International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.8, No.10 (2015), pp.375-388
http://dx.doi.org/10.14257/ijsip.2015.8.10.37

Data Driven Data Encoding for Low Power LDPC Applications

Suresh Dannana[1], Govinda Rao Tamminaina[2] and A. R. V. Satheesh Kumar[3]

[1,2,3]*Department of ECE, GMRIT College of Engineering*
Andhra Pradesh, India
[1]*Suresh.d@gmrit.org,* [2]*Govindarao.t@gmrit.org,*
[3]*Satish.adigarla401jo@gmail.com*

Abstract

The power consumption of network-on-chip (NoC) links start to compete with that of NoC routers. Reduce power consumption in NoCs, in both wires and logic, is to reduce the switching activity by means of coding schemes. The fundamental basic schemes of a NoC-based communicate are Network Interfaces (NIs), Links and routers. These links misuse power due to the switching activity (both self and coupling) induces by following data model traverse the links and routers. Three schemes join to reduce the dynamic power of the NoC data path through a decrease in the number of bit transitions. The transitions are of Type (I to IV) and correspondingly. In this paper, LDPC the encoder is replaced with data encoding schemes in control to reduce the dynamic power consumption in Low Density Parity Check Techniques (LDPC) and the power values are varied to the package of XC3S100E and TQ144 device is used to the comparing of NLDPC to LDPC, best results are evaluated in the proposed technique.

Keywords: *Inversion schemes, Types of transitions, BI, NoC, Low Power, NI, LDPC, NLDPC*

1. Introduction

"Network-on-a-Chip" (NoC) is a new representation for System-on-Chip (SoC) design. NoC based-systems contain multiple asynchronous clocking that many of today's complex SoC designs [4] use. The NoC solution brings a networking method to on-chip communications and claims around a threefold performance increase over usual bus systems. Network-on-Chip (NoC) is a communications essentially composed of routers interconnected by communication channels. It is proper to support the design paradigm. It provides asynchronous communication, scalability, reliability. The idea of network-on-chip (NoC) becomes more capable because of performance, power, and scalability requirements for a SoC device [4]. The power consumption in a NoC grows linearly with the amount of bit transitions in subsequent data packets sent through the interconnect design one way to reduce power consumption in NoCs, in both wires and logic, is to reduce the switching activity by means of coding schemes. The interconnect resistance (R) and interconnect capacitance (C) increases linearly with increase in communicate length [4].

2. Inversion Encoding Techniques

The data programming techniques are developed to reduce the power consumption caused by the transitions in the communicate on the chip [1]. These techniques are based on reducing the number of transitions by considering the types of transitions in the interconnects and by in view of them as discussed in the table-1, below and also consider the transitions as different types of inversions existing for us. The different types of

ISSN: 2005-4254 IJSIP

inversions available for us are odd Inversion, full inversion and even inversions. By reducing these inversions can manage the number of transitions in the interconnect reduces the power consumption caused by these transitions in the links.

Table 1. Table Shows Transitions Versus Types

TIME	NORMAL			ODD INVERTED			EVEN INVERTED		
t-1 t	TYPE-1			TYPE-2,3 and 4			TYPE-2,3 and 4		
	01,10 00,11	00,11,01,10 10,01,11,00	00,11 01,10	01,10 10,01	00,01,10,11 00,01,10,11	00,11 11,00	01,10 10,01	00,01,10,11 00,01,10,11	00,11 11,00
	T1*	T1**	T1***	TYPE-2	TYPE-4	TYPE-3	TYPE-2	TYPE-4	TYPE-2
t-1 t	TYPE-2 01,10 10,01			TYPE-1 01,10 11,00			TYPE-1 01,10 00,11		
t-1 t	TYPE-3 00,11 11,00			TYPE-1 00,11 10,01			TYPE-1 00,11 01,10		
t-1 t	TYPE-4 00,01,10,11 00,01,10,11			TYPE-1 00,11,01,10 10,01,11,00			TYPE-1 00,01,10,11 01,00,11,10		

By using these three schemes which are as follows

2.1. Encoding Scheme

The data encoding techniques are transparent with respect to the NoC fabrication. In encoding scheme there is no modification in links and routers architecture. Transition occurred in on chip interconnection is classified as four types. Type 1 transition occurred when one line switches and the other line unchanged, type 2 transition occurred when one line switches from high to low and other line switches from low to high, type 3 transition occurred when both lines are simultaneously switches and type 4 transition are no changed.

Encoding scheme based on inversion on odd bits condition, it consider the total link width of w bits. The header flit is not encoded, the w bits of the input is encoded and passed through link. The last bit indicates whether the odd bit inversion taken place or not. The generic block diagram is shown in Fig below 2.1. The encoder block E, it is inbuilt into the network interface, is responsible for inversion occurrence. To make the decision the first input flit is compared with previous encoded flit. The integration of w-1 bits and last one bit is w bit; represent the first input of the encoder. The previous encoded flit is given as a feedback that is the second input of the encoder. The encoder consists of three blocks (i) Transition block (ii) Majority voter (iii) odd bit inversion

