Design and Implementation of Area and Power Efficient Embedded Transition Inversion Coding For Serial Links

C.Jacob Ebbipeni[1] N.Bamakumari[2]
Department Of Electronics And Communication Engineering
Sri Venkateswara College Of Engineering And Technology, Thirupachur.

ABSTRACT
Advanced silicon technology offers possibility of integrating hundreds of millions of transistor into a single chip, which makes system on chip (SoC) design possible. With the continuous scaling of silicon technology, area and power dissipation of interconnects are one of the main bottlenecks for both on chip and off chip buses. 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 and energy dissipation. Therefore, and efficient coding method that reduce the switching activity is an important in serial interconnect design. Embedded Transition Inversion (ETI) is the coding method that reduces the switching activity and energy dissipation. Area is one of the important objective in integrated circuits. The main of this paper is reduce the area in ETI coding architecture. In this proposed method Alexander Phase Detector is replaced by Hogge Phase Detector. The proposed system done by using Xilinx 13.2 and ModelSim software to get efficient output. The proposed system result analysis shows better than the existing method.

INDEX TERMS :Serial Interconnection, Embedded Transition Inversion (ETI), Phase Encoder, Phase Decoder, Bit two inverter.

I. INTRODUCTION

Advanced silicon technology offers the possibility of integrating hundreds of millions of transistors into a single chip, which makes system-on-chip (SoC) design possible. With the continuous scaling of silicon technology, area and power dissipation of interconnects are one of the main bottlenecks for both on-chip and off-chip buses. Multiplexing parallel buses into a serial link enables an improvement in terms of reducing interconnect area, coupling capacitance, and crosstalk [1], 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 interconnect design. Many studies attempt to reduce the AF of parallel buses. For example, Stan and Burleson [2] introduced a bus-invert method that transmits the original or inverted pattern to minimize the switching activity. Researchers have proposed many techniques to improve the bus-invert coding method, such as the partial bus-invert coding [3] and weight-based businvert coding methods [4]. The schemes mentioned above use an extra channel to send the inversion indication signal. Kuo et al. [5] proposed the serial coding technique to solve the extra channel problem. They append extra information bits to the back of the original data word.

Although this approach resolves the area overhead problem, it increases data latency. Three level differential encoding is proposed for parallel bus [6] to enable multiple drivers at the transmitter and to recycle the same current and reduce power consumption [6]. Joint crosstalk avoidance code and error correction code are proposed to reduce the power in parallel bus [7]. Huang et al. [8] further proposed combining serializing bus with the joint crosstalk avoidance code and error correction code to reduce the power. Serialized low-energy transmission (SILENT) [1] is a coding method used in reducing the switching activity for serial links. This approach encodes every single bit in the parallel bus using the XOR gate, and multiplexes the encoded parallel buses into a serial link. The XOR operation sets an adjacent bit with the same value to zero. The greater the correlation is, the more zeros the encoder produces. This method is designed for data with strong correlation. Bharghava et al. [9] proposed the transition inversion coding (TIC) technique to reduce switching activity for random data and to detect errors. Their technique counts the transitions in the data word, and inverts the transition states if the number of transitions in a data word is more than half of the word length.

The scheme sets the current bit in the serial stream to be the same as the previous encoded bit when there is a transition. Otherwise, it is set to the inversion of the previous encoded bit. A transition indication bit is added in every data word. This extra bit not only increases the number of transmitted bits, but also increases the transitions and latency. Lee et al. [10]– [12] used serial links as communication channels on an on-chip network architecture for SoC. The serial links reduce the area of communication channels by 57% compared with a non-serialized approach [10]. This approach also reduces the switch activity because the coupling capacitance of the interconnect wires decreases [10]–[12]. Forward error correction (FEC) code is used to reduce the serial link power by trading off the FEC coding gain with specifications on transmit swing, analog-to-digital converter precision, jitter tolerance, receive amplification, and by enabling higher signal constellations [13]. By combining the single and differential signals and 8B/10B coding, a high speed phase tracking clock recovery is developed for serial link to reduce the power [14]. To utilize the relation between data in a link, the encoder reorders or shuffles the M bits in the parallel bus such that the encoded data are less correlated or silent [15]. Then the encoded data are serialized with one control bit. The silent coding with shuffling can reduce the power dissipation compared to the silent coding.

