TRANSITION INVERSION BASED LOW POWER DATA CODING SCHEME FOR BUFFERED DATA TRANSFER

[1]Dr. John Paul Pulipati, Principal, [2]Dr.Purushotham Naik, [3]P.Venkatapathi,
[4]D. Rajendra Prasad, [5]K. Rajeswar
[1]principal@mrce. in,Malla Reddy College of Engineering
[2]Professor, Dept. of ECE,Malla Reddy College of Engineering
[3]Assistant Professor, Dept. of ECE,Malla Reddy College of Engineering
[4]Assistant Professor, Dept. of ECE,Malla Reddy College of Engineering
[5]Assistant Professor, Dept. of ECE, Malla Reddy College of Engineering

Abstract— In this work the authors propose a data coding protocol that leads to power reduction for block data transfer in off-chip buses. I/O pads driving off-chip buses contribute to a major portion of power dissipation in chips. Also, block data transfer is preferred in most systems like caches, DMA etc. In this proposed work, the prior knowledge of the block of data to be transmitted, when it is stored in the buffer, is exploited in a serial fashion to reduce transitions on every bus line. Statistical analysis shows up to 31.9% reduction in transitions. Benchmark results show that it leads to 29% reduction in power consumption. The technique provides added error detection on the lines of parity bit technique, with similar average error detection capability.

1. Introduction

Increased integration and higher operating frequencies compound the problem of power dissipation in VLSI chips. One of the main contributors to power consumption is switching activity on the high-capacitance lines of an interconnection system, especially off-chip buses. Buses constitute an important resource for addressing and data transfer in the implementation of most electronic systems. The fact that the power consumed at the I/O pads accounts for a significant fraction of the total power consumed in VLSI systems has been independently established by many researchers [1-3].

Error detection is also of equal importance, as buses are more prone to error due to temperature variations, interference from neighboring sources, and ageing etc. Thus a low power data coding scheme supporting error detection is desired. Existing techniques make use of the data as it is placed on to the bus. This leads to a delay which can severely affect the operating frequency of the circuit. The proposed encoding technique, which is based on serial transition inversion, works on blocks of data for power reduction in off-chip buses. Numerous bus transmission protocols deal with blocks of data rather than individual data words. This is evident in DMA transfers and cache lines which are widely used in computer systems. In block data transfer the latency in data coding can be hidden since the block will be transmitted only after it is filled up. This brings the pipeline approach to the coding technique. Since Off- chip buses consume more power, keeping the encoding circuitry before the I/O pad reduces power consumed by it. The proposed technique is also called transition inversion in the following sections

2. Related work

One of the most often cited encoding methods is the bus-invert method [1]. Bus-invert selects between the original and the inverted pattern in a way that minimizes the switching activity on the bus. The resulting patterns together with an extra bus line (to notify whether the address or its

ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-12, 2017
DOI:10.21276/ijcesr.2017.4.12.31

complement has been sent) are signaled over the bus. Musoll et al. proposed the working zone method [3]. Their method takes advantage of the fact that data accesses tend to remain in a small set of working zones. Other encoding techniques include Asymptotic Zero-Transition Encoding [2], Gray coding (mostly for addresses), and other application specific encoding techniques [9,10]. Most of the existent techniques make use of the repeating patterns in address buses to reduce address bus transitions [6,8,9]. Frequent Value encoding (FVE) is another technique proposed in [13] also results in a significant reduction in transitions, but has not been considered here, due to the overhead involved.

There is no existent literature on bus coding methodologies for block data transfer, other than Serial Bus Invert [14], which encodes blocks of data rather than individual data words. Also a transition inversion based encoding technique [12] was developed for serial buses that serve as a basis for the proposed technique. The technique proposed in this work is compared with Serial Bus Invert, serial gray coding [15], and transition signaling [11], which is the reverse of gray coding. The last technique mentioned is similar to the algorithm proposed by us, but differs in the decision making process.

3. Proposed technique

In block data transfer, data is generally loaded onto a buffer and then is 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. Before transmission, the number of transitions on a line is counted. 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. 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. This has to be done on all lines. Since the data is sent as a block, the extra bit on each line will signal for all the data words.

