A Deadlock-Free Dynamic Reconfiguration Protocol for Distributed Routing on Interconnection Networks

Mohiadeen Abdul Kadhar
Department of Information Technology, National College of Engineering, India

Abstract: *In interconnection networks, reconfiguration protocol is necessary to remap and reconnect the network paths, so that the network remains connected. However, the reconfiguration process brings the deadlock problem and prevention of deadlock is a tedious task in this situation. In existing works, very little work have considered deadlock problem and further, they paid no attention to reduce packet loss rate. In this paper, we propose a Token-Based (TB) robust deadlock-free dynamic reconfiguration protocol. When a device observes topology changes or detects faulty nodes, it triggers the reconfiguration process and it becomes the Reconfiguration Controller (RC). Initially, HELLO message is transmitted by the RC to all devices for which they respond with a network status message. The RC constructs the new routing function based on the received network status messages. To synchronize the old and new routing functions, the RC distributes Reconfiguration Token (RT) in an ordered way. First, it distributes to the devices that surrounds the failed device and then to other devices. Every device holds the packet until it gets packet according to new routing function and then starts the transmission. By simulation, we show efficacy of our reconfiguration protocol. The network evaluation parameter like throughput, latency (delay) and pocket loss are measured in the high and low load scenarios in NS2 network simulator. We compare the results with existing protocol Overlapping Static Reconfiguration (OSR). Based on the simulation results we have proved that the proposed token based reconfiguration protocol is produces better efficiency in all aspects.*

Keywords: *Dynamic reconfiguration, RC, RT, deadlock, interconnection networks, distributed routing.*

Received August 7, 2012; accepted February 21, 2013; published online March 13, 2014

1. Introduction

1.1. Network Reconfiguration

In interconnection networks [1], topology changes and faults occurs frequently, it is essential to reconfigure. The reconfiguration protocol is necessary to update the routing table. Reconfiguration is required to remap and reconnect the network paths so, that the network remains connected. Node that detects a fault prompts the reconfiguration process. It disseminates control packets to remaining nodes to adjust the routing function [6]. The reconfiguration process requires local or global synchronization among nodes before its completion and routing function amendment on each node is asynchronous [15]. The process of reconfiguration is broadly divided into two classes namely, static reconfiguration [4] and dynamic reconfiguration [3]. Static reconfiguration is a simple technique. It halts data transmission until reconfiguration process completes. In this case, deadlock is avoided easily but halting transmission increases latency and the situation is worst for real time applications.

Conversely, dynamic reconfiguration allows data to be transmitted during reconfiguration process. In this circumstance, deadlock may occur due to the existence of both old and new routing functions [21].

1.2. Deadlock Freedom and Its Issues

An abnormal state of the networks that arise out of circular hold and wait dependencies on network resources is defined as deadlock [2]. When a network is in deadlock, it is held over packets indefinitely until and unless some action is taken to rectify it. Deadlock degrades the system reliability [14], availability and system performance [9]. We can classify the interconnection deadlock into three classes: Routing-dependent deadlocks [16], message-dependent deadlocks [17] and reconfiguration-induced deadlocks. Generally, cyclic dependency of resources causes routing dependent deadlocks. Turn models based routing functions can be used to avoid this type of deadlocks. When a device shares resource in the network, its message interactions and dependencies creates message dependent deadlocks [11]. Detaching the set of un routable packets from the network can assure deadlock freedom [18]. This can be done even when the reconfiguration process has inadequate channel resources in the network. During the reconfiguration process, deadlock can be avoided by an ideal design of the routing algorithm. While synchronizing old and new routing table, the routing function should possess deadlock free properties [8]. The network interconnect [13] should be designed efficiently. Inefficiency can lead to system crashes and unexpected system behavior, which is clearly unacceptable. Design issues that complicate the efficiency are described below:

1. When comparing deadlock detection/recovery phases, deadlock avoidance mechanisms require more network resources and makes implementation very complicated [19].
2. Apart from deadlocks, the reconfiguration process also suffers from message losses. During reconfiguration process, message will be lost, if the switch is deactivated [10].
3. Deadlock dependencies [5] can be occurred during and after reconfiguration due to the reconfiguration of a network's routing function [7].
4. Transmission of data packets during reconfiguration process increase congestion around the failed device [20].

