REDUCTION OF TRANSITIONS ON OFF-CHIP DATA BUSES USING ADAPTIVE BUS ENCODING

[1]Myakala Mounika, [2]Deepthi Amuru
[1]M.Tech student, [2]Asst. Professor
[1,2]Department of Electronics and Communication Engineering
[1,2]G.Narayanamma Institute of Technology and Science (for women), Hyderabad, India

***Abstract*:** In CMOS circuits more power is dissipated during charging and discharging the load capacitance. This can be lowered by minimizing the number of transitions inside CMOS circuit. This paper includes some encoding techniques to reduce the transition activity. An Adaptive Encoding technique is one among them. It aims to reduce transition activity in high capacitance off-chip data buses, since they are the main source of power dissipation. The entire architectures are programmed using Verilog. The coding is done on Xilinx ISE 14.7 and simulation is done using Xilinx ISim simulator and Modelsim.

***Index terms*: Power dissipation, high capacitance, switching activity, number of transitions, off-chip data buses.**

I. INTRODUCTION

Power optimization is important to achieve high reliability. The important requirement is to know for particular application how much power the circuit will dissipate. With the increase in speed and complexity, there is an increase in power consumption. Power consumption is proportional to switching activity. By reducing the bus switching, bus power can be reduced.

The chips between different functional blocks are called on-chips and the chips between different IC's or PCB are called off chips. In micro processor related systems load capacitance of off-chip buses is orders of magnitude greater than that of internal nodes. Power dissipation on these buses mainly occurs during signal transitions. Reducing these signal transitions will reduce the power dissipation.

Power dissipation is classified as static power dissipation and dynamic power dissipation. Static power dissipation is due to leakage currents and dynamic power dissipation is due to switching activity. Switching activity is the probability of transitions ref [4]. Dynamic power dissipated by CMOS circuit is given by,

$$P_{dyn} = \sum_{i=1}^{N} C_{Li} \times V_{dd}^2 \times f_{CLK} \times \propto_i \qquad (1)$$

where V_{dd} is the supply voltage, C_{Li} is load capacitance, f_{CLK} is chip clock frequency, $\propto_i$ is activity factor. To reduce the power dissipation one of the above factors must be minimized. V_{dd} cannot be lowered because of required threshold voltages of pMos and nMos transistors. Clock is turned off for the portions of chip which are not used. Load capacitance is the gate capacitance of transistor it is limited during fabrication according to the technology. Thus, reducing activity factor is the best way to reduce overall power. Lowering activity factor is reducing number of transitions (at behavioral domain) or reducing glitches (unwanted transitions) at logic level.

Transitions on the buses can be reduced by using some bus encoding techniques. In address buses, the addresses are consecutive and are sequential. In data buses the contents are random. Address and data bus encoding are classified as

1. Adaptive techniques. 2. Non adaptive techniques.

If the signal characteristics are known, then non adaptive techniques are used, these techniques are mainly used for address buses. Some of the non adaptive techniques include gray code, T0 code, Dual T0, T0_BI etc.

For data buses since the contents are random there is no prior knowledge of data available, therefore adaptive techniques are used for data buses. Adaptive techniques include Bus invert coding (BIC), Bus shift coding (BSC), Partial bus invert coding (PBIC), Adaptive partial bus invert coding (APBIC), FVMSB (Frequent Value Most significant Bit), FV-MSBLSB (Frequent Value Most significant Bit Least significant Bit) ref [1-4].

In this paper, proposed technique is Adaptive Bus Encoding (ABE), here data is observed over time. In observation window signal statistics are known from a set of bit lines which have highly correlated switching patterns in that window.

Rest of this paper is organized as follows. In section II, Bus Invert Coding (BIC) is defined. In section III, Bus Shift Coding (BSC) is explained. Section IV includes Adaptive Bus Encoding (ABE). Section V, includes clock gating of ABE. Section VI, include results and discussions. Section VII includes overall results. Section VIII includes, comparison of different encoding techniques. Section IX, concludes the paper.

II. BUS INVERT CODING (BIC)

Bus invert coding is also called "Starvation coding" and "limited weight coding". It is one of the adaptive encoding methods suitable for uncorrelated data patterns. It reduces the number of transitions from n to n/2. BIC codes the I/O which decreases the bus activity which in turn decreases I/O peak power dissipation by 50% and I/O average power dissipation by 25%. Buses have

very large capacitance associated with them, therefore dissipate lot of power. The devices in I/O pads are needed to be large, so that they can drive the large external capacitances. These large capacitances have two effects namely long delays and high power dissipation.

The simplified model has two types of nodes. Small capacitance nodes (internal nodes) for internal circuits and large capacitance nodes (external nodes) for I/O pins. Total power is the sum of power dissipated by the internal circuit and the power dissipated at the I/O. It is given by the equation below.

$$P_{chip} \propto C_{int}.N(transitions)_{int} + C_{I/O}.N(transitions)_{I/O} \quad (2)$$

Internal number of transitions, $N(transitions)_{int}$ are much larger than transitions on I/O pins, $N(transitions)_{I/O}$ this is because the number of internal nodes are more when compared to number of external nodes and, internal capacitance C_{int} is much smaller than external capacitance $C_{I/O}$. Coding is done on data in order to decrease number of transitions on large capacitance side i,e (I/O) ref [4].