The transitions in the bit stream, transmitted on a line, can be reduced by the aforementioned scheme. Each line is processed independent of each other. If there is a need for transition inversion, then the following steps are followed to obtain encoded data. Let the data bit that is to be transmitted next be bd and previous data bit be bdp. The previous transmitted bit is btp. The next transmitted bit will be

$$bt = btp \quad \text{if} \quad bd \neq bdp$$
$$= !btp \quad \text{if} \quad bd = bdp$$

In receiver the reverse logic needs to be applied. When the bit stream has been signaled as modified, then the following steps are followed to decode data. The previous and current received bits are assumed to be 'brp' and 'br' respectively. The previous decoded bit is assumed to be bdp. The current received bit will be

$$bd = bdp \quad \text{if} \; brp \neq br$$
$$= !bdp \quad \text{if} \quad brp = br$$

The encoding and decoding is done on the fly, to reduce performance losses. For example in the bit stream 10101011, the number of transitions is 6. Thus this stream is to be modified according to the algorithm described above. The first bit is transmitted as such, without any change. This is described in Table 1. The encoded data has only one transition.

Table 1: Sample coding process

Key: NT – No Transition, T – Transition, NC-No Change

Bit No.	1	2	3	4	5			
Bit stream	1	0	1	0	1			
Transition State	NC	T	T	T	T			NT
Encode state	NC	NT	NT	NT	NT			T
Encoded Bit stream	1	1	1	1	1			0

ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-12, 2017
DOI:10.21276/ijcesr.2017.4.12.31

Decode state	NC	T	T	T	T			NT
Decoded bit stream	1	0	1	0	1			1

A. Statistical Analysis of the Algorithm

A statistical analysis of the serial transition inversion (STI) algorithm has been carried out in two independent methods taking buffer depths of 8, 16, 32, and 64. The first analysis considers all combinations of the word and determines the original and modified transitions in the datasets. It is essentially a brute force approach. The data considered was a uniform distribution of all possible data patterns that is likely to be transmitted over the bus. For example, considering a buffer depth of 8, the number of possible data patterns of one bit stream is 256. The number of transitions in these data patterns was calculated along with the number of transitions in the data pattern after being modified, using the proposed algorithm. The second analysis was an analytical one. For an N-bit system, where the transitions are taken between the consecutive bits, maximum of (N-1) transitions are possible in the bit stream. Number of possibilities of 'i'

transitions= (N-1)Ci. Let Torg and Tmod be the number of transitions in the original data patterns and the number of transitions in the modified data patterns respectively. These entities can be calculated as follows: Total number of transitions Torg

Transition inversion is done when the number of transition is more than or equal to N/2. The number of transitions in the modified data will be (N-1-i) for

'i' transitions in the original data.

T_{mod} =

$$\sum_{i=0}^{N/2-1} {}^{N-1}C_i * 2 * i + \sum_{i=N/2}^{N-1} {}^{N-1}C_i * 2 * (N-1-i) \quad (2)$$

The average reduction can be calculated by taking the difference between the number of transitions in the modified data patterns, given by equation (2) and unmodified data patterns, given by equation (1). The reduction figures would be smaller if the decision bit is also considered. The results obtained by both the methods agree with each other and are shown in Table 2.

Table 2: Statistical Percentage Reduction

Word Length	8 Bits	16 Bits	32 bits	64 bits
% Reduction in transitions	31.25	20.95	13.54	9.78

B. Power Analysis of the Algorithm

The overall power reduction consists of the power reduction achieved by the transition reduction minus the power consumed by the extra circuitry. Unmodified dynamic power consumed by I/O pads is given by

$$p_{org} = \frac{1}{2} C_T V_{dd}^2 f \quad (3)$$

variation of this parameter with word length is shown in Figure 1. Bus invert leads to a reduction in bandwidth since it poses a delay in putting the encoded data. By following a depth based approach where most delays are hidden, bandwidth need not be reduced.