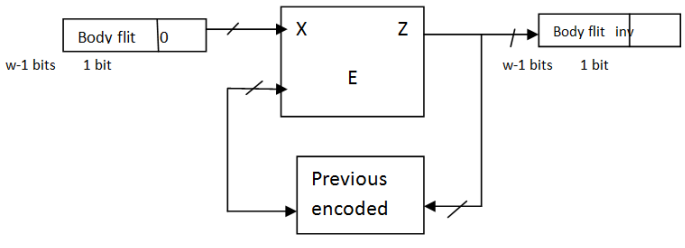


Figure 2.1. Encoder Architecture Scheme

2.2. Data Encoding Inversion Schemes

The Encoding scheme is to reduce power dissipation by minimizing the coupling transition activities and interconnection network. The dynamic power dissipated by the interconnects and drivers is

$$P = [T_{0\to1}(C_s + C_l) + T_c C_c]V^2 ddF_{clk} \quad 2.1$$

Where $T_{0\to1}$ is the number of 0 transitions in the bus in two consecutive transmissions, T_C is the number of correlated switching between physically adjacent lines, C_S is the line to substrate capacitance, C_l is the load capacitance, C is the coupling capacitance, V_{dd} is the supply voltage, and F_{clk} is the clock frequency. A Type I transition occurs when one of the lines switches when the other remains unchanged. In a Type II transition, one line switches from low to high while the other makes transition from high to low. A Type III transition corresponds to the case where both lines switch simultaneously. Finally, in a Type IV transition both lines does not change. The coupling transition activity T_c is a weighted sum of different types of coupling transition contributions.
Therefore

$$T_C = K_1T_1 + K_2T_2 + K_3T_3 + K_4T_4 \quad 2.2$$

Using (2.2), one may express (2.1) as

$$P = [T_{0\to1}(C_S + C_l) + (T_1 + 2T_2)C_C]V^2 ddF_{clk} \quad 2.3$$

$$P \propto T_{0\to1}C_S + (T_1 + 2T_2)C_C \quad 2.4$$

Calculate the occurrence probability for different types of transitions. Consider that flit (t − 1) and flit (t) refer to the previous flit which was transferred the link in the flit respectively. Consider only two adjacent bits of the physical channel. Sixteen different combinations of these four bits could occur (Table 1). The first bit is the value of the generic line of the link, whereas the second bit represents the value of its (i +1) t_h line. The number of transitions for Types I, II, III, and IV are 8, 2, 2, and 4, respectively. For a random set of data, each of these sixteen transitions has the same probability. Therefore, the occurrence probability in Types I, II, III, and IV are 1/2, 1/8, 1/8, and 1/4, respectively.

2.2.1. Odd Inversion of Scheme-1: In odd inversion of scheme 1, our main goal is to reducing the number of Type 1 transitions and Type 2 transitions. Type 1 transitions is converted into Type III and Type IV transitions and Type II transitions is converted into Type I transitions. This scheme compares the two data‘s based on to reducing the link power reduction by doing odd inversion or no inversion operation.

$$P' \propto T_{0\to1} + (K_1T_1' + K_2T_2' + K_3T_3' + K_4T_4')C_C \quad 2.5$$

i. Power Model

If the flit is odd inverted dynamic power on the link in the self-transition activity, and the coupling transition before being transmitted, the activity of Types I, II, III, and IV, respectively. The first bit is the value of the generic it_h line of the link, whereas the second bit represents the value of its (i + 1) t_h line. For each partition, the first line represents the values at time t − 1 (t).

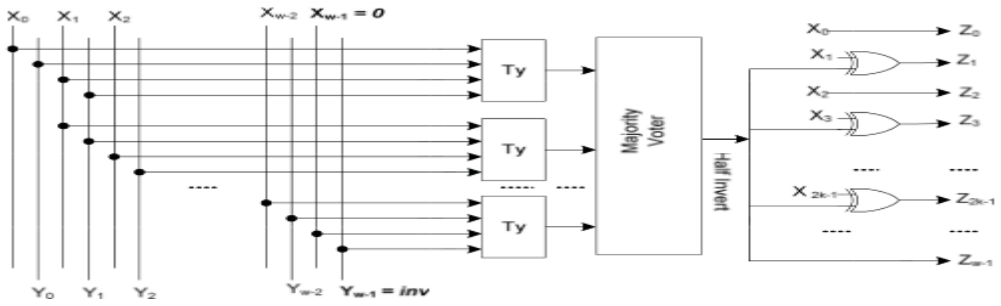


Figure 2.2. Encoder Architecture Scheme1 Internal View of the Encoder Block (E)

$$T_{0\to1} = T_{0\to1(odd)} + T_{0\to1(even)} \quad 2.6$$

$$\frac{1}{4}T_{0\to1}(odd)+T_1 + 2T_2 > \frac{1}{4}T_{0\to0}(odd) + T_2 + T_3 + T_4 + 2T_1 *** \quad 2.7$$

It can be approximate the exact condition as,

$$T_1 + 2T_2 > T_2 + T_3 + T_4 + 2T_1 *** \quad 2.8$$

The encoding scheme due to the error induced by the approximation but it simplifies the hardware implementation of encoder.

$$T_y = T_2 + T_1 - T_1 *** \quad 2.9$$

$$T_y > T_x \quad 2.10$$

$$T_y + T_x = w - 1 \quad 2.11$$

$$T_y > \frac{(w-1)}{2} \quad 2.12$$

2.2.2. Full inversion of Scheme-2: In full inversion of scheme II, our main goal is to reducing the number of Type II transitions. Type II transitions are converted into Type IV transitions. This scheme compares the two data‘s based on to reducing the link power reduction by doing full inversion or odd Full and odd inversion based this advanced encoding architecture consist of w-1 link width and one bit for inversion bit which indicate if the bit travel through the link is inverted or not. W bits link width is considered when there is no encoding is applied for the input bits.

i. Power Model

The odd inversion leads to power reduction when p’< p’’ and p’< p. the power p’’ is given by

$$P \propto \ddot{T_1} + 2T_4 \quad 2.13$$

$$T_2 + T_3 + T_4 + 2T_1 ***< T_1 + 2T_4 ** \quad 2.14$$

$$2(T_2 - T_4 **) < T_1 + 2T_4 ** \quad 2.15$$

Based on (2.12) and (2.15), the odd inversion condition is obtained as

$$2(T_2 - T_4 **)<2T_Y - w + 1\ T_y > \frac{w-1}{2} \quad 2.16$$

$$T_2 > T_4 \quad 2.17$$

$$2(T_2 - T_4 **)> 2T_y - w + 1T_2 > T_4 ** \quad 2.18$$

ii. Encoding Architecture of Full Inversion Scheme-2

The operating an encoder implementing the Scheme1in encoding architecture, which is based on the odd invert condition of and the full invert condition of is shown in (Fig. 2.3). Here again, the bit of the previously and the full invert condition of is shown in (Fig.2. 3). In this encoder, in addition to the block in the Scheme I encoder, we have the $T_2$and blocks which determine if the inversion based on the transition types and should be taken

place for the link power reduction. The second stage is formed by a set of 1s blocks which count the number of 1s in their inputs.

