International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-1 Issue-5, June 2012

High Speed Codec Design for Crosstalk Avoidance

Sowjanya Sunkara, T.Ravi Sekhar

Abstract*— The cross talk is dependent on the data transition patterns on the bus, patterns can be classified based on the severity of the crosstalk they impose on the bus. The general idea behind techniques that improve on-chip bus speed is to remove undesirable patterns that are associated with certain classes of crosstalk. Different schemes incur different area overheads since they requires additional wires, spacing between wires or both. We analyze the properties of the FPF-CAC and show that mathematically, a mapping scheme exists between the data words and code words. Our proposed CODEC design offers a near-optimal area overhead performance. An improved version of the CODEC is then presented, which achieves theoretical optimal performance. We also investigate the implementation details of the CODECs, including design complexity and the speed. Optimization schemes are provided to reduce the size of the CODEC and improve its speed.

***Index Terms*—CODEC, FPF-CAC, pruning,shielding.**

I. INTRODUCTION

As VLSI technology has marched into the deep sub-micrometer (DSM) regime, new challenges are presented to circuit designers. As one of the key challenges, the performance of bus based interconnects has become a bottleneck to the overall system performance. In large designs [e.g., systems-on chip (SoCs)] where long and wide global busses are used, interconnect delays often dominate logic delays. Once negligible, crosstalk has become a major determinant of the total power consumption and delay of on-chip busses. The impact of crosstalk in on-chip busses has been studied as part of the effort to improve the power and speed characteristics of the on-chip bus interconnects. In this paper, we offer a systematic CODEC construction solution for the forbidden-pattern-free crosstalk avoidance code (FPF-CAC). The mapping scheme we propose is based on the representation of numbers in the binary numeral system. We show that all data words can be coded to code words using encoder. We propose several different coding schemes that allow the CODECs to be constructed *for any arbitrary bus size*. With such a systematic mapping, the CODEC for a wider bus is constructed by a simple extension of the CODEC for a smaller bus. The first CODEC proposed in the paper is proven to have near-optimal area overhead performance. We further offer an improved coding scheme that achieves optimal overhead performance. We also propose modifications can be made to our near-optimal CODEC that will reduce the complexity and improve the delay performance of the CODEC. The theoretical lower bound of the area overhead for memory-based codes is lower compared to memory-less codes [1]. However, the memory-based CODECs are much more complex and the only known codeword generation method is an exhaustive search and pruning-based method. Several different types of memory-less CACs have been proposed. The code designs are discussed in [3]–[6]. These codes offer the same degree of delay reduction as the passive shielding technique, with much less area overhead (ranging from 44% to 62.5%). Unfortunately, none of the referred papers addresses the mapping between data words and code words for the CODECs. So far, all the CODEC design approaches are based on bus partitioning (which breaks a big bus into a number of small groups (lanes) and applies CAC coding on each group independently). Such an approach has to deal with the cross talk across the group boundaries. Several different schemes are proposed to handle this inter-group cross talk, such as group inversion and bit overlapping [4], [5]. In all cases, more wires are needed and therefore the overall area overhead is higher than the theoretical lower bound. due to this cornstalk problems like delay power dissipation, leakage currents will coming in to picture and leads to failure of the chip.

Manuscript published on 30 June 2012.
* Correspondence Author (s)
Sowjanya Sunkara*, pursuing MTECH,VLSI, ECE Department, Sree vidyanikethan Engineering college, Tirupathi, India.
T.RaviSekhar, Assistant professor, ECE Department, Sree vidyanikethan Engineering college, Tirupathi, India.

II. RELATED WORK

As the crosstalk is dependent on the data transition patterns on the bus, patterns can be classified based on the severity of the crosstalk they impose on the bus. Crosstalk patterns are classified as *0C*, *1C*, *2C*, *3C*, and *4C* patterns, respectively, as shown in Table I.

TABLE I
CLASSES OF CROSSTALK

Class	C_{eff}	Transition patterns
0C	C_L	$000 \rightarrow 111$
1C	$C_L(1+\lambda)$	$011 \rightarrow 000$
2C	$C_L(1+2\lambda)$	$010 \rightarrow 000$
3C	$C_L(1+3\lambda)$	$010 \rightarrow 100$
4C	$C_L(1+4\lambda)$	$010 \rightarrow 101$

Several methods have been proposed to eliminate/reduce crosstalk delay. Some of these methods are based on routing strategies [7, 9, 10, 11, 13], which employ various routing techniques to minimize crosstalk delay within a data-path or logic block. Repeater insertion techniques are used to reduce capacitance and resistance of long interconnections for decreasing wire delay [2, 12].

