned above employ *spatial redundancy* (that is, they use extra bus wires for encoding) and can be modeled as shown in Figure 5. In other words, $m > n$ in Figure 5 for spatial redundancy based techniques.

Recently, *temporal redundancy* (redundancy in time) based bus encoding techniques to minimize propagation delay and/or energy consumption are reported in the literature. In a temporal redundancy based bus encoding technique, a data item is encoded in such a way that the encoded data is transmitted in two or more successive cycles.

In [22], a bus encoding technique based on both spatial and temporal redundancy is proposed for on-chip delay and energy minimization. In this technique, data to be transmitted on the bus is classified into different crosstalk classes [5], [11]. The *temporal encoding* is applied to only those data that belong to certain crosstalk classes.

A bus encoding technique, based on temporal redundancy alone is proposed in [4], wherein each *32*-bit data to be transmitted is encoded as two 24-bit data and transmitted in two successive cycles. The coding technique does not directly eliminate all the class 5 and class 6 crosstalk patterns. The encoder circuit detects a class 5 or class 6 pattern in advance and transmits a *dummy* data on the bus to avoid the same. In contrast to the above-mentioned approaches, this paper proposes a temporal redundancy based coding technique which fully eliminates both class 5 and class 6 crosstalk transition patterns. The technique is a *memory-less self-shielding* one.

4 BUS ENCODING USING TEMPORAL REDUNDANCY

The main idea behind the proposed *temporal redundancy* based encoding scheme is to encode the original *n*-bit data packet which is to be transmitted, into two m-bit data packets, where, ($m < n$). The two encoded data packets are transmitted over *two consecutive cycles* on the bus.

Though it may seem that this technique will incur a large delay overhead, as the number of transmissions are doubled, it has been proved that this technique can reduce the bus delay significantly when compared to normal transmissions on the bus. The central idea is explained below.

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 [5]. During a normal (uncoded) transmission on a bus, any of the six classes of transition patterns, as shown in Table I, can appear on the bus. Thus, the clock period should be larger than the maximum possible delays of all such classes. From Table I, we can see that the Class 6 patterns induce the maximum delay, which is equal to CL.RT(1 + 4λ) . Thus, for a normal uncoded transmission over a bus

$$T_C \geq C_L \times R_T (\mathbf{1} + \mathbf{4}\lambda) \quad (1)$$

As mentioned earlier, for a temporal redundancy based encoding scheme, the size of the bus reduces from n to m, that is, m < n. By assuming that the new m-bit wide bus occupies the same interconnect area as that of the original n-bit wide bus, the spacing between the wires in the new bus can be increased. The following formula (Equation (2)) illustrates the increased spacing between the wires got by converting an *n*-bit bus to an *m*-bit bus.

$$nW + (n - \mathbf{1})S = mW + (m - \mathbf{1})S' \quad (2)$$

where, *W* is the width of a wire, *S* is the original spacing in the *n*-bit bus, and *S'* is the new spacing between adjacent wires in the m-bit bus. Increasing the space between adjacent wires, decreases the C_I and increases the C_L [5], which in turn decreases the value of λ.

The encoding technique proposed in this paper ensures that Class 5 and Class 6 crosstalk patterns are eliminated during the transmission over the bus. Therefore, as seen from Table I, the worst case delay pattern that can occur on the bus is due to the Class 4 transition pattern. In addition, as two encoded data are transmitted (temporal redundancy) for each of the original data, the proposed scheme shall result in a better performance compared to the normal transmission if and only if

$$C_L \times R_T(1 + 4\lambda) > 2 \times C_L' \times R_T (1 + 2\lambda') + \delta \tag{3}$$

where, *capacitance factor* $\lambda' = \frac{C_I'}{C_L'}$, C_L' and C_I' are the values of the ground and interwire capacitance, respectively, for the new *m*-bit bus and δ is the delay associated with the encoder and the decoder. The *self capacitance* C_L and *coupling capacitance* C_I are calculated using the equations presented in [23] (reproduced here as Equations (4) and (5)).

$$C_L = \varepsilon_{ox}\left(\frac{W}{H} + 2.217\left(\frac{S}{S+0.702H}\right)^{3.193} + 1.171\left(\frac{S}{S+1.51H}\right)^{0.7642} \times \left(\frac{T}{T+4.532H}\right)^{0.1204}\right) \tag{4}$$

$$C_I = \varepsilon_{ox}\left(1.14\frac{T}{S}\left(\frac{H}{H+2.059S}\right)^{0.094} + 0.743\left(\frac{W}{W+1.59S}\right)^{1.144} + 1.158\left(\frac{W}{W+1.874S}\right)^{0.16} \times \left(\frac{H}{H+0.98S}\right)^{1.179}\right) \tag{5}$$

where W is the wire width, H is the dielectric thickness, S is the interwire spacing, T is the wire thickness and ε is the permittivity. In the subsequent sections of this paper it is shown that the above inequality (Equation (3)) is indeed true for DSM buses.

5 CROSSTALK PREVENTING CODE (CPC) BASED TEMPORAL ENCODING

The encoding scheme presented in this paper is denoted as an (*n*, *p*, *q*, *r*)-encoding and is described as follows: Consider an *n*-bit data D = (x_1, x_2,...,x_n) which is to be transmitted on the bus. This n-bit data is partitioned in order into two data sets, namely, $D_1 = (x_1, x_2, \ldots, x_{n/2})$ and $D_2 = (x_{(n/2)+1}, \ldots, x_n)$ of n/2 bits each. The n/2 bits of D_1 are encoded into an m-bit code and transmitted over an m-bit bus, as described in the Algorithm 1 that follows. The values of p, q and r in the (n, p, q, r)-encoding are also defined in the Algorithm 1. As described in the step 4 of Algorithm 1, the value of m depends on the values of n, p, q and r.