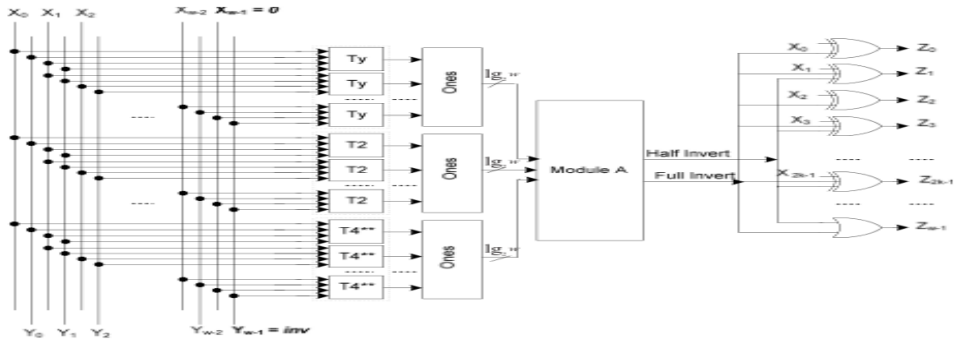


Figure 2.3. Encoder Architecture Scheme-2

2.2.3. Even Inversion of Scheme-3: In the encoding scheme-3, include even inversion to scheme-2. The reason is that odd inversion converts some of Type I (*T*1***) transitions to Type II transitions. As can be experiential from Table-1, achieve of even inversion on modify of transition types [1], if the flit is even inverted, the transitions indicated as *T*** 1 / *T*1*** in the table are converted to Type IV/Type III transitions. Therefore, the even inversion may reduce the link power dissipation as well [1].

i. Power Model

Similar to the analysis given for scheme1, approximate the condition p'''< p as

$$T_1 + 2T_2 > T_2 + T_3 + T_4 + 2T_1 \quad 2.19$$

During $T_e = T_2 + T_1 - T_1 *$ 2.20

ii. Encoding Architecture of Even Inversion Scheme-3

The encoding architecture (Fig.2.4) in even inversion schemeIII is same of encoder architecture in scheme I and II. Here adding the Te block to the scheme II. This is based on even invert condition, Full invert condition and Odd invert condition. It consists of w-1link width input and the w bit is used for the inversion bit. The full, half and even Inversion is performed means the inversion bit is set 1', otherwise it set as 0 .TheT_y, T_e and T_4 ∗∗blockdetermines the transition types T2, Te and T4**. The transition types are sending to the number of one's block. The Te block is determined if any of the detected transition of types T2, T1** and T1**. The ones block determines the number of ones in the corresponding transmissions of TY, T2, Te and T4**. This number of one's is given to the Module C block. This block check if odd, even, full or no invert action corresponding to the outputs '10', '01', '11' or '00'respectively, should be performed. The decoder architecture of scheme II and scheme III are same.

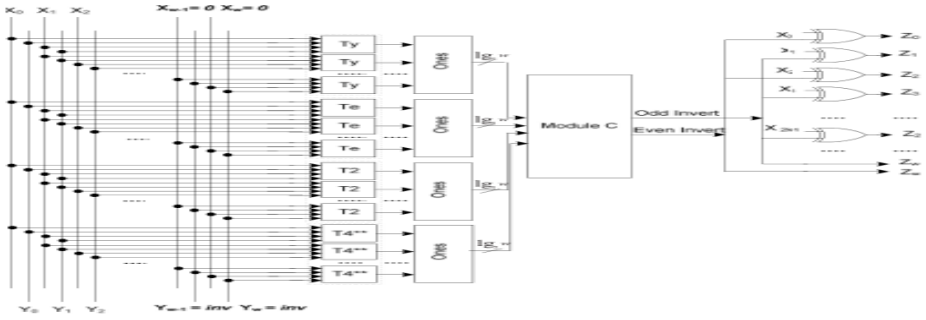


Figure 2.4. Encoder Architecture Scheme III

3. Proposed System

3.1. Low Density Parity Check Code (LDPC)

Low-Density Parity Check (LDPC) coding is a form of error coding introduced by Gallagher [5] that can complete performance lock to the Shannon limit, The coding system was introduced in the early 1960's[5]. The LDPC code is based on a set of one or more original LDPC codes. Each of the major codes is a logical linear block code. The important codes can contain various data code rates and packet sizes. Each LDPC code in the set of LDPC codes is defined by a matrix H of size *m*-by-*n*, where *n* is the duration of the code and *m* is the number of parity check bits in the code.[2] The number of logical bits is *k*=*n*-*m*.
The matrix H is defined as [5]

3.2. LDPC through Encoding

The encoding of a container at the transmitter generates parity-check bits k= (k_0, k_m-1) based on an information block s= (s_0, s_k-1) [5], and transmits the parity-check bits down with the in order block. Because the present symbol set to be encoded and transmitted is enclosed in the transmitted codeword, the information block is also known as consistent bits. The encoder receives the in order blocks = (s_0,.., s_k-1) and uses the matrix $\mathbf{H}_{bm}$ to establish the parity-check bits [5]. The extended matrix **H** is determined from the representation matrix $\mathbf{H}_{bm}$. Since the extended matrix **H** is a twofold matrix, data encoding of a packet can be performed with vector or matrix operations [5].

3.3. Modules and its Description

3.3.1. Organize Bits: In this module, arranged the words which are want to be error checking. These bits are arranged in bit by bit to shift register.

3.3.2. Check Parity: In this module, find the parity bit as a word to find error bits. There are N/4 numbers of Xor gates to get parity bit from shift register.

