IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735.Volume 8, Issue 3 (Nov. - Dec. 2013), PP 09-15
www.iosrjournals.org

Fibonacci Codes for Crosstalk Avoidance

[1]Sireesha Kondapalli, [2]Dr. Giri Babu Kande

[1]PG Student (M.Tech VLSI), Dept. Of ECE, Vasireddy Venkatadri Ins. Tech., Nambur, Guntur, AP, India
[2] Professor & Head, Dept. Of ECE, Vasireddy Venkatadri Ins. Tech., Nambur, Guntur, AP, India

Abstract: *In the deep sub micrometer CMOS process technology, the interconnect resistance, length, and inter-wire capacitance are increasing significantly, which contribute to large on-chip interconnect propagation delay. Data transmitted over interconnect determine the propagation delay and the delay is very significant when adjacent wires are transitioning in opposite directions (i.e., crosstalk transitions) as compared to transitioning in the same direction. Propagation delay across long on-chip buses is significant when adjacent wires are transitioning in opposite direction (i.e., crosstalk transitions) as compared to transitioning in the same direction. By exploiting Fibonacci number system, we propose a family of Fibonacci coding techniques for crosstalk avoidance, relate them to some of the existing crosstalk avoidance techniques, and show how the encoding logic of one technique can be modified to generate code words of the other technique.*
Keywords: *On-chip bus, crosstalk, Fibonacci coding.*

I. Introduction

The advancement of very large scale integration (VLSI) technologies has been following Moore's law for the past several decades: the number of transistors on an integrated circuit is doubling every two years and the channel length is scaling at the rate of 0.7/3 years. It was not long ago when VLSI design marched into the realm of Deep Submicron (DSM) processes, where the minimum feature size is well below 1 μm. These advanced processes enable designers to implement faster, bigger and more complex designs. With the increase in complexity, System on Chip (SoC), Network on Chip (NoC) and Chip-level Multiprocessing (CMP) based products are now readily available commercially.

Some major challenges in DSM technologies include design productivity, manufacturability, power consumption, dissipation and interconnect delay. High design cost and long turn-around time are often caused by the growth in design complexity. A high design complexity results from a growth in transistor count and speed, demand for increasing functionality, low cost requirements, short time-to market and the increasing integration of embedded analog circuits and memories. Poor manufacturability is often a direct result of reduction in feature size. As the feature size gets smaller, the design becomes very sensitive to process variation, which greatly affects yield, reliability and testability. To address these issues, new design flows and methodologies are implemented to improve the efficiency of the designs. IC foundries are adding more design rules to improve the design robustness. For many high densities, a high speed DSM design, power consumption is a major concern. Increasing transistor counts, chip speed, and greater device leakage are driving up both dynamic and static power consumption.

In the meanwhile, however, DSM technologies also present new challenges to designers on many different fronts such as (i) scale and complexity of design, verification and test; (ii) circuit modeling and (iii) processing and manufacturability. Innovative approaches are needed at both the system level and the chip level to address these challenges and mitigate the negative effects they bring.

II. On-Chip Crosstalk Avoidance

Capacitive crosstalk has become a major determinant of the total power consumption and delay of on-chip busses. Figure.1 illustrates a simplified on-chip bus model with crosstalk. In the figure, CL denotes the load capacitance, which includes the receiver gate capacitance and also the parasitic wire-to-substrate parasitic capacitance. CI is the inter-wire coupling capacitance between adjacent signal lines of the bus. In practice, this bus structure is typically modeled as a distributed RC network, which includes the non-zero resistance of the wire as well It has been shown that for DSM processes, *CI* is much greater than *CL* [61]. Based on the energy consumption and delay models given in the bus energy consumption can be derived as a function of the total crosstalk over the entire bus. The worst case delay, which determines the maximum speed of the bus, is limited by the maximum crosstalk that any wire in the bus incurs. It has been shown that reducing the crosstalk boosts the bus performance significantly different approaches have been proposed for crosstalk reduction in the context of bus interconnects. Some schemes focus on reducing the energy consumption, some focus on minimizing the delay and other schemes address both.

