ISSN: 0974-2115
www.jchps.com Journal of Chemical and Pharmaceutical Sciences

Design and implementation of area efficient embedded transition inversion coding for serial links

G.Sankar Babu[1], M.Anto Bennet[1*], G.Janakiraman[2], R.Kaushik Krishna[1], S.Jaya prakash[1], B.S.Jayavignesh[1]

[1]Department of Electronics and Communication Engineering VEL TECH, Chennai - 600062
[2]Department of Automobile Engineering, Dr.Mahalingam College of Engg & Technology
***Corresponding Author:E.Mail:bennetmab@gmail.com**

ABSTRACT

In all type of electronic circuits, reducing the power dissipation is one of the major topic of interest. The dynamic power dissipation contributes a significant fraction in the overall power dissipation in the VLSI interconnect. The proposed work in this paper is to reduce the number of bus lines of the conventional parallel bus by multiplexing onto a single line. The advantage of serial link interconnection is its reduced area and crosstalk. But it increases power dissipation and switching activity. Hence, the main target is to minimize the switching activity on the serial link. The proposed Embedded Transition Inversion Coding method uses an efficient encoding and decoding technique in order to minimize the transition activity and power consumption in serial links. This coding scheme is to solve the issue of the extra indication bit. This scheme eliminates the need of sending an extra bit by embedding the inversion information in the phase difference between the clock and the encoded data. In this phase, an efficient encoder has to be designed, which minimizes both transition activity; it turns to reduce the global power dissipation .

KEY WORDS: personal digital assistants (PDAs), Serializer, Check Transition Block, Inverter, Phase Decoder.

1. INTRODUCTION

Low power design is a major issue in current integrated circuit design, especially for portable systems like cellular phones and personal digital assistants (PDAs). With the continuous scaling of technology, the power consumption of on-chip buses is becoming very significant. Thus, on-chip buses should be designed and optimized to consume -reasonable power while providing sufficient performance.The total power dissipation consists of two components: (a) the static power dissipation, which is due to a leakage current of transistors during steady state. (b) The dynamic power dissipation, which has two components : short circuit and charge/discharge of capacitance power dissipation. The short circuit power dissipation is a function of the slew rate of the input voltage; the sharper the clock edge, the lower the short circuit power dissipation. Short circuit power occurs when both p-transistor and n-transistor is ON for short duration of time. Mathematically,

$$Vdd < |Vtp| + Vtn \quad \ldots\ldots\ldots\ldots (1)$$

Where Vtp and Vtn are threshold voltages for PMOS and NMOS transistors, respectively. Threshold voltage is a voltage at which channel formation occurs in a metal oxide semiconductor field effect transistor (MOSFET). Power dissipation is given by

$$P = C V_{DD}^2 F \quad \ldots\ldots\ldots\ldots (2)$$

Where V_{DD} yields to supply voltage and Voltage swing. It is tried to reduce both supply voltage and Voltage swing because voltage have highest impact on total power dissipation as show in equation (2). C is capacitance and F is operating frequency.

Interconnect - power is dynamic power consumption due to interconnect capacitance switching. The switching activity can be expressed as

$$P = \alpha C V_{DD}^2 F \quad \text{----------} \quad (3)$$

Where, Vdd, f, Ct, α represent drain voltage, Frequency of operation, line capacitance, switching activity respectively. Switching activity is the changes in values the data has within itself. The dynamic power is consumed whenever there is a change of value on the line. Any transition, either from 0→1 or 1→0 will consume this dynamic power. Switching activity is defined as the sum of self - transition and coupling transition. Self - transition is defined as the transition from 0 to 1 or 1 to 0 take place on the data bus with respect to previous data on it. A coupling transition is defined as the transition from 0 to 1 or 1 to 0 take place between adjacent data bus lines. Most of the power is being wasted on the data buses and long interconnects as dynamic power dissipation for charging and discharging of internal node capacitance and inter - wire capacitance. This increased inter-wire effect on interconnects not only increase the power but also deteriorate the signal integrity due to the inter wire capacitance. The coupling capacitance also depends on the data dependent transitions and the coupling effect will increase or decrease depending on the relative switching activity between adjacent bus wires. Hence reducing switching activity eventually reduces the power dissipation.