Algorithm 1 : Encode

---

begin

1) Partition D_1 in order into $\lfloor n/2p \rfloor$ number of p-bit data blocks $d_1, d_2, \ldots, d_{\lfloor n/2p \rfloor}$. Here, p need not be a factor of n. So, the remaining $(n/2 - \lfloor n/2p \rfloor * p)$ bits constitute the $\lceil n/2p \rceil^{th}$ data block;
2) Encode the p-bits in each block d_i, $1 \le i \le \lfloor n/2p \rfloor$ into a q-bit code;
3) Encode the g $(=(n/2 - \lfloor n/2p \rfloor * p))$ bits in the block $d_{\lceil n/2p \rceil}$ into a r-bit code;
4) All these $\lfloor n/2p \rfloor$ number of q-bit encoded data blocks and one r-bit encoded block are transmitted over the bus with a *shielding* wire placed between every two adjacent data blocks as shown in the Figure 6. The *shielding* wire carries a logical zero or a logical one signal permanently, and therefore, never toggles. There are at most $\lfloor n/2p \rfloor$ shielding wires as shown in Figure 6. Hence, the total number of wires in the new bus is

$$m = \left\lfloor \frac{n}{2p} \right\rfloor (q + 1) + r \tag{6}$$

end of Algorithm 1

Similar encoding is done for the n/2 bits in D_2 and these encoded bits are transmitted in the consecutive cycles following the transmission of the encoded D_1 bits. The *decoding* procedure is very straightforward as seen in the Figure 6, wherein, each of the *q*-bit data blocks are decoded into *p*-bit data blocks. The last *r*-bit data block is decoded into a *g*-bit data block, where, $g = (n/2 - \lfloor n/2p \rfloor * p)$.

From Table I, it is clear that, for a Class 5 or Class 6 crosstalk to occur; two adjacent lines in the bus should have *opposite* transitions. The presence of the non-toggling shielding wire between adjacent blocks ensures that no *Class 5 or Class 6 crosstalk can occur in any transmission that involves wires of two different blocks*. The above implies the following design objective.

Objective 1: *Design an encoding scheme that eliminates Class 5 and Class 6 crosstalks within the q-bit blocks and also within the r-bit block of the bus shown in the Figure 6.*

The CPC encoding scheme presented below achieves the above-mentioned objective.

Consider the set F_k, $k \geq 1$ of k-bit binary sequences generated by Algorithm 2. The first four sets F_1 to F_4 are shown in Table III. Consider the set F_3 in Table III. It is interesting to note that for any two arbitrary 3-bit strings $s, t \in F_3$, transmitting *s* on a 3-bit bus followed by *t* on the same shall not lead to any Class 5 or Class 6 transition patterns (i.e. two opposite transitions on adjacent lines). Similar is the case with the sequences in the sets F_2 and F_4. The following theorems prove some properties of the sets F_k, $k \geq 1$.

Algorithm 2: CPC Construction

---

begin

  $F_1 \Leftarrow \{0,1\}$

  for all $m \geq 1$

if m is odd then

$F_{m+1} \Leftarrow \{0x \mid \forall x \in F_m\} \cup \{11y \mid \forall 1y \in F_m\}$

else

$F_{m+1} \Leftarrow \{1x \mid \forall x \in F_m\} \cup \{00y \mid \forall 0y \in F_m\}$

end if

end of Algorithm 2

Theorem 1:

Let $|F_k|$ denote the number of elements in the set F_k. Then, $|F_k| = |F_{k-1}| + |F_{k-2}|$

Proof: Without loss of generality, let *k* be even. Then, Algorithm 2 implies that F_k is constructed by

1) taking all elements of F_{k-1} and prefix the same with a 0; **and**,
2) taking all elements of F_{k-1} starting with a 1 and prefix the same with 1. Since *k* is even, *k-1* is odd. Again from Algorithm 2, we see that all the strings that start with a 1 in F_{k-1} are those got by prefixing a 1 to all the strings in F_{k-2}.

The above two observations imply that $|F_k| = |F_{k-1}| + |F_{k-2}|$, for all even values of *k*. A similar proof holds good for odd values of *k*.

Theorem 2: For any two arbitrary *m*-bit strings *s*, $t \in F_m$, transmitting *s* on an *m*-bit bus followed by *t* on the same shall not cause any opposite transitions on adjacent lines.

Proof: The proof is by induction on *m*. The theorem is true for F_2. Let the theorem be true for all $i \leq k$. Consider F_{k+1}. Since F_{k+1} is constructed by prefixing *0's* and *1's* to the strings in F_k, by induction hypothesis, it is clear that, if at all opposite transitions should happen on adjacent wires during two consecutive transmissions of arbitrary strings in F_{k+1} on a (k+1)-bit bus, then, it must happen on the leftmost (most significant bit) two lines. This implies that there should exist at least two strings in F_{k+1} with the two most significant bits as *01* and *10*, respectively, so that if they are transmitted one after another, they cause opposite transitions on adjacent lines. But, this is never the case, as from Algorithm 2, it is clear that, F_{k+1} shall have strings starting with either 10 or 01, but *not both*. Hence, the theorem.

5.1 Design of the Encoder and the Decoder

Theorem 2 implies that, *if all the k-bit sequences transmitted over a k-bit bus belong to* F_k, *then, there would be no Class 5 or Class 6 transition patterns on the bus*. Theorem 2 suggests a solution to the *p*-to-*q* bit and *g*-to-*r* bit encoding as required by steps *2* and *3* of Algorithm 1 that also meets the objective 1 stated earlier.

The idea behind the design of the *p*-to-*q* encoding is to map each of the 2^p different *p*-bit inputs onto *unique* *q*-bit sequences in the set F_q. Since the mapping has to be one-to-one (injective), it is straightforward to see that *q* is the *smallest integer* satisfying $|F_q| \geq 2^p$. Similar technique is employed for the *g*-to-*r* encoding. As seen above, *r* is the *smallest integer* satisfying $|F_r| \geq 2^g$. This and Theorem 2 imply that class 5 and class 6 are avoided in all transitions over the *m*-bit bus of Figure 6. The Table IV gives the cardinality of Fq for different values of *q*, calculated using Theorem 1.