1.3. Problem Identification

The work in the area of deadlock free-reconfiguration protocol has attained less attention in the literature. In existing works, one or two works has considered static reconfiguration approach, which is incompetent when compare with dynamic approach. In [8] have proposed deadlock free reconfiguration protocol that follows static reconfiguration approach? With their approach, each input port has to wait until it processes Reconfiguration Token (RT), mean while they disable intra-switch ports, which forwards packets from the input ports. Thus, more packets will miss their deadline and their approach increase packet loss ratio. Epoch-based dynamic reconfiguration approach is proposed in [10]. They do not stipulate any specific method to distribute the new routing table. Their routing function distribution may lead to additional deadlock condition. To provide a complete solution for deadlock free reconfiguration protocol and to avoid the fore-mentioned problems, we propose to develop a Token-Based (TB) robust deadlock-free dynamic reconfiguration protocol.

2. Proposed System

In this paper, we propose a TB robust deadlock-free dynamic reconfiguration protocol. Node that triggers the reconfiguration process becomes the Reconfiguration Controller (RC). The RC sends HELLO message to all network devices. Each network device sends back Network-STATUS (NW-STATUS) message to the RC. NW-STATUS contains the status of their neighboring devices. Based on received NW-STATUS message, the RC constructs the new routing function and before distributing it to the devices, it sends RT. The RC first transmits RT to the devices that surround the failed node. Once completing the distribution of RT, the RC distributes new routing function to devices. By receiving the new routing function, each device updates its routing table and removes the old one. At that moment, it can transmit packets that follow new routing function. Then, it holds the input packets until it gets the packets to transmit according to new routing function and then transmits the packets.

2.1. Invoking Reconfiguration Process

Let RC be the RC and ND be a network device. The network includes number of ND's. Changes in topology and faulty node detection trigger the reconfiguration process. Node that triggers the reconfiguration process becomes RC. As an initial step of reconfiguration process, the RC periodically broadcasts HELLO message to all ND. While receiving the HELLO message, each ND sends back NW-STATUS message to RC as shown in Figure 1.

$$RC \xrightarrow{HELLO} ND$$

$$RC \xleftarrow{NW-STATUS} ND$$

Figure 1. Network signals.

NW-STATUS message describes the topology of the network. The NW-STATUS message consists of bandwidth and delay values of neighboring ND. General format of NW-STATUS message is shown below in Table 1.

Table 1. Format of NW_STATUS.

Device ID	Neighboring Node	Bandwidth	Delay

2.2. New Routing Function Creation

The RC waits for a time t_{rc} to receive all possible NW-STATUS messages from NDs. After the expiration of the timer, it checks for changes in current topology. If changes occur, it builds the new routing function RF_{new} based on NW-STATUS messages. Now, the network has two routing functions old and new routing functions, RF_{old}, RF_{new} like Table 2 and Table 3 respectively.

Table 2. Old routing table (RF_{old}).

Source ND	**Destination ND**	**Next Hop**	**Hop Count**
ND1	ND4	ND2	2
ND1	ND6	ND2	4
ND1	ND7	ND7	0
ND1	ND3	ND2	1

After t_{wait}, ND3 noticed ND5 as a faulty ND. ND3 invokes the reconfiguration process and becomes the RC. It sends HELLO message to ND1, ND2, ND4, ND6, and ND7. These NDs respond the RC by sending NW-STATUS message. The RC constructs the new routing table of RF_{new} considering received NW-

STATUS messages. The new routing table of RF_{new} will be in the following format.

Table 3. New routing table (RF_{new}).