2. LITERATURE REVIEW

This paper proposes SILENT: Serialized Low Energy Transmission Coding method to reduce the transmission energy of the serial communication by minimizing the number of transitions on the serial wire. Serial communication occupies less area due to less communication lines. Furthermore, the crosstalk problem can be avoided by wide spacing of serial lines. This technique reduces the number of transitions on the serial data line to cut down the energy consumption of the transmitter and wires.

On-chip source-synchronous serial communication has many advantages over multi-bit parallel communication in the aspects of skew, crosstalk, area cost, wiring difficulty, and clock synchronization. However, the serial wire tends to dissipate more energy than parallel bus due to the bit multiplexing. In this paper, we propose a novel coding method to reduce the transmission energy of the serial communication by minimizing the number of transitions on the serial wire. By applying this coding method, about 13% power reduction has been obtained on the overall on-chip networks. The main drawback of this work is this method is applicable only for serial links.

The data bus is partitioned into different clusters. Each cluster is of 5-bit width, with four data bit wide and one control bit. Bus invert (BI) method uses a control line called invert pin to differentiate between the transmission of original data and inverted data. As per BI method, if the number of transitions that are being transmitted are more than half of the bus width, then the original data is inverted and the control line is set to 'high', otherwise, the original data are transmitted and the control line is set to 'low'. Accordingly, this method finds the relationship of the original data and last encoding data (i.e., b(t) and B(t-1)), and then an extra control bit (i.e., the INV line) is obtained, and the unique decoding ability is achieved.

An encoding scheme for high-speed single-ended parallel transceiver system is presented. In the proposed scheme, three bits of information are transmitted across four pins using 3-level signaling. By using this coding method power reduction is achieved 68 %. The main disadvantage of this method is, it adds extra inversion signal during the data transfer. This extra bit increases transmitted data and also increased data transition.

In serial data transfer, data is generally loaded onto a buffer in parallel or serial fashion and then put on the bus serially. Before transmission, the number of transitions on a line is counted. This is just counting the transitions of the bit stream in that line. This can be done by a simple XOR gate between consecutive bits and counting the number of '1's. If the number of transitions is more than half the number of data words, the transitions states between the bits can be inverted. Each transition is made as a non-transition and vice versa. If not, the bit stream is transmitted as such. The system employs simple encoder/decoder, removes reference ambiguity, and reduces transmitter power line fluctuations all without sacrificing signal noise margin. Using current recycling, the power consumption of the proposed transmit drivers is reduced by 33% compared to the drivers in a conventional parallel link.

In case transition inversion is needed, the scheme operates by observing the transition states between any 2 bits and setting the encoded second bit to be the same as the previous encoded bit if there is a transition. If there is no transition, the previous encoded bit is inverted. The decision bit signifying transition inversion is transmitted before transmitting the encoded data. Also, the first bit of the bitstream is transmitted as such. This has to be done on all lines. Since the data is sent as a block, the extra bits on each line will signal for the respective bit streams. The decision bit signifying transition inversion is transmitted before transmitting the encoded data. This results in an overhead for the system. Switching activity reduction achieved is 15%. Even though the existing method reduces transition the extra transition indication bot increases the number of transmitted bit and also increases number of transition.

2.1. Proposed System: This proposed method reducing the number of bus lines of the conventional Parallel-Line Bus (PLB) architecture by multiplexing each m-bits onto a single line. This bus architecture, the Serial-Link Bus (SLB), transforms an *n*-bit conventional PLB into an n/m-line (serial link) bus. The advantage of SLBs is that they have fewer lines, less area occupation and reduced coupling capacitance. Serializing parallel buses increases switching activity. So an Embedded Transition Inversion encoding technique is proposed and to reduce the switch activity penalty and power dissipation due to serialization. The block diagram of proposed encoder and decoder is shown in figure 1 and figure 2 respectively.

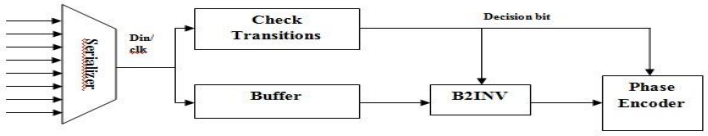

Fig.1. Block diagram for proposed ETI Encoder

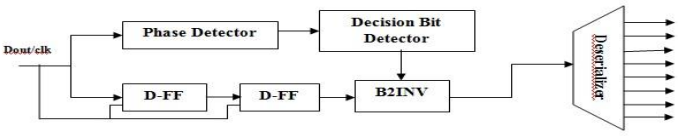

Fig.2.Block diagram for proposed ETI Decoder

2.2. Serializer: Multiplexing the data of m bus lines onto one line is done by serializer. This converts the parallel - line bus into serial links, which reduces the number of physical bus lines. For the same bus area, this reduction in the number of bus lines leads to a larger interconnects width and pitch. The larger line pitch reduces the coupling capacitance, while the wider interconnects reduce the resistivity, leading to a significant improvement in the interconnect energy dissipation and delay. By carefully selecting the number of serial links in the bus, a significant improvement in the overall energy dissipation can be achieved. This improvement increases as technology scales.

