FERNA: a Performance/Cost Aware Spare Switch Selection Algorithm for Fault Tolerant NoC Architecture

Majid Janidarmian[1], Ahmad Khademzadeh[2], Melika Tinati[1], Maryam Ghavibazoo[1], and Atena Roshanfekr[1]

Abstract—**In this paper a fast and efficient spare switch selection algorithm is presented in a reliable NoC architecture based on specific application mapped onto mesh topology called FERNA. Based on ring concept used in FERNA, this algorithm achieves best results equivalent to exhaustive algorithm with much less run time improving two parameters. Inputs of FERNA algorithm for response time of the system and extra communication cost minimization are derived from simulation of high transaction level using SystemC TLM and mathematical formulation, respectively. The results demonstrate that improvement of above mentioned parameters lead to advance whole system reliability that is analytically calculated. Mapping algorithm has been also investigated as an effective issue on extra bandwidth requirement and system reliability.**

Key words—**NoC, fault tolerant system, spare switch, mapping, reliability.**

I. INTRODUCTION

The International Technology Roadmap for Semiconductors (ITRS) predicts that chips with over 4 billion transistors operating at 10 GHz speeds will be commercialized before the end of the decade [1]. So, the number of Intellectual Properties (IPs) included in System on Chip (SoC) is increased persistently [2]-[4]. Future SoC designs will need efficient on-chip communication architectures that can provide efficient and scalable data transport among the IPs [5]. The on-chip interconnection architectures used in current SoCs are dedicated wires or shared medium buses which cause huge challenges in performance, power consumption, delay, and reliability, etc. [6]-[8].

Network on chips (NoC) have emerged as a feasible solution to handle above challenges. It solves diversified problems that SoC is confronted with , by adopting the methods of packet switching and routing and separating process units from communication infrastructure [9],[10]. It is a promising paradigm due to its advantages like greater reusability, scalability, and predictability in the electrical parameters in addition to its importance in bandwidth guarantee required applications [1]. The way in which the cores and switches are connected, defines the topology or architecture of the network-on-chip. Different NoC topologies can drastically affect the network characteristics such as average inter-IP distance, total wire length and communication flow distributions [11],[12]. A 2-D mesh topology consists of nodes that are arranged in a rectilinear grid where each node is bi-directionally connected to its adjacent neighbors [2], which is the simplest and most dominant topology for today's regular tile-based NoCs. Also mapping of IP cores on a given platform is one of the important aspects of NoC design [1]. That is, different mapping algorithms are presented to decide which core should be linked to which switch [13] most of which were implemented onto mesh-based NoC architecture.

As the scalability of chips increases, so does the probability of errors, hence making reliability a major issue in scaling chips [14]. In 2008, a fault tolerant mesh-based NoC architecture with the ability of recovering from single permanent failures by adding a redundant link between each core and one of its neighboring switches is presented as a system that significantly improves reliability and has a very little effect on performance [15]. In this architecture, only one spare switch should be selected among all possible alternative switches. This has an influential effect on overall performance in terms of the average response time and reliability of the system. Regarding to this work, in this paper a new fast and optimum algorithm based on performance measurement and extra communication cost is proposed to find a best configuration that also results in a more reliable system. Finally, to show mapping effect in extra communication cost and system reliability, a mapping algorithm which is one of the best concepts in this area has been used for experimental applications.

[1]Computer Engineering Department, Science and Research Branch, Islamic Azad University, Tehran, Iran, {jani, m.tinati, m.ghavi, a.roshan}@srbiau.ac.ir.
[2]Iran Telecommunication Research Center, Tehran, Iran, zadeh@itrc.ac.ir.

The paper is organized as follows: section 2 briefly reviews the fault tolerant NoC architecture and its characteristics. Some prerequisite definitions for our algorithm are explained in section 3. FERNA spare switch selection algorithm is presented in section 4 followed by performance result and then analytical results of system reliability and extra communication cost are calculated. Finally, we draw some conclusions in section 5.