D. Error Detection Analysis of the Algorithm

The proposed algorithm's propensity towards reducing the number of transitions can be used for detecting errors. This can be done by determining if the number of transitions in the received bitstream is more than half the bitstream length. If this count is more than half the bitstream length, the incoming data is incorrect. The proposed technique is compared with parity bit technique, as both have similar overhead i.e. addition of one bit to the bitstream. The parity bit detects all odd bit errors, but misses even bit flips, whereas, transition

ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-12, 2017
DOI:10.21276/ijcesr.2017.4.12.31

inversion can detect a certain percentage of any number of bit errors. Error analysis has been done by considering all combinations of the given word length that are transmitted over the bus. Fortransition inversion coding, the

possible combinations of the word are those in which the

Where, Vdd, f, Ct, α represent drain voltage, frequency of operation, line capacitance, switching activity respectively. If the power dissipation of the extra circuitry required for the coding process is taken into account then the equation given above has two extra terms on the right hand side, the encoder and decoder power dissipation respectively.

C. Performance Analysis of the Algorithm

The transition inversion algorithm needs an extra bit to be transmitted before the start of the block of data on all lines. This leads to a decrease in bandwidth utilization. For a system with buffer depth of 8, 9 bits are transmitted on one line. Therefore a frequency increase of 9/8 will be needed to maintain the same bandwidth utilization. The corresponding power consumed by I/O pads will increase linearly.

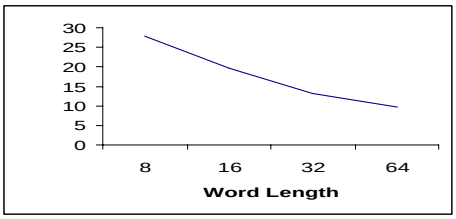

Figure 1: Effects of frequency scaling with word length

A performance metric is defined to take into account the scaling of the frequency and the reduction intransitions, and is calculated as their product. The variation of this parameter with word length is shown in Figure 1. Bus invert leads to a reduction in bandwidth since it poses a delay in putting the encoded data. By following a depth based approach where most delays are hidden, bandwidth need not be reduced.

D. Error Detection Analysis of the Algorithm

The proposed algorithm's propensity towards reducing the number of transitions can be used for detecting errors. This can be done by determining if the number of transitions in the received bitstream is more than half the bitstream length. If this count is more than half the bitstream length, the incoming data is incorrect. The proposed technique is compared with parity bit technique, as both have similar overhead i.e. addition of one bit to the bitstream. The parity bit detects all odd bit errors, but misses even bit flips, whereas, transition inversion can detect a certain percentage of any number of bit errors. Error analysis has been done by considering all combinations of the given word length that are transmitted over the bus. Fortransition inversion coding, the

possible combinations of the word are those in which the number of transitions is less than 4. All the combinations of bit errors right from one bit error to 8 bit errors have been checked for both the proposed technique and parity bit technique. The result of this analysis is shown in Table 3.

Table 3: Error Detection Analysis

No. of Bit errors	% of errors detected	
	Parity Coding	Proposed Technique
1	100	31.25
2	0	44.64
3	100	52.68
4	0	55.71
5	100	52.68
6	0	44.64
7	100	31.25
8	0	0

If all the bits are in error, then neither technique can detect the error, as in the proposed technique if all bits are flipped, the number of transitions remains the same. Calculation of statistical averages over the entire range of bit errors shows that the proposed technique and parity bit technique both have the same value of 50.2%. The average is calculated as the ratio of total number of errors detected to the total number of errors possible on the line. Thus the proposed technique can be used as a hint to upper layers of communication that an error has occurred since it cannot reliably detect all errors.

ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-12, 2017
DOI:10.21276/ijcesr.2017.4.12.31

4. Implementation strategies

In most block systems, the data buffer is present just before the transmission part. The core logic puts the data inside the buffer from one side and the transmission happens from the other side.

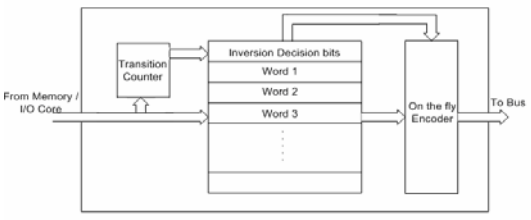

Figure 2: High Level Architecture

The transition counting happens when the data is being filled up in the buffer. The transition inversion decision is made depending on the count of the transitions and encoding done based on it. The bit stream is encoded on the fly as the data is put on the bus, as shown in Figure 2. In the receiver the decoder has to decode the incoming bit stream and recover the original data.