3.3.3. Detect Error: In this module, check the error bit from the parity word. For detecting and decoding more number of error bits maximum likelihood algorithm is working. Then, after this decoding use MLD algorithm.

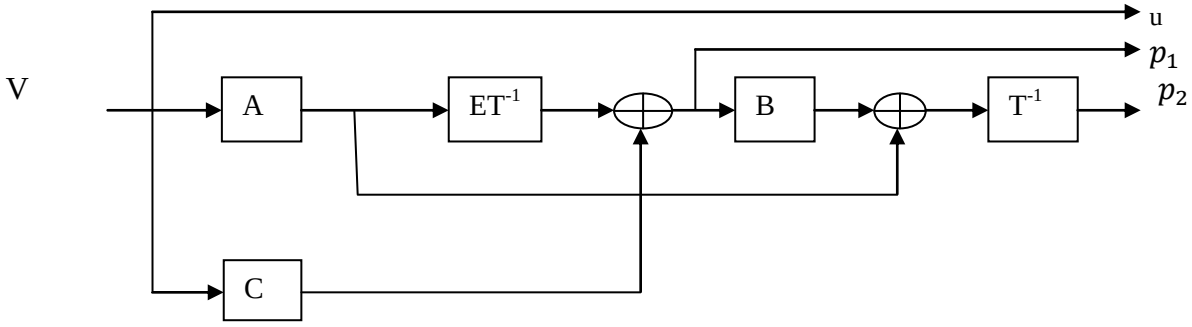


Figure 3. Block Diagram of the Encoder Architecture for the Block LDPC Code

Step 1) Compute $Au^T and Cu^T$
Step 2) Compute $ET^1(AU^T)$
Step 3) Compute by $P_1^T by P_1^T = ET^{-1}(Au^T) + Cu^T$

Step 4) Compute by $P_2^T byTP_2^T = Au^T + Bp_1^T$

As a result, the encoding procedures and the corresponding operations can be summarized below and illustrated in Figure 3.

3.3.4. Correct Error: Here module, correct the error bit by checking last bit from shift register and bit from majority logic circuit. Here, there was an error correction stage. In this error correction stage, XOR gate that changes error bit if there was change in bit.

3.3.5. Recover Output: Here module, if get error free output from shift register. In this, if there was no any error in the word, then it stop cycle checking and goes to next word.

Using two decoding techniques majority logic decoding and majority logic decoder/detector in LDPC to reduce the delay occurs in these techniques.

3.4. Majority Logic Decode

Majority-logic decode is an easy and helpful scheme for decode positive classes of block codes [6]. In exacting for decode positive classes of returning codes. Majority logic decode is a system to decode repetition codes, base on the theory that the largest number of occurrence
of a sign was the transmitted figure-4. It will increase the power use. Syndrome vector is oldest machinery, which is used to identify the error in the code word. Hamming code is one of the examples of syndrome decoder [6].

3.5. Majority Logic Detector/Decoder

The ML detector/decoder (MLDD) has been implementing using the Euclidean Geometry LDPC. EG-LDPC codes there are a subclass of codes that is one stage majority logic decoder (MLD). This method is very practical to generate and check all possible error combinations for data codes with small words and affected by a small number of bit flits [6]. When the size of code and the number of bit flit increases, it is difficult to systematically test all achievable combination [6].

4. Data Programming Techniques in LDPC Encoder

These data programming schemes are discussed in the above sections are replaced by these encoding systems in its place of the encoder in the LDPC block:
4.1. Performance of LDPC Scheme-1
4.2. Performance of LDPC Scheme-2
4.3. Performance of LDPC Scheme-3

4.1. Performance of LDPC Scheme-1

The programming structural design, which is based on the odd invert state defined, is shown in Figure 1. Consider a link width of *w* bits [1]. If no programming is used, the body flits are group in *w* bits by the NI and are transmitted via the link. One bit of the connection is used for the inversion bit, which indicate if the flit traverse the link has been inverted or not. This programming method is scheme-1 of data encoding techniques are used in low density parity check of majority logic decoding is replaced with our encoding scheme-1 technique.

4.2. Performance of LDPC Scheme-2

The idea of this encoder is parallel to those of the encoder implement scheme-1. The encoding structural design, which is base on the odd invert state and the full invert state, is shown in Fig. 2. In this encoder, in count to the *Ty* block in the system-1 encoder, the *T*2 and *T*4** blocks which conclude if the full inversion based on the transition types *T*2 and *T*4** have to be taken place for the link power reduction [1]. The top 1s block determines the number of transitions that odd inverting of pair bits leads to the link power reduction. The output of these blocks has the width of log2 *w*. The output of The middle 1s block and bottom 1s block identifies the number of transitions whose full inverting of pair bits leads to the link power reduction. The method which is mention above is LDPC using scheme-1 now is replaced with scheme-2 in the LDPC and thereby reducing some quantity of power compared from scheme-1.

4.3. Performance of LDPC Scheme-3

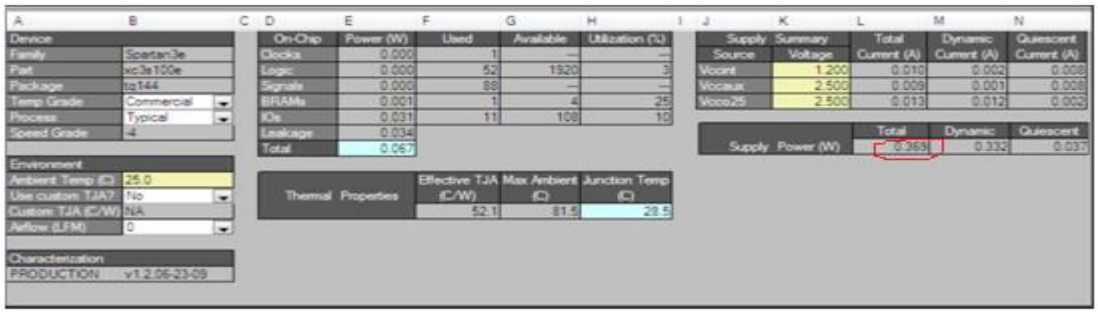
The functional ideologies of this encoder are parallel to those of the encoders implement scheme-1 and 2. The (inv = 0), the initial stage of the encoder determines the transition types while the second stage is formed by a set of 1s blocks which count the number of ones in their inputs. In the primary stage, added the *Te* blocks include to the which establish if any of the transition types of *T*2, *T*1**, and *T*1*** is detected for each pair bits of their inputs [1],[7].

