

# Delay and Energy Efficient Data Transmission for On-Chip Buses

Madhu Mutyam<sup>†\*</sup>

<sup>†</sup>International Institute of Information Technology  
Gachibowli, Hyderabad  
India  
mutyam@cse.psu.edu

Melvin Eze\* N. Vijaykrishnan\* Yuan Xie\*

\*Dept. of Computer Science and Engineering  
Pennsylvania State University  
University Park, PA 16802  
{eze,vijay,yuanxie}@cse.psu.edu

## Abstract

*On-chip buses in deep sub-micron designs consume significant amounts of power and have large propagation delays. Thus, minimizing power consumption and propagation delay are the most important design objectives. In this paper, we propose a technique for delay and energy efficient data transmission for on-chip buses and evaluate the effectiveness of our technique by focusing on the L1 cache address/data buses of a microprocessor using the SPEC2000 CINT benchmark suit. We show that our technique achieves 31% (30%) of delay improvement along with energy savings of 13% (9%) over the base case for data transmission on address (data) bus.*

## 1 Introduction

As VLSI fabrication technologies scaled down to the deep sub-micron region, the inter-wire capacitance ( $C_I$ ) becomes significant compared to the wire-to-substrate capacitance ( $C_L$ ). As  $C_I$  is the dominant capacitance in deep sub-micron era, it has two significant effects: large propagation delay due to opposite transitions on adjacent wires [5, 17, 19] and power dissipation associated with driving on-chip buses [17]. There are techniques to minimize the effects of opposite transitions on adjacent wires and/or power consumption due to capacitive coupling [6, 7, 8, 10, 9, 11, 12, 13, 14, 16, 18, 20]. Many of the techniques use *spatial redundancy* to minimize power or delay and hence have large area overhead. For example, *crossstalk avoidance codes* [18] eliminate opposite transitions on adjacent wires completely but require 46 wires to encode 32-bit data and a coding technique proposed in [16] is both area and energy-efficient and eliminates opposite transitions on adjacent wires completely but it requires 48 wires for encoding 32-bit data.

In this paper we exploit the *variable cycle transmission*

technique [9] to apply *temporal redundancy* in the process of minimizing both delay and energy for on-chip data transmission. Our encoding scheme requires *two* extra wires to encode *any* bit-width data and achieves 31% (30%) of delay improvement along with 13% (9%) of energy savings over the base case for data transmission on address (data) bus.

## 2 Analytical Models for Delay and Energy Consumption

In order to measure the actual effects of inter-wire capacitance in deep sub-micron technologies, analytical models for delay and energy consumption in deep sub-micron buses have been proposed in [12, 14, 15]. Throughout the paper we consider these analytical models. Let  $d_t$  be a  $n$ -bit data present on the bus. By considering  $\lambda = \frac{C_I}{C_L}$ , the propagation delay of the RC circuit, for transmitting a  $n$ -bit data  $d_{t+1}$ , can be calculated by [14]

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

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

where  $R_T$  is the total resistance and  $\Delta_k = d_{t+1}^k - d_t^k$ . Similarly, the total energy (due to self and coupling transitions) consumed during the transition from  $d_t$  to  $d_{t+1}$  is given by [12, 15]

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

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

For example, if  $d_t = 010$  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  $E(d_t, d_{t+1}) = 3C_L$ . From these two examples (and hence from the above two equations), it is clear that the transition patterns, such as  $0 \rightarrow 1$  (or  $\uparrow$ ),

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

**Table 1. Crosstalk classes (here  $\lambda = \frac{C_L}{C_T}$ )**

1 → 0 (or ↓), 0 → 0 (or -), and 1 → 1 (or -), of data not only dictate the propagation delay but also the energy consumption.

Since the propagation delay is determined by the transition pattern of the data, transition patterns are classified into six different classes [9, 14] based on the relative delay a wire w.r.t. its adjacent wires, which we call as *crosstalk classes*. Table 1 shows different crosstalk classes. From this table, it is clear that the worst-case crosstalk delay for transmitting data items, without any coding, is  $C_L R_T(1 + 4\lambda)$ .

### 3 Related Work

Bus encoding techniques to eliminate crosstalk classes 5 and 6 have been proposed in [6, 18]. Note that the technique proposed in [18] can also eliminate part of the crosstalk class 4. From Table 1, it is clear that as a result of eliminating crosstalk classes 5 and 6, the worst-case crosstalk delay becomes  $C_L R_T(1 + 2\lambda)$ . Though the worst-case delay can be reduced by 50%, these techniques are difficult to implement and have large area overhead (for a 32-bit data, the technique in [18] requires 46 wires without memory and 40 wires with memory and the technique in [6] requires 52 wires). Similar to the ideas of [6, 18], a new coding technique is proposed in [16] to obtain 50% reduction in delay along with 10% reduction in energy, but it also requires 48 wires to encode 32-bit data.