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

But these techniques cannot solve the problem of simultaneous transitions for opposite directions. In order to overcome this problem, a bus delay reduction technique was proposed in [8]. The main idea is to prevent simultaneous opposite transitions by skewing signal transition timing of adjacent wires. A simple technique to eliminate crosstalk delay is by placing a shielding wire between every adjacent wire. But this technique has a drawback of doubling the wiring area. Abstracted from the concept of shielding, Victor *et al.* have proposed a bus encoding technique [14] and gave *self-shielding codes* with/without memory to prevent crosstalk delay. This technique was proved theoretically far better than just placing
shielding wire between every adjacent wire. They showed that a 32-bit bus can be encoded with 40 wires using self shielding codes with memory or 46 wires with memory less self-shielding codes. Though this technique is theoretically sound, there is no simple approach to generate the self-shielding codes.

The self shielding codes can be further divided into two categories: memory-less and memory-based. The memory-based coding approaches generate a codeword based on the previously transmitted code and the current data word to be transmitted. On the receiver side, the data is recovered based on the received codeword's from the current and previous cycles. The memory-less coding approaches use a fixed code book to generate a codeword to transmit, solely based on the input data. The corresponding receiver decoder uses the current received codeword as the only input to recover the data. However, the memory-based CODECs are much more complex.

The memory less *forbidden pattern*-based crosstalk avoidance code was first proposed in [5]. The *forbidden patterns* are defined as 3-bit patterns "101" and "010". A code is ***forbidden pattern free***(FPF) if there is no forbidden pattern in any three consecutive bits. As examples, 1101110 is not forbidden pattern free; 1100110 is FPF. It has been shown in [5] that for a code that contains only FPF code words, the bus that transmits only these code words will experience maximum crosstalk of no greater than *2 .C*. Therefore, by encoding the data words to FPF code words, we can speed up the bus by 100%. This type of code is referred herein as FPF-CAC. The FPF-CAC can be generated using an inductive procedure [5]. The following is the inductive procedure that generates FPF code words, where "." is the concatenation operator.

Algorithm 1 FPF codeword generation

$S_2 = \{00, 01, 10, 11\}$
for $m \geq 3$ **do**
  $S_m = \{\}$;
  for $\forall V^i_{m-1} \in S_{m-1}$ **do**
    if $b^i_{m-1}b^i_{m-2} = 00$ or 11 **then**
      add $0 \cdot V^i_{m-1}$ and $1 \cdot V^i_{m-1}$ to S_m;
    else if $b^i_{m-1}b^i_{m-2} = 01$ or 10 **then**
      add $b^i_{m-1} \cdot V^i_{m-1}$ to S_m;
    end if
  end for
end for

Table II lists the code words of the 3-, 4-, and 5-bit FPF-CACs generated by Algorithm 1.

TABLE II
FPF-CAC CODEWORDS FOR 2-, 3-, 4-, AND 5 BIT BUSSES

2-bit	3-bits	4-bits	5 bit	
00	000	0000	00000	10000
01	001	0001	00001	10001
10	011	0011	00011	10011
11	100	0110	00110	11000
	110	0111	00111	11001
	111	1000	01100	11100
		1001	01110	11110
		1100	01111	11111
		1110		
		1111		

In this work, we propose a new memory less bus encoding technique and give a recursive procedure to generate crosstalk delay free binary *FPF code words*.

III. PROPOSED SYSTEM

This paper focuses on the overcome of the limitations of FPF-CAC. The lack of practical CODEC construction schemes has hampered the use of such codes in practical designs. This work presents guidelines for the CODEC design of the "forbidden pattern free crosstalk avoidance code" (FPF-CAC). We analyze the properties of the FPF-CAC and show that mathematically, a mapping scheme exists. In this paper, we offer a systematic CODEC construction solution for the forbidden-pattern-free crosstalk avoidance code (FPF-CAC). The mapping scheme we propose is based on the recursive procedure . We propose several different coding schemes that allow the CODECs to be constructed for any arbitrary bus size. With such a systematic mapping, the CODEC for a wider bus is constructed by a simple extension of the CODEC for a smaller bus. The first CODEC proposed in the paper is proven to have near-optimal area overhead performance. We further offer an improved coding scheme that achieves optimal overhead performance.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

We also propose modifications to our near-optimal CODEC that will reduce the complexity and improve the delay performance of the CODEC. The key contributions of this paper include the following.

