International Journal of Advanced Research in IT and Engineering

ISSN: 2278-6244

BIT TRANSITION REDUCTION TECHNIQUE FOR LOW POWER VLSI

S. Indira*

G. Senthil Kumar**

Abstract: *With ever increasing complexity of VLSI circuits and an increased focus on mobile computing, low power design techniques have become a must for all aspects of digital design Low power VLSI circuit design is one of the most important issues in present day technology. Developing low power circuits without affecting too much the performance (area, latency, and period) is a difficult task. For CMOS circuits, more power is dissipated as dynamic power for charging and discharging node capacitance. Low power design is obtained by minimizing the number of transitions inside CMOS circuits. Bus invert coding is a widely used technique for that. The proposed technique is superior to the bus invert coding technique. The simulation results show a considerable reduction on the number of transitions over and above that obtained with bus invert coding. Further, the proposed technique requires only 2 extra bits for the low power coding, regardless of the bit-width of the bus and does not assume anything about the nature of the data. The two extra bits are used to indicate that the data is in original format or inverted or left shifted or right shifted.*

The main idea in the proposed technique is to optionally shift the data bits by one bit position (either left-shift or right-shift) if the shifting reduces the number of bus transitions. The transition reductions are greater for the buses with arbitrary bus widths than buses whose widths area power of 2. The shift operations are inherently well suited for encoding buses with arbitrary widths which is generally the case for data buses, especially for the on chip data buses. This techniques is achieved by using one of the famous VLSI language called Verilog Hardware Description Language and simulation result is verified using tool "Modelsim- Altera 6.4a Starter Edition".

*Asst. Prof., Department of Electronics, S.N.R. Sons College, Coimbatore

**Prof. and Head, Department of Electronics, S.N.R. Sons College, Coimbatore

1. INTRODUCTION

Electronics took birth in 1897 when J.A. Fleming developed a vacuum diode. Electronics is a big bag of tricks. It finds applications almost everywhere. Tremendous innovations have been found in field of Electronics for the past few decades. The subsequent developments of 1980's paved the way for VLSI (Very Large Scale Integration) with number of transistors one to two magnitude higher. As a result, fabrication of any complex digital system within feasible cost had become a reality. Further, mixed signal circuits involving both digital and analog devices had become possible. The low cost consumer electronic devices (cell phone with camera, notepad, hand held low cost computing devices etc.) the required computational power of these applications is the driving force for the fast development of this field.

One of the ways of reducing power in a CMOS circuit is to reduce the number of transitions on the bus. Over the past few years, a number of coding schemes have been proposed for reducing the transition on a bus. For data buses, one popular coding scheme is the bus invert coding technique. This is a suitable technique for uncorrelated data patterns. Other variants of the bus invert coding include a partial bus coding technique and the decomposition approach.

Both these techniques have an area overhead to determine the suitable partition of the data bus. In addition, the decomposition approach can require up to p-1 extra lines on the bus where p is the number of partitions of the original data bus. The partial bus invert coding has the limitation that in-spite of the additional hardware, it might not utilize the bus invert coding for a subset of the bus lines, there by producing sub-optimal results. Further, the partial bus invert coding requires that one has the information apriority of the sequence of the memory address patterns on the bus.

The proposed technique does not have any additional area overhead in determining the transition correlations and transition probabilities. It does not need any prior knowledge of the address patterns. The data on the bus can be uncorrelated and completely random, just as was the case with the original bus invert coding. This scheme determines the number of bus lines that normally change state when the next output word is clocked onto the bus. When the number of transitions exceeds half the bus width, the output word is inverted or left shifted or right shifted before being clocked onto the bus. The number of extra bus lines

needed by this method is always 2 regardless of the bit width of the bus. The two extra bits are used to indicate that the data is original or inverted or left shifted or right shifted. The simulation results shows that shift invert coding technique provides considerable reduction on the number of transitions over and above that obtained with bus invert coding.