The four Ones blocks to establish the number of detect transitions for Ty, *Te*, *T*2, *T*4** blocks [1]. The method is mention above is LDPC using scheme-1 now is replaced with scheme-2 in the LDPC and thereby reducing some amount of power compared from scheme-2. Therefore, analyzed that scheme-3 is the one which is used in LDPC technique.

5. Results

The LDPC Coding for proposed methodology is written using Verilog HDL coding and simulated using Xilinx 14.2 software. Software power estimator device is XC3S100E and Spartan 3E family, package is TQ144. The power is reduced by having less number of flip- flops and slice LUT registers. The comparison of power results for all the schemes in existing and LDPC proposed is shown in graph-1. The LDPC encoder is implemented for matrix size 32 X 64. Results are compared in terms of total power and area figures as shown below. The errors present in the existing system are rectified by using LDPC technique. The quiescent power is not changed before encoding as the LDPC encoding techniques are not applied. The quiescent power values are reduced after the LDPC encoding techniques are applied**.**

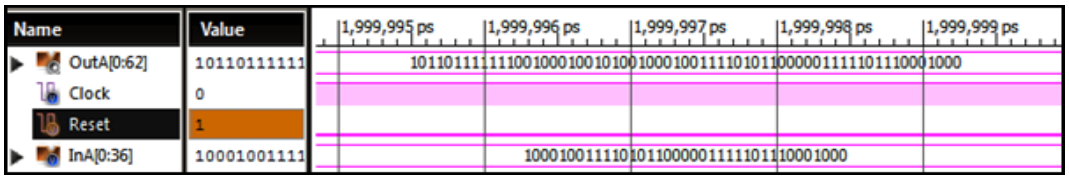
5.1. Power Report for NLDPC First Data



The power calculations for the normal LDPC are shown in the figure 5.1.It is observed that the power calculated is 0.369mW without using any inversion techniques. Comparing

the powers of NLDPC with the inversion techniques (ODD, FULL, EVEN) depending upon the transitions.

5.2. Simulation Result for Normal LDPC



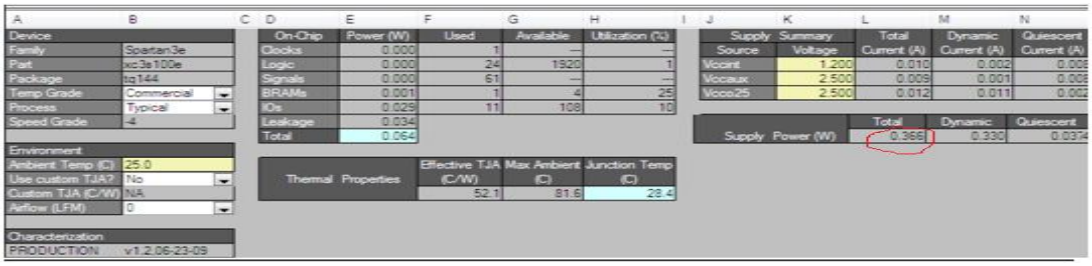
From the Figure 5.2, for the first input data 1000100111101011000001111101110001000the corresponding output is above fig 5.2, in case of NLDPC when reset is 1.So, it is observed that the received data is efficient due to correction of error using parity checking and power is also reduced

5.3 Summary of NLDPC First Data

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of 4 input LUTs	131	1,920	6%
Number of occupied Slices	66	960	6%
Number of Slices containing only related logic	66	66	100%
Number of Slices containing unrelated logic	0	66	0%
Total Number of 4 input LUTs	131	1,920	6%
Number of bonded IOBs	91	108	84%

According to the table Figure 5.3 shows the number of LUTS utilized in the NLDPC

5.4 Power Report For ODD LDPC First Data



The power calculations for the ODD LDPC are shown in the figure 6.3. In case of odd LDPC the output data shown in fig 6.4, received is for the given input data 1000100111101011000001111101110001000respectively. Comparing the powers of NLDPC and ODD LDPC, the ODD LDPC (0.366mW) gives efficient result then the NLDPC (0.369mW).

5.5. Simulation Result for ODD LDPC

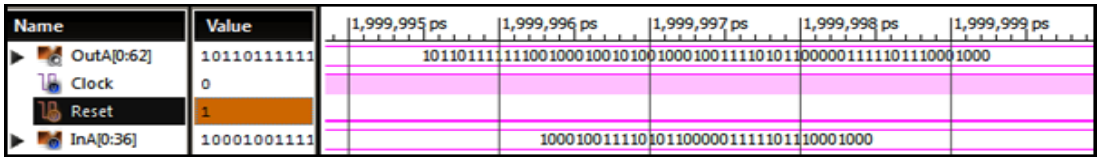


Figure5.5, for the first input data 1000100111101011000001111101110001000 the corresponding output is shown in above fig 5.5, in case of ODD LDPC when reset is 1.