1) We define a deterministic mapping scheme for the FPF-CAC.
2) Based on the mapping scheme, we propose coding algorithms that allow systematic CODEC constructions so that the CODEC for a wider bus is obtained as an extension of the CODEC for smaller bus.
3) We show that the CODEC gate count grows quadratically with bus size as opposed to the exponential growth for the existing approaches.

The following figure(1) shows the proposed CODEC design that encodes the data word to be forbidden pattern free using encoder. After transmission through the on chip buses it is retrieved back from the code word using decoder on the receiver side.

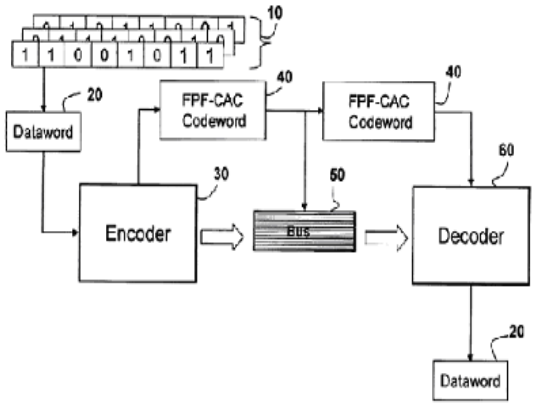

Fig1: Block diagram of our proposed system

IV. ADVANTAGES

A. FPF-CAC CODEC DESIGN:

As discussed in Section II, the 3C and 4C crosstalk classes can be avoided if the bus is encoded using the FPF code. We provided the recursive procedure for generating the code words and showed how to compute the total number of code words and the lower bound for the area overhead. However, the mapping scheme between the input data words and the output code words was not discussed, nor was it shown how a CODEC for the FPF-CAC can be constructed. Conceptually, the mapping between the data words and the code words is flexible, provided it can be reversed by the decoder. In the case when the size of the code book is not a power of two, a 1-to-1 mapping is not required. A 1-to-many mapping for certain data words may reduce the CODEC complexity further. When the data bus width is small, the CODEC can be implemented can do the mapping flexibility can be exploited to optimize the speed and/or the area of the CODEC. However, as the data bus width increases, the CODEC size grows exponentially.

V. SIMULATION RESULTS

Xilinx edition (MXE) tool will be used for simulation and results will be verified. Fig.2. Xilinx Synthesis technology (XST) will be used for FPGA synthesis. Fig.3. Timing analysis will be carried out to predict the maximum achievable clock speeds for chosen Xilinx spartan 3E FPGA device. The Xpower analysis for estimation of power in the new proposed circuit.

A. FINAL RESULTS:

Device utilization summary:

Selected Device : 3s400pq208-4

Number of Slices:	5	out of 3584	0%
Number of Slice Flip Flops:	3	out of 7168	0%
Number of 4 input LUTs:	10	out of 7168	0%
Number of bonded IOBs:	15	out of 141	10%
IOB Flip Flops: 6			
Number of GCLKs:	1	out of 8	12%

==========

TIMING REPORT

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
GENERATED AFTER PLACE-and-ROUTE.

Timing Summary:

Speed Grade: -4

Minimum period: 2.586ns (Maximum Frequency: 386.698MHz)
Minimum input arrival time before clock: 3.038ns
Maximum output required time after clock: 8.962ns
Maximum combinational path delay: 9.604ns

Timing Detail:

All values displayed in nanoseconds (ns)

==========

Timing constraint: Default period analysis for Clock 'clk'
Clock period: 2.586ns (frequency: 386.698MHz)
Total number of paths/destination ports:8/3

Delay: 2.586ns (Levels of Logic = 1)
Source: prs_state_FFd3 (FF)
Destination: prs_state_FFd3 (FF)
Source Clock: clk rising
Destination Clock: clk rising

Data Path: prs_state_FFd3 to prs_state_FFd3.Gate Net Cell:in->out fanout Delay Delay Logical Name (Net Name)