A bus encoding technique to minimize power consumption and eliminate crosstalk classes 5 and 6 is proposed in [10], which requires 55 wires to encode 32-bit data. Coupling-driven bus encoding technique [8] reduces power consumption by 30% on an average, but it may not eliminate crosstalk class 4 and above and hence it is not much advantageous from the delay perspective. Another bus encoding technique to minimize both energy and delay is proposed in [7], which can eliminate only crosstalk classes 4 and 6 (but not crosstalk class 5) so that the worst-case delay is still  $C_L R_T(1 + 3\lambda)$  and it requires 55 wires to encode 32-bit data.

Odd/even bus invert technique is proposed in [20] to minimize coupling energy. With half cycle delay of one of the adjacent wires, it can eliminate opposite transitions on adjacent wires and achieve 30% power savings. Though this technique is energy efficient, from the delay perspective

it is not advantageous because the half cycle delay can result in crosstalk class 4 patterns such as - ↑ - and - ↓ -. As a result, the net delay for receiving the data becomes  $C_L R_T(2 + 4\lambda)$ , which is more than that of crosstalk class 6. Another technique to minimize power consumption due to coupling transitions is proposed in [13], but it may not be easy to implement due to its complex codec circuitry.

A technique close to the topic of this paper is proposed in [9], where instead of considering the worst-case delay of  $C_L R_T(1 + 4\lambda)$ , the authors considered variable delay which is determined based on the data present on the bus and next data to be transmitted. Based on the crosstalk class of the next data w.r.t. the present data on the bus, the number of cycles required to transmit the next data is controlled dynamically and hence obtain significant performance benefits. This technique requires one extra wire, which acts as a shield wire between the actual data and the *ready*<sup>1</sup> signal. Since the data is transmitted as it is without any coding, this technique does not provide any energy benefits.

Contrast to the above mentioned works, here we propose a technique which eliminates crosstalk classes 5 and 6 with the help of temporal redundancy and uses variable delay for data transmission so that both delay and energy minimization can be achieved. Unlike the technique in [9], where for crosstalk class 6 it takes the delay of  $4C_L R_T(1 + \lambda)$ , our technique takes the delay of  $3C_L R_T(1 + \lambda)$  so that we can even get better performance benefits over the technique proposed in [9].

### 4 Our Approach

Basic idea behind our approach is to analyze the crosstalk class of next data w.r.t. the data present on the bus and based on the crosstalk class we either transmit the original data or encode the data into two new data items and transmit the encoded data. In either case, we use the necessary delay for data transmission.

#### 4.1 Area and Energy Efficient Crosstalk Analyzer

From Section 2, we know that the crosstalk class of next  $n$ -bit data w.r.t. the present data on the bus is determined by the maximum delay among all the wires  $k$ ,  $1 \leq k \leq n$ . So, in order to know the crosstalk class, first we have to find the delay of each of the  $n$  wires. In [9], a crosstalk analyzer is designed in such a way that whenever a next data is received, the analyzer determines the crosstalk class of the next data by comparing it with the present data on the bus. This process incurs extra hardware and consumes more energy as none of the computed data, in the process of finding

<sup>1</sup>the authors considered 33-bit data with 32-bit actual data and one *ready* signal

```

Input: present data  $d_t$  and next data  $d_{t+1}$ 
Output: encoded data items  $d'_{t+1}$  and  $d''_{t+1}$ 
if(crosstalk_class  $\geq$  5)
  begin
     $d'_{t+1} = d_{t+1}$ ;  $d''_{t+1} = d_{t+1}$ 
    for( $i = n$ ;  $i > 0$ ;  $i --$ )
      if( $(XORP[i + 1] \& XORN[i + 1] \& (d_t[i] \oplus d_{t+1}[i])) \parallel$ 
         $(XORP[i] \& XORN[i] \& (d_t[i - 1] \oplus d_{t+1}[i - 1]))$ )
         $d'_{t+1}[i] = 1$ ;  $d''_{t+1}[i] = 0$ 
  end

