International Journal of Engineering and Manufacturing Science.
ISSN 2249-3115 Volume 8, Number 1 (2018) pp. 65-76

http://www.ripublication.com

DESIGN AND IMPLEMENTATION ARCHITECTURE FOR RELIABLE ROUTER RKT SWITCH IN NOC

Nivedita S Naragundakar, Prof Arun Raj S R

Department of Electronics and Communication Engineering
U.B.D.T.C.E Davanagere, Karnataka, India
Email:niveditha9193@gmail.com

Abstract

A new reliable dynamic NoC we are proposing. The proposed NoC is a mesh structure of routers able to detect routing errors for adaptive routing based on the *XY* algorithm. Finally our aim is to Design and develop architecture for intelligent independent reliable routers like reliable router RKT-switch for implementation on FPGA.

1. Introduction

Network on Chip (NoC) is slowly being accepted as an important paradigm for implementing communication among various cores in a SoC. With the advances in integrated circuits (IC) manufacturing a constant attempt has been to design enormous amounts of networks on the same chip so as to accomplish more resourceful and optimized chips. Further an efficient routing algorithm can be useful to increase the efficiency of the networks embedded on the chips [1]. Fig 1 shows the conceptual view of a NOC where, each tile is composed of a resource(R) and a switch or router(S). The router is connected to the four neighboring tiles and its local resource via channels. Each channel consists of two directional point-to-point links between two routers or a router and a local resource.

Increasing complexity and the reliability evolution of SoCs, MPSoCs are becoming more sensitive to phenomena that generate permanent, transient, or intermittent faults. These faults may generate data packet errors, or may affect router behavior leading to data packet losses or permanent routing errors. A fault in a routing logic will often lead to packet routing errors and might even crash the router. The precise location of permanent faulty parts of the NoC must be determined, in order for them to be bypassed effectively by the adaptive routing algorithm.

By using adaptive routing algorithm, When a router receives a data packet, it compares its own address to the destination and source addresses. Then, the router checks its own position in the deterministic *XY* path of the NoC for the considered data packet. The router performing this checking is able to decide whether the switch

from which the packet was received made a routing error or not according to the correct *XY* path. However, this technique has a major drawback; it is unable to handle the bypass of faulty nodes and unavailable regions. For handling message routing errors in dynamic networks, a new faulty switch detection mechanism is required for adaptive or fault-tolerant routing algorithm

2. Design methodology:

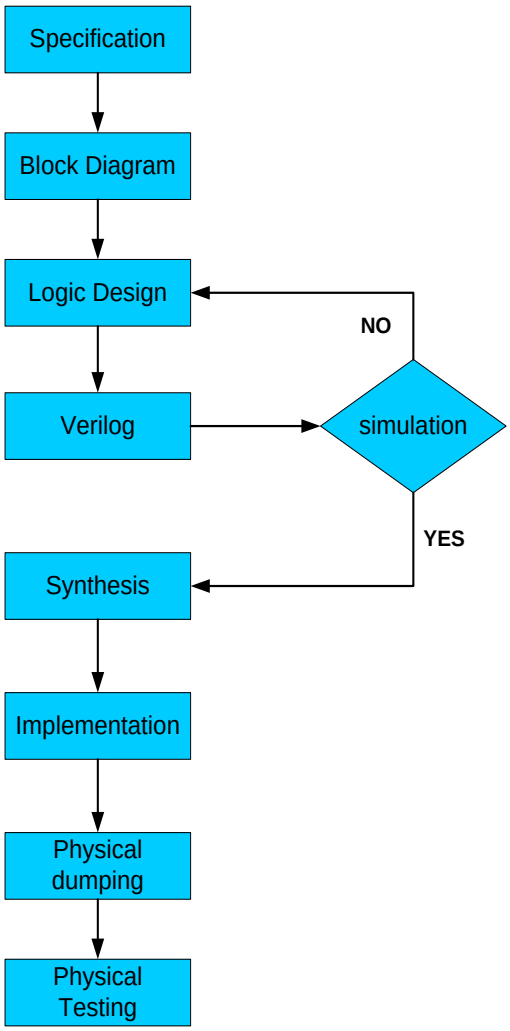


Fig: FPGA Data Flow

Above flow graph shows the method or procedure of the project, and each block present in the flow graph is briefly explained below:

Specification: Here **Spartan 3** tool will use to work on this project, it has the operating speed of 100MHZ on board, and it also has soft processor **Micro Blaze.**

And I work on tools like **Xilinx ISE 13.4** and also work on simulation software **ModelSim6.3c.**

Block Diagram: Design of intelligent independent reliable routers like reliable router RKT-switch. The proposed NoC is a mesh structure of routers able to detect routing errors for adaptive routing based on the *XY* algorithm

Logic Design: In this project logic design contains 4 router architecture includes loopback mechanism, I/O Buffers, FSM, Routing error detection and logic and hamming Logic(ECC).