Figure 1 represents BIC circuit. There are 16 XOR gates in which 8 gates are for inverting the bus content and remaining 8 for comparing present bus values with previous data value. Fig. 2 represents majority voter circuit. The digital voter is implemented as a tree of full-adders. Based on hamming distance majority voter circuit decides whether to invert or not the next data value.

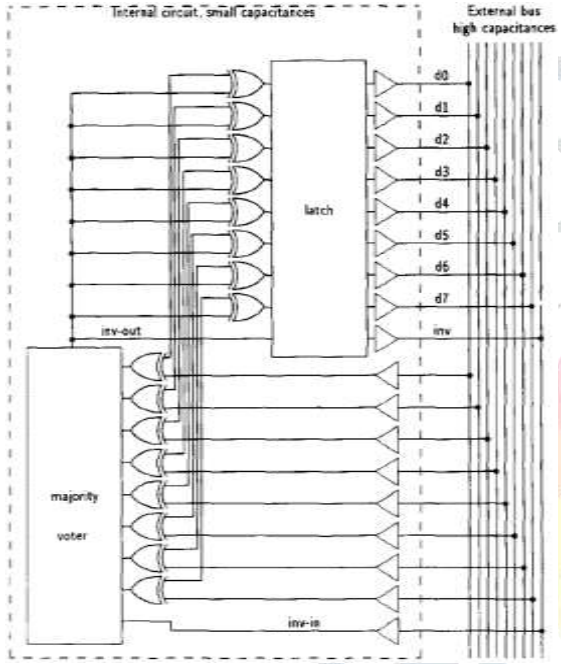

Fig 1: BIC circuit

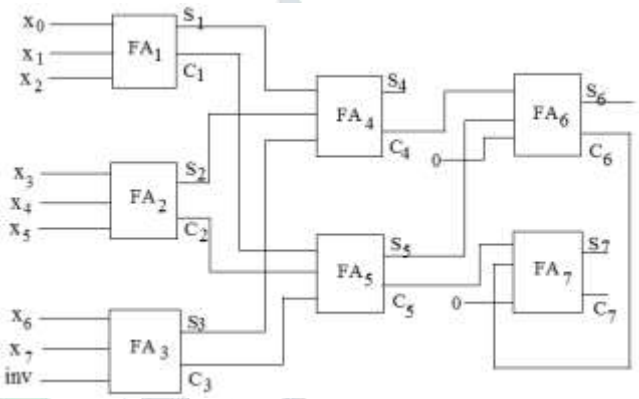

Fig 2: Majority voter circuit.

2.1 Operation of BIC:

1) Compute the hamming distance between present bus value (also considering present invert line) and next data value.
2) If hamming distance is greater than n/2, set invert=1, invert next word and send inverted word on the bus.
3) Otherwise, send the next word as it is, and set invert=0.
4) Receiver first looks at the invert line, if it is 1 the inverted value of data found on bus is taken, otherwise the same received value is taken.

BIC uses some extra circuitry on data paths, which require extra area. This method also requires high control bit count which makes this method expensive. Power saving in average cases using BIC is not satisfactory.

III. BUS SHIFT CODING (BSC)

BSC is modification of BIC. BIC reduces the peak power by 50% and average power is reduced by 25%. Power saving in average cases using BIC is not satisfactory. Therefore BIC is modified to reduce the average power, resulting Bus Shift Coding (BSC).

Bus Shift Coding shifts the data circularly in order to reduce transitions. Let a word is to be transmitted, and if the transitional difference between the present word and its consecutive word is more, it can be reduced by circularly shifting the word that is to be transmitted. This minimizes the activity factor.

Consider two words, present word as "presword" and next word. The following operation is to be performed to determine how many bits of next word are to be shifted.

3.1 Operation of BSC:

1) For x = 1 to n, shift the next word by i-bits on left and calculate hamming distance between the shifted word and present word.
2) Let h_{min} denote the number of bits that result in minimum hamming distance.
3) Shift next word by h_{min} bits, and transmit the shifted word.
4) Send h_{min} as a binary number to receiver on control lines to make it know how many bits are to be shifted to retrieve the original data.

These encoding schemes mentioned have the knowledge of the data statistics to be transmitted. For many applications, benchmark data streams are not available or the streams have non-stationary statistical parameters such as a time varying switching activity on the bus lines. For these cases static encoding schemes are inefficient. Rather, encoding methods which have an automatic adaptation to (time-varying) statistical parameters of data streams are needed.

IV. ADAPTIVE BUS ENCODING (ABE)

Proposed technique is ABE. Since the data characteristics are random in nature and can't be known earlier, adaptive encoding techniques are used to code data buses. To reduce the transitions the best way is to extract the signal statistics before encoding. This can be done by observing data over time. ABE encode data before transmitting it to off-chip bus so that average power dissipation is reduced. The technique is determined by observing the data characteristics of each bit line over fixed window sizes and formation of transition correlation among the bit lines. A cluster is formed with in an observation window with bit lines which are having more correlation among them.