Given the value of *n*, the choice of *p* is crucial. From the discussion above and Algorithm 1, it is obvious that the values of *q*, *g* and *m* depend on *p*. The value of *r* depends on *g* and hence, in turn depends on *p*. The following two important factors have to be considered for choosing *p*.

1) The p should be chosen such that, there is a significant reduction in m, the number of wires in the new bus. The reason is, from Equation (3) it is seen that, as the bus width decreases, the spacing between the lines increases and value of λ decreases, which shall result in better performance of the bus.
2) The p should be chosen such that, the encoder and decoder are not too complex to implement.

The construction of the encoder and decoder circuits can be understood through the following example. Consider transmission of a *32*-bit data (n = 32). The encoding scheme partitions the input into two *16*-bit data, encodes them, and transmits the encoded version one after another in two consecutive cycles. Given n = 32 and hence n/2 = 16, the Table V gives for each choice of p, the values of q, r and m (refer Figure 6).

From Table V we see that the minimum bus-width is 25, which is got by selecting p as 5, 7 or 8. In all these cases, the p-to-q en/decoder becomes complex in terms of delay and power consumption. Tables VI and VII compare different types of en/decoder circuits in terms of gate count, number of transmission lines and depth of the circuit. Even though some of the en/decoder circuits consume less number of bus transmission lines than the 3 to 4 en/decoder circuit, the corresponding circuit depth and the number of gates needed to realize them is significantly larger than what is incurred by a 3 to 4 en/decoder circuit. Therefore other types of en/decoder circuits consume more power and higher logic delay than the 3 to 4 en/decoder. This justifies the use of a 3 to 4 encoder and the corresponding decoder for transmission. Hence, p is chosen as 3, which implies that q=4, g=1, r = 1 and m = 26. The Table VIII shows the p-to-q (*3*-to-*4*) encoding using the sequences in F_4. The g-to-r (1-to-1) encoding is straightforward, wherein, the single bit is transmitted as it is. The Boolean equations for the *3*-

to-*4* encoder and decoder are given below. The Figure 7 shows the gate-level representation of the above encoder and decoder functions.

Encoder			**Decoder**		
O_1	=	a+~b.c	a	=	$O_3.O_1$
O_2	=	a.b	b	=	$O_4+\sim O_3.O_1$
O_3	=	a+b	c	=	$O_2+\sim O_1.O_3$
O_4	=	c.(a+b)			

6 EXPERIMENTAL VALIDATION AND RESULTS

This section presents the delay and peak-power analysis of a bus during both normal transmission and the proposed encoded transmission and compares the same. For experimental study, the normal bus is assumed to be *32*-bit wide, and the corresponding encoded bus is *26*-bit wide, as described in the previous section.

6.1 Synthesis of the Encoder-Decoder

A Verilog description of the encoder and decoder circuit shown in the Figure 7 was taken through the RTL2GDSII cycle of the Magma Blast Chip Version *4.1.57* design flow using the *CL013LV* (130nm technology) cell library of the TSMC. Table IX shows the delay, energy and area overheads respectively of the encoder and decoder, calculated using the above design process.

6.2 Peak-Power Analysis

The effect of the proposed encoding scheme on the peak-power consumption of on-chip buses was studied by simulating the SPEC2000 CINT [24] benchmark suite on the Simplescalar 3.0 [25] architectural simulator. The performance of different on-chip buses between the processor datapath and L1 I-cache/D-cache were studied. For each benchmark, the first 100 million instructions were fast-forwarded and simulation study was done on the next 100 million instructions. The following equation was used to determine the power consumption (*P*) during one data transmission.

$$P = \left(N_S \times C_L + N_C \times C_I\right) \times f \times V^2 + P_{enc} + P_{dec} \tag{7}$$

where, $N_{s,}$ is the total number of switching transitions, N_c is the total number of coupling transitions, f is the frequency, V is the value of the high voltage level, P_{enc} and P_{dec} are the power consumed by the encoder and decoder, respectively. The coupling transitions per every bus-line value were estimated as given in Table II. The switching transitions per every bus-line were computed. The values of N_c and N_s were computed by summing up the coupling and switching transitions over all the bus-lines respectively. The values of f and V are given in Table X. The values of P_{enc} and P_{dec} are taken from Table IX. Thus, the power for each transmission on the bus is computed and the maximum of this value computed over all transmissions is the Peak-Power. Figure 8 shows the percentage peak-power reduction due to the proposed encoding technique in comparison to the normal transmission, for both address and data buses in *45nm*, *65nm* and *90nm* technologies, respectively. The results show that on an average the proposed technique leads to a reduction in the peak-power consumption by 51% (28%), 51% (29%) and 52% (30%) in the data (address) bus for *90nm*, *65nm* and *45nm* technologies, respectively. Simulation results show that the peak power reduction is higher for the data bus than the address bus because the data transmitted on the latter are temporally correlated in nature when compared to the data transmitted on the former. In other words, the Hamming distance between any two consecutive transmissions in an address bus is less when compared to any two consecutive transmissions in a data bus.

6.3 Delay Analysis

The delay analysis was based on the technology parameters taken from the ITRS 2001 [9] for *90nm*, *65nm* and *45nm* technologies. The parameters are shown in Table X. In addition, the Predictive Technology Model [26] was used to calculate the resistance (R_T), self-capacitance (C_L), coupling-capacitance (C_I) and wire spacing (S) values for different technologies. The value of λ was calculated using the capacitance values. These computed values for a *32*-bit uncoded bus and the corresponding *26*-bit encoded bus are shown in Tables XI and XII, respectively. The wire spacing parameter (S') for the encoded bus shown in Table XII was calculated using Equation (8).

$$\mathbf{32}W + \mathbf{31}S = \mathbf{26}W + \mathbf{25}S' \tag{8}$$

Equation (8) was derived from the Equation (2) by substituting $n = 32$ and m = *26*.