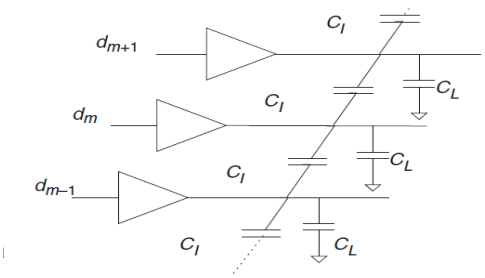


Figure 1 simplified on-chip bus model with crosstalk

The simplest approach to address the inter-wire crosstalk problem is to shield each signal using grounded conductors. Khatri et al. in [1, 2] proposed a layout fabric that alternatively inserts one ground wire and one power wire between every signal wire, i.e., the wires are laid out as . . . *VSGSVSGSVS* . . . , where *S* denotes a signal wire, *G* denotes a ground wire and *V* denotes a power wire. Any signal wire has a static (V or G) wire on each side, and hence, when it switches, it needs to charge a capacitance of value of 2*CI* .

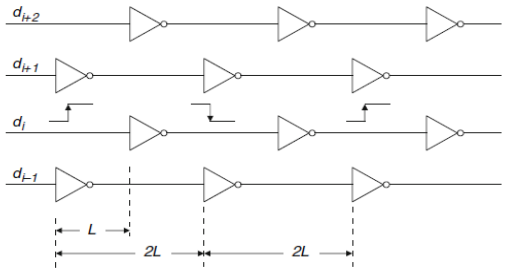


Figure 2 A Fabric Based Design

This fabric also enforces a design rule that metal wires on a given layer run perpendicular to wires on layers above or below. The fabric has the advantage of improved predictability in parasitic capacitance and inductance. It automatically provides a low resistance power and ground network as well. Such a fabric results in a decrease in wiring density. Even though the fabric appears to require a large area overhead, experimental results show that on average, the circuit size grows by only 3% if the circuit is implemented as a network of medium-sized programmable logic arrays (PLAs). In theworst case, however, the circuit size can be more than 200%of the conventional layout. Figure 2 illustrates such a fabric-based design. Khatri et al. [1] also provides discussions on wire removal in networks of Programmable Logic Arrays (PLA)

Crosstalk avoidance bus encoding techniques manipulate the input data before transmitting them on the bus. Bus encoding can eliminate certain undesirable data patterns and thereby reduce or eliminate crosstalk, with much lower area overhead than the aforementioned straightforward shielding techniques [36, 88, 90, 99, 100]. These types of codes are referred to as crosstalk avoidance codes (CACs) or self shielding codes. Depends on their memory requirements, CACs can be further divided into two categories: memory less codes and memory-based codes. Memory based coding approaches generate a codeword based on the previously transmitted code and the current data word to be transmitted [34, 100]. On the receiver side, data is recovered based on the received code words from the current and previous cycles. The memory less coding approaches use a fixed code book to generate code words to transmit. The codeword is solely dependent on the input data. The decoder in the receiver uses the current received codeword as the only input in order to recover the data.

Among all the memory less CACs proposed, two types of codes have been heavily studied. The first is called forbidden pattern free (FPF) code and the second type of code has the property that between any two adjacent wires in the bus, there will be no transition in opposite directions in the same clock cycle. Different names have been used in the literatures for the second type of codes. In this paper, we refer these codes as forbidden transition free (FTF) CACs. Both FPF-CACs and FTF-CACs yield the same degree of delay reduction as passive shielding while requiring much less area overhead. Theoretically, the FPF-CAC has slightly better overhead performance that the FTF-CAC. In practice, for large size bus, this difference is negligible.

III. Fibonacci Binary Numeral System

A numeral system is *"a framework where numbers are represented by numerals in a consistent manner"* [5]. The most commonly used numeral system in digital design is the binary numeral system, which uses powers of two as the *basis*. For a number *v*, its binary representation is defined in Eq. 1. The binary numeral system is *complete* and *unambiguous*, which means that each number has one and only one representation in the binary numeral system.