A serial link on-chip bus architecture is proposed to lower interconnect power [16]. Serialization reduces the number of wires and leads to a larger interconnect width and spacing. A large interconnect spacing reduces the coupling capacitance, while the wider interconnects reduce the resistivity. A significant improvement in the interconnect energy dissipation is achieved by applying different coding schemes and their proposed multiplexing techniques. However, the power reduction decreases when the degree of multiplexing increases. The embedded transition inversion (ETI) coding scheme to solve the issue of the extra indication bit [17]. 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. When there is an inversion in the data word, a phase difference is generated between the clock and data. Otherwise, the data word remains unchanged and there is no phase difference between the clock and the data. This ETI coding scheme reduces transition by 31% compared with the SE scheme. The improvement of transition reduction is 19% compared with that of the TIC.

The receiver side adopts a phase detector (PD) to detect whether the received data word has been encoded or not. Statistical analysis and experimental results show that the proposed coding scheme has low transitions for different kinds of data patterns. Increasing the interconnect spacing reduces the coupling capacitance for on-chip buses. We also study about different phase detector to optimizes the performance of the ETI and also reduced the power and the area of the ETI system. The remainder of this paper is organized as follows. Section II presents ETI coding scheme. In Section III describe about ETI Architecture that includes the ETI encoder and decoder. Experimental result shown in Section IV and Conclude in Section V.

II. ETI CODING SCHEME

Although many coding algorithms can reduce the switching AF, most of them are designed for specific applications, such as video streaming or strongly correlated data. The TIC is one of the methods developed for random data [9]. This method adds a transition indication bit to every data word to indicate if there is an inversion or not. This inversion coding is performed on every bit of two consecutive bits in the serial stream. The extra indication bit increases the switching activity. This paper proposes the ETI coding scheme that operates on a two-bit basis and removes all the transition indication bits. Fig. 1. n/m ETI serial links with n input bitstreams under degree of multiplexing m. Fig. 1. ETI coding scheme for one serial link, word length = WL, Nth = WL/2, and number of transition = Nt . An n/m ETI serial links with n input bitstreams under degree of multiplexing m is shown in Fig. 1. Each serial link has m input bitstreams that are multiplexed by a serializer, followed by the ETI encoding. The encoded stream is transmitted through the serial link and followed by the ETI decoding and a deserializer. The ETI coding scheme includes the inversion coding and phase coding as shown in Fig. 2.

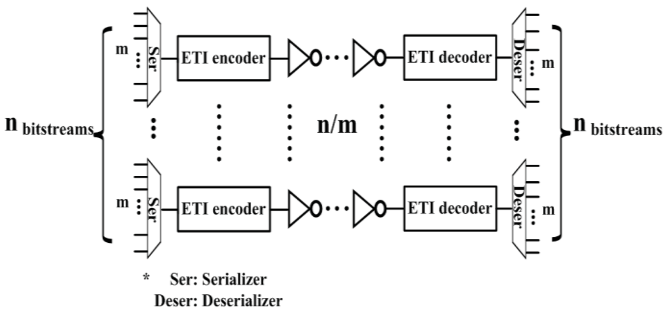

Fig 1. *n/m* ETI serial links with *n* input bitstreams under degree of multiplexing *m*.

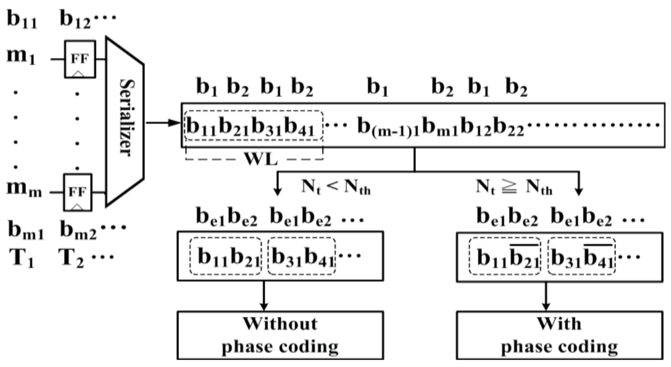

Fig 2. ETI coding scheme for one serial link, word length = WL, *N*th = *WL/*2, and number of transition = *Nt* .