To measure the performance of the proposed encoding, a *SPICE* model of a *3*-bit wide bus as shown in Figure 3 was developed. The model is a distributed R-C one. The lines were assumed to be capacitively coupled. The model was simulated using the Mentor Graphics *Eldo 6.2.2* Spice simulator for three different technologies, namely, *90nm*, *65nm* and *45nm* and for both the normal and the encoded bus models.

The technology parameters were taken from the Tables X, XI and XII. The delays due to the encoder and the decoder were taken from the Table IX to compute the total transmission delay of the encoded bus. The total percentage of delay reduction due to the proposed encoded transmission in comparison to the normal transmission was computed as given in Equation (9).

$$Delay\ \mathrm{Re}duction = \frac{Time_{uncoded} - Time_{coded}}{T_{uncoded}} \times \mathbf{100} \qquad (9)$$

The results obtained for *90nm*, *65nm* and *45nm* technologies are shown in Figure 9. The graph indicates that for a bus length of *10 mm* the proposed technique also achieves 17%, 31% and 37% reduction in the bus delay for *90nm*, *65nm* and *45nm* technologies, respectively, when compared to what is incurred by the data transmission on a normal bus. Table XIII compares the proposed approach with the other existing techniques in terms of interconnect spacing, redundancy type, number of bus lines, shielding protection, power and delay requirements. The comparison results show that the proposed approach is better than the existing ones reported in literature, in terms of all the parameters mentioned above. To sum up, the salient features of the proposed technique are as follows:

1) Basic en/decoder is a simple combinational circuit realized using 6 logic gates with a circuit depth of 3;
2) The technique widens the distance between bus lines without area penalty, resulting in less parasitic capacitance between bus lines;
3) The technique completely eliminates class 5 and class 6 without using extra wires for shielding;

4) Total number of bus lines required by the technique is less than the original bus width; and,
5) The technique is independent of the executing application that causes the bus transmission.

7 CONCLUSIONS

In this paper, a new temporal encoding scheme that uses self-shielding memory-less codes to minimize crosstalk was proposed. The paper presented a detailed comparison of the proposed scheme with existing encoding schemes. The proposed encoding scheme was tested with the SPEC2000 CINT benchmarks to study the peak power consumption. On an average, the proposed technique reduced the peak power consumption by 51% (28%), 51% (29%) and 52% (30%) in the data (address) bus for *90nm*, *65nm* and *45nm* Technologies, respectively. The experimental results got by a SPICE simulation of the DSM bus model show that the proposed technique achieves 17%, 31% and 37% reduction in the bus (*10mm*) delay for *90nm*, *65nm* and *45nm* technologies, respectively when compared to data transmission without any encoding. To the best of our knowledge, this is the only temporal encoding scheme reported in literature that eliminates Class 5 and Class 6 crosstalk patterns completely.

ACKNOWLEDGMENTS

This work was partially supported under the Fast-track scheme for young scientists supported by the Department of Science and Technology (DST), Government of India. We thank the anonymous referees for their valuable comments that have greatly enhanced the quality of this paper.

REFERENCES

[1] N. Magen, A. Kolodny, U. Weiser, and N. Shamir, “Interconnect-power dissipation in a microprocessor,” Proceedings of 6th International Workshop on System Level Interconnect Prediction, (**2004**), pp. 7-13.

[2] C. Duan and S. P. Khatri, “Exploiting crosstalk to speed up on-chip buses,” Proceedings of the Conference on Design Automation and Test in Europe, (**2004**), pp. 778-783.

[3] C. Duan, A. Tirumala, and S. P. Khatri, “Analysis and avoidance of crosstalk in on-chip buses,” Proceedings of the Hot Interconnects 9, IEEE CS Press, (**2001**), pp. 133-138.

[4] K. Najeeb, Vishal Gupta, V. Kamakoti, and M. Mutyam, “Delay and Peak Power Minimization for On-Chip buses Using Temporal Redundancy,” Proceedings of the ACM Great Lakes Symposium on VLSI, PA, USA, (**2006**), pp. 119-122.

[5] P. Sotiriadis and A. Chandrakasan, “Reducing bus delay in sub-micron technology using coding,” Proceedings of the Asia and South Pacific Design Automation Conference, (**2001**), pp. 109-114.

[6] B. Victor and K. Keutzer, “Bus encoding to prevent crosstalk delay,” Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, (**2001**), pp. 57-63.

[7] J. Yim and C. Kung, “Reducing cross-coupling among interconnect wires in deep-submicron datapath design,” Proceedings of the 36th ACM/IEEE Conference on Design Automation, (**1999**), pp. 485-490.

[8] Sachin Sapatnekar, “Timing,” Kluwer Publishers, (**2004**).

[9] International Technology Roadmap for Semiconductors, “http://public.itrs.net”.

[10] P. Sotiriadis and A. Chandrakasan, “Bus energy minimization by transition pattern coding in deep sub-micron technologies,” Proceedings of the IEEE/ACM International onference on Computer Aided Design, (**2000**), pp. 322-328.

[11] P. Sotiriadis and A. Chandrakasan, “A bus energy model for deep sub-micron technology,” IEEE Transactions on VLSI Systems, (**2002**), vol. 10(3), pp. 341-350.

[12] P. Sotiriadis and A. Chandrakasan, “Low power bus encoding techniques considering inter-wire capacitances” Proceedings of the IEEE International Conference Custom Integrated Circuits, (**2000**), pp. 507-510.

[13] M. Stan and W. Burleson, “Bus invert coding for low power i/o,” IEEE Transactions on VLSI system (**1995**), vol. 3, Issue 1, pp. 49-58

[14] C. G Lyuh and T Kim, “Low power bus encoding with crosstalk delay elimination,” Proceedings of the Annual IEEE International ASIC/SOC Conference, (**2002**), pp. 389-393.

[15] K. Kim, K. Baek, and N. Shanbag, “Coupling-driven signal encoding scheme for low power interface design,” Proceedings of the International Conference on Computer Aided Design, (**2000**), pp. 318-321.