Equation 1

$$v = \sum_{k=1}^{n} b_k \cdot 2^{k-1} \qquad b_k \in \{0,1\}$$

$$= \sum_{k=1}^{m} d_k \cdot f_k \qquad d_k \in \{0,1\}$$

The Fibonacci-based numeral system $N(Fm, \{0, 1\})$ is the numeral system that uses Fibonacci sequence as the basis. The definition of the Fibonacci sequence [3] is given in Eq. 2. A number v is represented as the summation of some Fibonacci numbers, and no Fibonacci number is in the summation more that once, as indicated in Eq. 2.

Equation 2

$$f_m = \begin{cases} 0 & \text{if } m = 0, \\ 1 & \text{if } m = 1, \\ f_{m-1} + f_{m-2} & \text{if } m \geq 2. \end{cases}$$

Similar to the binary numeral system, the Fibonacci-based numeral system is complete, and therefore any number v can be represented in this system. However, the Fibonacci-based numeral system is *ambiguous.* As an example, there are *six* 7-digit vectors in the Fibonacci numeral system for the decimal number 19: {0111101, 0111110, 1001101, 1001110, 1010001, 1010010}. For clarity, we refer to a vector in the binary numeral system as a *binary vector* or *binary code*; a vector in the Fibonacci numeral system is referred to as a *Fibonacci vector* or *Fibonacci code*. All the Fibonacci vectors that represent the same value are defined as *equivalent vectors.*

The n-bit binary vector can represent numbers in the range of $[0, 2n-1]$, and therefore a total of $2n$ values can be represented by n-bit binary vectors. we know that the range of an m-bit Fibonacci vector is $[0, fm+2-1]$, where the minimum value 0 corresponds to all the bits dk being 0, and the maximum value corresponds to all dk being 1. Hence a total of $fm+2$ distinct values can be represented by m-bit Fibonacci vectors.

We first propose a coding scheme that converts the input data to a forbidden pattern free Fibonacci vector. The code is near-optimal since the required overhead is no more than 1 additional bit, compared to the theoretical lower bound given The coding algorithm is developed based on a result that states that any number v can be represented in FNS, in an FPF manner. In order to prove this result, we first derive the following corollaries:

The following twom-bitFibonacci vectors are equivalent: dmdm−1 · · · *d*301 *and dmdm*−1 · · · *d*310. *In other words, dmdm*−1 · · · *d*301 ≡ *dmdm*−1 · · · *d*310.

Proof Since *f*2 = *f*1 = 1, it is obvious that the last two digits are interchangeable

```
\\ MSB stage:
if v ≥ f_{m+1} then
    d_m = 1;
    r_m = v − f_m;
else
    d_m = 0;
    r_m = v;
end if
\\ other stages:
for k = m-1 to 2 do
    if r_{k+1} ≥ f_{k+1} then
        d_k = 1;
    else if r_{k+1} < f_k then
        d_k = 0;
    else
        d_k = d_{k+1};
    end if
    r_k = r_{k+1} − f_k · d_k;
end for
\\LSB
d_1 = r_2;
return (d_m d_{m−1} - - - d_1);
```

Figure 3 CPF encoding algorithm

Input: d;
  $r_m = d$;
  for $k = m-1$ to 1 **do**
    if $r_{k+1} < f_k$ **then**
      $c_k = 0$;
    else
      $c_k = 1$;
    end if
    $r_k = r_{k+1} - f_k \cdot c_k$;
  end for
  $c_0 = r_1$;
Output: $c_{m-1}c_{m-2}\cdots c_0$;

Fig Nff encoding algorithm

Input: d;
  $r_m = d$;
  for $k = m-1$ to 1 **do**
    if $r_{k+1} < f_{2\lceil \frac{k-1}{2} \rceil}$ **then**
      $c_k = 0$;
    else
      $c_k = 1$;
    end if
    $r_k = r_{k+1} - f_{k-1} \cdot c_k$;
  end for
  $c_0 = r_1$;
Output: $c_{m-1}c_{m-2}\cdots c_0$;

Fig CRF encoding process

Figure. 3 show that an *m*-bit FPF vector is generated in *m* stages. Each stage outputs one bit of the output vector (*dk*) and the remainder (*rk*) that is the input to the following stage. In the *k*th stage, the input *rk*+1 is compared to two Fibonacci numbers *fk*+1 and *fk*. If *rk*+1 ≥ *fk*+1, *dk* is coded as 1; If *rk*+1 < *fk* , *dk* is coded as 0; If the value *rk*+1 is in between, *dk* is coded to the same value as *dk*+1. The remainder is computed as *rk*+1 − *dk* · *fk* . We shall refer the ranges [*fk*+1, *fk*+2), (*fk* , *fk*+1) and [0, *fk*) as the ***force-1 zone***, ***gray zone*** and ***force-0 zone*** of the *k*th stage respectively. The most significant bit (MSB) stage is slightly different from other stages since no bit proceeds it. It encodes *dm* by comparing the input *v* with only one Fibonacci number, *fm*+1.