Verilog: To work on this project Verilog language will use for the implementation of Design of like reliable router RKT-switch architecture in Xilinx ISE 13.4.

Synthesis: After combining, testing has to do, i.e whether the program is working properly or not, if yes it will continue with next process else it has to rewrite or correct. After correction again test the program, if its successfully working then it will be implement.

Implementation: All the process till testing will implement in this step.

Physical Dumping: In this step implementation of the project will dump into the FPGA.

Physical Testing: Finally the project will be test in the FPGA kit.

3. ERROR CORRECTION AND DETECTION

The error detection and correction which consists of Hamming Priority Encoder, XY Routing algorithm, Decoder and Random arbiter .

3.1 Hamming code

Hamming's development [Ham] is a very direct construction of a code that permits correcting single-bit errors. He assumes that the data to be transmitted consists of a certain number of *information bits* ***u***, and he adds to these a number of *check bits* ***p*** such that if a block is received that has at most one bit in error, then ***p*** identifies the bit that is in error (which may be one of the check bits). Specifically, in Hamming's code ***p*** is interpreted as an integer which is 0 if no error occurred, and otherwise is the 1-origined index of the bit that is in error. Let *k* be the number of information bits, and *m* the number of check bits used. Because the *m* check bits must check themselves as well as the information bits, the value of ***p***, interpreted as an integer, must range from 0 to m+k which is m+k+1 distinct values.