Source ND	Destination ND	Next Hop	Hop Count
ND1	ND4	ND2	2
ND1	ND6	ND2	2
ND1	ND7	ND7	0
ND1	ND3	ND2	1

2.3. Synchronizing Old and New Routing Tables

To synchronize both RF_{old} and RF_{new} routing functions, the RC distributes RT. The RC disseminates RT in an ordered way. It first sends RT to the NDs that surrounds the failure ND, and then passes the RT to other NDs. This is done to avoid further deadlocks in reconfiguration.

The below described Figure 2 represents the distribution of RT in the network. ND3 triggers the reconfiguration process and becomes RC. The RC finds ND5 as a faulty device. As a result, the RC first sends RT to ND5 and ND8, which surrounds the faulty ND (ND5). Subsequently, it sends RT to ND1, ND2 and ND7. As soon as the RC completes the distribution of RT, it disseminates RF_{new} in an ordered way. It follows the order used in RT distribution. Higher priority will be given to the NDs that surround the failure ND processes.

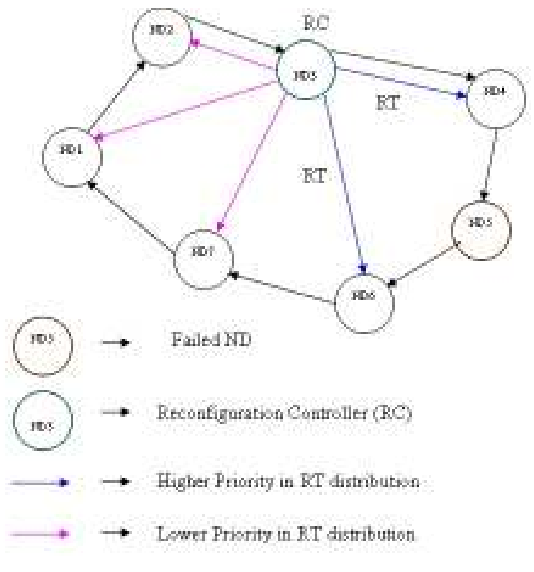
ND2
RC
ND3
ND4
RT
ND1
RT
ND5
ND7
ND6
ND5
Failed ND
ND3
Reconfiguration Controller (RC)
Higher Priority in RT distribution
Lower Priority in RT distribution

Figure 2. RT distribution.

2.4. Transmitting Data with New Routing Function

While processing the RT, the ND will receive its RF_{new} from the RC. Once a ND is given the RF_{new}, it updates its routing table with RF_{new} and removes the RF_{old}. At that time, ND can transmit packet only with new routing function. It holds the input packets until it gets the packets to transmit according to new routing function and then transmits the packets. To differentiate the new routing function and old routing function, the header of the data packet is modified to contain another field called Differentiator (DIFF). It is a Boolean value, zero represents that the packet is transmitted using old routing function, and one represents that the packet is followed by new routing function. The header of the data packet is shown below in Table 4.

Table 4. The header of the data packet.

A	B	C	D	C	E	C	C	F	DIFF

In the above Table 4, A indicates command; it tells whether the packet is a request or a response. B denotes the version number of routing information protocol. C symbolize unused bit and it usually set to zero. D is the address family identifier and it specifies the type of address being transmitted. E is the IP address of the entry. F denotes metric which generally represents hop count value. DIFF is our newly added field and is used to represent the type of routing function used, whether RF_{new} or RF_{old}. The overall deadlock free reconfiguration process is described below.

2.5. Executing Process

- The node that detects the topology change or failure of the link on the forwarding path does the following action immediately for initiating the reconfiguration process:
  1. The detected node becomes the RC and it will broadcast the HELLO messages to all the neighbor nodes of the faulty node.
  2. The message received nodes will respond the status to RC by NW-STATUS messages.
  3. The RC looks for changes in NW-STATUS and constructs the RF_{new}.
  4. RC makes a RT and it will send RT along with RF_{new} for synchronize RF_{old} and RF_{new}.
  5. RC first sends the above to the neighbors of the faulty node then update the same to all the nodes of the whole network.
- Each node has to do the following operations on the network:
  1. Continue RF_{old} until RT receives.
  2. Once RT receives with RF_{new}, multicast RT to all nodes and update to RF_{new}.
  3. Disable the casting of RT and follow RF_{new} completely.
- At each input of every node should perform the following on the network:
  1. Continue RF_{old} until RT receives.
  2. After receives RT node change from RF_{old} to RF_{new}.