1) Inversion Coding: Define the word length (WL) as the number of bits in a data word and a threshold Nth as half of WL. A transition is defined as a bit changing from zero to one or from one to zero. For example, the bitstream “0100” has two transitions while “0101” has three transitions. When the number of transitions Nt in a data word exceeds the threshold Nth, 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, as Fig. 3 shows. Every two bit in the serial stream is combined as a base to be encoded. In this case, the b11b21 is a base and the b31b41 is another base. The 2-bit in a base is denoted as b1b2 and the encoded output is denoted as be1be2. When the Nt in a data word is less than Nth, b1b2 remains unchanged. Otherwise, we perform the inversion coding and the phase coding. For the inversion coding, the bitstreams “01” and “10” are mapped to “00” and “11,” respectively. The bitstreams “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 The inversion decoding operation for the decoded output bd1 bd2 is Since this operation is on a two-bit basis and only the second bit is inverted, it is called bit-two inversion (B2INV).

2) Phase Coding: The ETI coding uses the phase difference between the data and the clock to encode the indication information. Table I shows the corresponding output data word after TIC, ETIpre, and ETI. The ETIpre has the same data word as the TIC, except that it removes the extra bit bex. Removing the bex leaves eight sets of data words that are exactly the same. For example, there are two “1000” data words after the ETIpre coding. Within every data word duration, the phase difference between the data and the clock distinguishes these two data words, as Fig. 4 illustrates. Same Dout “1000” in Fig. 3 and 4 is obtained from Din “1000” and “1101” without and with inversion. A half clock cycle difference between Dout and CK is shown in Fig. 3, indicating that Din has been encoded.

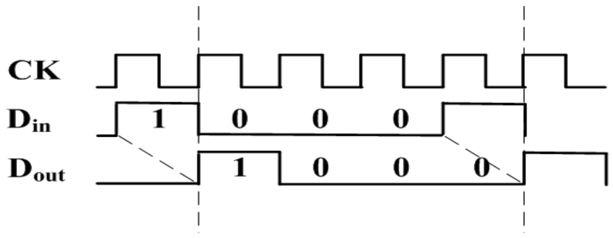

Fig 3. Din without encoding

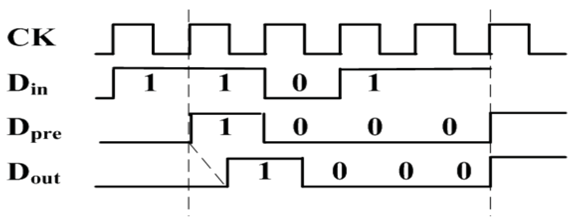

Fig 4. Data with encoding

This approach is able to identify whether Dout has been encoded or not as long as there is a half cycle delay between the Dout and CK. 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. The corresponding waveforms are shown in Fig. 5 for these two data words. The first bit of Dout in the "1000" and "0111" is aligned with CK and the duration of the bit is only half of the clock cycle.