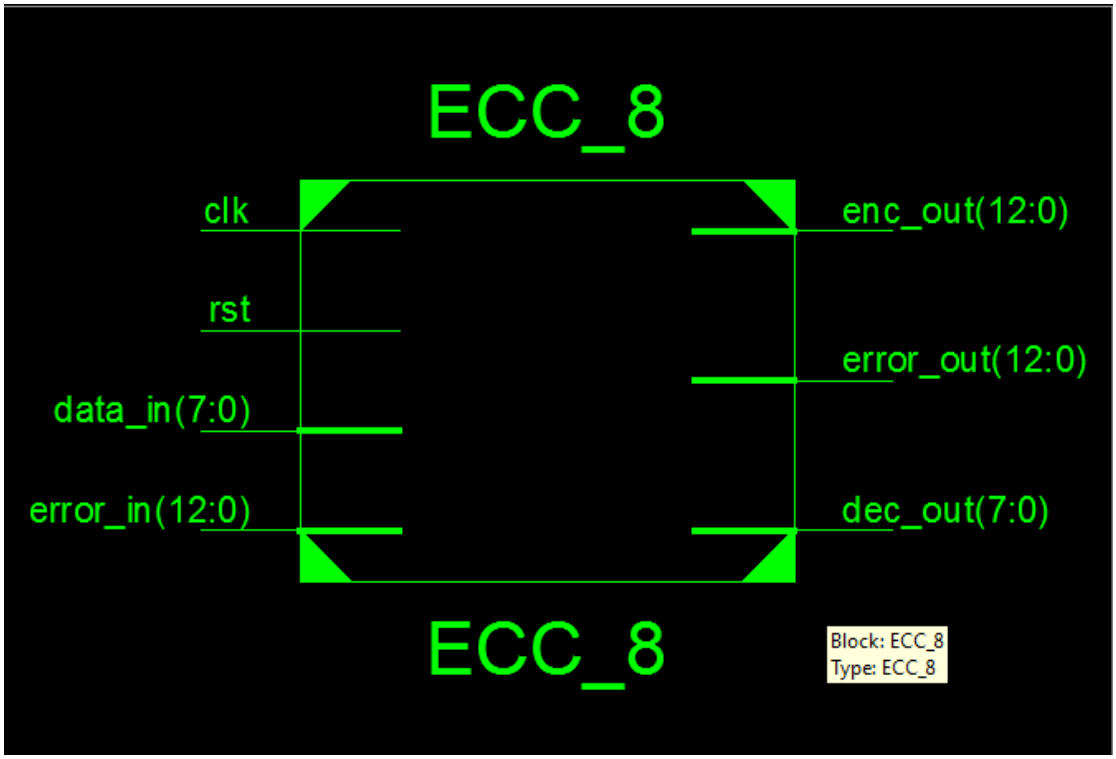


Fig3.1 block diagram of ECC

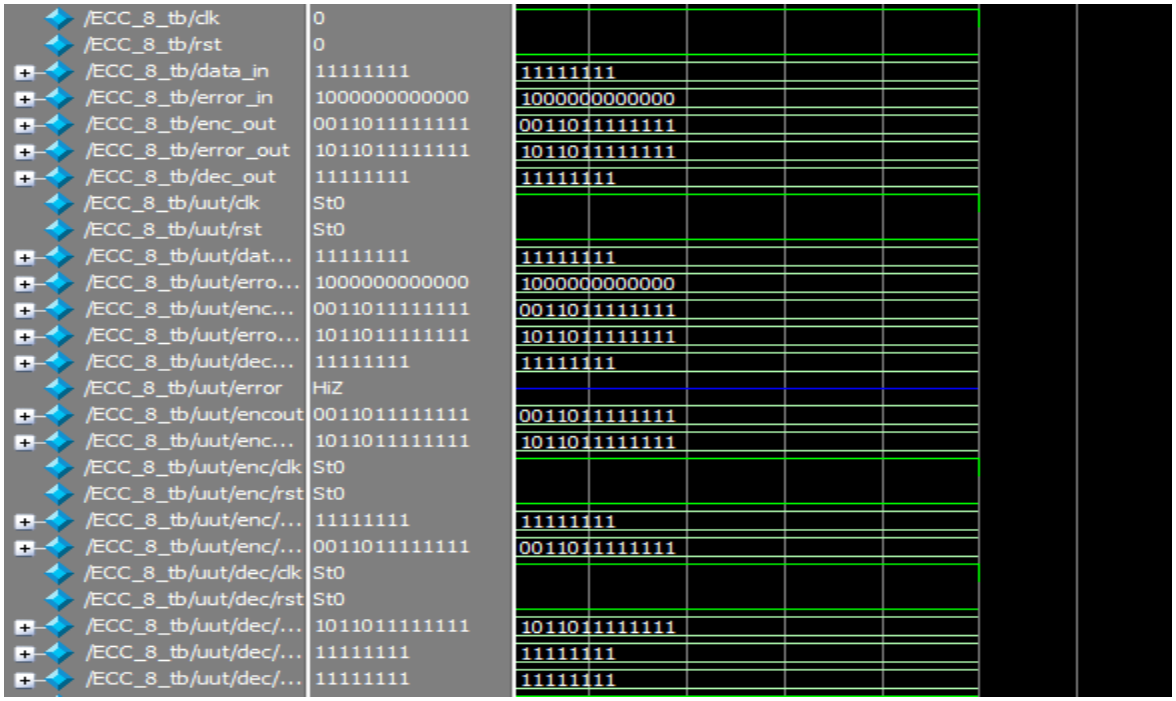
3.2 XY Routing

XY routing is one of the type of Dimension order routing (DOR) which is a typically a minimal turn algorithm and is more suitable for networks using mesh or torus topology. XY routing algorithm routes packets first in x-direction (or horizontal direction) to the correct column and then in y- direction (or vertical direction) to the receiver. In XY routing the addresses of the routers are their xy-coordinates. One of the advantages of XY routing is that it never runs into deadlock or livelock .

3.3 Random Arbiter:

The Random Arbiter plays a vital role on the router in order to take decision for servicing a packet. The packets are serviced randomly from different directions of the network without any packet stacking. In the proposed arbiter there are five independent requests req_0 to req_4 and these requests are input to the arbiter. The arbiter will process and generate the grants GNT_0 to GNT_4 (service) for incoming packet's requests in a random order on the router.