3. Multicast RT to all neighbor nodes.

- At each output of every node should perform the following on the network:
  1. Trasmit the token from output of the node to input of the neighbor nodes.
  2. Pocket forwarding by RF_{new} routing method.
  3. Disable the casting of RT and follow RF_{new} completely.

3. Results

3.1. Simulation Model and Parameters

In this section, we examine the performance of our TB Robust Deadlock-Free Dynamic Reconfiguration Protocol with an extensive simulation study based upon the ns-2 network simulator. We compare our results with the Overlapping Static Reconfiguration protocol (OSR) Olav Lysne *et al.* [8]. We have used mesh and hybrid-ring topologies in our simulation which are depicted in Figure 3 and 10, respectively. The Distance Vector (DV) routing is used for establishing the shortest paths between the source and destination. The Exponential traffic is used with packet size 512 bytes and there are totally 2 flows. The link bandwidth and link delay is set as 5 Mb and 10 ms respectively.

3.2. Performance Metrics

In the simulation experiments, we have used a low-load scenario with rate 2 Mb and high-load scenario with rate 5Mb. We measure the following metrics. Packet Loss, Throughput (bytes) and Delay in seconds at various time intervals. The results are described in the next section.

3.2.1. Hybrid-Ring Topology

In this hybrid ring topology, simulation setup in figure 3, the link 4-10 is down from 2.5 to 3.5 seconds and the link 15-10 is down from 5.5 to 6.5 seconds.

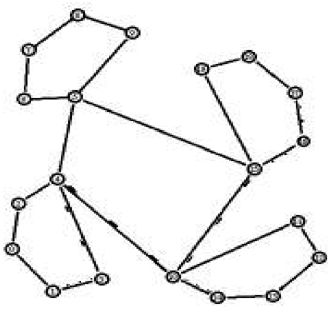

Figure 3. Simulation topology for hybrid ring.

3.2.1.1. High Load Scenario

The throughput, packet loss and delay are measured at various time intervals, for a high-load scenario with traffic rate as 5 Mb and outputs are shown in Figures 4, 5 and 6 respectively.

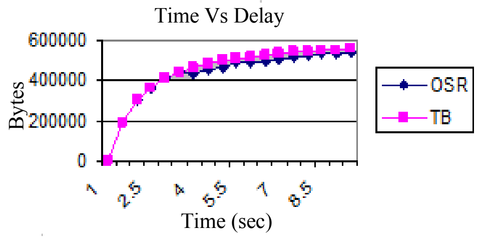

Figure 4. Throughput for high load scenario.

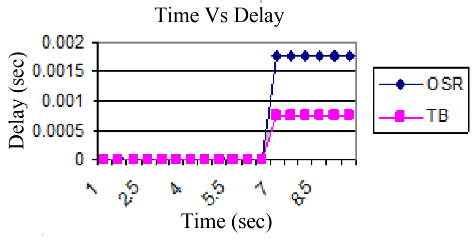

Figure 5. Delay for high load scenario.

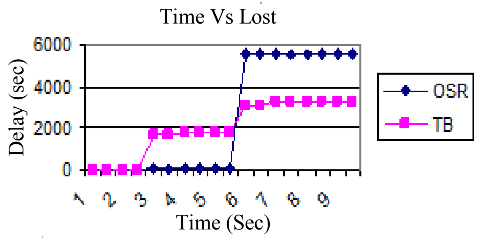

Figure 6. Packet lost for high load scenario.

3.2.1.2. Low Load Scenario

The throughput, packet loss and delay are measured at various time intervals, for a low-load scenario with traffic rate as 2 Mb.

Time vs Throughput are measured in the OSR and TB protocols about 10 sec for both high and low load scenarios. Our proposed TB protocol yields the higher throughput than the OSR. Likewise delay and packet loss also measured for 10 sec, but TB produces the minimum delay and minimum packet loss. Based on the output we plot the graphs shown in Figures 7, 8 and 9.