4.1 Concept of Adaptive Bus Encoding

In an observation window, one bit line is selected as a basis line, the line that is selected as basis line should have maximum correlated switching transitions with the remaining lines. The lines which have maximum correlation with the basis are clustered together. XOR all the lines in cluster with basis line, it leads to maximum switching savings.

Let d_i^t and d_i^{t-1} denote signal values on bit line b_i at two different times 't' and 't −1'. Transition T_i^t at particular time 't' on bit line "b_i" can be represented as

$$T_i^t = \begin{cases} 1, & \text{if } d_i^t \neq d_i^{t-1} \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

At time t, XOR operation is performed between bit lines b_i (which is chosen as basis) and b_j (a bit line other than basis). The result of this XOR operation leads to three scenarios of switching. They are switching savings, switching increase, and no switching. Let symbol $\propto_{i,j}^t$ denote the three switching scenarios.

$$\alpha_{i,j}^t = \begin{cases} 1, & \text{if } T_i^t = 1, T_j^t = 1 \\ -1, & \text{if } T_i^t = 1, T_j^t = 0 \\ 0, & \text{if } T_i^t = 0, T_j^t = 0/\ 1 \end{cases} \quad (4)$$

or

$$\alpha_{i,j}^t = T_i^t \times (2 \times T_j^t - T_i^t) \quad (5)$$

The above equation is for one clock cycle. This is done for 'W' transitional clock cycles. The below equation says weather line b_j is to be included in cluster C_i or not.

$$\propto_{i,j} = \sum_{t=0}^{W-1} \alpha_{i,j}^t \quad (6)$$

$b_j \in C_i$ iff $\propto_{i,j} > 0$

If say i^{th} line is chosen as basis for one particular observation window. The total switching savings (ρ_i) from that observation window due to the selection of i^{th} bit line as basis is defined by Eq.7.

$$\rho_j = \sum_{j=0, j\neq i}^{N-1} \frac{1}{2}(\text{sgn}(\propto_{i,j}) + 1) \times \propto_{i,j} \quad (7)$$

Similarly calculate ρ_i for every bit line [for i= 0 to $(N-1)$] assuming it as basis. After calculating ρ_i for all lines choosing each line as basis, max (ρi) [for i = 0, 1. . . $(N-1)$] defines the basis among all bit lines. It is given by Eq.8.

$$b_k = \text{argmax}_i(\rho_i),\ 0 \leq i \leq (N-1) \quad (8)$$

From Eq.8, it can be noted that k^{th} line is basis for that observation window.

4.2 Algorithm of Adaptive Bus Encoding.

Procedure for selecting bit line as basis for every observation window can be written as follows

1) Find the total number of switching transitions (s_{ini}) in all the bit lines
2) For i= 0 to (N − 1)
   a) Choose i^{th} line as basis b) For j = 0 to (N − 1), j≠ i
3) Put j^{th} line in cluster if j^{th} line has more switching transitions, then j^{th} line is XORed with basis line
4) XOR all the clustered bit lines with the basis line
5) Find number of switching transitions (s_i) in this modified set
6) Savings $(\rho i) = s_{ini} - s_i$
7) If b_k= argmax$_i$ (ρi), 0 ≤ i≤ (N − 1), then the k^{th} bit-line is basis line for that observation window.
8) end

Flow chat in fig.3 is diagrammatic representation of the algorithm of ABE technique.

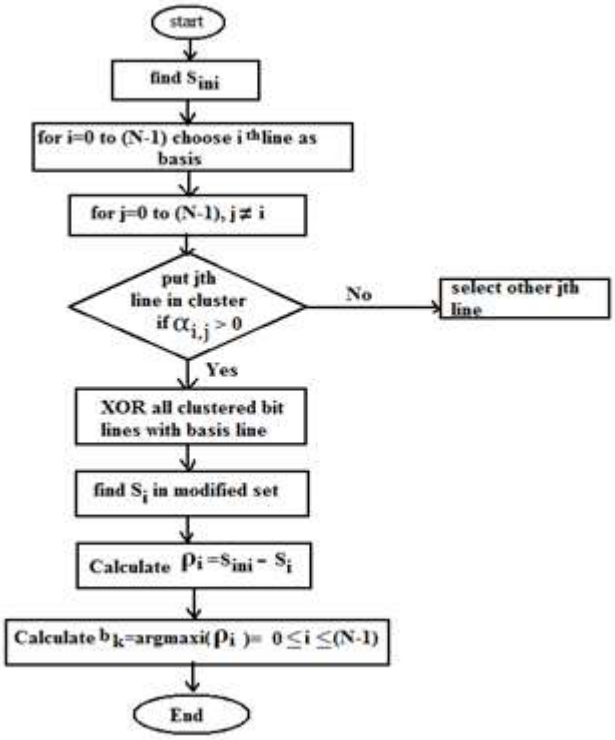

Fig 3: Flow chart for choosing basis line.