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

```
  FDC:C->Q      4         0.720         1.112   prs_state_FFd3
(prs_state_FFd3)
  LUT3_L:I1->LO 1     0.551     0.000   prs_state_FFd3-In1
(prs_state_FFd3-In)
  FDC:D 0.203         prs_state_FFd3
    ----------------------------------
 Total 2.586ns
 (1.474ns logic,1.112ns route)
 (57.0% logic, 43.0% route)

========================================
CPU : 6.51/8.78 s | Elapsed :7.00/9.00 s

Total memory usage is 111328 kilobytes

Number of errors   :    0 (   0 filtered)
Number of warnings :    0 (   0 filtered)
Number of infos    :    0 (   0 filtered)
```

B. TIMING REPORTS:

Data Sheet report:

All values displayed in nanoseconds (ns)

Setup/Hold to clock clk

Setup to Source	Hold to clk (edge)	clk (edge)	Internal Clock(s)	Phase
ED<0>	1.730(R)	0.641(R)	clk_BUFGP	0.000
ED<1>	1.762(R)	0.419(R)	clk_BUFGP	0.000
ED<2>	2.415(R)	0.057(R)	clk_BUFGP	0.000
ED<3>	2.852(R)	0.363(R)	clk_BUFGP	0.000
start_enc	2.177(R)	0.930(R)	clk_BUFGP	0.000

Clock clk to Pad

Destination	clk (edge) to PAD	Internal Clock(s)	Phase
do<0>	7.382(R)	clk_BUFGP	0.000
do<1>	7.382(R)	clk_BUFGP	0.000
do<2>	7.382(R)	clk_BUFGP	0.000
do<3>	7.382(R)	clk_BUFGP	0.000
do<4>	8.671(R)	clk_BUFGP	0.000
do<5>	7.382(R)	clk_BUFGP	0.000
enc_done	10.249(R)	clk_BUFGP	0.000

Clock to Setup on destination clock clk

SourceClock	Src:Rise|Dest:Rise	Src:Fall|Dest:Rise	Src:Rise|Dest:Fall	Src:Fall|Dest:Fall
clk	2.048			

C. ENCODER SIDE:

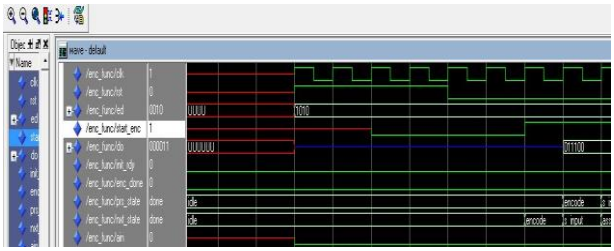

Fig.(2)

D. DECODER SIDE:

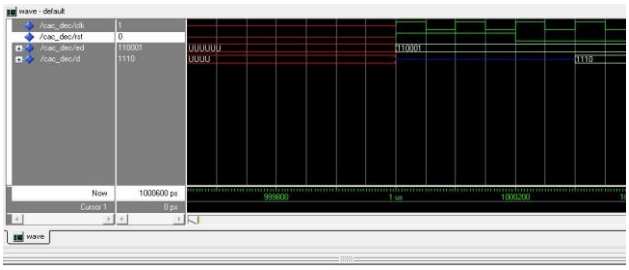

Fig.(3)

VI. CONCLUSION

The first solution to this problem is that we showed that data can be coded to a forbidden pattern free vector. 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. Our systemic coding scheme allows the code design of arbitrarily large busses without having to resort to bus partitioning. We further synthesized, and implemented the near-optimal CODEC in Xilinx and Modelsim. The reported results show that the size of our CODEC is several orders of magnitude lower than a previously reported design for a 12-bit bus. We also investigated the possibility of combining our approach with bus partitioning in very large busses to address the propagation delay issue as well as to reduce the total size of the CODEC. We compared the average bus energy consumption of un coded and FPF coded busses. Our experimental results show that FPF coding offers on average 20% power saving. Even though this work is strictly limited to one class of crosstalk avoidance code (the FPF-CAC), we believe that the approach can be easily adapted to other varieties of crosstalk avoidance codes as well.

REFERENCES

[1] Duane .C, Cordero. V and Khatri. S. P. “Efficient On-Chip Crosstalk Avoidance CODEC Design”, IEEE Transactions on VLSI Systems, April 2009, pp 551 – 560.

[2] H.B. Bakoglu, *Circuits, Interconnections and Packaging for VLSI*, Addison-Wesley, 1990.

[3] Madhu Mutyam, ”Preventing Crosstalk Delay using Fibonacci Representation”, Intl Conf. on VLSI Design, 2004, pp 685-688.