3.4 Simulation results of ECC



4. SYNTHESIS OF ONE NODE

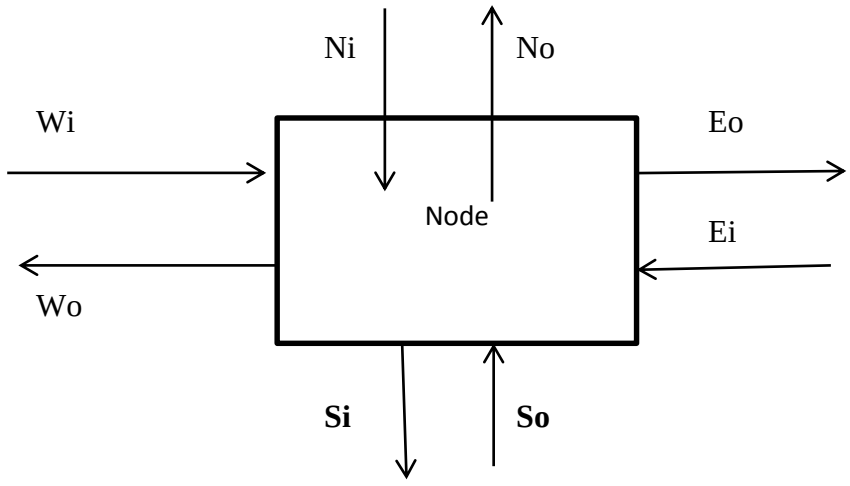
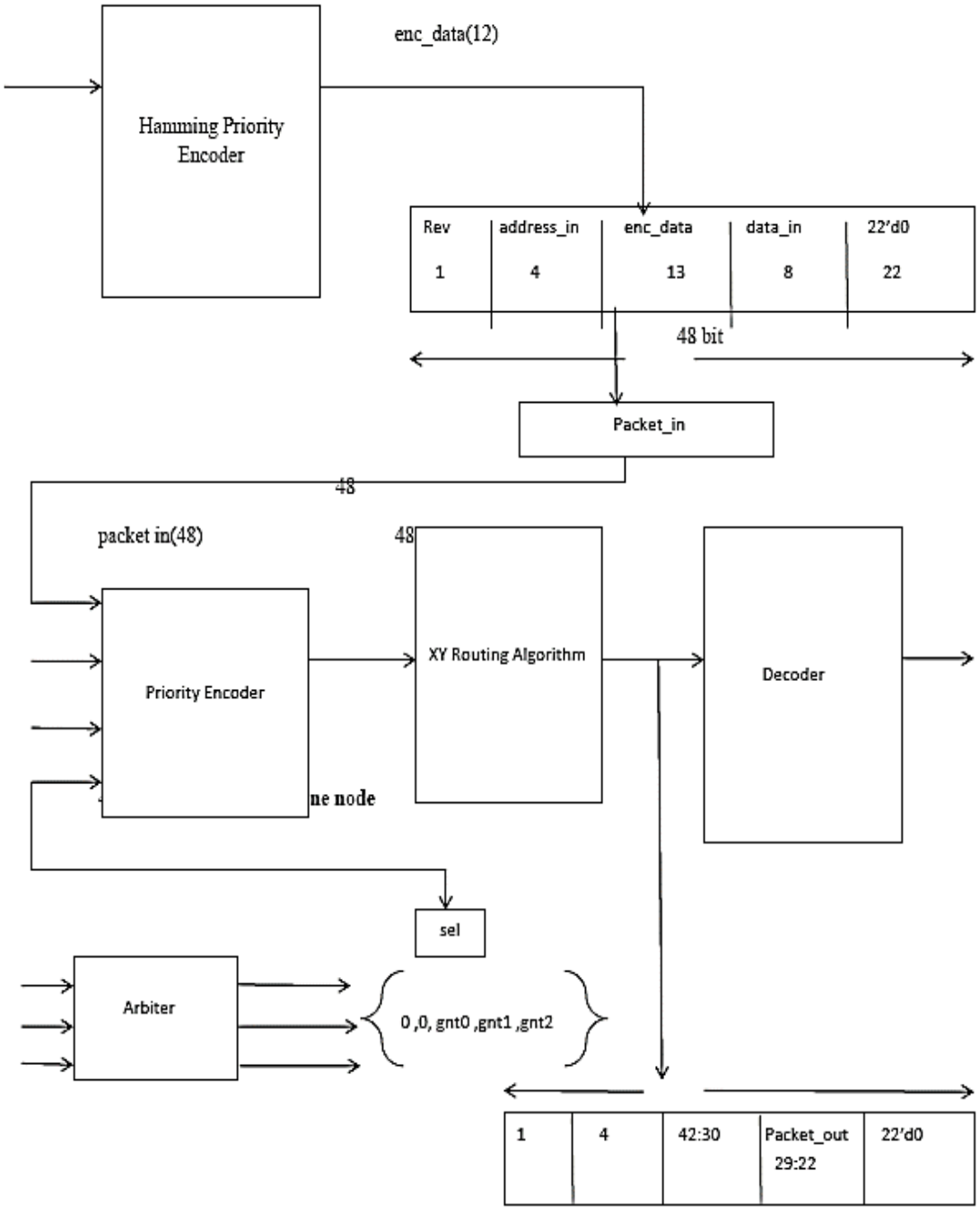
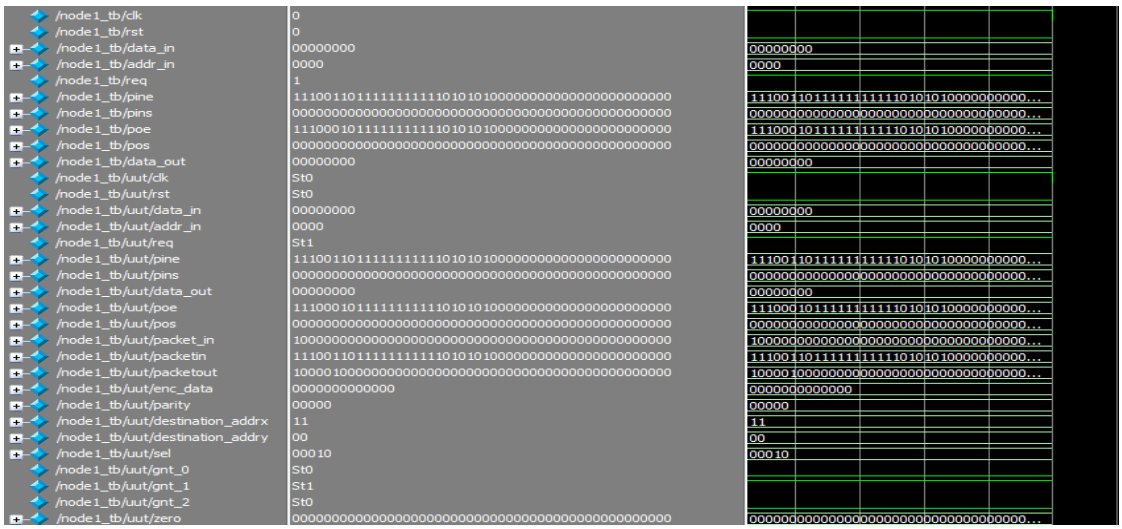