BLOCK DIAGRAM

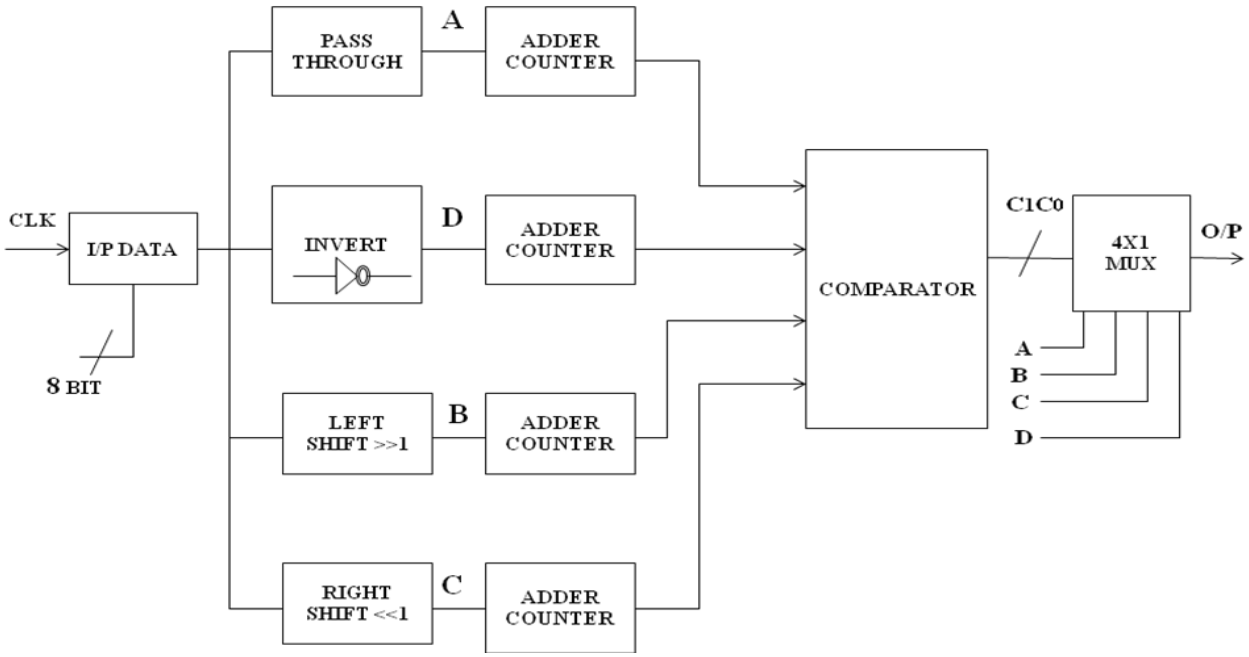


Fig1:Block diagram

The hardware realization for the proposed shift invert coding is shown in the figure. The inputs to the encoder are D^{K+1} and B^K where K denotes the time instance K. The bit width of D^{K+1} is eight and the bit width of B^K is ten since it includes the two control signals (C_1C_0) used to indicate the mode of encoding .

The blocks named **"Pass through" , "Left shift", "Right shift" and "invert"** indicate the four encoding schemes in our proposed technique. Each of these blocks has ten XOR gates. The basic functionality of an XOR gate is to produce an output of one whenever the inputs differs thus the XOR gate basically captures the transition between B^K and D^{K+1}.

The block " Pass Through" compares the un encoded(default) data bits D^{K+1} with the existing value on the bus ,B^K. The block "Left shift" compares the circular left shifted data bits with the bus contents B^K. It does not require any additional hardware. It can be done by mere re-adjustments of the data bits D^{K+1}. The block "Right shift" compares the circular right shifted data bits with the existing value on the bus B^K. This can be done by the re-adjustments of

the data bits D^{K+1} without any additional hardware requirements. The block termed "Invert" is used to get the transitions of the bit inverted original input bits. The inverter shown to the left of the inverter block is an eight bit inverter.

2. ENCODER UNIT

Each of the XOR blocks take 2, 10 bit inputs and generate one 10 bit output, one of the 2, 10 bit inputs is the signal B^K on the bus at time instant K. The signal B^K includes the two control bits (C_1C_0) used to indicate the scheme used at time instant K. The other input to the XOR blocks is the eight bit data signal D^{K+1} in some encoded form (depending on the encoding scheme). The bit width of D^{K+1} is only 8, and the other bits would come based on the encoding scheme used at instant K+1.