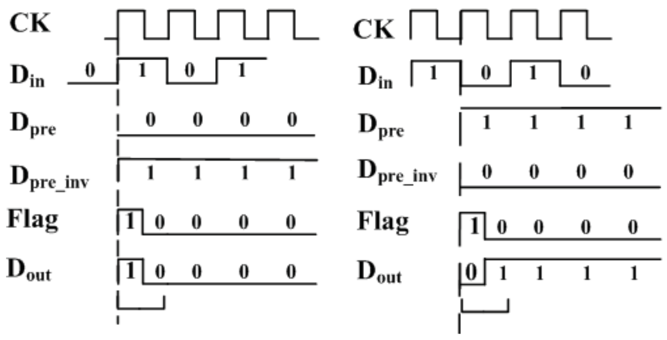

Fig 5. Waveform example for special data words

III. ETI ARCHITECTURE

A. ETI Encoder

The overall architecture of the ETI scheme is shown in Fig. 6. We add the ETIpre block and the TIC architecture for clarity. The ETIpre does not provide the decision bit information so it cannot be decoded in the receiver. The ETIpre encoder is shown by the dashed box in the ETI encoder in Fig. 6. The TIC counts the transitions in the data word then uses this information to perform encoding. The transition indication bit is added to every data word to indicate whether there is an inversion or not. The decoder adopts the transition indication bit to perform the decoding, as shown in Fig. 6. In the ETI encoder part, the input data Din are stored in the buffer to wait until the check transition operation is completed. The transition and threshold in a data word are used to set the decision bit. The decision bit is used to control the encoding process in the B2INV and the phase encoder block.

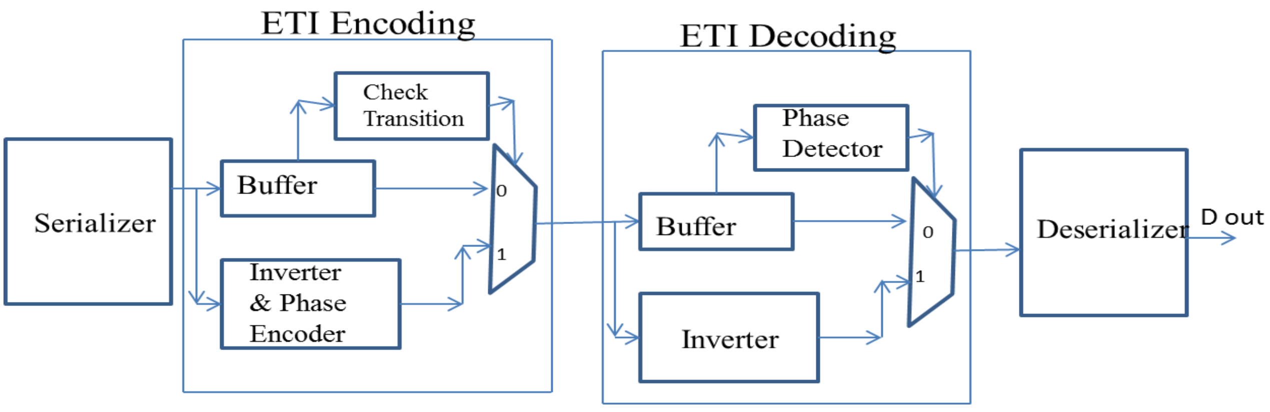

Fig 6. Architecture of the modified ETI scheme

The ETI encoder includes the check transitions block, buffer, B2INV, and phase encoder. The check transition block is shown in Fig. 7.

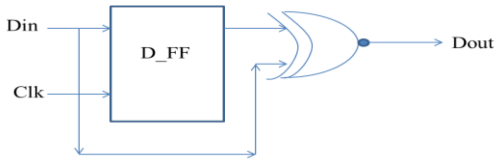

Fig 7. Check Transition Block

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. 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 bitstream 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 $Nt \geq N\text{th}$. 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 Nth 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 Nth, 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 $b11b21$ is a base and the $b31b41$ is another base. The 2-bit in a base is denoted as $b1b2$ and the encoded output is denoted as $be1be2$. When the Nt in a data word is less than Nth, $b1b2$ 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

$$be1 = b1$$

$$be2 = b2, \quad \text{with } Nt < N\text{th}$$

$$!b2, \quad \text{with } Nt \geq N\text{th}\ .$$

The inversion decoding operation for the decoded output $bd1$ $bd2$ is

$$bd1 = be1$$

$$bd2 = be2, \quad \text{with } Nt < N\text{th}$$

$$!be2, \quad \text{with } Nt \geq N\text{th}\ .$$

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. 8.

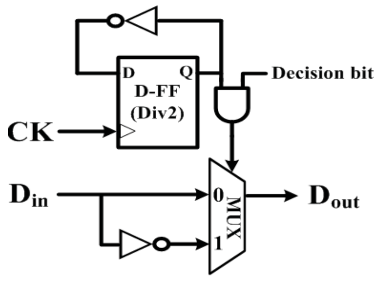

Fig 8. Bit Two Inverter

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. 9. is obtained from *D*in "1000" without inversion.