Multiplexing parallel buses into a serial link enables an improvement in terms of reducing interconnect area, coupling capacitance, and crosstalk, but it may increase the overall switching Activity Factor (AF) and energy dissipation. Therefore, an efficient coding method that reduces the switching AF is an important issue in serial interconnects design.

Table.1.Parallel data steam

Buffer Data	Bit Pattern
Data 1	D1,8 D1,7 D1,6 D1,5 D1,4 D1,3 D1,2 D1,1
Data 2	D2,8 D2,7 D2,6 D2,5 D2,4 D2,3 D2,2 D2,1
ata 3	D3,8 D3,7 D3,6 D3,5 D3,4 D3,3 D3,2 D3,1
Data 4	D4,8 D4,7 D4,6 D4,5 D4,4 D4,3 D4,2 D4,1
Data 5	D5,8 D5,7 D5,6 D5,5 D5,4 D5,3 D5,2 D5,1
Data 6	D6,8 D6,7 D6,6 D6,5 D6,4 D6,3 D6,2 D6,1
Data 7	D7,8 D7,7 D7,6 D7,5 D7,4 D7,3 D7,2 D7,1
Data 8	D8,8 D8,7 D8,6 D8,5 D8,4 D8,3 D8,2 D8,1

In block data transfer, data is generally loaded onto a buffer and then put on the bus. Each line in the bus is a serial line that will transmit one particular bit position of all data words that are put on the bus. A typical block can be as shown in Table 1.The buffer mostly will be able to hold a larger data than just one block of data but transmission will still happen with a granularity of one block. The bits taken in the vertical direction form a bitstream that travels in a line. Taking the bus into account, it is a collection of bitstreams running in parallel. The columns in the table represent the lines of the bus. The rows represent each data element of the buffer. When transmitting, the bits from each element travel in parallel across the lines. In the perspective of a line, bits of all data elements of one position are transmitted sequentially. This forms bitstreams on all lines composed of corresponding bits of all data elements. In the ETI encoder part, the input data *D*in are stored in the buffer to wait until the check transition operation is completed.

2.3. Check Transition Block:

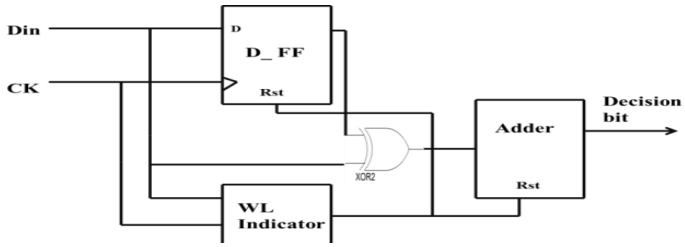

Fig.3.Check Transition Block Diagram

The check transition circuit is built by counting the transitions between consecutive bits in the bitstream. A transition between two bits is found in a simple manner by performing the equivalence operation of XOR (Exclusive OR) between them. The proposed circuit using a simple XOR gate between consecutive incoming bits of the bit stream is shown in Fig. 3. The WL indicator block counts the length of the data word and generates a high signal at the first bit of the data word. This signal is used to reset the adder and the D-flip-flop (D-FF). The D-FF stores the previous bit that is used to XOR with the current bit for transition checking. Before transmission, the number of transitions on a line is counted. This is just counting the transitions of the bit stream in that line. This can be done by a simple XOR gate between consecutive bits and counting the number of '1's. The adder block calculates the number of transition in a data word and sets the decision bit to high when the $Nt \geq N$th. If the decision bit is set to 1 the input data becomes inverted. Word length (WL) defines the number of bits in a data word and a threshold *N*th defines half