A. Decision Circuit and Encoder

The decision circuit is a simple XOR gate between consecutive bits of the input bit stream as shown in Figure 3. The counter needs to count only up to half the number of maximum transitions.

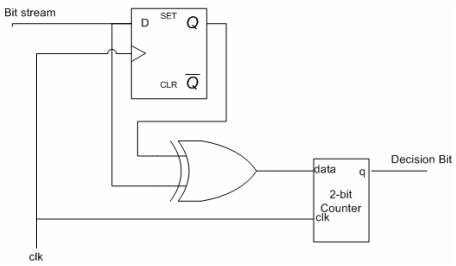

Figure 3: Decision Circuit (Transition Counter)

This circuit can also be implemented with double edge triggered circuits to further optimize at the encoder stage.

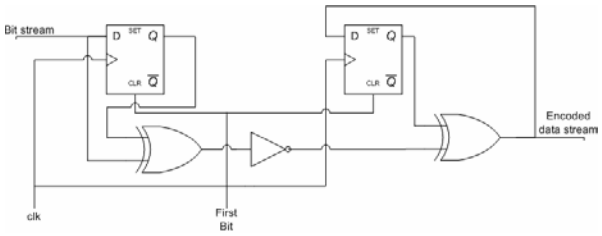

Figure 4: Encoder Circuit

The transition counter works in parallel to buffer loading, and is thus masked. The on the fly encoder is shown in Fig 4.

B. Decoder

The decoder shown in Fig 5 performs XOR between consecutive bits to determine transition state, inverts the received bit if required to recover the data.

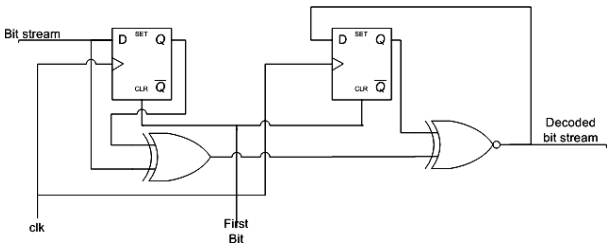

Figure 5: Decoder Circuit.

C. Complexity Analysis

The main components of both systems are the decision circuit, encoder and decoder.

Decision circuit: With increase in bus width, the space complexity increases exponentially (O(N2)) for bus invert decision circuit. A comparable parameter in the proposed technique is buffer depth which leads to a linear increase (O(N)) in circuit complexity. Time complexity increases linearly (O(N)) in bus invert, while in the proposed technique it is constant(O(1)).

Encoder and Decoder: With increase in bus width, the bus invert encoder/decoder circuit complexity increases linearly (O(N)). For the proposed technique, the circuit complexity is constant (O(1)). Time complexity is constant (O(1)) in both bus invert and the proposed technique.

5. Experimental results

For experimental analysis, the algorithm was applied on random image data and SPEC2000 benchmark binaries.

ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-12, 2017
DOI:10.21276/ijcesr.2017.4.12.31

Random Image Data:

For this analysis of the algorithm seven images were taken and their RGB values were ran through the algorithm. The images were a mix of both smooth and detailed features. The results are tabulated in Table 4. These do not include the power dissipated by the encoder and decoder circuitry.

Table 4: A comparison of transition reduction for Bus invert and the Proposed Technique

#	Original no. of transitions	Bus Invert Coding		Proposed technique	
		transitions	% reduction	transitions	% reduction
1	120160	86296	28.18	72212	39.9
2	127770	94776	25.82	85454	33.11
3	74666	61746	17.3	53502	28.34
4	165678	119578	27.83	119908	27.62
5	111909	81645	27.04	70978	36.58
6	66189	49251	25.59	46769	29.34
7	159620	121466	23.9	114163	28.48

It is clear from the above table that STI performs much better than bus invert.

With SPEC2000 benchmarks:

SPEC2000 benchmark binaries traces were run with the proposed technique and compared with bus invert and gray coding. The 26 binaries were run with varying the buffer depth and bus widths with the values 8,16,32,64. The averages for a given combination of bus width and buffer depth were taken and have been plotted. The results for the proposed technique and bit invert are showed in Figure 6 and 7 respectively.