The decoder is a straightforward implementation of Eq. 2 which converts the Fibonacci vector back to the binary vector The correctness of Algorithm can be proven by showing that if after the *k*th stage, the partially generated output vector *dm* · · · *dk*+1*dk* is FPF, then adding the output of the (*k* − 1)th stage, *dk*−1 will not introduce a forbidden pattern

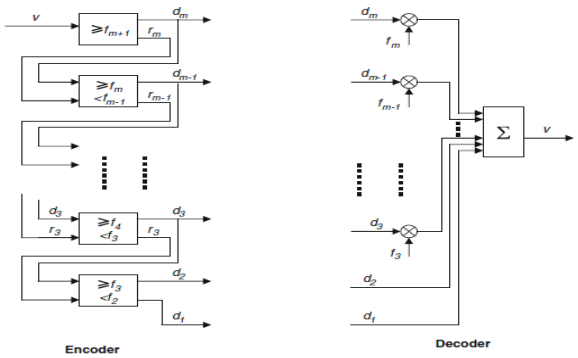


Figure 4 Encoder and Decoder Implementation

Table : 7-bit near-optimal FPF code books

Input Decimal value	CODE-1 f_6 8	f_5 5	f_4 3	f_3 2	f_2 1	f_1 1	CODE-2 f_6 8	f_5 5	f_4 3	f_3 2	f_2 1	f_1 1
20*	1	1	1	1	1	1						
19*	1	1	1	1	1	0						
18*	1	1	1	1	0	0						
17*	1	1	1	0	0	1						
16*	1	1	1	0	0	0						
15	1	1	0	0	1	1						
14	1	1	0	0	0	1						
13	1	1	0	0	0	0						
12	0	1	1	1	1	1	1	0	0	1	1	1
11	0	1	1	1	1	0	1	0	0	1	1	0
10	0	1	1	1	0	0	1	0	0	0	1	1
9	0	1	1	0	0	1	1	0	0	0	0	1
8	0	1	1	0	0	0	1	0	0	0	0	0
7	0	0	1	1	1	1						
6	0	0	1	1	1	0						
5	0	0	1	1	0	0						
4	0	0	0	1	1	1						
3	0	0	0	1	1	0						
2	0	0	0	0	1	1						
1	0	0	0	0	0	1						
0	0	0	0	0	0	0						

data-word	Fibonacci codeword $\mathcal{NFF}_4$				$\mathcal{RF}_4$				$\mathcal{CRF}_4$			
4 2 1	5	3	2	1	3	2	1	1	3	2	1	1
0 0 0	0	0	0	0	0	0	0	0	0	0	0	0
0 0 1	0	0	0	1	0	0	0	1	0	0	1	0
0 1 0	0	0	1	0	0	1	0	0	0	0	1	1
0 1 1	0	1	0	0	0	1	0	1	1	0	0	0
1 0 0	0	1	0	1	0	1	1	1	1	0	1	0
1 0 1	1	0	0	0	1	1	0	0	1	0	1	1
1 1 0	1	0	0	1	1	1	0	1	1	1	1	0
1 1 1	1	0	1	0	1	1	1	1	1	1	1	1