[16] Z. Khan, T. Arslan, and A.T. Erdogan, “A novel bus encoding scheme from energy and crosstalk efficiency perspective for AMBA based generic SoC systems.” Proceedings of the International Conference on VLSI Design, (**2005**), pp. 751-756.

[17] Y. Zhang, J. Lach, K. Skandron, and M.R. Stan, “Odd/even bus invert with two-phase transfer for buses with coupling,” Proceedings of the International Symposium on Low Power Electronics and Design, (**2002**), pp. 80-83.

[18] Tiehan Lv, Jorg Henkel, Haris Lekatsas, and Wayne Wolf, “A dictionary-based en/decoding scheme for low-power data buses,” IEEE Transactions on Very Large Scale Integration Systems, (**2003**), vol. 11, No 5, pp. 943-951.

[19] Tiehan Lv, Jorg Henkel, Haris Lekatsas, and Wayne Wolf, “Enhancing signal integrity through a low-overhead encoding scheme on address buses,” Proceedings of the conference on Design, Automation and Test in Europe, (**2003**), p. 10542.

[20] M. Mutyam, “Preventing crosstalk delay using Fibonacci representation,” Proceedings of the 17th International Conference on VLSI Design, (**2004**), pp. 685-688.

[21] S.R. Sridhara, A. Ahmed, and N.R. Shanbhag, “Area and energy-efficient crosstalk avoidance codes for on-chip buses,” Proceedings of the IEEE International Conference on Computer Design, (**2004**), pp. 12-17.

[22] M. Mutyam, M. Eze, N. Vijayakrishnan, and Y. Xie, “Delay and energy efficient data transmission for on-chip buses,” Proceedings of the IEEE Computer Society Annual Symposium on Emerging VLSI Technologies and Architectures, (**2006**), pp. 355-360.

[23] S.C Wong, G.Y Lee, and D.J Ma, “Modeling of Interconnect Capacitance, Delay, and Crosstalk in VLSI,” IEEE Transactions on Semiconductor Manufacturing, (**2000**), vol. 13, pp. 108-111.

[24] SPEC CPU2000 Benchmark, “http://www.spec.org/”.

[25] D. C. Burger and T. M. Austin, “The Simplescalar tool-set, version 2.0.,” Technical Report 1342, Dept. of Computer Science, UW, (**1997**).

[26] Predictive technology model, “http://www-device.eecs.berkeley.edu/ptm/interconnect.html”.

.

BIOGRAPHIES

V. Kamakoti received his Ph.D Degree from the Department of Computer Science and Engineering, Indian Institute of Technology, Madras, India in 1995. Currently, he is working as an Associate Professor in the same Department. His research interests include Software Aspects of VLSI Design, Computer Architecture and High Performance Computing.

Madhu Mutyam received his Ph.D Degree from the Department of Computer Science and Engineering, Indian Institute of Technology, Madras in 2003. Currently, he is working as Assistant Professor at International Institute of Information Technology, Hyderabad. India. His research interests include Unconventional Models of Computing, Theoretical Computer Science and Computer Architecture.

K. Najeeb is pursuing his Ph.D Degree in Computer Science from the Department of Computer Science and Engg., Indian Institute of Technology, Madras. His research interests include Low Power VLSI Design and Computer Aided Design of VLSI circuits.

Vishal Gupta received his B.Tech from the Department of Computer Science and Engineering, Indian Institute of Technology, Madras in 2006. His research interests include GALS based Design, Low Power VLSI Design and Computer Architecture.