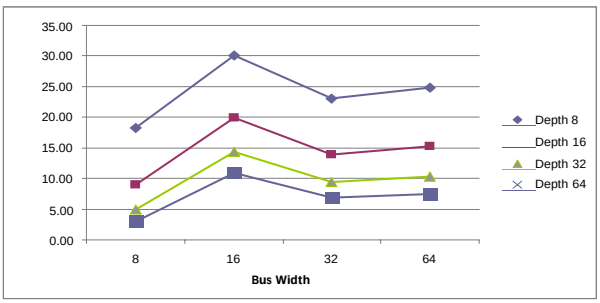

Figure 7: Transition reduction in Bus invert

It can be observed that the buffer depth does not make any changes to bus invert. Also with increase in buffer depth, the transition reduction reduces for the proposed technique. A similar observation can be made for bus invert when bus width is increased.

With increasing bus widths in present VLSI systems, the proposed technique will perform better.

The increase in buffer depth leads to a lesser reduction.

It can be offset by splitting the block into sub-blocks of smaller depths. Depending on the system, a compromise between power reduction and bandwidth utilization can be found. The benchmark files were also run using Gray Coding technique which is the reverse of transition signaling. The results are shown in Figure 8.

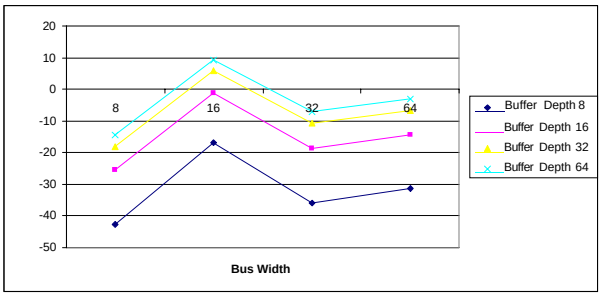

Figure 8: Transition reduction in Gray Coding

It can be seen that there is not much reduction in transitions, with some of the data points showing an increase in the number of transitions. Gray code does not show any reduction in transition since it is an N-bit to N-bit mapping. Whatever data tuples are in input set are exactly the same in the output set. So it is not possible to get any benefits out of Gray code. The same is the case with transition signaling.

Overall Power Analysis

The proposed system was designed in RTL and analyzed with Synopsys synthesis tools. A bus operating at 100MHz was assumed with its I/O voltage levels at 3.3v. The internal circuitry was modeled on 180nm process technology. The circuitry was simulated by feeding the SPEC2000 benchmark trace files as input for a buffer depth of 8. The power consumed by the circuitry is shown in Table 5.

Table 5: Power dissipation of encoding/decoding circuitry for transition inversion

Circuitry	Decision circuit	Encoder	Decoder
Power consumed	28.7µW	28.9µW	28.6µW

Assuming the parameters stated above and the

ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-12, 2017
DOI:10.21276/ijcesr.2017.4.12.31

activity factor for the benchmarks to be 0.5, the power was found to be 27.23mW. The reduction in power consumption is linearly dependent on the activity factor reduction. A reduction of 30% activity leads to a reduction of power by 8.17mW. The total power consumed by the extra circuitry is 86.2 µW leading to a

net power reduction of 8.08mW which corresponds to 29.7 % reduction in power.

C. Overall Delay Analysis

The proposed technique does not involve circuitry of multiple stages thus leading to less delay. The delay performance of the proposed technique and bus invert in terms of propagation speed is compared in Table 6.

Table 6: Comparison of encoder delay in Bus Invert and Transition Inversion

Technique	Proposed Technique	Bus Invert
Delay	1.2ns	3.3ns

For calculating the delay due to the proposed technique only the encoder is considered. The decision circuit is not taken as it will be part of the buffer loading delay and will not contribute to encoding. The decision circuit delay was found to be 0.2ns since it involves only the XOR gate. The flip flops delays will be masked by the sequential loading thus giving a pipelined approach. For the bus invert technique, the decision circuit delay is also taken into account because encoding has to be complete before the next data word arrives. Thus before the next data arrives the counting of the hamming distance should have been done and the data encoded. Overall the encoding delay of the proposed technique is considerably lesser compared to that of the bus invert.