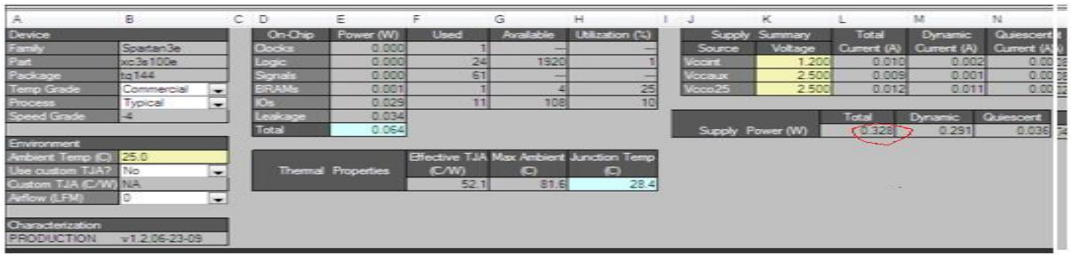
So, it is observed that the received data is efficient due to correction of error using parity checking and power is also reduced.

5.6. Summary of ODD LDPC

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**N**
Number of 4 input LUTs	97	1,920	5%	
Number of occupied Slices	49	960	5%	
Number of Slices containing only related logic	49	49	100%	
Number of Slices containing unrelated logic	0	49	0%	
Total Number of 4 input LUTs	97	1,920	5%	
Number of bonded IOBs	102	108	94%	

According to the Table 5.6 shows the number of LUTS utilized in the NLDPC and the number of IOB's is constant.

5.7. Power Report for NLDPC Second Data



The power calculations for the normal LDPC are shown in the figure 5.7. It is observed that the power calculated is 0.328mW without using any inversion techniques. Comparing the powers of NLDPC with the inversion techniques (ODD, FULL, EVEN) depending upon the transitions.

5.8. Simulation Result for Normal LDPC

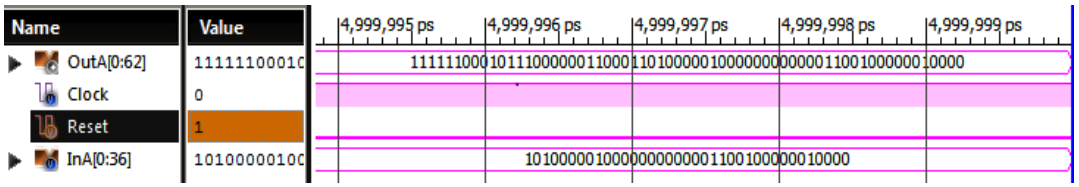


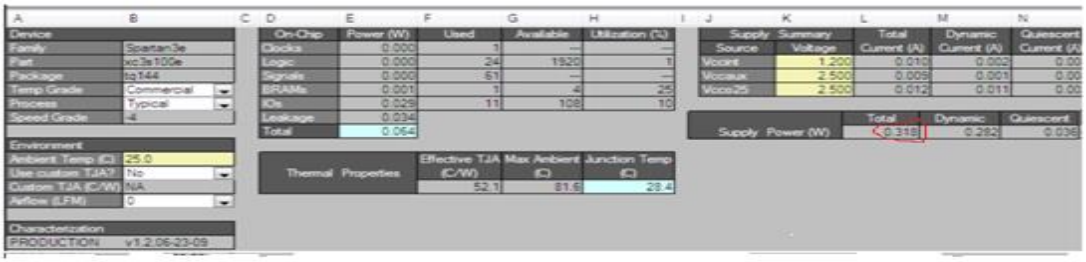
Figure 5.8, for the second input data 10100000100000000000011001000000010000 the corresponding output is shown in above fig 5.8, in case of NLDPC when reset is 1. So, it is observed that the received data is efficient due to correction of error using parity checking and power is also reduced

5.9. Summary of NLDPC Second Data

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	
Number of 4 input LUTs	131	1,920	6%	
Number of occupied Slices	66	960	6%	
Number of Slices containing only related logic	66	66	100%	
Number of Slices containing unrelated logic	0	66	0%	
Total Number of 4 input LUTs	131	1,920	6%	
Number of bonded IOBs	91	108	84%	

According to the table 5.9 shows the number of LUTS utilized in the NLDPC and the number of IOB's is constant.

5.10. Power Report for FULL LDPC Second Data



Device		On-Chip	Power (W)	Used	Available	Utilization (%)
Family	Spartan3e	Clocks	0.000	1	---	---
Part	xc3s100e	Logic	0.000	24	1920	1
Package	tq144	Signals	0.000	61	---	---
Temp Grade	Commercial	BRAMs	0.001	1	4	25
Process	Typical	IOs	0.009	11	108	10
Speed Grade	-4	Leakage	0.034			
		Total	0.064			

Environment		Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
Ambient Temp (C)	25.0		52.1	81.6	28.4
Use custom TJA?	No				
Custom TJA (C/W)	NA				
Airflow (LFM)	0				

Characterization	
PRODUCTION	v1.2,06-23-09

Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.200	0.010	0.002	0.00
Vccaux	2.500	0.009	0.001	0.00
Vcco25	2.500	0.012	0.011	0.00

	Total	Dynamic	Quiescent
Supply Power (W)	0.318	0.282	0.036

The power calculations for the FULL LDPC are shown in the figure 5.10. In case of full LDPC the output data shown in below fig 5.12, received is for the given input data 10100000100000000000011001000000010000 respectively. Comparing the powers of NLDPC and FULL LDPC, the FULL LDPC (0.318mW) gives efficient result then the NLDPC (0.328mW).