Table 1 shows impacts on b j while performing bi ⊕ b j and probabilities of different outcomes at different switching times where "p" denotes the local transitional probability.

Table 1: Probabilities of different switching scenarios

T_i^t	T_j^t	Different scenario	Probability χ
1	1	Switching savings	$\chi_1=p_i \times p_j$
1	0	Switching increase	$\chi_2=p_i \times (1-p_j)$
0	0	No switching change	$\chi_3=(1-p_i) \times (1-p_j)$
0	1	No switching change	$\chi_4=(1-p_i) \times p_j$

Probability(χ) of the number of switching savings in any particular window "W" from bit line "b_j" is expressed as

$$\chi_{W,k}^{i,j} = \sum_n \left(\binom{W}{n-1+k} \times \binom{W+1-n-k}{n-1} \right) \times Z \qquad (9)$$

where $Z=(\chi_1^{k+n-1} \times \chi_2^{n-1} \times \chi_5^{W-k-2(n-1)})$, and summation limits are different depending on "k". If "k" is odd, the limits are from n =1to ((W/2)-(k-1/2)) and if k is even, the limits range from n=1to ((W/2)-(k/2)+1). Probablility of saving $\chi_{W,k}^{i,j}$ says whether the bit line b_j is there in cluster along with b_i or not. Cluster formation in every observation window is done by the comparision of local swiching probability "p_i" of bit line b_i with mean switching probability. It is given by equation 10.

$$p_i \geq \frac{1}{N} \sum_{j=1}^{N} p_j \qquad (10)$$

From above equation it can be noted that if local swiching probability "p_i" is greater than mean value, then we can say that the bit line is present with in the cluster.

4.3 Block diagram of Adaptive Bus Encoding (ABE)

Block diagram of this technique has three main blocks namely encoder block, Serial Input Parallel Output block and decoder block. They are shown in Fig.4.

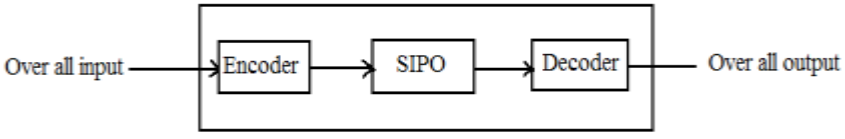

Fig 4: Block diagram of Adaptive Bus Encoding

A. Encoder block

Encoder block of ABE consists of decision block, delay block, XOR bank and multiplexer. Decision block consists of eliminator, cluster formation, and basis selection unit. Encoder block of ABE is shown in Fig.5.

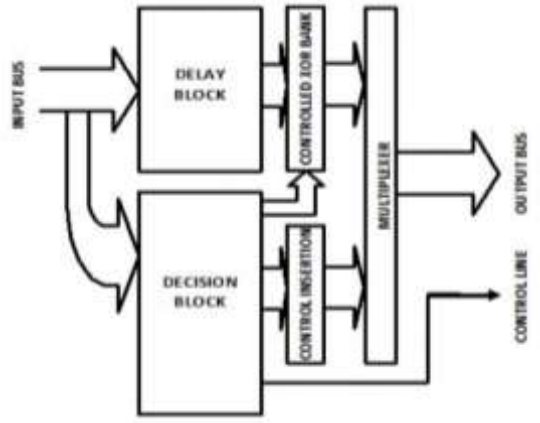

Fig 5: Encoder block

i) Decision block:

Decision block consists of eliminator block, bit wise saving computational unit and Basis selection unit which are connected as shown in Fig.6.

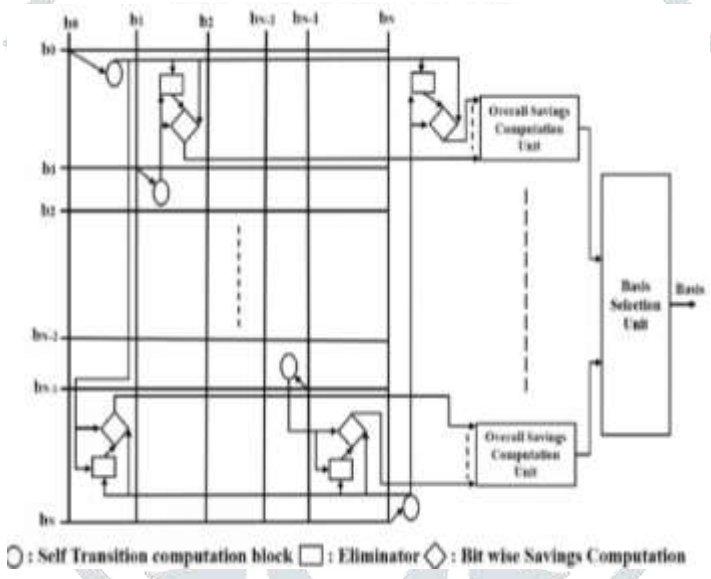

Fig 6: Decision block

a) Eliminator block:

Eliminator block eliminates unnecessary switching by reducing the internal node switching of encoder. Before computing the switching scenario ($\alpha_{i,j}$), some bit lines can be eliminated from cluster by this block. Eliminator block also gathers potential bit lines as basis among all the other bit lines, thus decreasing the internal switching count. Remove bit lines which have switching probability less than 0.25 for being chosen as basis. Because these lines have less probability of being selected as the basis; and eliminating them earlier leads to power savings in encoder.

