Congestion Free On-Chip Network Design Using An Agent-Based Management Method

Vivek Raj .K
Student
Department of ECE
SJCIT, Chickballapur-562101, Karnataka, India.

Shashi Raj .K
Assistant Professor
Department of ECE
DSCE, Bangalore-560078, Karnataka, India

Abstract

Congestion in on-chip networks may cause many drawbacks in multiprocessor systems including throughput reduction, increase in latency, and additional power consumption. Furthermore, conventional congestion control methods, employed for on-chip networks, cannot efficiently collect congestion information and distribute them over the on-chip network. In this paper, we present an on-chip network architecture that incorporates a novel agent-based management method to enhance the reliability and performance of network-based Chip Multi-Processor (CMP) and System-on-Chip (SoC) designs against traffic congestion on the network.
Keywords: on-chip network, reliability, permanent fault, routing algorithm, SOCs, CMP.

I. INTRODUCTION

As is predicted by the Moore's law, over a billion transistors could be integrated on a single chip in the near future. In these chips, hundreds of functional Intellectual Property (IP) blocks and a large amount of embedded memory could be placed together to form a Chip Multi-Processor (CMP) [1]. By increasing the number of IPs and memories in a single chip, the traditional bus-based architectures are not useful anymore and a new communication infrastructure is needed. Network-on-Chip (NoC) has been addressed as a solution for the communication requirement of CMPs [1]. The performance and efficiency of NoC largely depend on the underlying routing technique which decides the direction a packet should be sent.

Routing algorithms are used in NoCs in order to determine the path of a packet from a source to a destination. Routing algorithms are classified as deterministic and adaptive algorithms. Implementations of deterministic routing algorithms are simple but they are not able to balance the load across the links in a non-uniform or burst traffic [2][3]. The simplest deterministic routing method is dimension-order routing which is known as XY or YX algorithm. The dimension-order routing algorithms route packets by crossing dimensions in strictly increasing order, reducing to zero the offset in one direction before routing in the next one. Adaptive routing has been used in interconnection networks to improve network performance and to tolerate link or router failure. In adaptive routing algorithms, the path a packet travels from a source to a destination is determined by the network condition. So they can decrease the probability of routing packets through congested or faulty regions.

Adaptive routing algorithms can be decomposed into routing and selection functions. The routing function supplies a set of output channels based on the relative position of the current and destination nodes. The selection function chooses an output channel from the set of channels given by the routing function [4].

The selection function can be classified as either congestion- oblivious or congestion-aware scheme [5]. In congestion-oblivious algorithms, such as Zigzag [6] and random [7], routing decisions are independent of the congestion condition of the network. This policy may disrupt the load balance since the network status is not considered. Unlike congestion-oblivious methods, in congestion-aware algorithms, such as DyXY [8], HAMUM [9], BARP [10], GOAL [11] and GAL [12], the selection is usually performed using the congestion status of the network [4]. Most of congestion-aware algorithms consider local traffic condition for decision making in which each router analyses the congestion condition of its own and adjacent routers to choose an output channel. Routing decisions based on local congestion information may lead to an unbalanced distribution of traffic load. Therefore, routing algorithms based on local congestion information are efficient when the traffic is mostly local, i.e. cores communicate with those close to them [13].

In EDXY [14], the congestion information of a router is propagated to its row and column nodes via separate wires. Non-local information provided by these wires is used only when the packet is one row or column away from the destination. In the Neighbor-on-Path (NoP) approach [5], the locality decision is extended to 2-hop neighbors. So, the routing decision is performed based on the congestion information of the nodes within 1-hop and 2-hop of the current node. However, it suffers from the recursive nature of the routing algorithm, resulting in a high hardware overhead and increased router complexity. Moreover, it is not well scalable to N-hop neighbors since the router complexity increases non-linearly for larger N values (N>2) while the required chip area and power consumption increase significantly. In this paper, an Agent-based Network-on-Chip (A NoC) method is proposed to determine the congested areas and fault nodes in the network and route packets through the less congested fault free areas based on the local/non-local node information. The proposed agent-based management architecture consists of distributed agents in a hierarchical structure, which is especially suitable for large CMPs with tens or hundreds of processing elements. In this structure, the agents in each level of hierarchy have the same tasks to gather, manage and distribute the traffic and fault information.

This paper is organized as follows. In Section II, the preliminaries about the on-chip network and the Dynamic-XY routing algorithm are explained. In Section III, the proposed agent-based Network-on-Chip and congestion-aware selection method is discussed. The results are reported in Section IV while the summary and conclusion are given in the last section.

II. PRELIMINARIES

A. ON-CHIP NETWORK

In this paper, we consider a two-dimensional (2D) mesh topology with wormhole switching technique [4][15][16]. Networks with 2D- mesh topology offer massive parallelism and are more scalable than many other approaches in CMP interconnection [4].