```

**Table 2. Encoder**

a crosstalk class, is *reused* in the next stage. Generally, in the process of finding a crosstalk class, the analyzer checks whether or not there is any opposite transition on the adjacent wires (this is needed for crosstalk classes 4, 5, and 6). If  $d_1$  is the present data and  $d_2$  is the next data, then to know whether or not a wire  $k$ ,  $1 \leq k < n$ , has an opposite transition w.r.t. wire  $k + 1$ , it is enough to take the logical *AND* of  $d_1[k] \oplus d_1[k + 1]$ ,  $d_2[k] \oplus d_2[k + 1]$ , and  $d_1[k] \oplus d_2[k]$ . In this process, we can compute the *XOR* values of adjacent bits of  $d_2$  only once but use them twice (i.e., for finding crosstalk classes of  $d_2$  w.r.t.  $d_1$  and  $d_3$  w.r.t.  $d_2$ ). As a result, the hardware and the energy consumption for the crosstalk analyzer is significantly reduced compared to the similar circuit given in [9]. We obtain the hardware reduction at the cost of two  $(n - 1)$ -bit registers, i.e., *XORP* and *XORN*, to store the *XOR* values of adjacent bits of present  $n$ -bit data and next  $n$ -bit data, respectively. Whenever, a next data  $d_2$  is received, we only compute *XORN* and move the previous *XORN* (that of  $d_1$ ) to *XORP* with the possible exception when the next data  $d_2$  is in crosstalk class 5 or 6 w.r.t. the present data  $d_1$ . Since temporal redundancy is used for crosstalk classes 5 and 6, in such situations, *XORP* gets the *XOR* values of adjacent bits of  $d_2''$  (defined in the next subsection).

Our design requires 35 wires for transmitting a 33-bit data, which consists of 32-bit actual data and one *ready* signal. The additional wires are: one *temporal\_redundancy* signal for crosstalk class 5 or 6 and one *shield* wire between the *ready* signal and the actual data to prevent opposite transitions. There is no need of a shield wire between the *temporal\_redundancy* signal and the *ready* signal as the *temporal\_redundancy* signal makes  $0 \rightarrow 1$  transition only when the *ready* signal makes  $0 \rightarrow 1$  transition and it makes  $1 \rightarrow 0$  transition when the *ready* signal makes  $1 \rightarrow 1$  transition.

## 4.2 Variable Cycle Transmission with Temporal Redundancy

We now present our bus encoding technique, i.e., variable cycle transmission with temporal redundancy (VCTR technique). In order to facilitate variable cycle transmission, we use two different clocks with periods  $C_L R_T(1 + \lambda)$

```

Input: present data  $z_{t+1}$  and previous data  $R_1$  and  $R_2$ 
Output: decoded data  $d_{t+1}$ 
if(ready)
  if(temporal_redundancy)
    for( $i = n$ ;  $i > 0$ ;  $i --$ )
      if( $R_2[i] == z_{t+1}[i]$ )
         $d_{t+1}[i] = z_{t+1}[i]$ 
      else
         $d_{t+1}[i] = \sim R_1[i]$ 
  else
     $d_{t+1} = z_{t+1}$ 

```

**Table 3. Decoder**

and  $2C_L R_T(1 + \lambda)$ . Whenever a next data  $d_{t+1}$  is received, our crosstalk analyzer determines the crosstalk class of  $d_{t+1}$  w.r.t. the present data  $d_t$  on the bus. If the crosstalk class of  $d_{t+1}$  w.r.t.  $d_t$  is from the set  $\{1, 2, 3, 4\}$ ,  $d_{t+1}$  is transmitted with a delay of  $C_L R_T(1 + \lambda)$  (or  $2C_L R_T(1 + \lambda)$ ) if the crosstalk class is  $\leq 3$  (or 4) and the *ready* signal is *set* and the *temporal\_redundancy* signal is *reset*. If the crosstalk class is either 5 or 6, unlike delaying the transmission by the required number cycles (as proposed in [9]), we encode  $d_{t+1}$  into two new data items  $d'_{t+1}$  and  $d''_{t+1}$  (using the coding technique as given in Table 2) in such a way that the crosstalk class of  $d'_{t+1}$  w.r.t.  $d_t$  is from the set  $\{1, 2, 3, 4\}$  and that of  $d''_{t+1}$  w.r.t.  $d'_{t+1}$  is 3 (formal proof is given in the next section). Note that the encoder (given in Table 2) takes different conditions for the boundary values, i.e., when  $i = 1$ , it checks only  $(XORP[2] \& XORN[2] \& (d_t[1] \oplus d_{t+1}[1]))$  and when  $i = n$ , it checks  $(XORP[n] \& XORN[n] \& (d_t[n - 1] \oplus d_{t+1}[n - 1]))$ . Thus, we transmit  $d'_{t+1}$  with a delay of  $2C_L R_T(1 + \lambda)$  and  $d''_{t+1}$  with a delay of  $C_L R_T(1 + \lambda)$ . As a result, the net delay for transmitting two new data items becomes  $3C_L R_T(1 + \lambda)$ . During the transmission of  $d'_{t+1}$ , we *reset* both *temporal\_redundancy* signal and *ready* signal, whereas for  $d''_{t+1}$  transmission, we *set* both *temporal\_redundancy* signal and *ready* signal. Clearly, for both crosstalk classes 5 and 6, our approach takes  $3C_L R_T(1 + \lambda)$  cycles whereas the technique given in [9] takes  $3C_L R_T(1 + \lambda)$  and  $4C_L R_T(1 + \lambda)$  cycles, respectively. So, our approach not only provides energy benefits (due to elimination of coupling transitions of crosstalk classes 5 and 6) but also performance benefits.