II. FAULT TOLERANT NOC ARCHITECTURE

As the number of transistors on a chip increases, the problems associated with deep sub-micron will become more pronounced and may therefore pose problems of link and/or switch failures that have exacerbated reliability issues in on-chip interconnects [16],[17]. This argument strengthens the notion that chips need to be designed with some level of built-in fault tolerance [14]. Two different kinds of errors are probable to occur in a NoC: transient and permanent errors. Transient failures can occur on a chip for many reasons: alpha particles emitted by trace uranium and thorium impurities in packages and also high-energy neutrons from cosmic radiations can cause soft errors in semiconductor devices. Similarly, low energy cosmic neutrons interacting with isotope boron-10 can cause soft errors. These events, generally called single-event upsets, can affect the storage elements of a chip such as latches, memory and registers [18]. These faults are treated by error detection and correction coding techniques and retransmitting data. Crash or permanent failures can occur due to electro migration of a conductor or a connection failure permanently halting the operation of some modules. These faults like Gaussian noise on a channel and alpha particles strikes on memory and logic elements can cause one or more bits to be in error but do not cause permanent failures [19]. Generally permanent errors do not disappear as the time passes, hence the retransmitting solution does not solve the problem. Therefore dynamic rerouting concept would be helpful. By assuming a faulty switch in a mesh-based NoC, it is obvious that the core directly connected to the faulty switch is inaccessible and the rerouting technique is not helpful any more. Hence, the fault tolerant NoC architecture proposed in [15] recovers from switch failures by adding a redundant link between each core and one of its neighboring switches, and applying modifications to the NoC components in addition to using a rerouting strategy.

To minimize the hardware cost, only one of the possible spare switches for a core is chosen. On the other hand, with respect to possible locations for each core as shown in Fig.1, there are two constraints to select a spare switch.

1. Each switch is limited to be linked by only one core.
2. Each core is located in neighborhood of its local and adjacent switches and all cores should be placed in different locations.

The used spare switch selection algorithm of [15] is Exhaustive algorithm that tries all valid placements

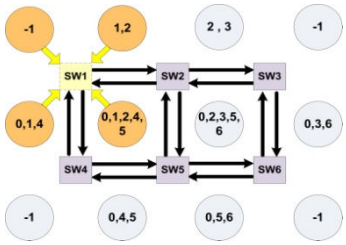


Figure1.Possible locations for each core

satisfying constraint 2 and eliminates the options among which constraint 1 is not satisfied. Using Exhaustive algorithm is absolutely non-efficient for NoCs with many cores because of its exponential time complexity but it can achieve perfect results. Hence, Greedy spare switch selection algorithm based on back tracking concept was provoked in [20] to reduce run time. Although run time of Greedy method is better than Exhaustive algorithm, its time complexity is still exponential and necessarily does not lead to the perfect results for the whole system. For solving these problems a new spare switch selection algorithm called FERNA is introduced whose time complexity is polynomial and results in perfect configuration for fault tolerant NoC architecture. Greedy and FERNA spare switch selection algorithms need to determine priorities of switches based on performance/cost parameter as inputs explained in the following section.

III. PREREQUISITE OF SPARE SWITCH SELECTION ALGORITHMS

In the fault tolerant NoC Architecture mentioned before, specific application is early mapped onto mesh topology and then spare switch selection algorithm is applied based on inputs that determine worth of spare switches to improve performance/cost.

A. *Mapping Problem*

As mapping of IP cores on a given platform is one of three aspects of NoC design [1], it is considered as a complicated problem which is better to be solved by innovative, combinational and flexible methods. Different mapping algorithms are supposed to decide which core should be connected to which switch [13]. Each application is divided to several IP cores. The manner of relation between cores is shown by a core graph.

Definition1: The core graph is a directional graph G(V,E), in which each vertex $v_i \in V$ shows a core, and a directional edge $e_{i,j} \in E$ illustrates connection between v_i and v_j. The weight of $e_{i,j}$ that is shown as $comm_{i,j}$, represents the bandwidth requirement/volume of the communication from v_i to v_j.