Fig : representation of one node

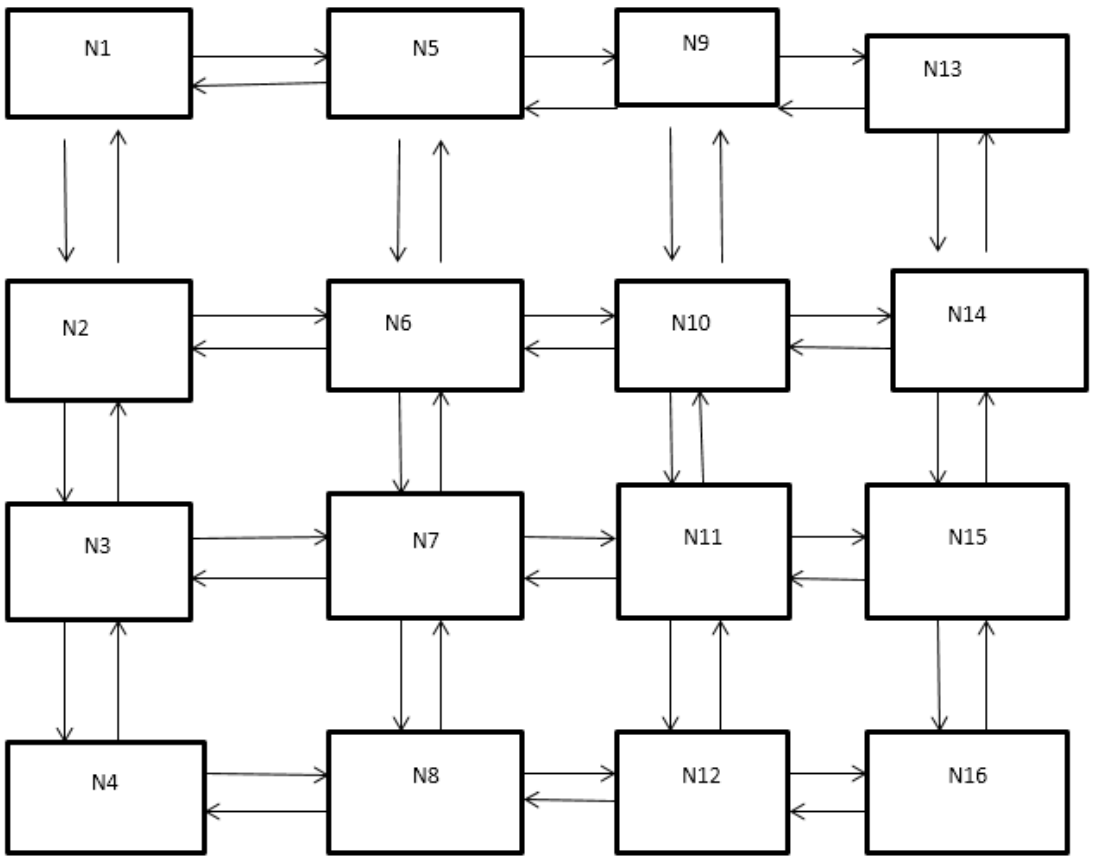
4.1 Block diagram



4.2 Simulation result of one node

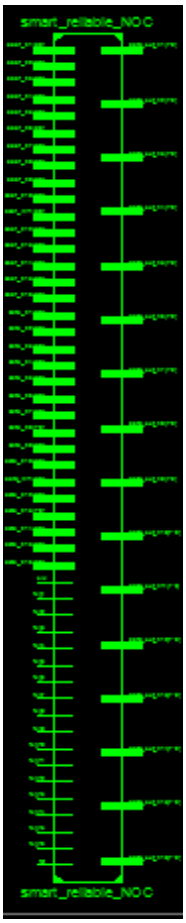


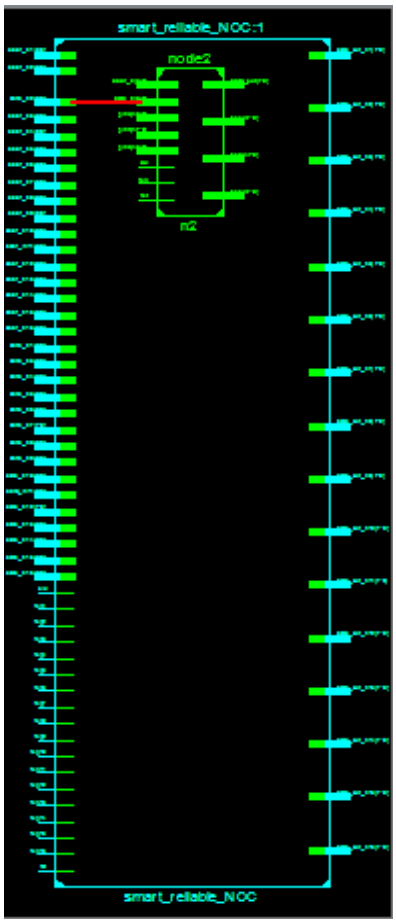
5. NOC OF 4X4



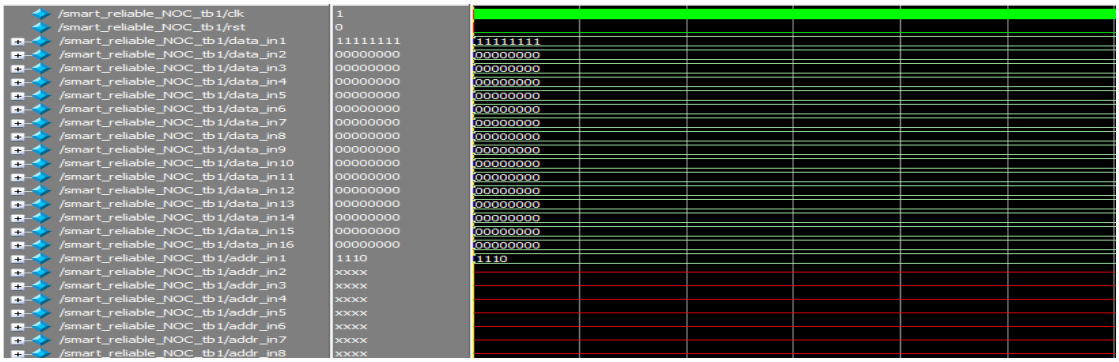
5.1 Result and Analysis

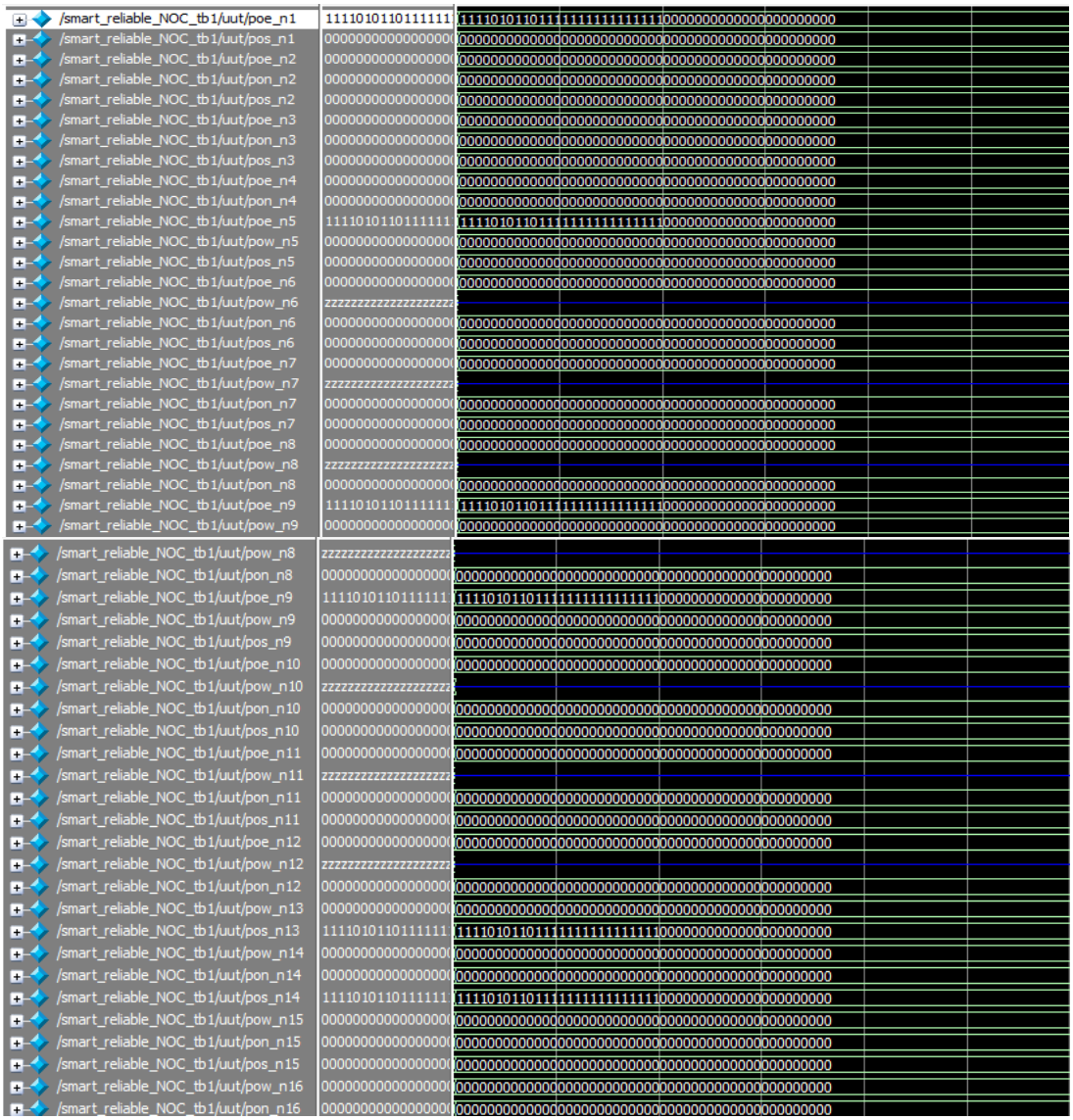
1.Top Module





5.2 Simulation Result of 4x4




/smart_reliable_NOC_tb1/uut/poe_n1
/smart_reliable_NOC_tb1/uut/pos_n1
/smart_reliable_NOC_tb1/uut/poe_n2
/smart_reliable_NOC_tb1/uut/pon_n2
/smart_reliable_NOC_tb1/uut/pos_n2
/smart_reliable_NOC_tb1/uut/poe_n3
/smart_reliable_NOC_tb1/uut/pon_n3
/smart_reliable_NOC_tb1/uut/pos_n3
/smart_reliable_NOC_tb1/uut/poe_n4
/smart_reliable_NOC_tb1/uut/pon_n4