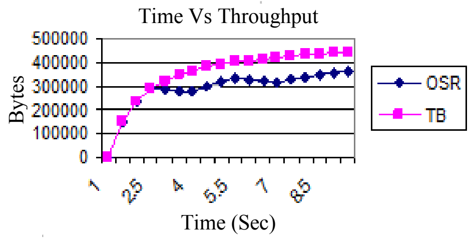

Figure 7. Throughput for low load scenario.

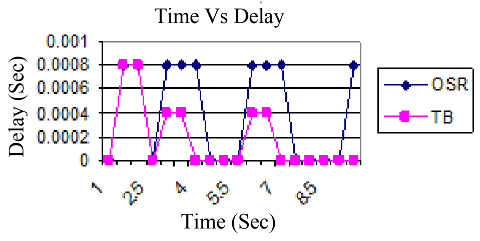

Figure 8. Delay for low load scenario.

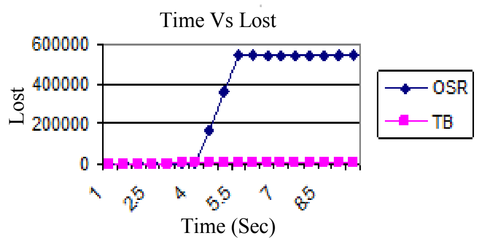

Figure 9. Packet lost for low load scenario.

3.2.2. Mesh Topology

In this Mesh topology, simulation setup in Figure 10, the link 9-13 is down from 2.5 to 3.5 seconds and the link 10-14 is down from 5.5 to 6.5 seconds.

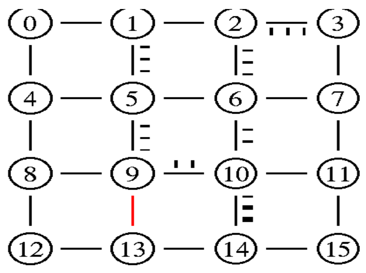

Figure 10. Mesh topology.

3.2.2.1. High Load Scenario

The throughput, packet loss and delay are measured at various time intervals, for a high-load scenario with traffic rate as 5Mb. Based on the output we plot the graphs shown in Figures 11, 12 and 13 respectively.

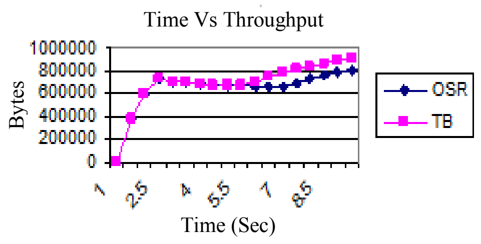

Figure 11. Throughput for high load scenario.

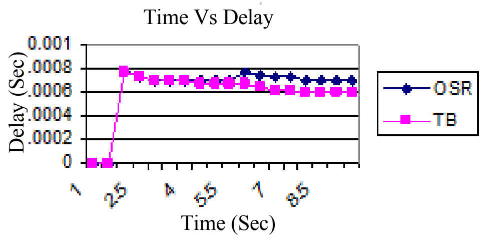

Figure 12. Delay for high load scenario.

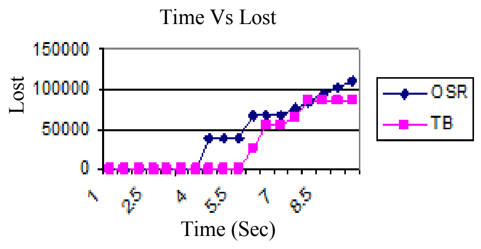

Figure 13. Packet lost for high load scenario.

3.2.2.2. Low Load Scenario

The throughput, packet loss and delay are measured at various time intervals, for a low-load scenario with traffic rate as 2 Mb and outputs are shown in Figures 14, 15 and 16.