Decoding the data at the receiver end is a simple process (as shown in Table 3). At the receiver end, we store the value on the bus in a  $n$ -bit register  $R_1$  when the *ready* signal is *set* and the *temporal\_redundancy* signal is *reset* and the value on the bus is stored in another  $n$ -bit register  $R_2$  when both *ready* signal and *temporal\_redundancy* signal are *reset*. We use the values of these two registers to decode the actual data. When the *ready* signal is *set* and the *temporal\_redundancy* signal is *reset*, there is no need of decoding (because, this situation arises only when the trans-

mitted data is in crosstalk class from the set  $\{1, 2, 3, 4\}$ , so we consider whatever the data present on the bus as the next data. On the other hand, if both *ready* signal and *temporal\_redundancy* signal are *set*, we decode the data using the decoding algorithm given in Table 3. Our decoder is a simple bit-wise comparison circuit so the delay due to the decoder can be ignored.

### 4.3 Correctness of our approach

We now prove that our encoding technique works correctly for any data and minimizes the delay and the energy consumption. Let  $CC(d_i, d_j)$  denote the crosstalk class of  $d_i$  w.r.t  $d_j$ ,  $T(d_i, d_j)$  denote the delay for transmitting  $d_j$  after  $d_i$ , and  $E(d_i, d_j)$  denote the energy consumption during the transition from  $d_i$  to  $d_j$ .

**Theorem 1** For any pair of  $n$ -bit data  $d_t$  and  $d_{t+1}$ , if  $CC(d_{t+1}, d_t) \geq 5$ , then  $d_{t+1}$  can be encoded as  $d'_{t+1}$  and  $d''_{t+1}$  such that

- $CC(d'_{t+1}, d_t) < 5$  and  $CC(d''_{t+1}, d'_{t+1}) = 3$
- $T(d_t, d'_{t+1}) + T(d'_{t+1}, d''_{t+1}) \leq T(d_t, d_{t+1})$
- $E(d_t, d'_{t+1}) + E(d'_{t+1}, d''_{t+1}) \leq E(d_t, d_{t+1})$

*Proof.* To prove  $CC(d'_{t+1}, d_t) < 5$ :

Let  $tp(a, b)$  be the transition pattern over  $\{\uparrow, \downarrow, -\}$  of length  $n$  when a  $n$ -bit data  $a$  is transmitted after another  $n$ -bit data  $b$ . Assume that  $tp(d_{t+1}, d_t) = p_1 \cdots p_n$ ,  $p_i \in \{\uparrow, \downarrow, -\}$ ,  $1 \leq i \leq n$ . Since  $CC(d_{t+1}, d_t) \geq 5$ , there must be at least 2 wires which have opposite transitions w.r.t. one or both of their adjacent wires. For simplicity, assume that there are exactly 2 wires, i.e., wire  $i$  and wire  $i+1$ , which have opposite transitions w.r.t. each other. Note that the proof can be easily extended to any number of wires having opposite transitions w.r.t. one or both of their adjacent wires. So, there is no pattern of type  $\uparrow\uparrow$  or  $\uparrow\downarrow$  in both  $p_1 \cdots p_i$  and  $p_{i+1} \cdots p_n$ . Let  $tp(d'_{t+1}, d_t) = p'_1 \cdots p'_n$  and  $tp(d''_{t+1}, d'_{t+1}) = p''_1 \cdots p''_n$ . Since there is no pattern of type  $\uparrow\uparrow$  or  $\uparrow\downarrow$  in both  $p_1 \cdots p_i$  and  $p_{i+1} \cdots p_n$ , according to our encoding technique, for  $1 \leq j \notin \{i, i+1\} \leq n$ ,  $p'_j = p_j$  and  $p''_j = -$ , and for  $j \in \{i, i+1\}$ ,  $p'_j = \downarrow$ , and if  $p_j = \uparrow$ ,  $p'_j = p_j$ , and if  $p_j = \downarrow$ ,  $p'_j = -$ . Hence,