/smart_reliable_NOC_tb1/uut/poe_n5
/smart_reliable_NOC_tb1/uut/pow_n5
/smart_reliable_NOC_tb1/uut/pos_n5
/smart_reliable_NOC_tb1/uut/poe_n6
/smart_reliable_NOC_tb1/uut/pow_n6
/smart_reliable_NOC_tb1/uut/pon_n6
/smart_reliable_NOC_tb1/uut/pos_n6
/smart_reliable_NOC_tb1/uut/poe_n7
/smart_reliable_NOC_tb1/uut/pow_n7
/smart_reliable_NOC_tb1/uut/pon_n7
/smart_reliable_NOC_tb1/uut/pos_n7
/smart_reliable_NOC_tb1/uut/poe_n8
/smart_reliable_NOC_tb1/uut/pow_n8
/smart_reliable_NOC_tb1/uut/pon_n8
/smart_reliable_NOC_tb1/uut/poe_n9
/smart_reliable_NOC_tb1/uut/pow_n9
/smart_reliable_NOC_tb1/uut/pow_n8
/smart_reliable_NOC_tb1/uut/pon_n8
/smart_reliable_NOC_tb1/uut/poe_n9
/smart_reliable_NOC_tb1/uut/pow_n9
/smart_reliable_NOC_tb1/uut/pos_n9