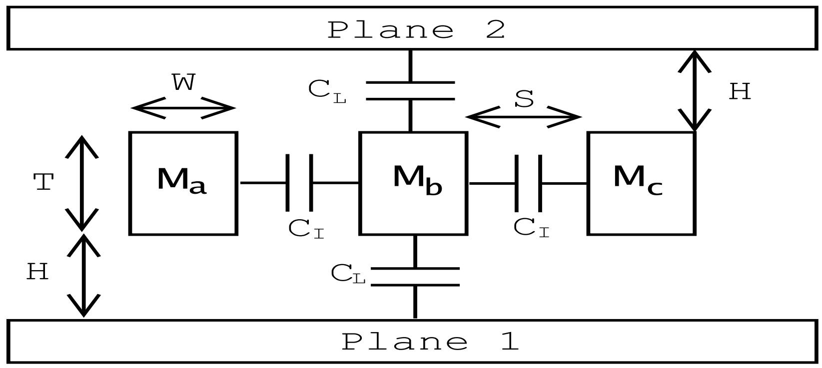


Figure 1. Coupling and Self Capacitance in Interconnects

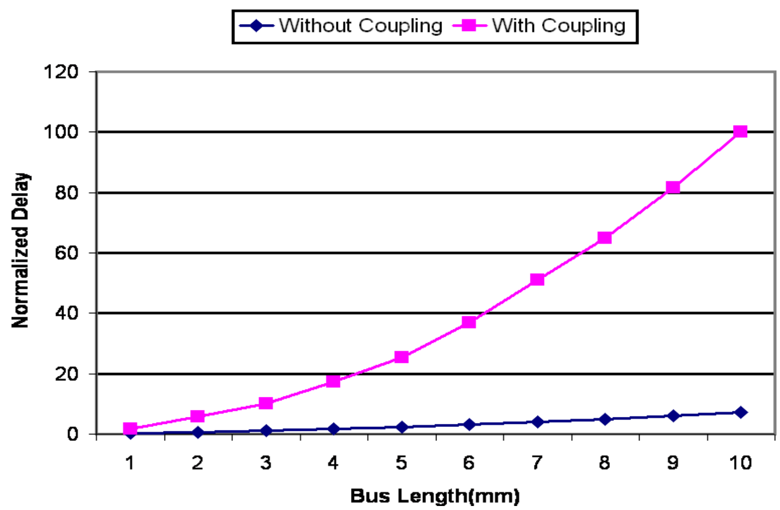


Figure 2. Effect of Coupling Capacitance on Delay

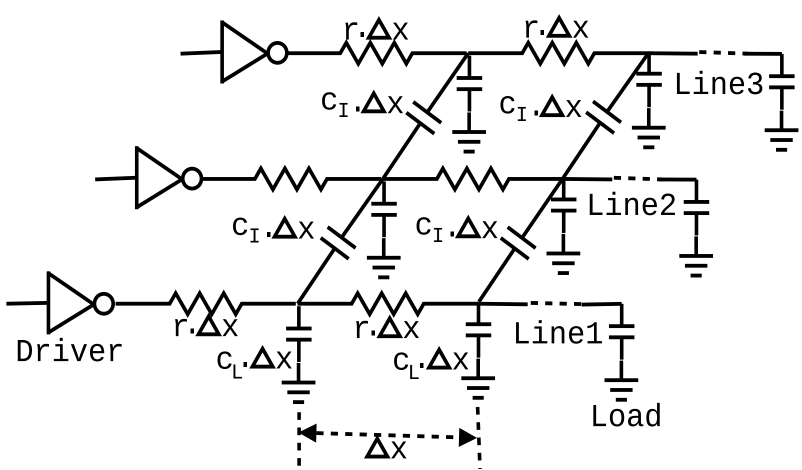


Figure 3. Interconnect modeling for a Sub-Micron Bus [5]