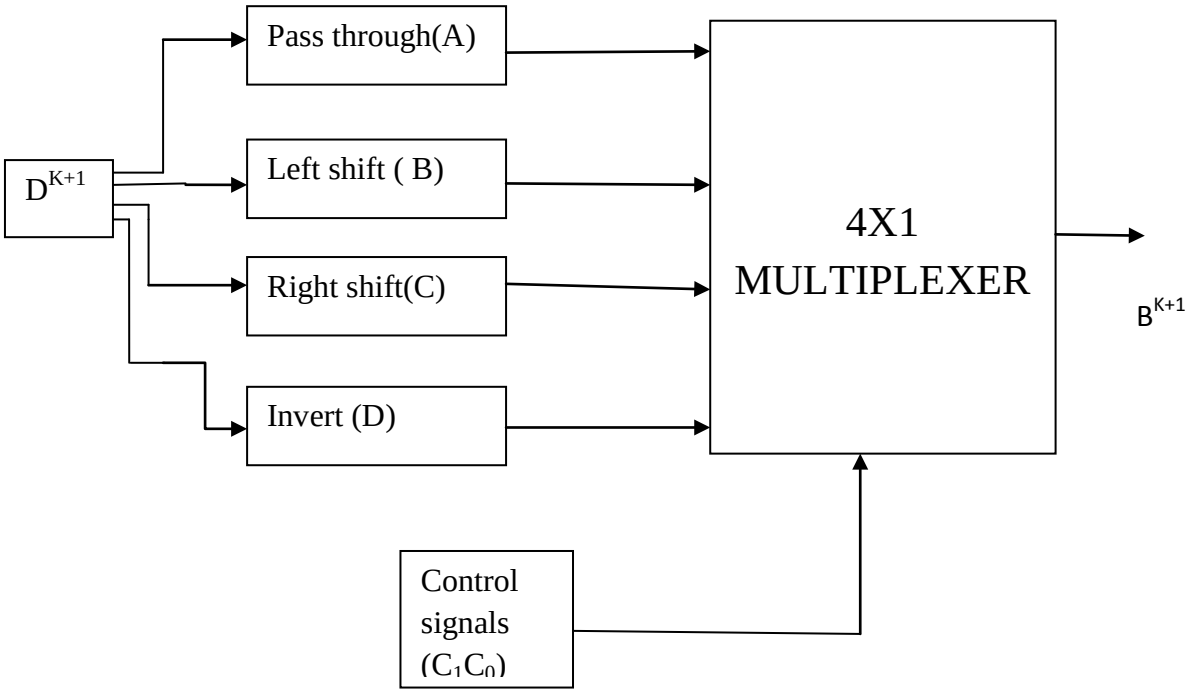


Fig2:Block diagram of selecting encoding scheme

3. BIT REPRESENTATION

S. No.	Encoded form	Control signals(C_1C_0)
1.	Default (no encoding)	00
2.	Left shift	01
3.	Right shift	10
4.	Invert	11

Table 1 Coding scheme.

The above table shows a bit representation to indicate the encoding schemem used. For example, if the input data is left shifted before we send it over the bus, the two control bits will be assigned the value 01. Likewise, the other bit assignments can be obtained from the table

4. ADDER COUNTER UNIT

The output bit of each of the XOR'ed 10-bit output is then added using a counter structure termed "10:4 ADDER COUNTER". The total number of 1's in the XOR output indicates the number of positions in which the two input vectors differ. The 10:4 ADDER COUNTER block takes 10 bits as input and generates a 4-bit output that indicates the total number of 1's in the input. This block consists of six full adder and three half adder circuit.