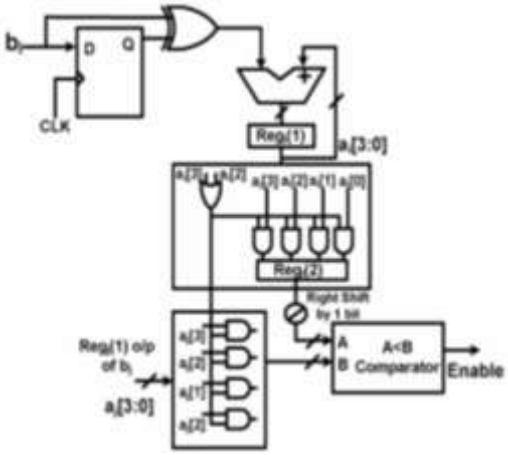

Fig 7: Eliminator block

In fig.7 Reg $_i$ or Reg $_j$ stores the self-switching count. The number of these registers is equal to bus width. Let b_i be the basis bit line which is given to D-flipflop. Self switching count from register Reg $_i$ is right shifted by 1-bit, say it as "A". Self switching count from register Reg $_j$ be "B". Comparator compares A and B. Output of comparator is enabled signal. It becomes high only when "A< B", when this condition is satisfied then computation of $\propto_{i,j}$ is enabled. $\propto_{i,j}$ is positive when switching count of b_j is greater than half of the switching count of basis.

Enabled output is same as d_{in} but it is delayed version of d_{in}. Enable line enables the computation of $\propto_{i,j}$.

b) Bit wise saving computational block:

From Fig.8 it can be seen that basis line b_i and bit line b_j are given to two D-flipflops. The enable output of comparator in eliminator block enables the computation of $\alpha_{i,j}$. This enabled output is applied to AND gate and other input of AND gate is from shifter. The 4-bit register here stores the combined transition count of basis line b_i and bit line b_j. For subtractor one of the inputs is the switching count of basis for that particular window.

.

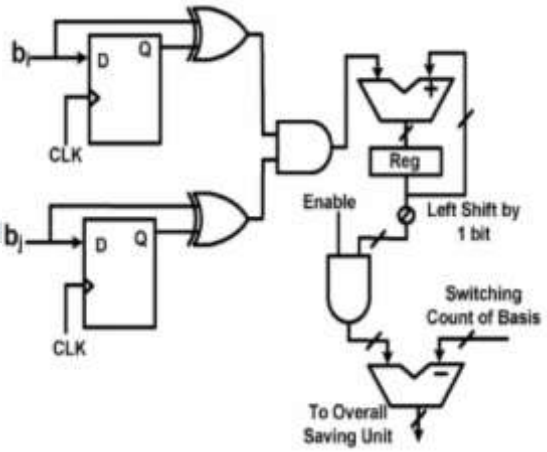

Fig 8: Bit wise saving computation unit.

Output from this block is given to overall saving unit. Overall saving computation unit computes total switching savings (ρ_i) when i^{th} line is chosen as basis line [eq (8)]. Basis selection unit (BSU) takes output from overall saving unit and selects one of the bit lines as basis line

In encoder architecture, the output of XOR bank is encoded output. This is applied to multiplixer along with decision block output including control insertion. Output of encoder will be encoded output or enabled output based on selection line of multiplexer.

B. DECODER BLOCK

Decoder recovers the data back to its original form. For this reason at beginning of encoded data the cluster information is sent to decoder as control signal for every observation window. Before decoding, the decoder extracts this cluster information by observing the transition between control signal and encoded data of previous observation window. This information is kept in register for each bit line until the end of decoding for current window. Fig.9 shows the decoder block of ABE.

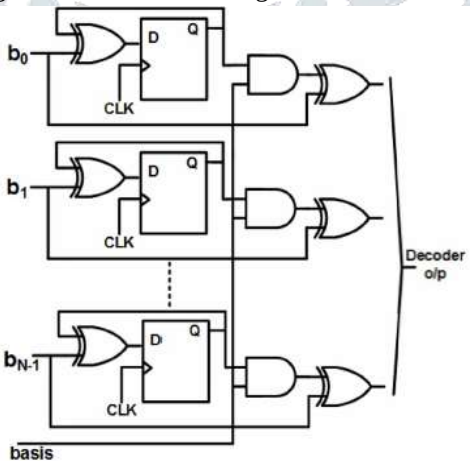

Fig 9: Decoder block

V. CLOCK GATING OF ABE

Switching states consume more power. Power dissipation is proportional to switching frequency of the device. By reducing number of switchings power dissipation can be reduced. Switching is saved by using gated clock instead of normal clock. Clock gating is a technique used for reducing power dissipation by reducing the clock power consumption by eliminating ideal clock cycles. It saves power by adding more logic to the circuit.