We display an IP core along with a switch connected to it by a Resource Network Interface (RNI) as a node, and the manner of node links and bandwidth between nodes would be determined by NoC topology graph.

Definition2: The NoC topology graph is a directional graph T(N,L) in which each vertex $n_i \in N$ represents a node in the NoC topology, and its directional edge that is shown by $l_{i,j} \in L$ shows a physical link from n_i to n_j and for each $l_{i,j}$ a bandwidth $BW(l_{i,j})$ is considered. Paying attention to used routing algorithm, $r_{i,j}$ shows routing path between n_i and n_j. Each $r_{i,j}$ includes a number of links out of existing links in NoC topology, and these links are defined with $L(r_{i,j})$. The definitions are demonstrated in Fig.2.

The core graph mapping G(V,E) on NoC topology graph T(N,L) is defined by a one to one mapping function (1).

$$\text{map}: V \longrightarrow N, \text{s.t.map}(v_i) = n_j, \{\forall\ v_i \in V, \exists\ n_j \in N,\ |V| \le |N|\} \quad (1)$$

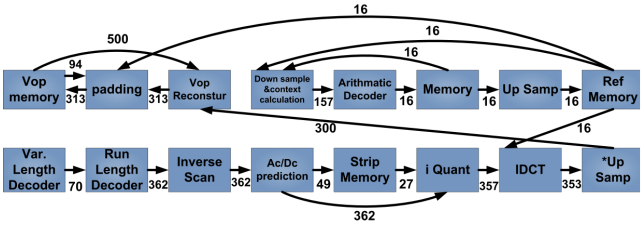
Choosing the best alternative switches for all cores should be done based on our goals after mapping.

B. *Spare Switch Selection based on average response time of the system*

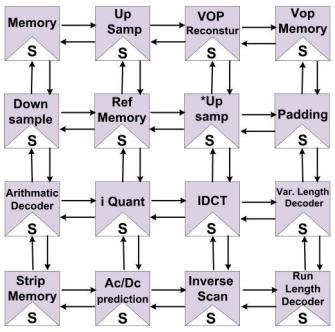
In [20], the fault tolerant NoC architecture has been implemented in SystemC TLM 2.0 library saying its details. The worse case response time of the destination cores and the average response time of the system are considered as the evaluation parameters. Using (2), one can evaluate delay for sending one bit on communication channel from node i to node j in mesh topology:

$$T_{bit}^{n_i,n_i} = \sum_{k=1}^{n_{hops}} (T_{S_k bit} + T_{fifo_k bit}) + (n_{hops} - 1) \times T_{lbit} \quad (2)$$

$T_{bit}^{n_i,n_i}$ indicates transmission delay of one bit between two nodes. T_{Sbit} and T_{lbit} represent forwarding delay of one bit in switch and link, respectively. $T_{fifo_k bit}$ is waiting time in the input FIFO of the switches. n_{hops} is the number of switches that one bit has to pass to reach from the source to the destination. Finally, delay for transmitted data could be calculated by (3).



(a)



(b)

Figure2. (a) Core Graph, (b) Topology Graph

$$T_{total}^{n_i,n_i} = \text{Data size} \times T_{bit}^{n_i,n_i} \quad (3)$$

In this simulation model, the weight of $e_{i,j}$ that is shown as $comm_{i,j}$, represents the volume of the communication from v_i to v_j. Consequently, the average response time of the system is estimated by maximum delays of last bits arriving at the destination cores.

The NoC simulation model is implemented using SystemC TLM and has been run for each switch failure and the average response time of the system is calculated for different possible spare switches.

After performing above simulation, we need to choose only one spare switch for each core in terms of constraints. Thus, all calculated delays are used as inputs of spare switch selection algorithm. Different algorithms try to find better alternative switches for all cores based on the average response time of the system.