Figure 5 FPF Code book & all implementations code book

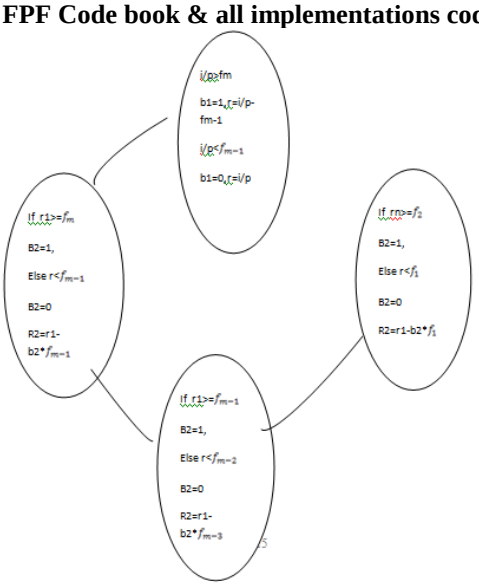


Figure 6 FSM Chart Implementation

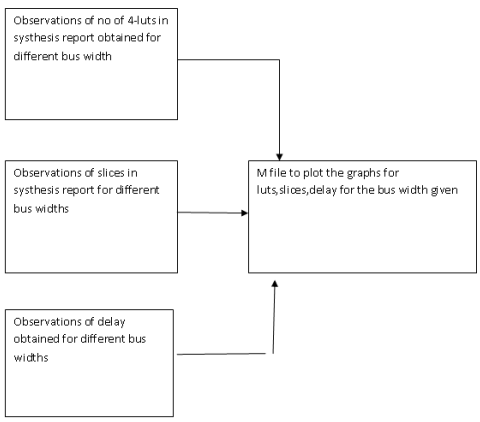


Figure 7 Matlab Design

IV. Results and Conclusions

We proposed a family of Fibonacci coding techniques for crosstalk avoidance by exploiting Fibonacci number system. Figure 8 & 9 Show the Simulated Waveform and RTL schematic of the proposed FTF Codec.
Crosstalk avoidance codes are shown to be able to reduce the inter-wire crosstalk and therefore boost the maximum speed on the data bus. They have the advantage of consuming less area overhead than shielding techniques. Even though several different types of codes have been proposed in the past few years, no mapping scheme was given which facilitates the CODEC implementation. Compounded by the nonlinear nature of the CAC, the lack of a solution to the systematic construction of the CODEC has hampered the wide use of CAC in practice. In this paper, we give what we believe is the first solution to this problem. We showed that data can be coded to a forbidden pattern free vector in the Fibonacci numeral system. We first give a straightforward mapping algorithm that produces a set of FPF codes with near-optimal cardinality. The area overhead of this coding scheme is near the theoretical lower bound. The CODEC based on this coding scheme is systematic and has very low complexity. The size of the CODEC grows quadratically with the data bus size as opposed to exponentially in a brute forced implementation. Our systemic coding scheme allows the code design of arbitrarily large busses without having to resort to bus partitioning

We showed the inter-dependency among the proposed techniques and provided a formal procedure to convert a codeword set into another codeword set. We also related our proposed techniques with some of the existing crosstalk avoidance coding techniques. The proposed techniques eliminate crosstalk completely, but not inductance. The worst-case inductance occurs when adjacent lines transition in the same direction. We plan to come up with a suitable mechanism to minimize the inductance effects using Fibonacci codes in future.
We can further propose an improved coding scheme which yields a set of FPF codes with maximum cardinality. The area overhead of this optimal coding scheme matches the theoretical lower bound. We gave the corresponding modification in the CODEC design as well. This paper also discusses issues associated with CODEC implementations. We proposed a modified coding scheme that eliminates the MSB stage in the encoder and simplifies the decoder side as well. The modification reduces the total gate count and improves the CODEC speed.

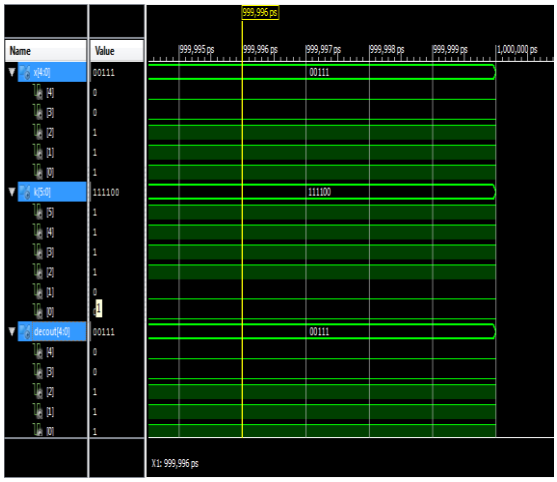
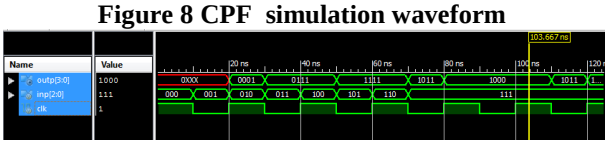