*D*out "0100" in Fig. 9 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. 9, 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 9.

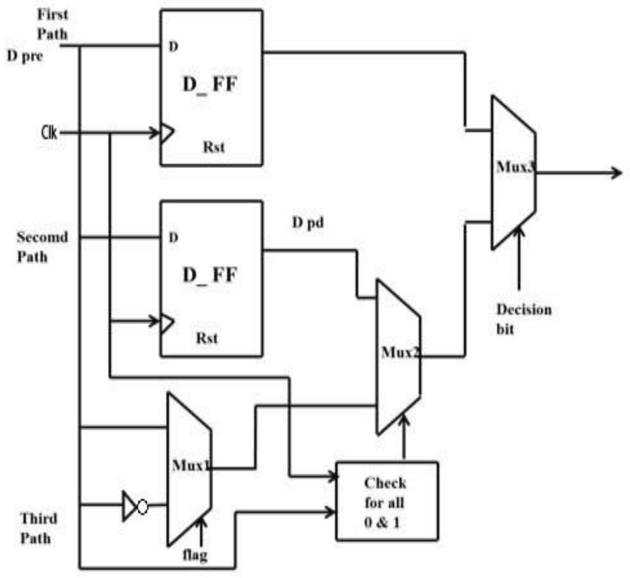

Fig 9. Phase Encoder

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.

B. ETI Decoder

The ETI encoder generates the phase difference between the clock and the data word. Normally, a PD identifies an early or delayed phase. A variety of PDs could detect the phase difference. This paper adopts the commonly used Hogge PD [18]. The Hogger PD architecture is shown in Fig. 10.

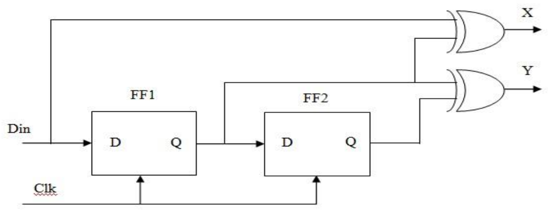

Fig 10. Hogge Phase Decoder

The PD is controlled by the clock CK and input data Din. When the clock CK and input data Din are valid, the PD is activated to identify the phase relation between the clock and the data. The PD can determine whether a data transition exists from the condition that the clock leads or lags the data.

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 interconnect 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. The disadvantage of this method is added in the revised paper. 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.

IV. EXPERIMENTAL RESULT

The simulation result of the ETI scheme with Hogge phase detector is shown in the Fig 11. And the ETI scheme with Alexander PD and ETI scheme with Hogge PD comparison table is shown in Table 1.

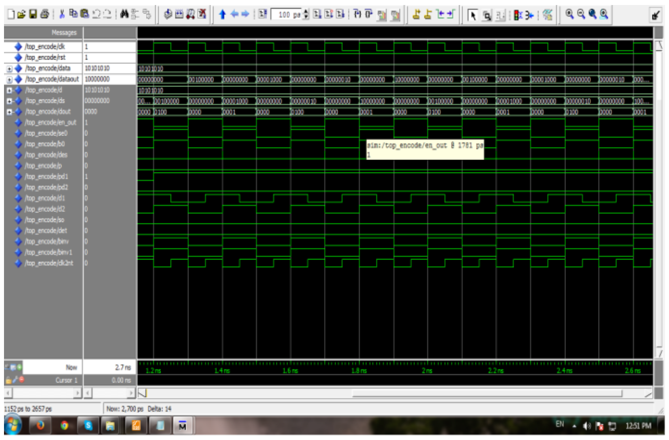

Fig 11. Simulation Result of the Proposed ETI Scheme

Spartan 3e Xc2s100e-4tq144	AREA		DELAY(ns)	POWER(W)
	SLICE	LUT		
EXIT_ETI	37	59	9.061	0.654
PROPOSED_ETI	34	41	7.652	0.571

Table 1. Comparison of ETI Schemes

V. CONCLUSION

The ETI scheme is reduces the extra bit used in the TIC scheme and reduces the crosstalk, energy dissipation. ETI scheme uses the phase difference between the data and clock to indicate the bit inversion. In this paper the proposed ETI scheme is reduces the area and power compare with the existing ETI scheme.