/smart_reliable_NOC_tb1/uut/poe_n10
/smart_reliable_NOC_tb1/uut/pow_n10
/smart_reliable_NOC_tb1/uut/pon_n10
/smart_reliable_NOC_tb1/uut/pos_n10
/smart_reliable_NOC_tb1/uut/poe_n11
/smart_reliable_NOC_tb1/uut/pow_n11
/smart_reliable_NOC_tb1/uut/pon_n11
/smart_reliable_NOC_tb1/uut/pos_n11
/smart_reliable_NOC_tb1/uut/poe_n12
/smart_reliable_NOC_tb1/uut/pow_n12
/smart_reliable_NOC_tb1/uut/pon_n12
/smart_reliable_NOC_tb1/uut/pow_n13
/smart_reliable_NOC_tb1/uut/pos_n13
/smart_reliable_NOC_tb1/uut/pow_n14
/smart_reliable_NOC_tb1/uut/pon_n14
/smart_reliable_NOC_tb1/uut/pos_n14
/smart_reliable_NOC_tb1/uut/pow_n15
/smart_reliable_NOC_tb1/uut/pon_n15
/smart_reliable_NOC_tb1/uut/pos_n15
/smart_reliable_NOC_tb1/uut/pow_n16
/smart_reliable_NOC_tb1/uut/pon_n16