C. *Spare Switch Selection based on extra communication cost*

If the weight of $e_{i,j}$ that is shown as $comm_{i,j}$, represents the bandwidth requirement of the communication from v_i to v_j, the communication cost is calculated by (4)

$$commcost = \sum_{k=1}^{|E|} \left(vl(d^k) \times \text{dist}(\text{source}(d^k), \text{dest}(d^k))\right) \quad (4)$$

The following part of this section depicts how we can use bandwidth requirement as inputs for spare switch selection algorithms. The communication cost is calculated when all switches work correctly. If a switch fails, communication cost is increased because of some rerouting paths. If SW_i is failure point and SW_s is spare switch of $Core_i$, total extra cost of links is calculated as follows:

Three types of paths need to be rerouted because of SW_ifailure:

$$\forall\ m,n = 1,2,\dots,n^2 \text{ and } m,n \neq i$$
$$type\ 1: source\ (d^k) = SW_i \text{ and destination } (d^k) = SW_m$$
$$type\ 2: source\ (d^k) = SW_m \text{ and destination } (d^k) = SW_i$$
$$type\ 3: source\ (d^k) = SW_m \text{ and destination } (d^k) = SW_n, \text{if } SW_i \epsilon\ S(r_{m,n})$$

Bandwidth of each link is obtained as (5):

$$\forall\ link\ l_{a,b},$$
$$BW(l_{a,b}) = \sum_{k=1}^{|E|} vl(d^k) \times f\left(l_{a,b}, r_{source(d^k),dest(d^k)}\right) \quad (5)$$

$$f(l_{a,b}, r_{i,j}) = \begin{cases} 1, if\ l_{a,b} \in L(r_{i,j}) \\ 0, \text{otherwise} \end{cases}$$

Fraction of each bandwidth link that is in routing path of these types can be reused in rerouting.

$$rr_i(D) = d^k \text{ s should be rerouted while } SW_i \text{ is failure point}; k = 1,2,\dots,|E|;$$
$$Av_{i,s}\left(BW\ (l_{u,v})\right) = \text{useable bandwidth for rerouting; } l_{u,v} \epsilon\ L(\text{Routing Path of } rr_i(D))$$

$$\forall\ link\ l_{a,b},\quad TERB_{i,s}(l_{a,b}) = Total\ Extra\ Requirment\ Bandwidth\ of\ link_{a,b} = \sum_{\forall\ d^k, d^k \epsilon\ rr_i(D)} vl(d^k) \times f\left(l_{a,b}, rerouting_{source(d^k),dest(d^k)}\right) \quad (6)$$

$$f(l_{a,b}, rerouting_{i,j}) = \begin{Bmatrix} 1, if\ l_{a,b} \epsilon\ L(rerouting_{i,j}) \\ 0, otherwise \end{Bmatrix} \quad (7)$$

$$ECL_{i,s}(l_{a,b}) = Extra\ Cost\ of\ Link_{a,b} = \begin{cases} 0, & Av_{i,s}\left(BW\left(l_{a,b}\right)\right) \geq TERB_{i,s}(l_{a,b}), \\ Av_{i,s}\left(BW\left(l_{a,b}\right)\right) = Av_{i,s}\left(BW\left(l_{a,b}\right)\right) - TERB_{i,s}(l_{a,b}) & \\ TERB_{i,s}(l_{a,b}) - Av_{i,s}\left(BW\left(l_{a,b}\right)\right), & otherwise \\ Av_{i,s}\left(BW\left(l_{a,b}\right)\right) = 0 & \end{cases} \quad (8)$$

$$\text{T}ECL_{i,s} = Total\ Extra\ Cost\ of\ links = \sum_{\forall\ l_{a,b}\ \epsilon\ L(Rerouting\ Path\ of\ rr_i(D))} ECL_{i,s}(l_{a,b}) \quad (9)$$

Total extra cost of links is calculated for different possible spare switches while a switch fails. After these calculations, we need to choose only one spare switch for each core. Thus, all calculated TECLs are used as inputs of spare switch selection algorithm. Different algorithms try to find better alternative switches for all cores based on extra communication cost shown below:

$$Final\ ECL(l_{a,b}) = Max\ \left(ECL_{i,candidated}(l_{a,b})\right);\ i = 1,2,\dots,n^2 \quad (10)$$

$$Extra\ commcost = \sum_{\forall\ link\ \epsilon\ L} Final\ ECL(link_{a,b}) \quad (11)$$

IV. FERNA ALGORITHM

In this section we present a novel method for spare switch selection in a fault tolerant NoC architecture that was perused before and named FERNA[1].