REFERENCES

[1] K. Lee, S. J. Lee, and H. J. Yoo, "SILENT: Serialized low energy transmission coding for on-chip interconnection networks," in *Proc. IEEE Int. Conf. Comput.-Aided Design Conf.*, Nov. 2004, pp. 448–451.

[2] M. R. Stan and W. P. Burleson, "Bus-invert coding for low-power I/O," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 3, no. 1, pp. 49–58, Mar. 1995.

[3] Y. Shin, S. I. Chae, and K. Choi, "Partial bus-invert coding for power optimization of application-specific systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 9, no. 2, pp. 377–383, Apr. 2001.

[4] R. B. Lin and C. M. Tsai, "Weight-based bus-invert coding for lowpower applications," in *Proc. Int. Conf. VLSI Design*, Jan. 2002, pp. 121–125.

[5] C. H. Kuo, W. B. Wu, Y. J. Wu, and J. H. Lin, “Serial low power bus coding for VLSI,” in *Proc. IEEE Int. Conf. Commun., Circuits Syst.*, Jun. 2006, pp. 2449–2453.

[6] S. Zogopoulos and W. Namgoong, “High-speed single-ended parallel link based on three-level differential encoding,” *IEEE J. Solid-State Circuits*, vol. 44, no. 2, pp. 549–558, Feb. 2009.

[7] S. R. Sridhara and N. R. Shanbhag, “Coding for reliable on-chip buses: A class of fundamental bounds and practical codes,” *IEEE Trans. Comput.- Aided Design Integr. Circuits Syst.*, vol. 26, no. 5, pp. 977–983, May 2007.

[8] P. T. Huang, W.-L. Fang, Y.-L. Wang, and W. Hwang, “Low power and reliable interconnection with self-corrected green coding scheme for network-on-chip,” in *Proc. 2nd ACM/IEEE Int. Symp. Netw. Chip*, Apr. 2008, pp. 77–84.

[9] R. Abinesh, R. Bharghava, and M. B. Srinivas, Transition inversion based low power data coding scheme for synchronous serial communication,” in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI Conf.*, May 2009, pp. 103–108.

[10] S. J. Lee, S. J. Song, K. Lee, J. H. Woo, S. E. Kim, B. G. Nam, and H. J. Yoo, “An 800 MHz star-connected on-chip network for application to systems on a chip,” in *IEEE Int. Solid-State Circuits Conf. Technol. Dig.*, Feb. 2003, pp. 468–469.

[11] K. Lee, S. J. Lee, and S. E. Kim, “A 51 mW 1.6 GHz on-chip network for low-power heterogeneous SoC platform,” in *Proc. IEEE Int. Solid- State Circuits Conf.*, Feb. 2004, pp. 152–518.

[12] K. Lee, S. J. Lee, and H. J. Yoo, “Low-power network-on-chip for highperformance SoC design,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 2, pp. 148–160, Feb. 2006.

[13] C.T. Chiu, W.C. Huang, C.H. Lin, W.C. Lai, and Y.F Tsao," Embedded Transition Inversion Coding With Low Switching Activity for Serial Links*", IEEE Trans. Very Large Scale Integration (Vlsi) Systems,* vol. 21, no. 10, Oct. 2013.

[14]. R. Abinesh, R. Bharghava, and M. B. Srinivas, “Transition inversion based low power data coding scheme for synchronous serial communication,” in Proc. IEEE Comput. Soc. Annu. Symp. VLSI Journal., May 2009, pp. 103–108.

[15]. Brajesh Kumar Kaushik n, Deepika Agarwal, Nagendra G. Babu “Bus encoder design for reduced crosstalk, power and area in coupled VLSI interconnects” Microelectronics Journal. Elsevire April 2013

[16]. J. Bhasker “VHDL Primer”, third edition.

[17]. M. Ghoneima, Y. Ismail, M. Khellah, J. Tschanz, and V. De, “Serial-link bus: A low-power on-chip bus architecture,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 56, no. 9, pp. 2020–2032, Sep. 2009.