Fig.10 shows the block diagram of gated clock. Clock is disabled for that portion of the circuitry where data is similar, so that the flip-flops in them do not have to switch. When not being switched, the switching power consumption goes to zero, thus the power dissipation is reduced. This gated clock is given as clock to all the blocks. When the gate is high, the operation is enabled, and when it is low the operation is disabled.

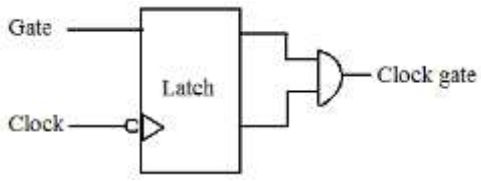

Fig 10: Clock gating

VI. RESULTS AND DISCUSSIONS

i) BIC results

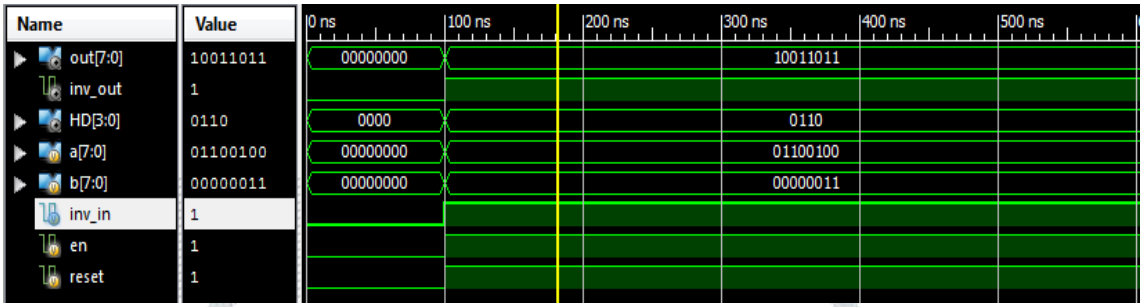

Fig 11: Simulation result of BIC

In fig.11 let "a=01100100" and "b=00000011" be two 8-bit input data. HD is the sum of hamming distance between "a" and "b" and the invert line which is 6. As the hamming distance between a and b is > $\frac{n}{2}$ the invert output is 1, and the output is inverted form of "a".

ii) BSC results

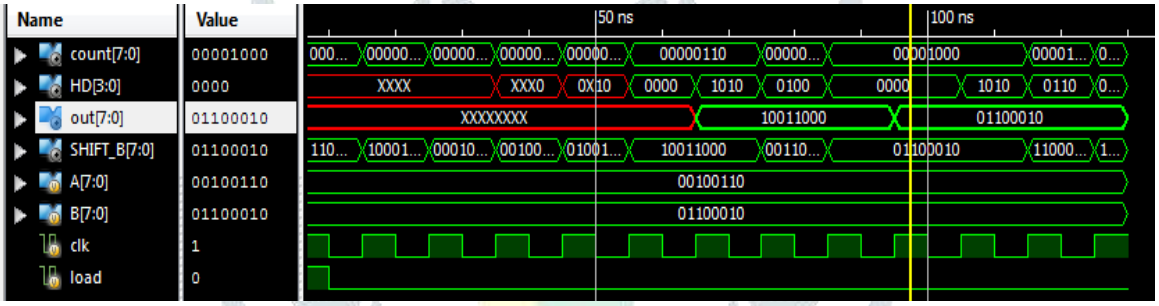

Fig 12: Simulation result of BSC

In fig.12 present word= 00100110 and next word= 01100010, the hamming distance is 2. To get the minimum hamming distance (i,e is 0) shift the next word by 4 bits on the left. Transmit the shifted word on data bus and 100 (binary equivalent of 4) on the control lines.

iii) Results of ABE

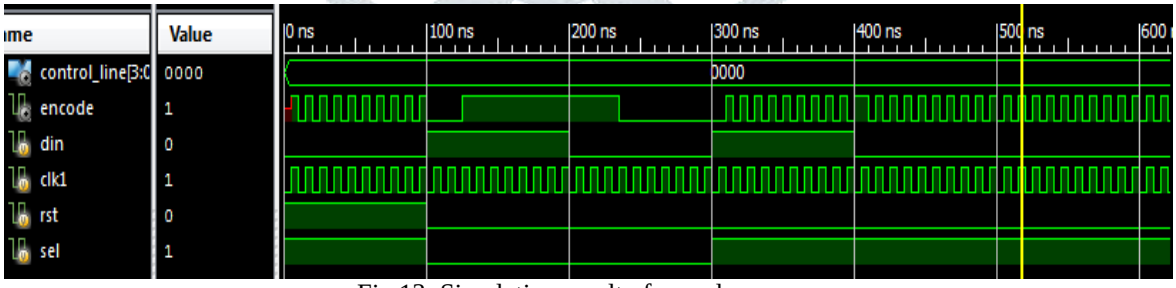

Fig 13: Simulation result of encoder

Encoded output depends on selection line of multiplexer. When sel=0 the output is from decision block and when sel=1 the output is from xor bank. When d_{in} is low the encoded output is "1-0",when d_{in} is high the encoded output is " 0-1", these are the cases when sel=1, when sel=0 the output is enabled one with some delay.