A. Proposed Algorithm

In above mentioned fault tolerant NoC architecture, wherein each core is connected to two switches and each switch is spared by one core, as depicted in Fig.3, some rings of cores and switches are essentially formed in an overall spare selection set regarding constraints. We exploit this concept to find an expedient solution. Our proposed algorithm takes each individual core and selects the best spare switch according to results of above mentioned SystemC simulation and extra communication cost formulation. The algorithm continues with a core whose directly connected switch was selected as a spare of previous core. A simple rule is employed so that the rings contain all switches and cores:

Rule: A selected switch spare should not be the one that causes the remaining switches and cores to get locked in which they could not participate in any selection ring. This is acquired surveying possible locations of a core to be fixed on and the switches that each core can take as its spare. This algorithm has a polynomial time complexity and can be resulted in a short time. So to have the best result, the starting switch is each n^2 individual core in n×n mesh and the best result is chosen among all solutions. In case that a ring is formed and still there are remaining cores without spares, the algorithm attends the highest ranked cores according to the following ranking equation of the cores:

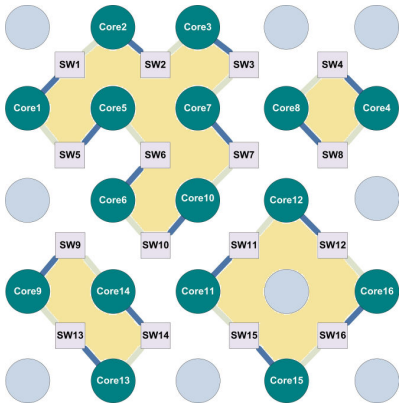


Figure 3. FERNA algorithm applied to 4×4 mesh-based NoC

$$Ranking(C_i) = \sum_{\substack{\forall j=1,2,\dots|v| \\ i \neq j}} (comm_{i,j} + comm_{j,i}) \quad (12)$$

$comm_{i,j}$ and $comm_{j,i}$ are bandwidth requirements of the communication from v_i to v_j and v_j to v_i respectively which were explained before.

One wavering situation may occur in which a core may have only two locations left for taking spare and being spared from. Although the neighbors adjacent to this locations may have other choices for spare selection but they have to be linked from these locations even if according to ranking formulation it is not the turn of these cores to be linked.

B. Results of spare switch selection algorithms decreasing average response time

According to the results of SystemC TLM simulation previously represented, in this section we apply Greedy and FERNA algorithms on three applications greedily mapped on mesh, MP3 and H262 encoder and decoder (MMS) presented in [20], Video Object Plane Decoder with12 cores (VOPD) and Multi-Window Displayer (MWD) used in [20]. Greedy algorithm selects spares with respect to cores ranking and in cases that there was no choice to select a spare for a core, the algorithm uses back tracking mechanism. By this mechanism, the algorithm selects other spare selection choices to achieve an acceptable result. However, FERNA algorithm starts with higher-ranked core and then selects the best switch for this core according to the simulation results. Next core to start with this time, is the core directly connected to recent switch. This mechanism will continue to form a ring with regard to constraints. The results show that FERNA obtains the best result rendered by Exhaustive algorithm in much lesser time than Greedy and Exhaustive algorithm . The system performance of three