of WL. A transition is defined as a bit changing from zero to one or from one to zero. For example, the bit stream "0100" has two transitions while "0101" has three transitions. When the number of transitions *Nt* in a data word exceeds the threshold *N*th, the bits in the data word should be encoded. Otherwise, the data word remains the same. When an encoding is needed in a data word, this method checks every two-bit in the data word. Every two bit in the serial stream is combined as a base to be encoded. In this case, the *b*11*b*21 is a base and the *b*31*b*41 is another base. The 2-bit in a base is denoted as *b*1*b*2 and the encoded output is denoted as *be*1*be*2. When the *Nt* in a data word is less than *N*th, *b*1*b*2 remains unchanged. Otherwise, we perform the inversion coding and the phase coding. For the inversion coding, the bit streams "01" and "10" are mapped to "00" and "11," respectively. The bit streams "00" and "11" are mapped to "01" and "10," respectively. For the phase coding, we embed the inversion information in the phase difference between the clock and the encoded data.

The inversion encoding operation can be expressed as

*be*1 = *b*1

*be*2 = *b*2, with *Nt* < *N*th

!*b*2, with *Nt* ≥ *N*th.

The inversion decoding operation for the decoded output *bd*1 *bd*2 is

*bd*1 = *be*1

*bd*2 = *be*2, with *Nt* < *N*th

!*be*2, with *Nt* ≥ *N*th.

2.4. Inverter: The bit stream is encoded if a transition inversion is needed. This is done as the data is being put on the bus. This can be done in an on-the-fly manner since the encoder need to only process the current and next bit. The decision bit is used to control the encoding process in the B2INV and the phase encoder block. When the decision bit is set to zero, the B2INV passes the non-inverted bit stream. Otherwise, the bit stream is encoded. This encoder needs to operate only for those cases where a transition inversion is needed. The D-FF on the incoming bit stream calculates the transition state just as the decision circuit did during the loading of the block. Once the transition state is known, it is inverted to generate an inverted state if the decision was to invert the transition. This inverted transition state is used to manipulate the next bit in such a way that the next bit will be in the inverted transition state in correspondence to the current bit. The inverter block is shown in the Fig.4.

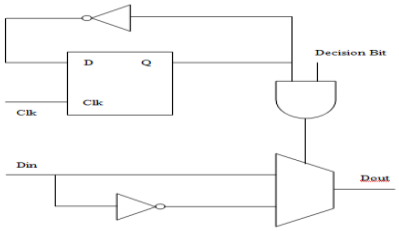

Figure.4.Inverter Block Diagram

2.5. Phase Encoder: Within every data word duration, the phase difference between the data and the clock distinguishes these two data words. Same *D*out "1000" in Fig. 5 is obtained from *D*in "1000" without inversion.

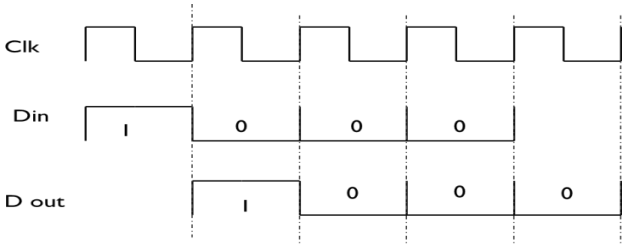

Figure.5.Input Data without inversion

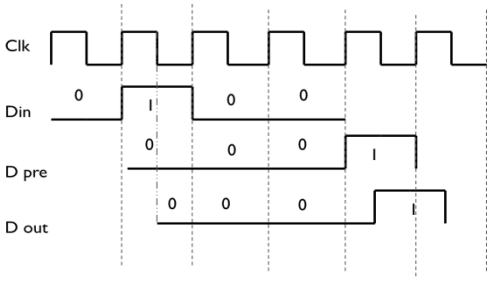

Figure.6.Input Data with inversion

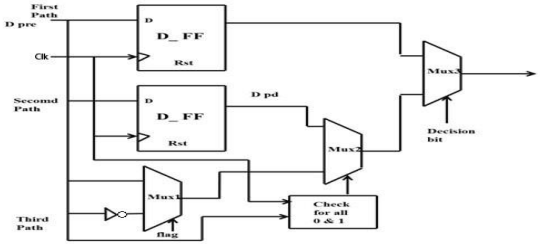

Figure.7.Phase encoder Block Diagram

*D*out "0100" in Fig. 6 is obtained from *D*in "1000" with inversion. A half clock cycle difference between *D*out and Clk, indicating that *D*in has been encoded. The *D*out and Clk are aligned in Fig. 5, indicating that *D*in has

not changed. This approach is able to identify whether *D*out has been encoded or not as long as there is a half cycle delay between the *D*out and Clk. Although the phase difference can distinguish most of the data words of ETIpre.