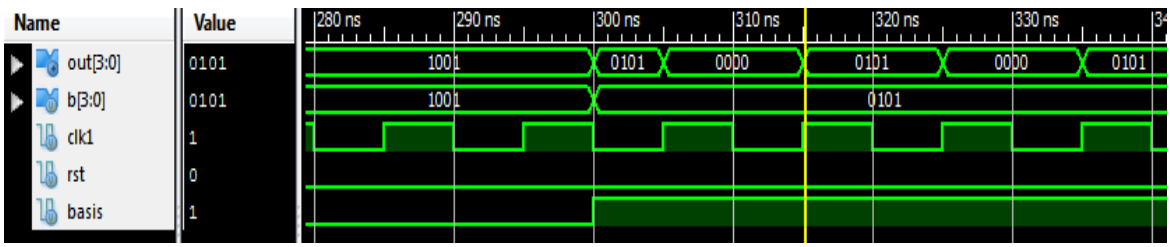

Fig 14: Simulation result of Decoder.

The basis line from encoder is given to decoder. The decoder output not only depends on input but also on basis line.

iv) Results of ABE after clock gating

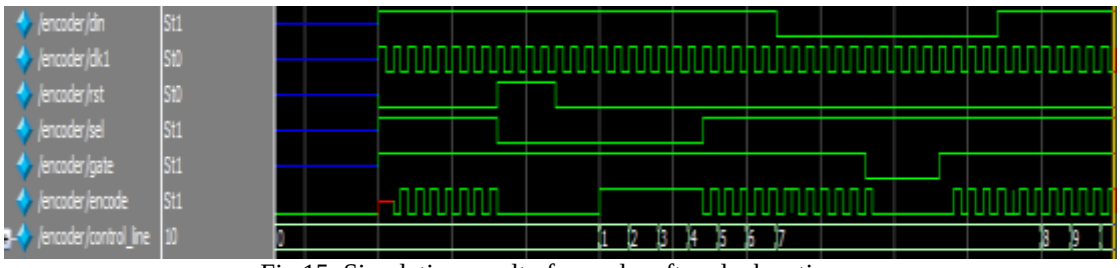

Fig 15: Simulation result of encoder after clock gating

In fig.15 when gate is high then only the operation occurs. When input "d_{in}" is "0" then the encoded output is "1-0",when d_{in} is "1" then the encoded output is " 0-1". When sel=0 the output is from decision block, it is an enabled output. When sel=1the output of encoder is from XOR bank, it is encoded output. The output is delayed here.

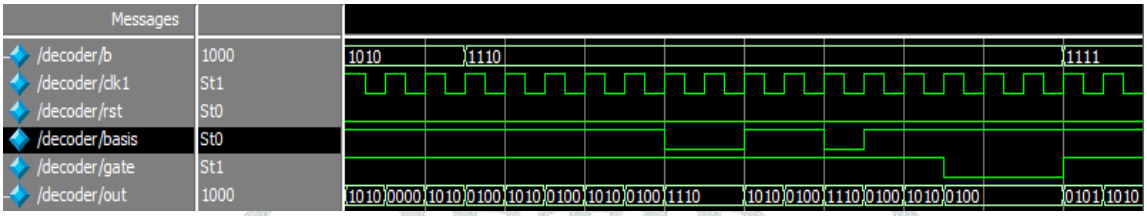

Fig 16: Simulation result of decoder after clock gating

The line which is chosen as basis is applied to decoder along with encoded output. In above figure when basis is high then output is decoded one. When basis is low decoded output is same as decoder input "b". These all operations are performed only when gate is high.

VII. OVERALL RESULTS:

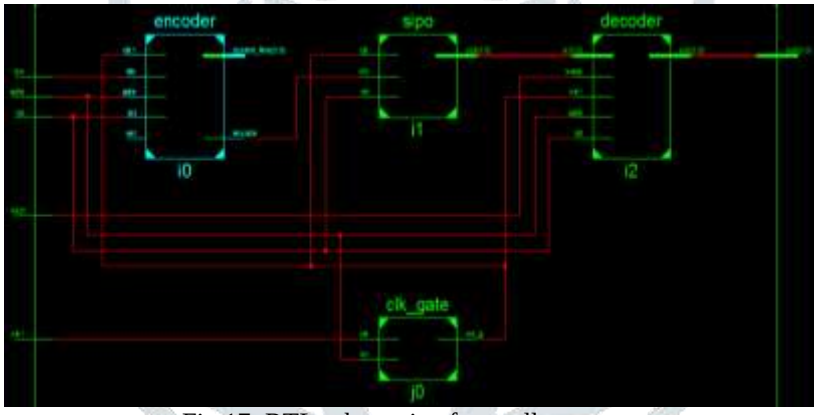

Fig 17: RTL schematic of overall process.