[1] Finding Efficient Rings in NoC Architecture

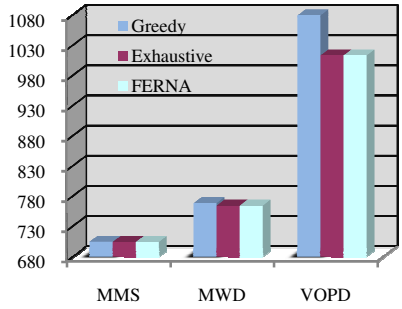


Figure 4. Average response time for each application/algorithm

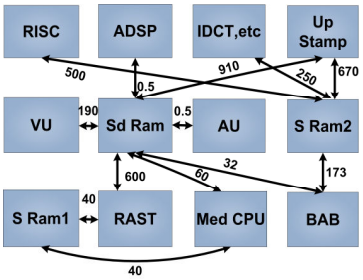


Figure 5. Core graph of MPEG-4

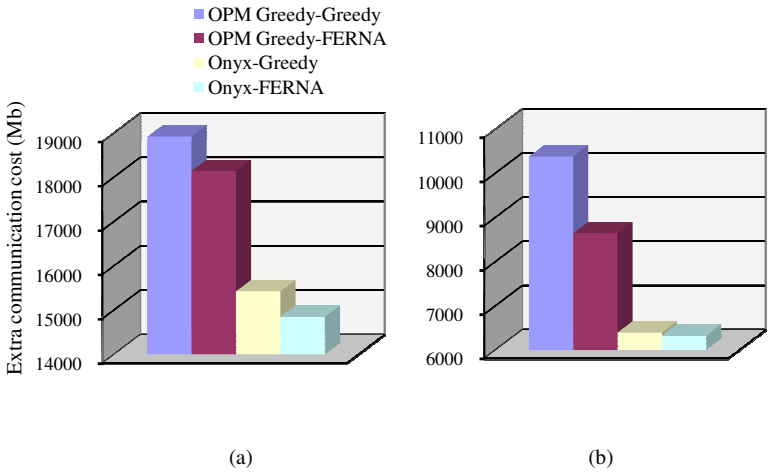
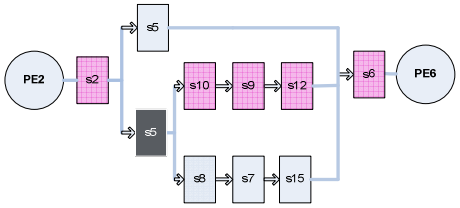


Figure 6. Extra communication cost , (a) MPEG-4 , (b) VOPD , In guide box, mapping algorithms come first followed by spare switch selection algorithms



$$R_{2,6}|_{SW_5\ fails} = R_2R_5R_6 + 0.5 \times R_2 \times (1-R_5)R_8R_7R_{15}R_6 + 0.5 \times R_2 \times (1-R_5)R_{10}R_9R_{12}R_6$$

Figure 7. Reliability diagram

algorithms is depicted in Fig.4.

C. Results of spare switch selection algorithms decreasing extra communication cost

In this section, considering the results of mathematical calculation of extra communication cost, we apply the FERNA and Greedy algorithms on two real applications, VOPD (with 16 cores) and MPEG-4 which in [13] were evaluated considering bandwidth requirement. Core graphs of these applications are represented in Fig.2(a) and Fig.5 respectively. The used mapping algorithm is Optimized Greedy which has better results comparing to Greedy algorithm. As the results show, FERNA algorithm has lesser extra communication cost than Greedy algorithm 17% in VOPD and 4% in MPEG-4. Also to demonstrate the influence of mapping algorithm on extra bandwidth, Onyx as a competent mapping algorithm in decreasing extra bandwidth cost was selected to map the applications and we compared it to Optimized Greedy algorithm. Fig.6 depicts the whole results. According to the results, Onyx mapping algorithm preciously decreases the bandwidth requirement even comparing to a proper mapping algorithm like Optimized Greedy. Improvement remedy of extra communication cost, utilizing Onyx in VOPD and MPEG-4 versus Optimized Greedy are 27% and 18% respectively, while FERNA is the spare switch algorithm.

D. Reliability of a Fault Tolerant Architecture

In the fault tolerant NoC system, we could provide a spare switch for each core with mentioned algorithms. In this architecture when a switch fails, the core will be accessible and also the network will not be blocked due to absence of faulty switch. Preventing unessential hardware complexity and redundancy, one of the eight neighbor switches is selected as a spare switch for each core. The spare switches cause an improvement in system reliability and will make the system fault tolerant confronting with switch failures.