This method cannot be used for "0000" or "1111" because there is no transition inside the data word. Under the inversion condition for these two data words, the "0000" and "1111" change to "1000" and "0111". The first bit of *D*out in the "1000" and "0111" is aligned with Clk and the duration of the bit is only half of the clock cycle.

The phase generator is used to generate phase difference between the encoded data (*D*pre) and the clock (Clk) at each data word. Depending on the encoded data, there are three types of phase encoding: the one cycle delay, the half cycle delay and the special data word. The half cycle delay and the special data word are shown by the second and the third path Fig 7. In the special data word, the predefined Flag signal is used to present the special data pattern. The "check all 0 and all 1" block is used to identify the special data word when the encoded data (*D*pre) are "0000" or "1111." If the encoded data are not the special data word, the second path is selected from the MUX1. Otherwise, the third path is selected. The decision bit then selects the data from the first path or the output of the MUX1.

2.6. Phase Decoder: The main aim of decoder is to detect whether the data from the encoder is encoded or not. If the data is encoded in the encoder the inversion operation takes place in the decoder. Otherwise the data from the encoder is not inverted. This can be done by checking the phase difference between Clk and data. Normally, a Phase detector identifies an early or delayed phase. A variety of Phase detector could detect the phase difference. This paper adopts the commonly used Alexander Phase detector. The Alexander Phase detector architecture is shown in Fig.8, which uses three consecutive clock edges to generate four sampling signals (S0, S1, S2, and S3). The Phase detector is controlled by the clock Clk and input data *D*in. When the clock Clk and input data *D*in are valid, the Phase detector is activated to identify the phase relation between the clock and the data. The Phase detector can determine whether a data transition exists from the condition that the clock leads or lags the data.

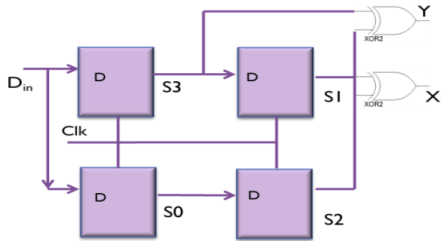

Figure.8.Phase detector block diagram

The basic waveform is shown in Fig. 9 to judge the un-inverted, inverted, no transition, or the special data word. If the clock leads the data (early conditions), the signal S1 ⊕ S2 is high and the S2 ⊕ S3 is low. Conversely, if the clock lags the data (late conditions), the signal S1 ⊕ S2 is low and S2 ⊕ S3 is high. Thus, S1 ⊕ S2 and S2 ⊕ S3 could provide the clock and data relation as shown in Table.2.

Table.2.Truth table for Phase decoder

S1 ⊕ S2	S2 ⊕ S3	Clock	Coding State
High	Low	Early	Has Not Been Encoded
Low	Low	No Transition	
Low	High	Late	Has Been Encoded
High	High	Special Data	

The early signal is for the un-inverted data and the no transition represents the un-encoded the "all zero or all one" data word. The late signal represents the case in which data have been inverted in encoder. The last case is for the encoded the "all zero or all one" data word. The decision bit is generated based on the phase information on the S1 ⊕ S2 and S2 ⊕ S3.

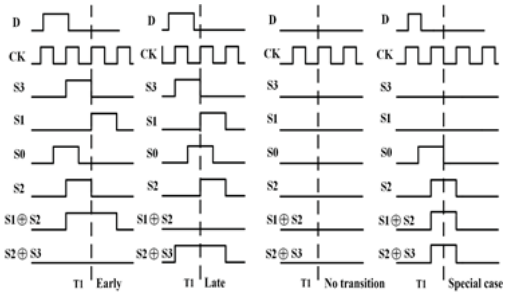

Figure.9.Phase Detector waveform

The decision bit is used in the B2INV for the decoding. The decoding operator in the B2INV is the same as that in the encoder. Two D-FFs are added in the front of the B2INV block for buffering and alignment. A larger

bandwidth is needed in the ETI coding scheme due to phase shift. The last bit has half the pulse width of the other bits so that the inter connect has twice the bandwidth. It means that the serial link needs to run at a much frequency. The higher clock frequency leads to problems, such as buffering, clock synchronization, and design complexity. The other way is to wait an extra bit to check the transition information but that would lower the overall bit rate.