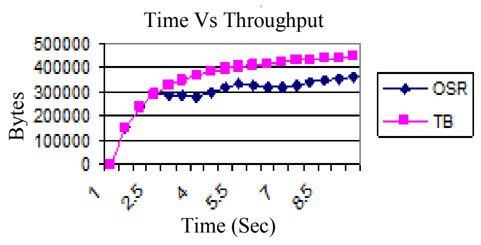

Figure 14. Throughput for low load scenario.

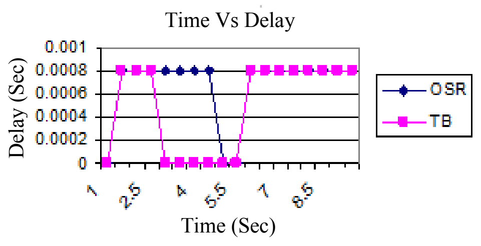

Figure 15. Delay for low load scenario.

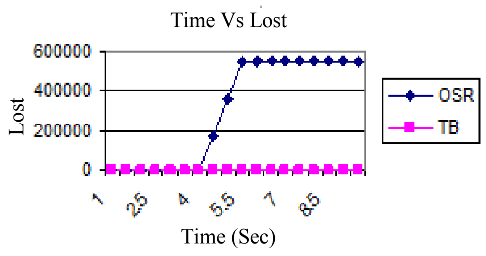

Figure 16. Packet lost for low load scenario.

4. Discussion and Conclusions

In both high and low load scenarios, time vs. throughput are measured up to first 10 sec for both OSR and TB protocols. In high load situation, up to first 7 sec both protocols are produced the same result. TB produced the higher throughput than the OSR after 7sec. Likewise delay and packet loss also measured for 10 sec, TB and OSR protocols produced the similar delay up to 6 sec, after that TB produced minimum delay and minimum packet loss In low load situation, up to 3 sec both protocols are produced the same result, after 3 sec the TB protocol produced the higher throughput than the OSR. Likewise delay and packet loss also measured for 10 sec, both protocols produced similar delay up to 3 sec then TB produces minimum delay and minimum packet loss. In this paper, we have proposed an autonomous, sovereign, independent, monarchic reconfigurable protocol with robustness. This protocol provides optimization to the reconfiguration induced deadlock problems. This paper should be the base for any new enhanced versions of the deadlock free reconfiguration situations. Based on our simulation results we have proved that our proposed reconfiguration protocol alleviates deadlocks completely during reconfiguration process and attains more throughputs with reduced packet loss and delay, when compared with the existing protocols.

References

[1] Duato J. and Yalamanchili S., *Interconnection Networks, An Engineering Approach*, Morgan Kaufmann Publishers, San Francisco, USA, 2005.

[2] Duato J., Lysne O., Pang R., and Pinkston T., "A Theory for Deadlock-free Dynamic Network Reconfiguration," *IEEE Transactions on Parallel and Distributed System*s, vol. 16, no. 5, pp. 412-427, 2005.

[3] Fernández L., García J., and Casado R., "On Deadlock Frequency During Dynamic Reconfiguration in NOWs," *in Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing*, London, UK, pp. 630-638, 2001.

[4] Jacksona C. and Hollisa S., "A Deadlock-Free Routing Algorithm for Dynamically Reconfigurable Networks-on-Chip," *ACM Journal Microprocessors & Microsystem*s, vol. 35, no. 2, pp. 139-151, 2011.

[5] Kim K. and Shin K., "Self-Reconfigurable Wireless Mesh Networks," *IEEE/ACM Transactions on Networking*, vol. 19, no. 2, pp. 393-404, 2011.

[6] Kinsy M., Cho M., Wen T., Suh E., Dijk M., and Devadas S., "Application-Aware Deadlock-Free Oblivious Routing," *in Proceedings of the 36th Annual International Symposium on Computer Architecture*, New York, USA, pp. 208-219, 2009.

[7] Kumar N., Vig R., and Bagai D., "Up-Down Routing Based Deadlock Free Dynamic Reconfiguration in High Speed Local Area Networks," *Global Journal of Computer Science and Technology*, vol. 11, no. 7, pp. 61-70, 2011.