$$\begin{aligned} tp(d'_{t+1}, d_t) &= p'_1 \cdots p'_{i-1} p'_i p'_{i+1} p'_{i+2} \cdots p'_n \\ &= p_1 \cdots p_{i-1} p'_i p'_{i+1} p_{i+2} \cdots p_n \end{aligned}$$

We know that either  $p'_i = p_i$  or  $p'_{i+1} = p_{i+1}$ . In either case,  $p_1 \cdots p_{i-1} p'_i p'_{i+1} p_{i+2} \cdots p_n$  does not have a pattern of type  $\uparrow\uparrow$  or  $\uparrow\downarrow$ . Hence,

$$CC(d'_{t+1}, d_t) < 5.$$

To prove  $CC(d''_{t+1}, d'_{t+1}) = 3$ :

We know that

$$\begin{aligned} tp(d''_{t+1}, d'_{t+1}) &= p''_1 \cdots p''_{i-1} p''_i p''_{i+1} p''_{i+2} \cdots p''_n \\ &= \cdots \downarrow \downarrow \cdots \end{aligned}$$

From Table 1, it is clear that

$$CC(d''_{t+1}, d'_{t+1}) = 3.$$

To prove  $T(d_t, d'_{t+1}) + T(d'_{t+1}, d''_{t+1}) \leq T(d_t, d_{t+1})$ :  
Since  $CC(d'_{t+1}, d_t) < 5$  and  $CC(d''_{t+1}, d'_{t+1}) = 3$ ,

$$\begin{aligned} T(d_t, d'_{t+1}) &= 2C_L R_T (1 + \lambda) \\ T(d'_{t+1}, d''_{t+1}) &= C_L R_T (1 + \lambda) \end{aligned}$$

But  $CC(d_{t+1}, d_t) \geq 5$ . So,

$$T(d_t, d_{t+1}) \geq 3C_L R_T (1 + \lambda)$$

Hence,

$$T(d_t, d'_{t+1}) + T(d'_{t+1}, d''_{t+1}) \leq T(d_t, d_{t+1}).$$

To prove  $E(d_t, d'_{t+1}) + E(d'_{t+1}, d''_{t+1}) \leq E(d_t, d_{t+1})$ :

For simplicity, as in the case of the first proof, assume that there are exactly 2 wires, i.e., wire  $i$  and wire  $i+1$ , which have opposite transitions w.r.t. each other. Let  $d_t^i$  be the value of  $i^{th}$ -wire at time instant  $t$ . Assume that  $d_t^i d_t^{i+1} = 01$  and  $d_{t+1}^i d_{t+1}^{i+1} = 10$  so that  $d_{t+1}^i d_{t+1}^{i+1} = 11$  and  $d_{t+1}^i d_{t+1}^{i+1} = 00$ . Note that the proof can be easily extended to the case where  $d_t^i d_t^{i+1} = 10$  and  $d_{t+1}^i d_{t+1}^{i+1} = 01$ . Then

$$\begin{aligned} E(d_t, d_{t+1}) &= C_L(x_1 - \lambda d_{t+1}^{i-1}) + C_L(\lambda d_{t+1}^{i+2} + x_2) \\ &\quad + C_L(1 + 3\lambda - \lambda(d_{t+1}^{i-1} - d_t^{i-1})), \end{aligned}$$

where  $C_L(x_1 - \lambda d_{t+1}^{i-1})$  is the energy consumption of first  $(i-1)$  wires (negative component is the effect of wire  $i$  on wire  $(i-1)$ ),  $C_L(1 + 3\lambda - \lambda(d_{t+1}^{i-1} - d_t^{i-1}))$  is the energy consumption of wire  $i$  and wire  $(i+1)$  including the effects of their other adjacent wires, and  $C_L(\lambda d_{t+1}^{i+2} + x_2)$  is the energy consumption of last  $(n-i-1)$  wires including the effect of wire  $(i+1)$  on wire  $(i+2)$ .