We send both data and clock from TX to RX in our proposed scheme. In our simulation, we assume the channel length of the clock and data links is equal and the phase skew between them is negligible. A PLL or CDR is usually needed in the receiver to resynchronize the de-skew data. Only one global PLL or CDR is needed in the serial link to recover the de-skew data then perform the ETI decoding and de-multiplexing.

3. RESULTS ANALYSIS

The design is simulated using Xilinx 13.1 ISE software. VHDL is more preferred because this technique can reduce cost and time, easy to troubleshoot, and high references availability. The input and the output waveform generated during the simulation are given below [Fig. 10-12].

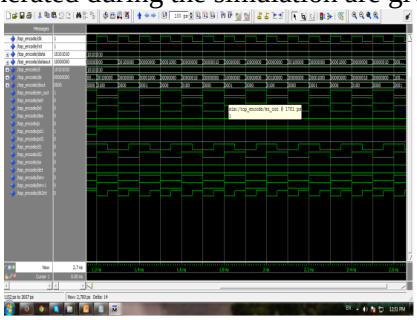

Fig.10.Simulation Output of ETI System

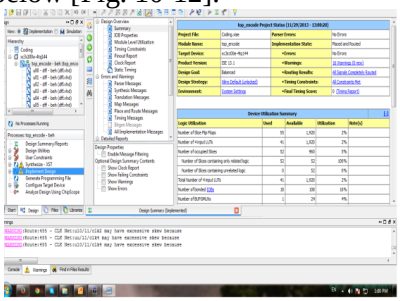

Fig.11.Result of Area Analysis

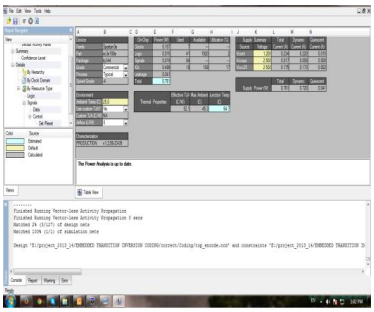

Fig.12.Result of Power Analysis

4. CONCLUSION

The proposed efficient Embedded Transition Inversion encoding scheme encodes the data efficiently depending on the transitional activity. The switching activity for the encoded data is minimum as compared to previous method. The main aim of the proposed technique is to save power dissipation of a serial link by minimizing transition on the data lines. This scheme eliminates the need of sending an extra bit by embedding the inversion information in the phase difference between the clock and the encoded data thereby reducing the extra bit transition and switching activity. The proposed technique is suitable for most of the Low Power VLSI applications**.** In addition to this work trying to reduce the area and power in future.

REFERENCES

Abinesh R, Bharghava R and Srinivas MB, Transition inversion based low power data coding scheme for synchronous serial communication, in Proc. IEEE Comput. Soc. VLSI Journal, 2009, 103–108.

Anto Bennet M, Sankar Babu G, Suresh R, Mohammed Sulaiman S, Sheriff M, Janakiraman G, Natarajan S, Design & Testing of Tcam Faults Using T_H Algorithm, Middle-East Journal of Scientific Research, 23 (08), 2015, 1921-1929.

Anto Bennet M, Manimaraboopathy M P, Maragathavalli P, Dinesh Kumar T R, Low Complexity Multiplier For Gf(2m) Based All One Polynomial, Middle-East Journal of Scientific Research, 21 (11), 2014, 2064-2071.

Anto Bennet M, Power Optimization Techniques for sequential elements using pulse triggered flip flops, International Journal of Computer & Modern Technology, 1 (1) , 2015, 29-40.

Brajesh Kumar, Kaushik N, Deepika Agarwal, Nagendra G. Babu, Bus encoder design for reduced crosstalk, power and area in coupled VLSI interconnects, Microelectronics Journal, Elsevier, 2013.

Lee K, Lee SJ, and Yoo HJ, Low-power network-on-chip for high performance SoC design, IEEE Trans. Very Large Scale Integr, (VLSI) Syst., 14 (2), 2006, 148–160.

Zogopoulos S and Namgoong W, High-speed single-ended parallel link based on three-level differential encoding, IEEE J, Solid-State Circuits, 44 (2), 2009, 549–558.