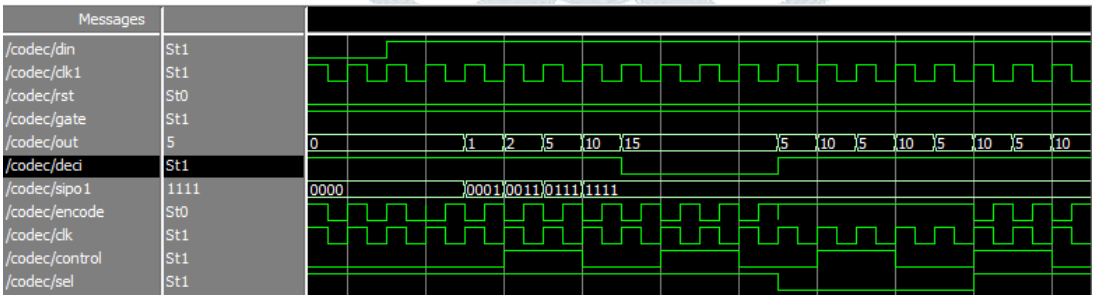

Fig 18: Simulation result of overall process

Above result is the combination of all the blocks. Codec output is the output obtained from decoder.

VIII. COMPARISON OF DIFFERENT ENCODING TECHNIQUES

Table 2: Comparison table

Techniques	Number of slices used	Total power (mW)	Delay (ns) from source to destination
BIC	16	55.57	16.095
BSC	33	116.67	5.400
Proposed technique (ABE) before clock gating	13	41.03	9.073
Proposed technique (ABE) after clock gating	19	27.61	6.698

From the table.2 it can be observed that there is drastic decrease in total power of proposed technique (ABE) after applying clock gating when compared to other techniques. If the ratio of off-chip data-bus capacitance to on-chip node capacitance is approximately 400, then we get good power savings from our proposed algorithm and architecture.

IX. CONCLUSION

Adaptive bus encoding technique does not require prior knowledge of input data. The existing techniques cannot tolerate when data changes suddenly, here ABE is advantageous. Clustering is adaptive since cluster of lines have more correlated switching patterns and are XORed with basis line which gives maximum switching saving. Switchings are saved by using gated clock instead of normal clock. From comparison table it is concluded that proposed technique (Adaptive Bus Encoding with clock gating) performs better than BIC, BSC and ABE before clock gating. Manual clock gating (person operating the clock) is replaced by automatic clock gating. In future the study is to be made on automatic clock gating, where it is designed such that the hardware turn ON the clock if needed and turn OFF if not required.

REFERENCES

[1] R. Siegmund, C. Kretzschmar, and D. Muller, "Adaptive bus encoding technique for switching activity reduced data transfer over wide system busses," in *Proc. Int. Workshop Power Timing Modeling Optim. Simulation*, 2000, pp. 66–75.

[2] M. Alamgir, I. I. Basith, T. Supon, and R. Rashidzadeh, "Improved bus-shift coding for low-power I/O," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Lisbon, Portugal, May 2015, pp. 2940–2943.

[3] Y. Shin, S.-I. Chae, and K. Choi, "Partial bus-invert coding for power optimization of system level bus," in *Proc. Int. Symp. Low Power Electron. Design*, Aug. 1998, pp. 127–129.

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

[5] Sumantra Sarkar, Ayan Biswas, Anindya Sundar Dhar, and Rahul M.Rao" Adaptive bus encoding for transition reduction on Off-chip buses with dynamically varying switching characteristics", in IEEE Trans , Very Large Scale Integration (VLSI) Syst, 2017, pp.1063-8210.

[6] Naveena Pai G1, M.B.Anandaraju2, Naveen.K.B "Design and Synthesis of Bus Invert Encoding and Decoding Technique Using Reversible Logic". Vol. 3, Issue 4, pp.143-146, , Jul-Aug 2013.

[7] Jayapreetha Natesan and Damu Radhakrishnan* Department of Electrical and Computer Engineering State University of New York 75 S. Manheim Blvd., New Paltz, New York 12561 "A Novel Bus Encoding Technique for Low Power VLSI" natesa76@newpaltz.edu.

[8] Nandita Srinivasana, Navamitha.S.Prakasha, Shalakha.Da, Sivaranjani.Da, Swetha Sri Lakshmi.Ga*, B.Bala Tripura Sundari "Power Reduction by Clock Gating Technique" Science Direct SMART GRID Technologies, August 6-8, 2015.

[9] Prabhat K. Saraswat, Ghazal Haghani and Appiah Kubi Bernard. Advanced Learning and Research Institute, ALaRI, University of Lugano, Switzerland "A Low Power Design of Gray and T0 Codecs for the Address Bus Encoding for System Level Power Optimization". prabhat.saraswat@alari.ch.

[10] Jagrit Kathuria, M. ayoubkhan, Arti Noor " A review of clock gating techniques" MIT international journal of electronics and communication engineering V0l, 1 no.2, pp 106-114, 2 Aug 2011.