$$\begin{aligned} E(d_t, d'_{t+1}) &= C_L(x_1 - \lambda d_{t+1}^{i-1}) + C_L(x_2) + C_L(1 + \lambda \\ &\quad - \lambda(d_{t+1}^{i-1} - d_t^{i-1}) - \lambda(d_{t+1}^{i+2} - d_t^{i+2})) \end{aligned}$$

$$E(d'_{t+1}, d''_{t+1}) = C_L(\lambda d_{t+1}^{i-1}) + C_L(0) + C_L(\lambda d_{t+1}^{i+2})$$

If we assume that  $E(d_t, d'_{t+1}) + E(d'_{t+1}, d''_{t+1}) > E(d_t, d_{t+1})$ , after simplification, we obtain the following inequality

$$\lambda(d_{t+1}^{i-1} - d_{t+1}^{i+2} + d_t^{i+2}) > 2\lambda.$$

Since  $\lambda > 0$ ,

$$d_{t+1}^{i-1} - d_{t+1}^{i+2} + d_t^{i+2} > 2,$$

which is always false. Hence,

$$E(d_t, d'_{t+1}) + E(d'_{t+1}, d''_{t+1}) \leq E(d_t, d_{t+1}).$$

Layer	H ( $\mu\text{m}$ )	W ( $\mu\text{m}$ )	S ( $\mu\text{m}$ )	T ( $\mu\text{m}$ )	C <sub>L</sub> (fF/mm)	C <sub>J</sub> (fF/mm)
MET 5	1.04	0.52	0.52	0.7	34.317	80.478

Table 4. Wire parameters

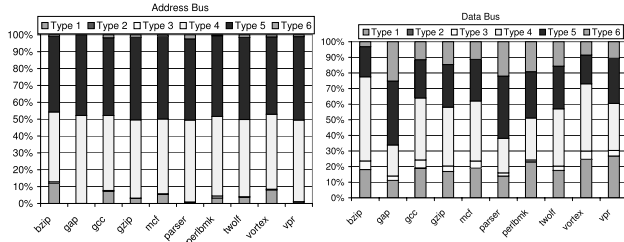


Figure 1. Distribution of transition patterns

## 5 Experimental validation of our approach

We now experimentally validate our approach. We design the crosstalk analyzer, the encoder, and the decoder in Verilog and synthesize it using the Synopsys Design Compiler with  $0.16\mu\text{m}$  *oki* technology library. We model three different wire lengths, i.e., 2mm, 5mm, and 10mm, to capture short, medium, and long wires, respectively, in the target technology. The Berkeley interconnect model [1] is used to calculate the ground and coupling capacitance of the interconnects. In our experimental results, we use the parameters of metal 5 (shown in Table 4).

We use SimpleScalar 3.0 [4] and the SPEC2000 CINT [2] benchmark suite to simulate the performance of different on-chip buses between the processor datapath and L1 I-cache/D-cache. Transition pattern distributions of both address and data buses are given in Figure 1.

We compare the variable cycle transmission with temporal redundancy (VCTR) technique with the base case (i.e., without any coding), the variable cycle transmission (VCT) method [9], the crosstalk prevention coding (CPC) technique [18], and the shielding (SHD) method [3]. We consider two delays,  $C_L R_T(1 + \lambda)$  (single cycle) and  $2C_L R_T(1 + \lambda)$  (two cycles; note that while absolute delay is only  $C_L R_T(1 + 2\lambda)$ , but a variable cycle transmission scheme requires two-cycle delay to be a multiple of the single cycle period) for the VCTR technique, and based on the output of the crosstalk analyzer, we consider the appropriate delay. Note that the delay for the base case is  $C_L R_T(1+4\lambda)$ , for the CPC technique and shielding method the delay is  $C_L R_T(1 + 2\lambda)$ , and for the VCT technique the delays are multiples of  $C_L R_T(1 + \lambda)$  based on the crosstalk class type.