Figure 8 CPF simulation waveform



Nff simulation waveform

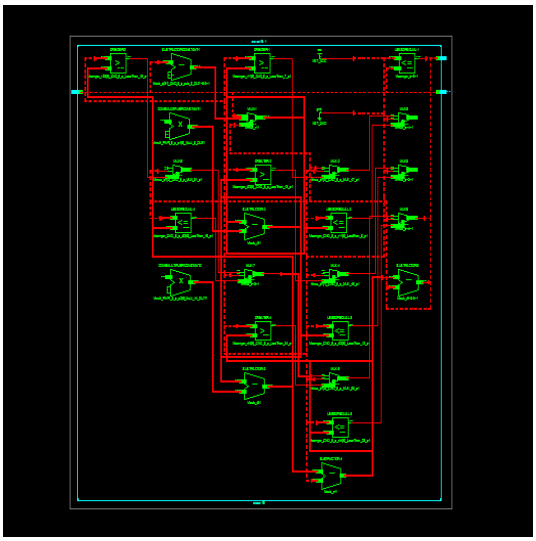


Figure 9 RTL Schematic of the Designed Model

Acknowledgements

The authors would like to thank the anonymous reviewers for their comments which were very helpful in improving the quality and presentation of this paper.

References

[1] S.P. Khatri "Cross-talk Noise Immune VLSI Design using Regular Layout Fabrics", PhD thesis, University of California at Berkeley, Berkeley, California, 1999.

[2] S.P. Khatri, A. Mehrotra, R.K. Brayton, Ralf H.J.M. Otten, and A.L. Sangiovanni-Vincentelli. "A novel vlsi layout fabric for deep sub-micron applications", In Proceedings of Design Automation Conference, pages 491–496. IEEE, 1999.

[3] F. Caignet, S. Delmas-Bendhia, and E. Sicard, "The challenge of signal integrity in deep-submicrometer CMOS technology," Proc. IEEE, vol. 89, no. 4, pp. 556–573, Apr. 2001.

[4] D. Pamunuwa, L.-R. Zheng, and H. Tenhunen, "Maximizing throughput over parallel wire structures in the deep submicrometer regime," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 11, no. 2, pp. 224–243, Apr. 2003.

[5] R. Arunachalam, E. Acar, and S. Nassif, "Optimal shielding/spacing metrics for low power design," in Proc. IEEE Comput. Soc. Annu. Symp. VLSI, 2003, pp. 167–172.

[6] B. Victor and K. Keutzer, "Bus encoding to prevent crosstalk delay," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design, 2001, pp. 57–63.

[7] C. Duan, A. Tirumala, and S. Khatri, "Analysis and avoidance of crosstalk in on-chip buses," in Proc. Hot Interconnects, 2001, pp. 133–138.

[8] P. Subramanya, R. Manimeghalai, V. Kamakoti, and M. Mutyam, "A bus encoding technique for power and cross-talk minimization," in Proc. IEEE Int. Conf. VLSI Design, 2004, pp. 443–448.

[9] M. Stan and W. Burleson, "Limited-weight codes for low power I/O," in Proc. IEEE/ACM Int. Workshop Low Power Design, 1994, pp. 209–214.

[10] M. Mutyam, "Preventing crosstalk delay using Fibonacci representation," in Proc. IEEE Int. Conf. VLSI Design, 2004, pp. 685–688.

[11] C. Duan, C. Zhu, and S. Khatri, "Forbidden transition free crosstalk avoidance codec design," in Proc. Design Autom. Conf., 2008, pp. 986–991.

[12] C. Duan, V. C. Calle, and S. Khatri, "Efficient on-chip crosstalk avoidance codec design," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 17, no. 4, pp. 551–560, Apr. 2009.