5.11. Summary of FULL LDPC

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**N**
Number of 4 input LUTs	97	1,920	5%	
Number of occupied Slices	49	960	5%	
Number of Slices containing only related logic	49	49	100%	
Number of Slices containing unrelated logic	0	49	0%	
Total Number of 4 input LUTs	97	1,920	5%	
Number of bonded IOBs	102	108	94%	

According to the figure 5.11 shows the number of LUTS utilized in the NLDPC and the number of IOB's is constant

5.12. Simulation Result for FULL LDPC

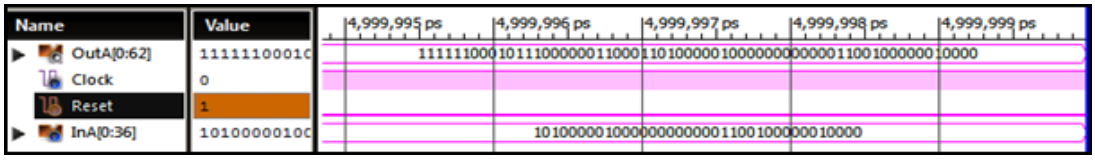
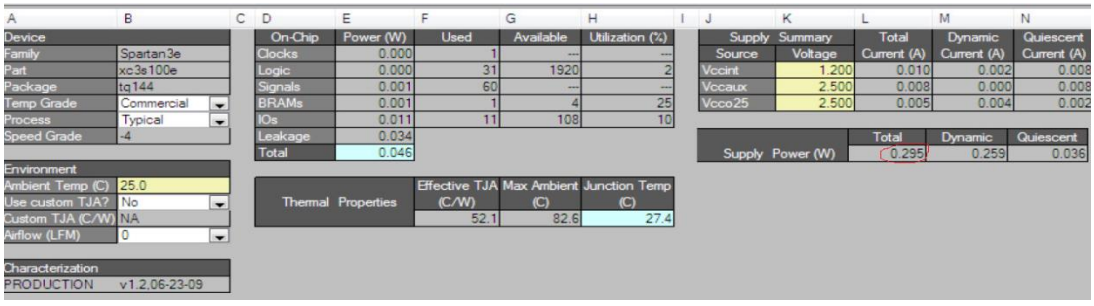


Figure 6.8, for the second input data 10100000100000000000011001000000010000the corresponding output is shown in above fig 6.8. In case of FULL LDPC when reset is 1. So, it is observed that the received data is efficient due to correction of error using parity checking and power is also reduced

5.13. Power Report for NLDPC Third Data



Device		On-Chip	Power (W)	Used	Available	Utilization (%)
Family	Spartan3e	Clocks	0.000	1	---	---
Part	xc3s100e	Logic	0.000	31	1920	2
Package	tq144	Signals	0.001	60	---	---
Temp Grade	Commercial	BRAMs	0.001	1	4	25
Process	Typical	IOs	0.011	11	108	10
Speed Grade	-4	Leakage	0.034			
		Total	0.046			

Environment		Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
Ambient Temp (C)	25.0		52.1	82.6	27.4
Use custom TJA?	No				
Custom TJA (C/W)	NA				
Airflow (LFM)	0				

Characterization	
PRODUCTION	v1.2,06-23-09

Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.200	0.010	0.002	0.008
Vccaux	2.500	0.008	0.000	0.008
Vcco25	2.500	0.005	0.004	0.002

	Total	Dynamic	Quiescent
Supply Power (W)	0.295	0.259	0.036

The power calculations for the normal LDPC are shown in the figure 5.13. It is observed that the power calculated is 0.295mW without using any inversion techniques. Comparing the powers of NLDPC with the inversion techniques (ODD, FULL, EVEN) depending upon the transitions.

5.14. Simulation Result for Normal LDPC

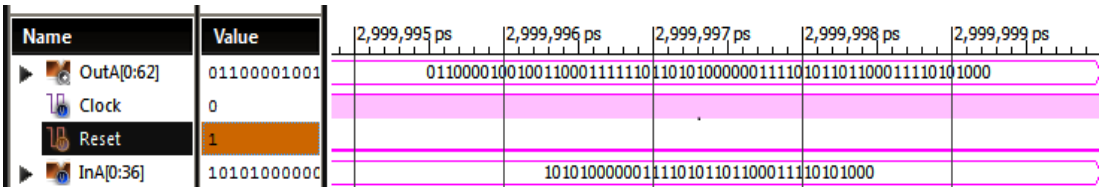


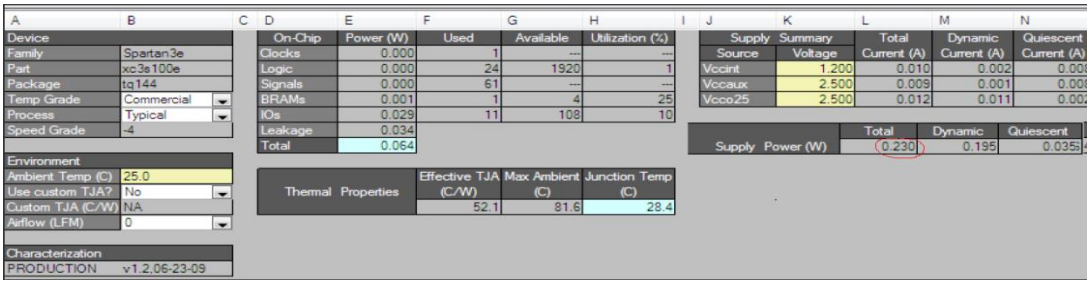
Figure 5.14, for the third input data 10101000000111101011011000111101010 00 the corresponding output is shown in above fig 5.14. In case of NLDPC when reset is 1. So, it is observed that the received data is efficient due to correction of error using parity checking and power is also reduced.

5.15. Summary of NLDPC THIRD DATA

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**N**
Number of 4 input LUTs	97	1,920	5%	
Number of occupied Slices	49	960	5%	
Number of Slices containing only related logic	49	49	100%	
Number of Slices containing unrelated logic	0	49	0%	
Total Number of 4 input LUTs	97	1,920	5%	
Number of bonded IOBs	102	108	94%	

According to the figure 5.15 shows the number of LUTS utilized in the NLDPC and the number of IOB's is constant.

5.16. Power Report for EVEN LDPC Third Data



]The power calculations for the EVEN LDPC are shown in the figure 5.16. In case of even LDPC the output data shown in fig 5.16, received is for the given input data 10101000000111101011011000111101010 00 respectively. Comparing the powers of NLDPC and EVEN LDPC, the EVEN LDPC (0.230mW) gives efficient result then the NLDPC (0.295mW).