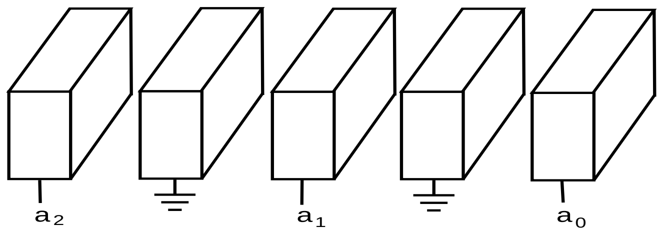


Figure 4. Use of Shielding wires to eliminate Crosstalk

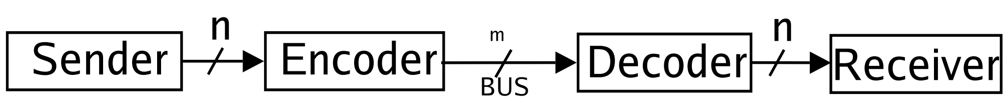


Figure 5. Model for Bus Encoding Schemes

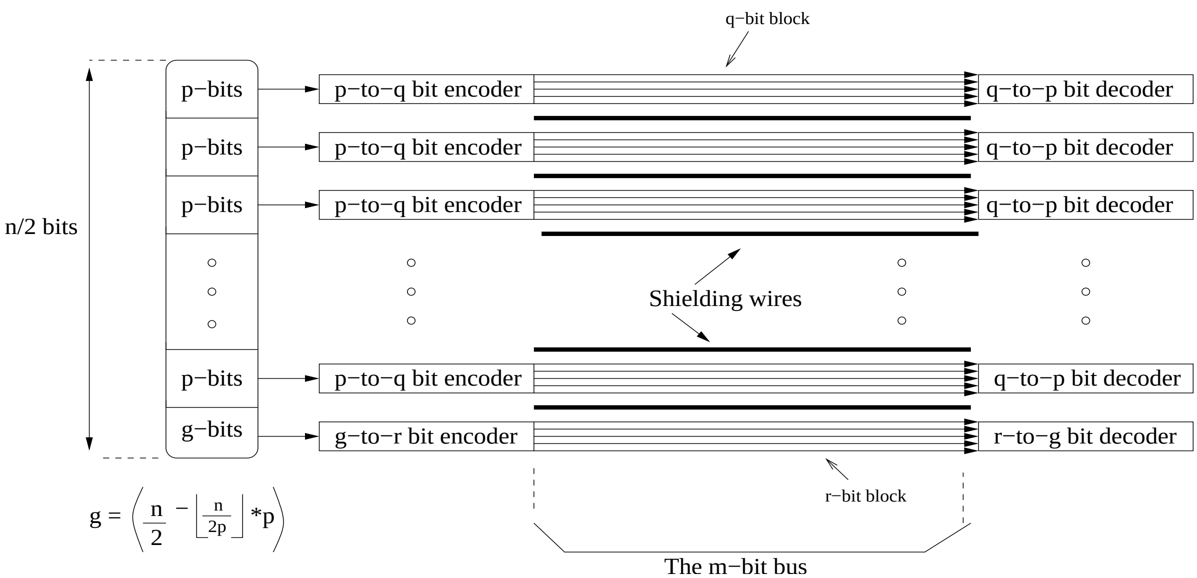


Figure 6. The proposed encoding technique

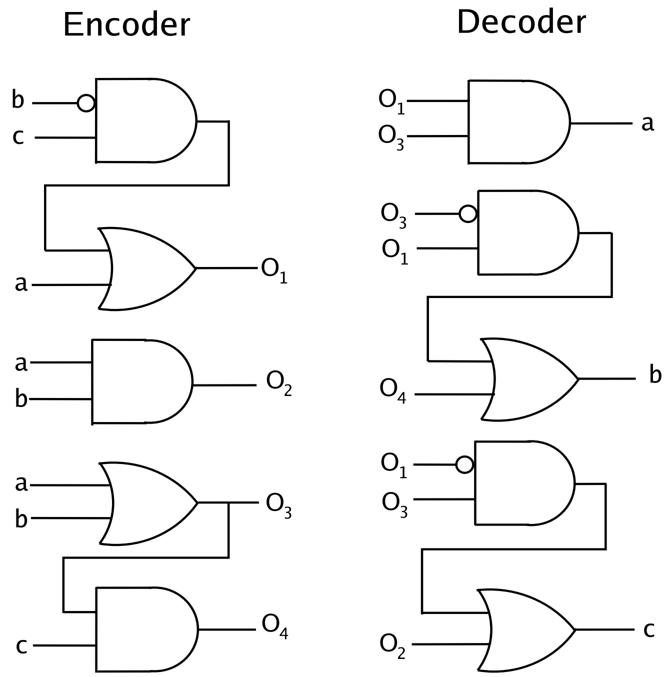
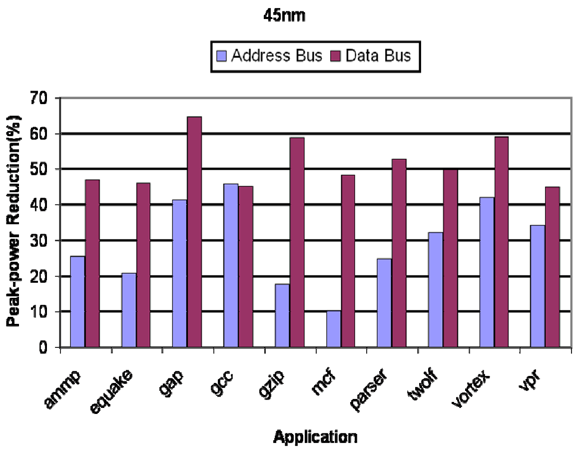
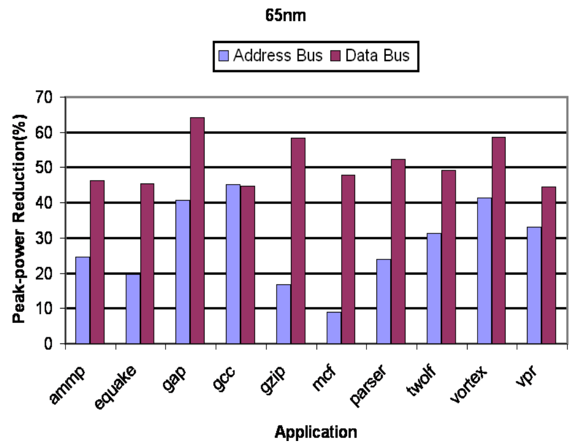


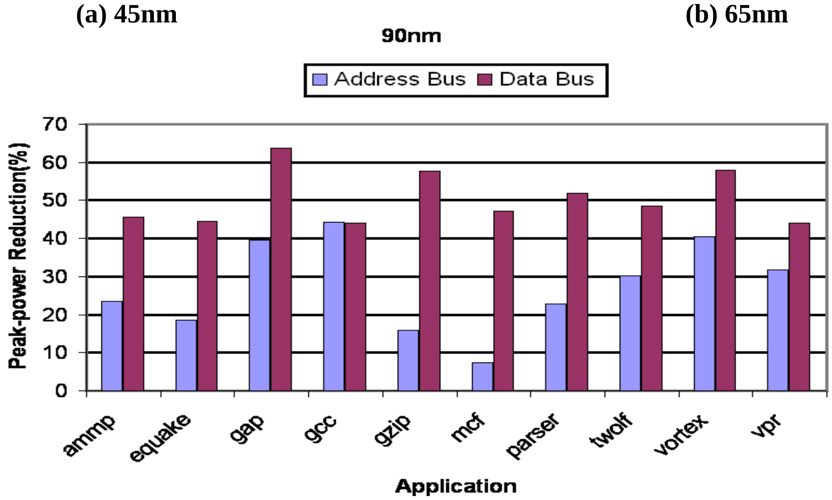
Figure 7. Logic for the Encoder and Decoder



(a) 45nm



(b) 65nm



(c) 90nm

Figure 8. Percentage Reduction in Peak-Power for different technologies

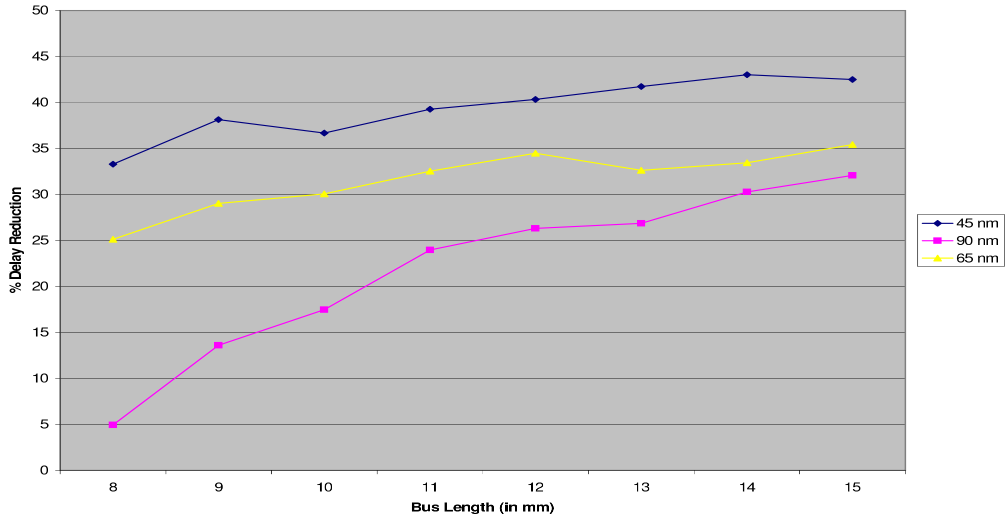


Figure 9. Delay Reduction Percentages obtained from Spice Analysis

---

TABLE I

CROSSTALK CLASSES (HERE CAPACITANCE FACTOR $\lambda = \frac{C_I}{C_L}$) [5], [12]

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

TABLE II

COUPLING TRANSITIONS AND CORRESPONDING TRANSITION PATTERNS [5], [12]