In the VCT method, the delay between the next data item and the present data item is determined by the crosstalk class of the next data w.r.t. the present data item. Note that in this method, no encoding is applied on the transmitted

Ori. code	000	001	010	011	100	101	110	111
CPC code	0000	0001	0100	0101	0111	1100	1101	1111

Table 5. CPC codes

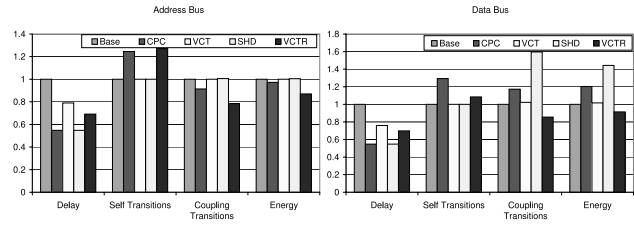


Figure 2. Normalized delay and energy

data. Though by using the CPC method, one can get 46-bit crosstalk-free codes for any 32-bit data, it is very difficult to implement the method. As a result, here we consider 3-to-4 CPC coding, where every 3-bit data is encoded with 4-bit CPC code. Table 5 gives a set of CPC codes for 3-bit data. In order to prevent adjacent bits of different 4-bit encoded words to form crosstalk classes 5 and 6, we use shield wires between every pair of 4-bit encoded words. Hence, for a 33-bit bus we use 55 wires. In the SHD method, one shield, i.e.,  $V_{da}$  or *Ground*, wire is inserted between every two wires so that the crosstalk classes 5 and 6 are completely eliminated and hence the worst-case delay becomes  $C_L \cdot R_T(1 + 2\lambda)$ .

Main characteristics of different methods are given in Figure 2. The values present in this table are averaged across all the benchmarks and normalized w.r.t. the base case. Compared to the VCT technique, our technique achieves better performance in both address and data buses because of two reasons: 1) in the case of crosstalk class 6, the VCT technique considers a delay of  $4C_L R_T(1 + \lambda)$ , whereas our technique encodes the data into two data items such that the net delay becomes  $3C_L R_T(1 + \lambda)$ ; 2) after encoding, for each 01 (or 10),  $d''_{t+1}$  gets 00 so that this can minimize the chances of getting crosstalk classes 5 or 6 for  $d_{t+2}$  w.r.t.  $d''_{t+1}$  (though we have not experimentally evaluated). The second reason is especially true, if same adjacent wires are repeatedly getting opposite transitions. This can be seen in address bus, where few LSB bits will change most of the time. That is the reason why, though there are less number of crosstalk class 6 transitions (from Figure 1), our technique outperforms the VCT technique. From the delay perspective, our technique achieves 31% (30%) savings for address (data) bus with just two extra wires. Though the CPC technique and SHD technique give more delay savings, they require 22 and 32 extra wires, respectively. From the energy perspective, our technique achieves 13% (9%) savings for address (data) bus compared to the base case. Since coupling transitions play a major role in the total energy, though self transitions for our technique are more than that of the CPC technique in the address bus case,

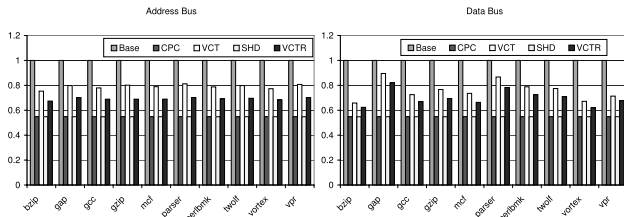


Figure 3. Benchmark-wise normalized delay

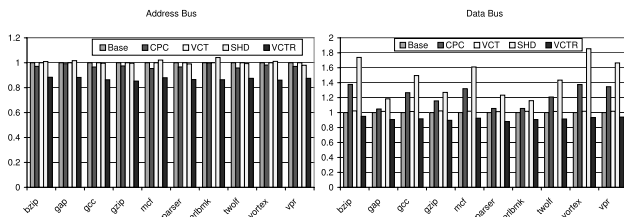


Figure 4. Benchmark-wise normalized energy

because of less number of coupling transitions, the energy consumption of our technique is less than that of the CPC technique. Since no encoding is applied in the VCT technique, its energy consumption is same as that of the base case. In the case of data bus, the CPC technique consumes more energy than that of the base case. The SHD technique consumes more energy than the base case because of more coupling transitions. Benchmark-wise normalized delay and energy consumption (w.r.t. the base case) of different techniques are given in Figure 3-4.

Table 6 gives the normalized area (w.r.t. the base case) and the power overhead. One can see from the table that our technique incurs an area overhead of 33%, 16%, and 11% for 2mm, 5mm, and 10mm bus, respectively. This is due to the codec (i.e., crosstalk analyzer, encoder, and decoder) and two extra wires. The absolute area overhead of our codec is 0.018mm<sup>2</sup>. One can observe that as the length of the interconnects increases the relative area overhead of our technique decreases. Our codec can run at a maximum clock speed of 660 MHz and has power overhead of 2.16mW.