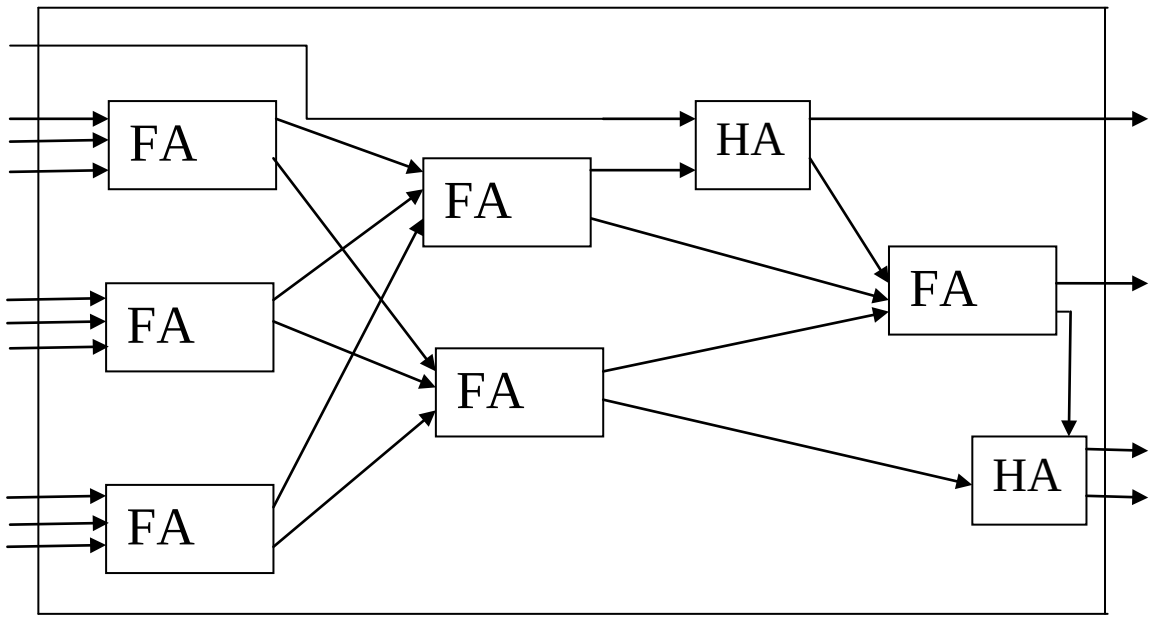


Fig3: 10:4 ADDER COUNTER CIRCUIT

The 10:4 ADDER COUNTER will generate an output that can be an output that can be anywhere in the range[0…10]. This implies that 10:4 ADDER COUNTER will generate a 4 bit output. These 4 bits indicate the total number of transitions on the bus for each type of encoding. The data bus is an 8 bits wide and adder counter block that took 10 input bits and generated a 4 bit output. In general, for a w bit data, it needs a (w+2) bit adder counter block that generates a $[\log_2(w+2)]$ bit output.

The proposed technique consists of four adder counter block. The input for each block from pass through, left shift, right shift and invert which are 10 bit data.The outputs from the four 10:4 adder counter blocks are sent to a 4-way 4- bit comparator which finds the mode that

has the least number of transitions. The comparator then generates the two control bits C_1C_0 that indicates the mode chosen for decoding at time instant K+1. The encoded data that has the least number of transitions and the control bits C_1C_0 are then sent over the data as B^{K+1}.

The comparator which compares four four bit data from adder counter block and generates value 00 if the data from pass through is less than other three blocks. It gives the value 01 if the data from left shift is less than other three blocks. Then it gives 10 if the data from right shift is less than other three blocks. Then it gives 11 if the data from inverted block is less than other three blocks.

The output from the comparator is two bit data which are control signals C_1C_0. This must be given to the 4x1 multiplexer. It selects any one of the four eight bit inputs given to multiplexer and passed to the output. The output is considered as previous data and given back to the input. In the next clock cycle, 8 bit input data is given and compared to previous data for the number of transitions. The cycle repeats for the number of data's to be transmitted.

In the reception, according to the control signals the received data to be decoded to original data. In the 10 bit data , lower eight bits considered as original data. The upper two bits are control signals which indicates the encoded form.

In the proposed technique, there is no assumptions on the nature of data on the bus. Completely random values for the data bus D^K were generated. The bit width and the time for simulation were passed as inputs. The bus invert coding using the same randomly generated data so as to compare the performance of shift invert coding with the respect to the bus invert coding.

The bit width increases, the shift invert coding results in a larger reduction in the average number of transitions per cycle. Thus, as applications go towards, 64- bits and beyond, the shift invert coding scheme will be more useful than the traditional bus invert coding. The number of extra lines for the shift inverts coding remains at 2 irrespective of the width of the data bus. The reduction in the power savings obtained by shift invert coding can more than offset this small increase in the hardware requirements.

5. SIGNIFICANCE OF SHIFT INVERT CODING

CMOS VLSI is intrinsically low power technology. The special interest in low power CMOS design is relatively recent and is due to an increased interest in portable applications. The power dissipated in a CMOS circuit can be classified into static power and dynamic power. The static power is drawn from voltage source attached to V_{DD} pins of a chip. The static power dissipated by a CMOS VLSI gate is in the nanowatt range.

CMOS logic dissipated less power than NMOS logic because CMOS dissipates power only when switching ("Dynamic Power"). The dynamic power is required to charge and discharge load capacitances when transistor switch. In one complete cycle of CMOS logic, current flows from V_{DD} to load capacitance to charge it and flows from the charged load capacitances to ground during discharge. In one complete Charge/ Discharge cycle total of $Q=C_LV_{DD}$ transferred from V_{DD} to ground. Multiply by switching frequency on load capacitances to get current used and multiply by voltage again to get characteristics switching power dissipated by CMOS device $P=CV^2f$.