6. Conclusions

In this paper an encoding technique has been presented that reduces power dissipated on off-chip data buses for block data transfer. The technique involves inverting the transition states on every line of the bus if the transitions exceed the number of non- transitions. The modification status is signaled as an extra word, thus avoiding the use of an extra line. The average reduction obtained in terms of transitions is 31.9% while the net power reduction after the extra power circuitry is taken into account is 29.7%. This is achieved without using an extra bus line. The compromise is in bandwidth utilization which can be adjusted by choosing a proper block length. This technique can also be applied to synchronous serial buses. The presence of a parity bit like error detection mechanism in addition to low power gives it an additional advantage.

7 References

[1] M. R. Stan, W. P. Burleson. “Bus-Invert Coding for Low Power I/O”, IEEE Transactions on Very Large Integration Systems, Vol. 3, No. 1, pp. 49-58, March 1995.

[2] L. Benini, G. De Micheli, E. Macii, D. Sciuto, C. Silvano. “Asymptotic Zero-Transition Activity Encoding for Address Buses in Low-Power Microprocessor-Based Systems”, IEEE 7th Great Lakes Symposium on VLSI, Urbana, IL, pp. 77-82, Mar. 1997.

[3] E. Musoll, T. Lang, and J. Cortadella. “Working- Zone Encoding f or reducing the energy in microprocessor address buses”. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 6, no. 4, Dec 1998

[4] E. Musoll, T. Lang, and J. Cortadella. “Exploiting the locality of memory references to reduce the address bus energy”, Proceedings of ISPLED, pp. 202-207, Monterey CA, August 1997.

[5] M. Stan and W. Burleson. “Low-power encodings for global communications in CMOS VLSI”, IEEE Trans. on VLSI Systems., pages 444–455, 1997.

[6] L. Benini, G. D. Micheli, E. Macii, M. Poncino, and S. Quer. “Power optimization of core-based systems by address bus encoding”, IEEE Transactions on Very Large Scale Integration, 6(4), Dec. 1998.

[7] Tomas Akenine-Moller, Jacob Strom. “Graphics Processing Units for Handhelds”, Proceedings of the IEEE Vol. 96, No. 5, pp. 779-789 May 2008.

[8] L. Benini, G. D. Micheli, E. Macii, D. Sciuto, and

C. Silvano. “Address bus encoding techniques for system- level power optimization”, In IEEE

Design Automation and Test Conference in Europe, pp 861–866, Paris, Feb. 1998.

[9] Giuseppe Ascia, Vincenzo Catania, Maurizio Palesi. “A Genetic Bus Encoding Technique for Power Optimization of Embedded Systems”, J.J. Chico and E. Macii (Eds.) PATMOS 2003, LNCS 2799, pp. 21–30, 2003. Springer-Verlag Berlin Heidelberg 2003.

[10]C. Su, C. Tsui, and A. Despain. “Saving power in the control path of embedded processors”, IEEE Design and Test of computers, 11(4):24–30, 1994

[11]Wei-Chung Cheng, Massoud Pedram. “Memory Bus Encoding for Low Power: A Tutorial”. ISQED 2001: 199-204

[12]Abinesh R., Bharghava R., M.B. Srinivas, “Transition Inversion Based Low Power Data Coding Scheme for Synchronous Serial Communication”, isvlsi, pp.103-108, 2009 IEEE Computer Society Annual Symposium on VLSI, 2009

[13]Jun Yang, Rajiv Gupta, Chuanjun Zhang. “Frequent value encoding for low power data buses”. ACM Trans. Design Autom. Electr. Syst. 9(3): 354-384 (2004)

[14]Saneei M, Afzali-Kusha A, Navabi Z. “Serial Bus Encoding For Low Power Applications”, International Symposium on System-On-Chip, pp. 1-4, November 2006.

[15]Kangmin Lee, Se-Joong Lee, Hoi-Jun Yoo. “SILENT: serialized low energy transmission coding for on- chip interconnection networks”. ICCAD 2004: 448-451