To calculate the reliability of the system, simple serial and parallel rules are used, hence for each pair of source and destination nodes in core graph, the reliability of the path is production of reliability of the switches which are met through the unique path according to XY routing algorithm. Then the system reliability is calculated again, each time assuming each individual switch fails and accordingly a new rerouting path is accounted contemplating XY algorithm and the additional spare switches. The final analytical formulation for system reliability is as (13):

$$R_{NOC} = \prod_{(i,j)\in CTG}[\prod_{k=1}^{S} R_k + \sum_{k=1}^{S}(1-R_k)\cdot(R_{i,j}|SW_k Fails)] \qquad (13)$$

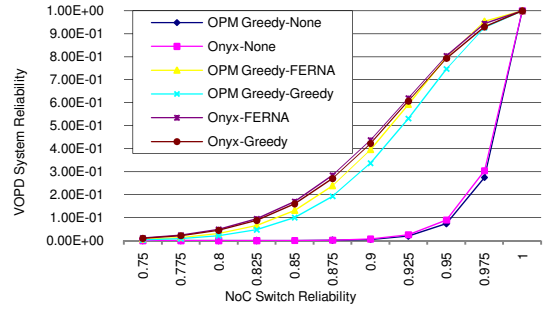
in which, (i,j) represents each pair of source and destination nodes in core graph. "S" shows the number of switches in each routing path and $R_{i,j}$ is the reliability of the path from i to j. We use this formulation given in [15], and calculate the reliability of systems with different mappings and spare switch selection algorithms. Table1 illustrates one sample of FERNA spare switch selection and Fig.7 shows the reliability calculated for this configuration in a special case.

In this figure a case is indicated in which deciding to reroute from source to destinations, two different rerouting paths are begotten. In such case the reliability of source to destination path is calculated as probability of selecting each of two paths multiplied to the reliability of each path.

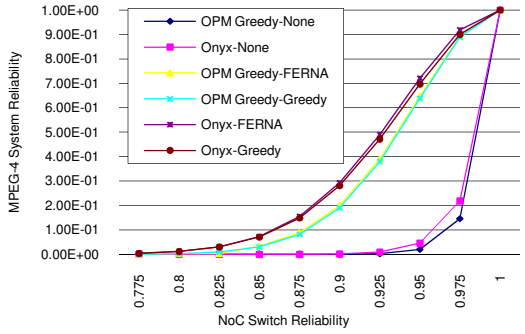
Analytical results of system reliability in different cases are shown in Fig.8. According to that, FERNA is expected to have better results comparing to Greedy algorithm. Meanwhile, using an optimum and appropriate mapping algorithm like Onyx improves system reliability besides extra communication cost.

Table 1. Spare switch selection for VOPD

Faulty Switches	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Alter Switches	9	1	2	3	4	5	6	7	10	8	15	11	12	16	13	14



(a)



(b)

Figure 8. System Reliability (a) Mapped VOPD core graph (b) Mapped MPEG-4 core graph, In guide box, mapping algorithms come first followed by spare switch selection algorithms

V. CONCLUSIONS

In this paper a heuristic method for spare link selection in a fault tolerant application specific mesh-based NoC architecture is presented namely FERNA. This algorithm results in the best configuration in lesser time versus previous algorithms considering performance, extra communication cost and system reliability. Also the Onyx mapping effect on extra bandwidth requirement and system reliability was evaluated. The results demonstrate that a perfect configuration is acquired using a convenient mapping algorithm and applying FERNA spare switch selection algorithm. Implementing FERNA on other NoC topologies, utilizing different routing techniques and assigning more than one spare switches for each core are some interesting subjects in this area for future works.

REFERENCES

[1] International Technology Roadmap for Semiconductor (ITRS), 2007 edition, http://www.itrs.net/.

[2] W. Dally, B. Towles. “Route packets, not wires: onchip interconnection networks,” Proc. the Design Automation Conference, Las Vegas, NV, pp.684-689,2001.

[3] A. Hemani, A. Jantsch, S. Kumar et al. “Network on a chip: an architecture for billion transistor era,” Proc. IEEE NorChip Conference, pp.166-173,2000.