5.17. Summary of EVEN LDPC THIRD DATA

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**N**
Number of 4 input LUTs	97	1,920	5%	
Number of occupied Slices	49	960	5%	
Number of Slices containing only related logic	49	49	100%	
Number of Slices containing unrelated logic	0	49	0%	
Total Number of 4 input LUTs	97	1,920	5%	
Number of bonded IOBs	102	108	94%	

According to the above figure 5.17 shows the number of LUTS utilized in the NLDPC and the number of IOB's is constant.

5.18. Simulation Result for EVEN LDPC

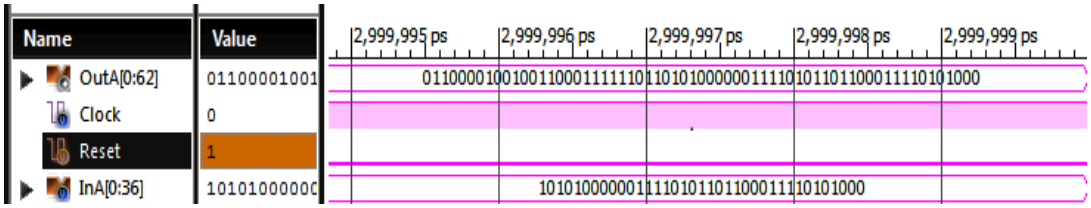


Figure 5.18, for the third input data 1010100000011110101101100011110101000 the corresponding output is shown in fig 5.18. In case of FULL LDPC when reset is 1. So, it is observed that the received data is efficient due to correction of error using parity checking and power is also reduced.

5.19. Comparison of Normal LDPC and ODD LDPC

Table 2. Comparison of Existing and Proposed Systems

PARAMETERS	Normal LDPC	ODD LDPC
POWER	0.369mW	0.366mW
DELAY	4.283ns	4.281ns
NUMBER OF LUT'S	131	97
MEMORY	2010708Kbytes	199276Kbytes

5.20. Comparison of Normal LDPC and FULL LDPC

Table 3. Comparison of Existing and Proposed Systems

PARAMETERS	Normal LDPC	ODD LDPC
POWER	0.328mW	0.318mW
DELAY	4.276ns	4.274ns
NUMBER OF LUT'S	131	97
MEMORY	201235Kbytes	206258Kbytes

5.21. Comparison of Normal LDPC and EVEN LDPC

Table 4. Comparison of Existing and Proposed Systems

PARAMETERS	Normal LDPC	ODD LDPC
POWER	0.295mW	0.230mW
DELAY	4.283ns	4.270ns
NUMBER OF LUT'S	97	97
MEMORY	212564Kbytes	212246Kbytes

When the comparison of both systems in this paper, so observed the above tables (5.19, 20 and 21). to reduce the power dissipation due to the sc-1 to sc-3 (sc-scheme) and reaming parameters (LUTs, Slices) is also reduced to the data encoding techniques for low power VLSI. The better results for Low Density Parity Checker Encoding Techniques comparison of Data Encoding Techniques.

6. Conclusion

In this paper, a set of new data encoding schemes aimed at reducing the power dissipated by the links of a NoC. In Proposed system, Data programming techniques which are used in the place of encoders in LDPC which reduces the power utilization by eliminating the transitions as discussed before. Hence analyzed the power consumption for these three schemes and compared their power and area performances. The encoders implementing the proposed schemes have been assessed in terms of power dissipation and silicon area. The impacts on the performance, power, and area metrics have been studied using a cycle- and bit correct NoC simulator under both synthetic and real traffic scenarios. As compared to the previous encoding inversion schemes proposed in the literature, the justification behind the proposed schemes is to minimize not only the switching activity, but also the coupling switching activity which is mainly responsible for link power dissipation in the deep submicron meter technology.

References

[1] N. Jafarzadeh, M. Palesi, A. Khademzadeh and A. A. Kusha, "Data Encoding Techniques for Reducing Energy Consumption in Network- on- Chip" IEEE transactions on very large scale integration (VLSI) systems, vol. 22, no. 3, (**2014**), pp. 675-685.
[2] S. Anusuyahdevi and S. Jayashri, "Performance Analysis of an Efficient Low Power NOC Router System Using Gray Encoding Techniques", (ijircce), vol. 2, no. 12, (**2014**), pp. 7463-7470.
[3] W. Wolf, Fellow, A. A. Jerraya and G. Martin, "Multiprocessor System on- Chip (MPSoC) Technology", IEEE, vol. 27, no. 10, (**2008**), pp. 1701-1713.
[4] S. R. Mullainathan and S. Ramkumar, "Switching Reduction Through Data Encoding Techniques in NoC", IEEE, vol. 23, no.76, (**2014**), pp. 38-41.
[5] L. Benini and G. De MSicheli, "Networks on chips: A new SoC paradigm", Computer, vol. 35, no. 1, (**2002**), pp. 70–78.
[6] G. Vlantis, "Low Density Parity Check Code", doc: IEEE, (**2006**), pp. 1-13.
[7] P. N. Manjunatha, T. S. Bharath Kumar and M. Z. Kurian, "Architecture for Low Density Parity Check Encoder", (IJAIEM), vol. 3, no. 3, (**2014**), pp. 414-417.
[8] M. Sakthivel, M. Karthick Raja, K. R. Ragupathy and K. Sathish Kumar, "Performance Comparison of EG - LDPC codes with maximum likelihood Algorithm over non-binary LDPC codes", (IJCSITY) vol. 2, no. 2, (**2014**), pp. 43-53.
[9] T. Tuan, "A Tutorial on Low Density Parity-Check Codes", The University of Texas at Austin, (**2003**), pp. 1-15.
[10] C. N. Kharkar, M. M. Jadhav and A. M. Sapkal, "FPGA Implementation of linear LDPC encoder", vol. 2, no. 11, (**2013**).
[11] T. J. Richardson and R. L. Urbanke, "Efficient Encoding of Low Density Parity-Check Codes", IEEE, vol. 47, no. 2, (**2001**), pp. 638-656.