5.3 Design Summary

Device Utilization Summary			
Slice Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	18	207,360	1%
Number used as Flip Flops	18		
Number of Slice LUTs	29	207,360	1%
Number used as logic	29	207,360	1%
Number using O6 output only	29		
Number of occupied Slices	28	51,840	1%
Number of LUT Flip Flop pairs used	47		
Number with an unused Flip Flop	29	47	61%
Number with an unused LUT	18	47	38%
Number of fully used LUT-FF pairs	0	47	0%
Number of unique control sets	2		
Number of slice register sites lost to control set restrictions	6	207,360	1%
Number of bonded IOBs	57	960	5%
IOB Flip Flops	13		
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Average Fanout of Non-Clock Nets	2.25		

5.4 Timing Sumaary

Minimum input arrival time before clock: 1.455ns

Maximum output required time after clock: 4.985ns

Maximum combinational path delay: 5.208ns

Applications:

- SMART antennas
- On chip network topologies
- Wireless LAN

Conclusion:

In this project, here I proposed new reliable router RKT Switch, Which is 4X4 mesh network topology. We are using for error detecting and correcting hamming codes, which is single bit correction technique. We are using for routing Adaptive Modified XY routing algorithm, where the main difficulty is to distinguish the bypasses of an unavailable component in the NOC (due to the use of the adaptive algorithm) from Real routing errors. Here I am not using any separate loopback module to process data or get back the data. The proposed design optimized in terms of area (LUT's and Slices) and works at high speed.

Future work:

In future, we focus on evaluating accurately the impact of faulty detection blocks and improving the routing error detection mechanisms, by protecting the DAI links and routing detection blocks against errors.

REFERENCES

[1] K. Sekar, K. Lahiri, A. Raghunathan, and S. Dey, "Dynamically configurable bus topologies for high-performance on-chip communication," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 10, pp. 1413–1426, Oct. 2008.

[2] J. Shen and P. Hsiung, *Dynamic Reconfigurable Network-on-Chip Design: Innovations for Computational Processing and Communication*, J. Shen and P. Hsiung, Eds. Hershey, PA, USA: IGI Global, 2010.

[3] G.-M. Chiu, "The odd-even turn model for adaptive routing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 11, no. 7, pp. 729–738, Jul. 2000.

[4] Y. M. Boura and C. R. Das, "Efficient fully adaptive wormhole routing in n-dimensional meshes," in *Proc. 14th Int. Conf. Distrib. Comput. Syst.*, Jun. 1994, pp. 589–596.

[5] C. Bobda, A. Ahmadinia, M. Majer, J. Teich, S. Fekete, and J. van der Veen, "DyNoC: A dynamic infrastructure for communication in dynamically reconfigurable devices," in *Proc. Int. Conf. Field Program. Logic Appl.*, Aug. 2005, pp. 153–158.

[6] T. Pionteck, R. Koch, and C. Albrecht, "Applying partial reconfiguration to networks-on-chip," in *Proc. Field Program. Logic Appl. Int. Conf.*, Aug. 2006, pp. 1–6.

[7] S. Jovanovic, C. Tanougast, and S. Weber, "A new high-performance scalable dynamic interconnection for fpga-based reconfigurable systems." in *Proc. Int. Conf. Appl.-Specific Syst., Archit. Process.*, Jul. 2008, pp. 61–66.