Power is consumed because of charging and discharging of a coupling and load capacitance due to transition of a signal on data bus. Reducing the transition activity or switching activity on the on-chip data buses is the one of the attractive way of reducing power dissipation. Switching activity on the data bus can be reduced by employing bus encoding techniques. Several bus encoding techniques have been proposed to reduce power consumption during bus transmission

The main idea in the proposed technique is to optionally shift the data bits by one bit position (either left shift or right shift). If the shifting reduces the number of bus transitions. The left shift operation on a W-bit data as follows.

$d_1^{(LS)} = d_{i-1}$

$d_0^{(LS)} = d_{w-1}$ $\quad 1<= I < W$

it is a circular left shift operation. So there is no loss of information from the original data.

The right shift operation is defined similar to the left shift as follows.

$d_1^{(RS)} = d_{i+1}$

$d_{w-1}^{(RS)} = d_0 \quad 0<= I < W-1$

For Example

Let B^K = 1010 1101 is an eight bit data

ISSN: 2278-6244

And the new data at time K+1

D^{K+1} = 1001 0110

The number of transitions between previous data and new data are **five.**

In the existing Bus Invert Coding technique, the new data are inverted to see whether it is beneficial i.e. whether the number of 0 to1 and 1 to 0 transitions are reduced to send inversion of new data over the bus.

B^K = 1010 1101

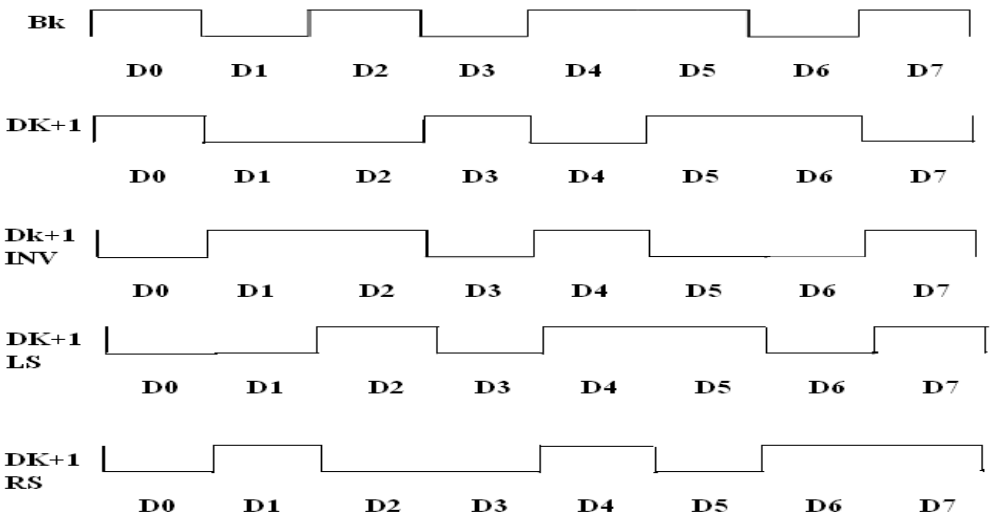
$D^{K+1(INV)}$ = 0110 1001

The number of transitions N^{INV} between B^K and $D^{K+1(INV)}$ is **three.** Since N^{INV} < N in the Bus Invert Coding Technique, $D^{K+1(INV)}$ will be sent over the bus at time K+1.

In the proposed technique, the new data should be left shifted or right shifted to reduce the number of transitions. The new data is left shifted and denoted as $D^{K+1(LS)}$ at time K+1.

B^K = 1010 1101

$D^{K+1(LS)}$ = 0010 1101



The number of transitions N^{LS} between B^K and $D^{K+1(LS)}$ is **one** which is better than three transitions that one gets from the inverted data $D^{K+1(INV)}$. Thus , in this case, it is clear that by sending left shifted data, we can reduce the number of transitions even further than the reduction obtained from sending the inverted data.

The rationale behind using the shift operation is simple. By shifting the data at time K+1, it is possible that the bit values could match in more places with the existing data on the bus at time k than if the data bits were either inverted or left unchanged. The matching of bits in

more places implies fewer transitions, thereby giving better solution. The above example reveals a left shift operation is better than inversion. Similarly, the data can be right shifted which reduces the number of transitions. Finally, we conclude that one can further reduce the number of transitions either by performing a left shift or right shift operations.