No. of Coupling Transitions	Transition Patterns (–*: 1→1)
-2	(↑,–*,↑)
-1	(↑,–*,–),(–,–*,↑)
0	(–,–,–),(–,↓,–),(↓,–,–),(–,–,↓),(–,–,↑),(↑,–,–),(↑,–,↓),(↓,–,↑), (↑,–,↑),(↓,–,↓),(↑,↓,–),(–,↓,↑),(↓,↓,–),(–,↓,↓),(↑,↓,↑), (↑,↓,↓),(↓,↓,↑),(↓,↓,↓),(↑,↑,↑)
1	(↓,–*,–),(–,–*,↓),(↑,↑,–),(–,↑,↑)
2	(↓,–*,↓),(–,↑,–),(↓,↑,↑),(↑,↑,↓)
3	(↓,↑,–),(–,↑,↓)
4	(↓,↑,↓)

TABLE III

CONSTRUCTION OF 4-BIT CROSSTALK PREVENTING CODES

F_1	F_2	F_3	F_4
0	0 0	0 0 0	0 0 0 0
1	0 1	0 0 1	0 0 0 1
	1 1	1 0 0	0 1 0 0
		1 0 1	0 1 0 1
		1 1 1	0 1 1 1
			1 1 0 0
			1 1 0 1
			1 1 1 1

TABLE IV

SIZE OF SET Fq

q	F_q Set Size	q	F_q Set Size
1	2	6	21
2	3	7	34
3	5	8	55
4	8	9	89
5	13	10	144

TABLE V

PARAMETERS FOR CPCT ENCODING

p	q	g	r	m
1	1	0	0	31
2	3	0	0	31
3	4	1	1	26
4	6	0	0	27
5	7	1	1	25
6	9	4	6	26
7	10	2	3	25
8	12	0	0	25

TABLE VI

ENCODER COMPLEXITY WITH RESPECT TO *p* AND *q*

CPCT Parameters		Encoder Type	Circuit Size (#gate count)	No. of bus transmission lines	Circuit Delay (Depth of the circuit)
p	q				
3	4	3 to 4 encoder	6	26	3
5	7	5 to 7 encoder	46	25	15
7	10	7 to 10 encoder	312	25	21
8	12	8 to 12 decoder	567	25	24

TABLE VII

DECODER COMPLEXITY WITH RESPECT TO *p* AND q

CPCT Parameters		Decoder Type	Circuit Size (#gate count)	Circuit Delay (Depth of the circuit)
p	q			
3	4	4 to 3 decoder	7	3
5	7	7 to 5 decoder	118	14
7	10	10 to 7 decoder	434	19
8	12	12 to 8 decoder	663	21

TABLE VIII

CPCT Encoding

Data a b c	Code $O_4O_3O_2O_1$	Data a b c	Code $O_4O_3O_2O_1$
0 0 0	0 0 0 0	1 0 0	0 1 0 1
0 0 1	0 0 0 1	1 0 1	1 1 0 1
0 1 0	0 1 0 0	1 1 0	0 1 1 1
0 1 1	1 1 0 0	1 1 1	1 1 1 1

TABLE IX

DELAY, POWER AND AREA OVERHEAD OF THE ENCODER AND DECODER

Circuit	Power	Delay	Area
Encoder	0.8μw	0.260ns	18μm^2
Decoder	0.9μw	0.270ns	21μm^2

TABLE X

TECHNOLOGY PARAMETERS FOR GLOBAL WIRES DERIVED FROM [9]

Node (nm)	90	65	45
Width (nm)	237	160	103
Spacing (nm)	237	160	103
Thickness (nm)	498	325	236
Height (nm)	498	325	236
Dielectric(ε_r)	2.8	2.5	2.1
V_{DD} (V)	1	0.7	0.6
f_{clk} (GHz)	3.99	6.73	11.51

TABLE XI

DELAY ANALYSIS OF UNCODED BUS

Node	90nm	65nm	45nm
S(μm)	0.237	0.16	0.103
R(Ω/mm)	198.448	475.624	905.051
C_I (fF/mm)	92.025	85.666	74.522
C_L (fF/mm)	26.458	22.577	18.259
λ	3.48	3.79	4.08

TABLE XII

DELAY ANALYSIS OF CODED BUS

Node	90nm	65nm	45nm
S'(μm)	0.351	0.237	0.152
R(Ω/mm)	198.448	475.624	905.051
C_I (fF/mm)	63.796	59.39	51.646
C_L (fF/mm)	32.782	28.058	22.752
λ	1.95	2.12	2.27

TABLE XIII COMPARISON OF CODING SCHEMES FOR A 32-BIT BASIC TRANSMISSION

Approach/ Reference Papers	Widen the spacing between bus lines	Spatial/ Temporal Redundancy	No. of Bus Lines	Shielding Protect-ion	Elimin-ation of Class 5 & Class 6	Power reduction in comparison to a 32-bit Basic bus	Delay reduction in comparison to a 32-bit Basic bus
Basic bus	NA	NA	32	no	no	NA	NA
Shielding Bus	no	spatial	64	high	yes	yes	yes
[18]	no	spatial	33	no	no	25%	-----
[19]	no	both	33	medium	yes	yes*	yes*
[6]	no	spatial	46	low	partial	no*	yes*
[21]	no	spatial	48	medium	yes	10%	yes*
[3]	no	spatial	52	medium	yes	yes*	yes*
[22]	no	both	35	medium	yes	51(57)% (160 nm)#	24(38)% (160 nm)#
[14]	no	spatial	55	medium	yes	23% (180 nm)#	yes*
[13]	no	spatial	33	low	no	25%	no

Approach/ Reference Papers	Widen the spacing between bus lines	Spatial/ Temporal Redundancy	No. of Bus Lines	Shielding Protect-ion	Elimin-ation of Class 5 & Class 6	Power reduction in comparison to a 32-bit Basic bus	Delay reduction in comparison to a 32-bit Basic bus
[16]	no	spatial	55	low	class 6	yes*	no
[4]	yes	temporal	24	medium	yes	17(43%) (65 nm)#	12(5%) (65 nm)#
Our approach	Yes	temporal	26	medium	yes	30(52%) (45 nm)#	37(37%) (45 nm)#

a(d%) (k nm): Address bus a% , data bus d%, implemented using a k nanometer technology

* Percentage value not reported