## 6 Conclusion

By combining the ideas of variable cycle transmission and temporal redundancy, we proposed a technique for delay and energy efficient data transmission for on-chip buses and evaluated our technique by focusing on the L1 cache address/data buses of a microprocessor and by using the SPEC2000 CINT benchmark suit. From the results, it is clear that our method achieves 31% (30%) of delay improvement along with 13% (9%) energy savings over the

Method	# of wires	Normalized Area			Extra Power
		2mm	5mm	10mm	
Base	33	100	100	100	—
VCT	34	119	109	106	3.35mW
VCTR	35	133	116	111	2.16mW
CPC	55	177	171	169	1.06mW

Table 6. Codec overhead

base case for data transmission on address (data) bus.

## 7 Acknowledgements

Madhu Mutyam was supported by a BOYSCAST fellowship from the Department of Science and Technology (DST), New Delhi, India. This work was also supported in part by grants from NSF (Career No. 0093085), SRC (Grant No. 00541), and GSRC (MARCO/DARPA).

## References

- [1] Berkeley predictive technology model. <http://www-device.eecs.berkeley.edu/~ptm/interconnect.html>
- [2] SPEC CPU2000 Benchmark. <http://www.spec.org>
- [3] R. Arunachalam, E. Acar, and S.R. Nassif. Optimal shielding/spacing metrics for low power design. In *Proceedings of IEEE Computer Society Annual Symposium on VLSI*, pages 167-172, 2003.
- [4] D.C. Burger and T.M. Austin. The SimpleScalar tool-set, version 2.0, Technical Report 1342, Department of Computer Science, UW, 1997.
- [5] F. Caignet, S. Delmas-Bendhia, and E. Sicard. "The Challenge of Signal Integrity in Deep-submicrometer CMOS Technology". *IEEE*, 89(4), 2001, pp. 556-573.
- [6] C. Duan, A. Tirumala, and S.P. Khatri. "Analysis and Avoidance of Crosstalk in On-chip Buses". In *Hot Interconnects*, 2001, pp. 133-138.
- [7] 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". In *VLSI Design*, 2005, pp. 751-756.
- [8] K. Kim, K. Baek, N. Shanbhag, C. Liu, and S. Kang. "Coupling-driven Signal Encoding Scheme for Low-power Interface Design". In *ICCAD*, 2000, pp. 318-321.
- [9] L. Li, N. Vijaykrishnan, M. Kandemir, and M.J. Irwin. "A Crosstalk Aware Interconnect with Variable Cycle Transmission". In *DATE*, 2004, pp. 102-107.
- [10] C.-G. Lyuh and T. Kim. "Low Power Bus Encoding with Crosstalk Delay Elimination". In *ASIC/SOC*, 2002, pp. 389-393.
- [11] K.N. Patel and I.L. Markov. "Error-correction and Crosstalk Avoidance in DSM Buses". In *SLIP*, 2003, pp. 9-14.
- [12] P. Sotiriadis and A. Chandrakasan. "Low Power Bus Coding Techniques Considering Inter-wire Capacitances". In *CICC*, 2000, pp. 507-510.
- [13] P. Sotiriadis and A. Chandrakasan. "Bus Energy Minimization by Transition Pattern Coding (TPC) in Deep Sub-micron Technologies". In *ICCAD*, 2000, pp. 322-328.
- [14] P. Sotiriadis and A. Chandrakasan. "Reducing Bus Delay in Sub-micron Technology using Coding". In *ASPAC*, 2001, pp. 109-114.
- [15] P. Sotiriadis and A. Chandrakasan. "A Bus Energy Model for Deep Submicron Technology". *TVLSI*, 10(3), 2002, pp. 341-350.
- [16] S.R. Sridhara, A. Ahmed, and N.R. Shanbhag. "Area and Energy-efficient Crosstalk Avoidance Codes for On-chip Buses". In *ICCD*, 2004, pp. 12-17.
- [17] D. Sylvester and C. Hu. "Analytical Modeling and Characterization of Deep-submicrometer Interconnect". *IEEE*, 89(5), 2001, pp. 634-664.
- [18] B. Victor and K. Keutzer. "Bus Encoding to Prevent Crosstalk Delay". In *ICCAD*, 2001, pp. 57-63.
- [19] J. Yim and C. Kung. "Reducing Cross-coupling among Interconnect Wires in Deep-submicron Datapath Design". In *DAC*, 1999, pp. 485-490.
- [20] Y. Zhang, J. Lach, K. Skandron, and M.R. Stan. "Odd/even Bus Invert with Two-phase Transfer for Buses with Coupling". In *ISLPED*, 2002, pp. 80-83.