It is obvious that shifting left or right will not always reduce the number of transitions. Depending on the values of B^K and D^{K+1} , it is possible that either the inverted data $D^{K+1(INV)}$ or may be even the unmodified/original data D^{K+1} gives the least transitions when sent on the bus. For each new data that needs to be sent over the bus, we evaluate the transitions N, N^{INV}, N^{LS}, and N^{RS} between B^K and D^{K+1}, $D^{K+1(INV)}$, $D^{K+1(LS)}$ and $D^{K+1(RS)}$ respectively. We then choose the encoding that results in the least number of transitions.

The steps of the proposed technique are

Inputs are D^{K+1} and B^K

Output is $B^{K+1(SHIFT\_INV)}$

N: number of transitions between D^{K+1} and B^K

N^{INV}: number of transitions between $D^{K+1(INV)}$ and B^K

N^{LS}: number of transitions between $D^{K+1(LS)}$ and B^K

N^{RS}: number of transitions between $D^{K+1(RS)}$ and B^K

B^{K+1} : one of (D^{K+1} ,$D^{K+1(INV)}$,$D^{K+1(LS)}$, $D^{K+1(RS)}$) depending on minimum of (N, N^{INV}, N^{LS}, N^{RS})

Any one of the above data can be sent over the bus. With the data , additional two bits are added to indicate the coding that was used. This will be used to decode the bus value appropriately at the receiving end. Thus, in shiftinv coding, the width of the bus $w' = w+2$, where w is the width of the data vector and we use 2 additional bits as compared to 1 additional bit in default Bus invert coding.

6. CONCLUSION

The Shiftinv Coding is a new method for bus encoding using left shift and right shift operations. This technique is a simple yet efficient scheme that enhances the Bus Invert Coding technique. This technique reduces the number of transitions in the data and dynamic power dissipation by charging the various capacitances. On completely random data, the simulation results suggest that the proposed ShiftInv Coding reduces the average number of transitions over and above that given by the existing technique Bus Invert Coding.

The proposed coding scheme does not have too much area overhead and it does not assume any correlation between the data values. Further, the proposed technique uses only two extra lines regardless of the width of the data bus. The technique will reduce the total bus energy and not just the number of bus transitions. From the simulation results, the savings are more for smaller bus widths and less for larger bus widths. Shiftinv coding is an extension of the Bus Invert coding does not perform well for larger bus widths.

Since the coding for ShiftInv coding has been done using Verilog HDL language and the output is verified successfully. In future, it can be modified for special purposes applications in which one may get even more savings.

REFERENCES:

[1] M. R. Stan and W. P. Burleson, "Bus-Invert coding for lowpower I/O", IEEE Trans. on VLSI, March 1995, vol. 3, pp. 49-58.

[2] S. Ramprasad, N. R. Shanbag, and I. N. Hajj, **"**A coding framework for low power address and data busses"**,** IEEE Trans. on VLSI Systems, June 1999, vol. 7, pp. 212-221.

[3] C. L. Su, C. Y. Tsui, and A. M. Despain, **"**Saving power in the control path of embedded processors**",** IEEE Design and Test of Computers, 1994, vol.. 11, no. 4, pp. 24-30.

[4] L. Benini, G. De Micheli, E. Macii, D. Sciuto, and C. Silvano,**"**Asymptotic zero-transition activity encoding for address buses in low-power microprocessor-based systems**",** Great Lakes VLSI Symposium, Urbana IL, March 1997, pp. 77-82.

[5] L. Benini, G. De Micheli, E. Macii, M. Poncino, and S. Quer,**"**System-level power optimization of special purpose applications: The beach solution**",** *Proc.* Int. Symp. Low Power Electronics Design, August 1997, pp. 24-29.

[6] S. Hong, U. Narayanan, K.S. Chung, and T. Kim, **"**Bus-Invert Coding for Low-Power I/O – A Decomposition Approach**",** Proc. 43rd IEEE Midwest Symp. Circuits and Systems, August 2000.

[7] Y. Shin, S.I. Chae and K. Choi, **"**Partial Bus-Invert Coding for Power Optimization of Application-Specific Systems**",** IEEE Trans. on VLSI Systems, April 2001, vol. 9, pp. 377-383.

[8] P. P. Sotiriadis and A. Chandrakasan, **"**Bus energy minimization by transition pattern coding (TPC) in deep submicron technologies", Proc. 2000 IEEE/ACM Int. Conf. Computer-Aided Design, November 2000, pp. 322-328.

[9] Tina Lindkvist, Jacob Lofvenberg, Oscar Gustafsson "Deep Sub-Micron Bus Invert Coding", Proceedings of the 6th Nordic Signal Processing Symposium - NORSIG 2004 June 9 - 11, 2004, Espoo, Finland.