[4] S.R. Sridhara, A. Ahmed, and N. R. Shanbhag, ”Area and Energy-Efficient Crosstalk Avoidance Codes for On-Chip busses”, Proc. of ICCD,2004, pp 12-17.

[5] C. Duan, A. Tirumala and S. P. Khatri, “Analysis and Avoidance of Crosstalk in On-chip Bus”, HotInterconnects, 2001,pp 133-138.

[6] Bret Victor and K. Keutzer,"Bus Encoding to Prevent Crosstalk Delay",ICCAD, 2001, pp 57-63.
[7] T. Gao and C.L. Liu, "Minimum Crosstalk Channel Routing," *IEEE Transactions on Computer-Aided Design*, vol. 15, no. 5, pp. 465-474, 1996.
[8] K. Hirose and H. Yasuura, "A Bus Delay Reduction Technique Considering Crosstalk," *Proceedings. Design, Automation and Test in Europe Conference and Exhibition 2000*, pp. 441-445, 2000.
[9] H.-P. Tseng, L. Scheffer, and C. Sechen, "Timing and crosstalk Driven Area Routing," *Proceedings. 35th Design Automation Conference*, pp. 378-381, 1998.
[10] A. Vittal and M. Marek-Sadowska, "Crosstalk Reduction VLSI," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, no., pp. 290-298, 1997.
[11] T. Xue, E. Kuh, and D. Wang, "Post Global Routing Crosstalk Synthesis," *IEEE Transactions on Computer Aided Design*, vol. 16, no. 12, pp. 1418-1430, 1997.
[12] T. Sakurai, "Closed-Form Expressions for Interconnection Delay Coupling, and Crosstalk in VLSI's," IEEE *Transactions on Electron Devices*, vol. 40, no. 12, pp. 118-124, 1993.
[13] H. Zhou and D.F. Wong, "Global Routing with Crosstalk Constraints," *Proceedings. 35th Design Automation Conference*, pp. 374-377, 1998.
[14] B. Victor and K. Keutzer, "Bus Encoding to Prevent Crosstalk Delay," *Proceedings. ICCAD-2001 International Conference on Computer-Aided Design 2001*, pp.57-64, 2001.
[15] K. Kim, K. Baek, N. Shanbhag, C. Liu, and S.-M.Kang, Coupling driven signal encoding scheme for low-power interface design, Proc. Of IEEE/ACM International Conference on Computer-Aided Design, Nov 2000.
[16] H. Kaul, D. Sylvester and D. Blauuw, "Active shielding of RLC global interconnects", Proc. of the 8th ACM/IEEE international workshop on Timing issues in the specification and synthesis of digital systems, 2002, pp 98-104.
[17] D. Li, A. Pua, P. Srivastava, and U. Ko, "A Repeater Optimization Methodology for Deep Sub- Micron, High-Performance Processors," *Proceedings. International Conference on Computer Design, VLSI in Computers and Processors*, pp. 726-731, 1997.
[18] A.B. Kahng, S. Muddu, E. Sarto, and R. Sharma, "Interconnect Tuning Strategies for High-Performance ICs,"*Proceedings. Design, Automation and Test in Europe Conference and Exhibition 1998*, pp. 471-478, 1998.
[19] A. Apostolico and A.S. Fraenkel, "Robust Transmission of Unbounded Strings Using Fibonacci Representations," *IEEE Transactions on Information Theory*, IT- 33, pp. 238-245, 1987.
[20] H.B. Bakoglu, *Circuits, Interconnections and Packaging for VLSI*, Addison-Wesley, 1990.
[21] A.S. Fraenkel, "Systems of Numeration," *The American Mathematical Monthly*, vol. 92, pp. 105-114, 1985.

AUTHORS

Ms.S.Sowjanya, Student, is currently Pursuing her M.Tech VLSI., in ECE department of Sree Vidyanikethan Engineering College, Tirupati. She has completed graduation in Electronics and Communication Engineering, from G.Pulla Reddy engineering college, Kurnool . Her research areas are VLSI, Digital IC Design, and ASIC Design.

Mr.T.Ravisekhar, M.Tech., is currently working as an Assistant Professor in ECE department of Sree Vidyanikethan Engineering College, Tirupati. He has completed M.Tech in VLSI Design, in Satyabhama University. His research areas are Low power VLSI Design, & VLSI Signal Processing.