[8] Lysne O., Montanana J., Flich J., Duato J., Pinkston T., and Skeie T., "An Efficient and Deadlock-Free Network Reconfiguration Protocol," *IEEE Transactions on Computers*, vol. 57, no. 6, pp. 762-780, 2008.

[9] Mohammad Hassan., "Collaborative and Integrated Network and Systems Management: Management Using Grid Technologies," *the International Arab Journal of Information Technology*, vol. 10, no. 5, pp. 503-510, 2013

[10] Montanana J., Flich J., and Duato J., "Epoch-Based Reconfiguration: Fast, Simple, and Effective Dynamic Network Reconfiguration," *in Proceedings of IEEE International Symposium on Parallel and Distributed Processing*, Miami, USA, pp. 1-12, 2008.

[11] Murali S., Meloni P., Angiolini F., Atienza D., Carta S., Benini L., De-Micheli G., and Raffo L., "Designing Message-Dependent Deadlock Free Networks on Chips for Application-Specific Systems on Chips," *in Proceedings of IEEE International Conference on Very Large Scale Integration*, Nice, France, pp. 158-163, 2006.

[12] NetworkSimulator., available at: http: //www.isi.edu/nsnam/ns, last visited 1995.

[13] Robles-Gomez A., Bermu´dez A., and Casado R., "A Deadlock-Free Dynamic Reconfiguration Scheme for Source Routing Networks Using Close Up*/Down* Graphs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 10, pp. 1641-1652, 2011.

[14] Sem-Jacobsen F. and Lysne O., "Topology Agnostic Dynamic Quick Reconfiguration for Large-Scale Interconnection Networks," *in Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, Ottawa, Canada, pp. 228-238, 2012.

[15] Taktak S., Desbarbieux J., and Encrenaz E., "A Tool for Automatic Detection of Deadlock in Wormhole Networks on Chip," *ACM Transactions on Design Automation of Electronic Systems*, vol. 13, no. 1, pp 1-22, 2008.

[16] Turner Y. and Tamir Y., "Deadlock-Free Connection-Based Adaptive Routing with Dynamic Virtual Circuits," *ACM Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 13-32, 2007.

[17] ValadBeigi M., Safaei F., and Pourshirazi B., "DBR: A Simple, Fast and Efficient Dynamic Network Reconfiguration Mechanism Based on Deadlock Recovery Scheme," *international Journal of VLSI Design & Communication Systems*, vol. 3, no. 5, pp. 13-26, 2012.

[18] Warnakaulasuriya S., "Characterization of Deadlocks in Interconnection Networks," *in Proceedings of the 11th International Parallel Processing Symposium*, Genva, Switzerland, pp. 80-86, 1999.

[19] Yoshinaga T. and Nishimura Y., "Performance Evaluation of Dynamic Network Reconfiguration Using Detour-UD Routing," *in Proceedings of IEEE Innovative Architecture for Future Generation High-Performance Processors and Systems*, Hawaii, USA, pp. 110-118, 2005.

[20] Zhou M. and Fanti M., *Deadlock Resolution in Computer-Integrated Systems*, CRC Press, USA, 2004.

[21] Ziade H., Ayoubi R., Velazco R., and Idriss T., "A New Fault Injection Approach to Study the Impact of Bitflips in the Configuration of SRAM-Based FPGAs," *the International Arab Journal of Information Technology*, vol. 8, no. 2, pp. 155-162, 2011.

Mohiadeen Abdul Kadhar got Bachelor degree in electronics and communication engineering discipline and Master degree in Digital Communication and Network Engineering by Madurai Kamaraj University in the year 1990 and 2002 respectively. Presently he is working as associate professor in National College of Engineering, Maruthakulam, Tirunelveli-627151, Tamilnadu, India. He is a Research Scholar of Anna University Chennai, Tirunelveli Region, and Tamilnadu, India. His current research area is deadlock-free network reconfiguration. He is a life member of ISTE, CSI and IEEE.