[4] T. Ye. “On-chip multiprocessor communication network design and analysis,” PhD Dissertation, Stanford University, 2003.

[5] A.E. Kiasari, S. Hessabi, H. Sarbazi-Azad, “PERMAP: A performance-aware mapping for application-specific SoCs,” Application-Specific Systems, Architectures and Processors, , pp.73-78 , Jul. 2008.

[6] L. M. Ni and P. K. McKinley, “A survey of Wormhole Routing Techniques in Direct Networks”, IEEE on Computers, vol. 26, pp.62-76, Feb. 1993.

[7] I. Leijten et al., “Stream Communication between Real-Time Tasks in a High-Performance Multiprocessor”. Design, Automation and Test in Europe, 1998., Proceedings Publication. pp. 125-131,Feb 1998 .

[8] A. Hemani, A. Jantsch, *S.* Kumar, A. Postula, J. Oberg, M. Millberg, and D. Lindqvist, “Network on Chip: An architecture for billion transistor era”, Proc. of the IEEE NorChip Conference,pp. 68 - 73 , Nov. 2000.

[9] J. Hu, R. Marculescu.” Energy- and performance-aware mapping for regular NoC architectures”. IEEE Trans. On Computer-Aided Design of Integrated Circuits and Systems, pp. 551-562,Apr. 2005 .

[10] K. Goossens. “Formal methods for networks on chips”. Proc. Fifth International Conference on Application of Concurrency to System Design, p.188-189,Jun. 2005 .

[11] K. Yalamanchili, A. Pasalapudi, D. Majeti, V. Sunitha, “Design of Optimal Architectures Using Homogeneous Routers for Application Specific Network on Chip,” Emerging Trends in Engineering and Technology, pp.873-877,Jul.2008.

[12] C. Wu, Y. Li, S. Chai, “Design and Simulation of a Torus Structure and Rout Algorithm for Network on Chip,” 7th International Conference on, , pp. 1289-1292, Oct. 2007.

[13] M. Janidarmian, A. Khademzadeh, M. Tavanpour, “Onyx: A new heuristic bandwidth-constrained mapping of cores onto tile-based Network on Chip”, IEICE Electron. Express, Vol. 6, No. 1, pp.1-7, January 2009.

[14] M. Ali, M. Welzl, S. Hessler, S. Hellebrand, “A Fault tolerant mechanism for handling Permanent and Transient Failures in a Network on Chip,” Information Technology.Fourth International Conference on, pp. 1027-1032,Apr.2007 .

[15] F. Refan, H. Alemzadeh, S. Safari, P. Prinetto, Z. Navabi, “Reliability in Application Specific Mesh-Based NoC Architectures,” On-Line Testing Symposium, 2008. IOLTS apos;08. 14th IEEE International Volume , Issue , 7-9 ,pp.207 – 212 ,Jul. 2008.

[16] B. Vermeulen, J. Dielissen, K. Goossens,K. Ciordas, “Bringing Communication Networks On Chip: Test and Verification Implications,” IEEECommunications Magazine, Vol. 41, No. 9, pp. 74-81. Sep.2003.

[17] R. Ho, K. Mai, M. Horowitz, “The Future of Wires,” Proceedings of the IEEE, Vol. 89, No. 4, pp. 490 – 504,Apr.2001.

[18] S.K. Shukla, R.I. Bahar, “Nano, Quantum and Molecular Computing, Implications to High Level Design and Validation,” Kluwer Academic Publishers, Boston, 2004.

[19] M.L. Bushnell, V.D. Agarwal, ”Essentials of Electronic Testing for Digital, Memory and Mixed-Signal Circuits,” Kluwer Academic Publishers, USA, 2000.

[20] F. Renan, H. Alemzadeh, P.Kabiri, P. Prinetto, Z. Navabi, “Application Specific Configuration of a Fault-tolerant NoC Architecture,” Proc. Electronics Conference, 2008. BEC 2008. 11th International Biennial Baltic,pp.179-182,Oct.2008.