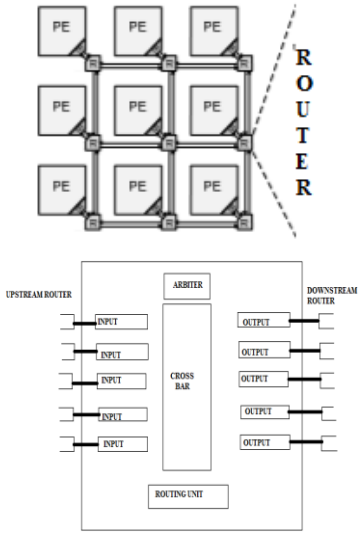

Fig. 1: 2D-Mesh NoC.

Besides, a mesh is well suited to a variety of applications including matrix computation, image processing and problems whose task graphs can be embedded naturally into the topology [17]. A 2D-mesh NoC based system is shown in Fig. 1. The NoC consists of Routers (R), Processing Elements (PE), and Network Interfaces (NI). PEs may be intellectual property (IP) blocks such as CPU or DSP core, video stream processor, high-bandwidth I/O unit or embedded memory. Each core is connected to the corresponding router port using the network interface. A packet is divided into smaller segments called flits which are routed successively until they reach their destination [15]. The first flit, called the header flit, holds the routing and control information and sets up the routing path for all subsequent flits associated with the packet. The header flit always goes first to allocate a path for the remaining flits (data). The remaining flits simply follow the path in a pipeline fashion.

B. THE DYXY ROUTING ALGORITHM

In the Dynamic XY (DyXY) routing algorithm, if there are multiple shortest paths available between the current and destination node, a packet can always be forwarded to either X or Y direction. Therefore, this routing algorithm needs a mechanism to guarantee deadlock avoidance. The DyXY method utilizes one virtual channel along the Y direction to guarantee deadlock freedom. In this method, the network is partitioned into increasing and decreasing sub networks as shown in Fig. 2. The increasing sub network covers the +X direction and half of the channels in the Y direction, while the decreasing sub network contains the rest of the channels. Therefore, if the destination node is to the right of the source, the packet will be routed through the increasing sub network. Similarly, if the destination node is to the left of the source, the packet will be routed through the decreasing sub network. In the case that the source and destination has the same Y address, the packet can be routed using both sub networks [15]. Since DyXY method has no restrictions to send a packet through the X or Y direction, we have chosen this adaptive method in this paper.

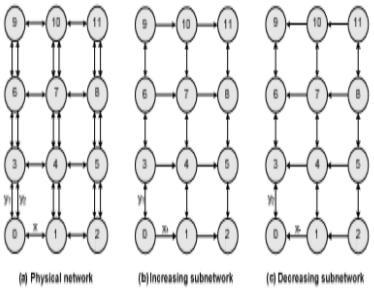

Fig. 2: (a) A 3×4 mesh physical network and the corresponding (b) increasing and (c) decreasing sub networks.

III. AGENT-BASED MANAGEMENT METHOD

To enhance the performance of a on-chip network with a large number of components, a scalable management method can be beneficial. Thus, we propose a management method that is agent-based and hierarchical to be more profitable for scalable on-chip networks.

A. There are two types of agents in the proposed management structure:

Cell agent: Each node or cell includes an agent called the cell agent which collects, manages and distributes the traffic information related to the components of its node. In addition, it updates the LFR and RFR.
Cluster agent: Each cluster that includes a number of nodes is controlled by a cluster agent. A cluster agent configures the cell agents inside the cluster by sending the new information which is obtained from the other cell agents inside the cluster or other cluster agents.

The incorporated agent hierarchy is shown in Fig. 3.1. This agent hierarchy differs from that of proposed in the previous works. This is due to the fact that in the proposed structure, for faster reconfiguration the cell agents communicate with their neighbor cell agents even if they are situated inside different clusters. This is a real case because in general, in a CMP, a task may require more than a cluster to be run. On the other hand, the clusters running a common task are not necessarily neighbor clusters. However, the routers should be aware about

Their neighbors to select the best path for sending the packets to their destinations and for faster awareness their cell agents should exchange the required traffic information.

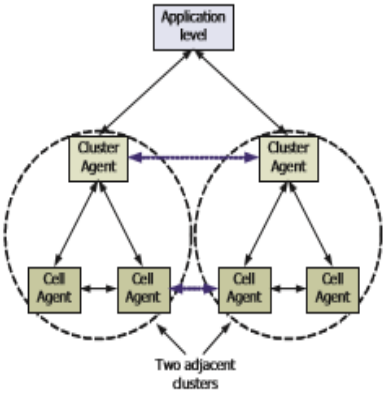

Fig. 3.1: Hierarchical agents in two neighbor clusters

B. Interconnections for Hierarchical Agents

A small 3×3 network with an agent in each node is shown in Fig. 3.2. In this figure, R, NI and PE correspond to the router, network interface and core or processing element, respectively. In addition, the agents can be cell or cluster agents but the number of cluster agents is much less than the number of cell agents. For example, in a regular mesh network each 3×3 sub- network can be a cluster with a cluster agent in the center. The proposed agent-based management structure uses two types of communications: a physically separate network for only peer to peer communication between the cell agents, and the baseline data network for control packet communication with a higher priority compared to data packets. The former is shown in Fig. 3.2 between the agents and the latter is the network connecting the NoC routers. The control packets including the reconfiguration and traffic information are exchanged between the cell agents and the cluster agents in addition to the communications between the cluster agents.

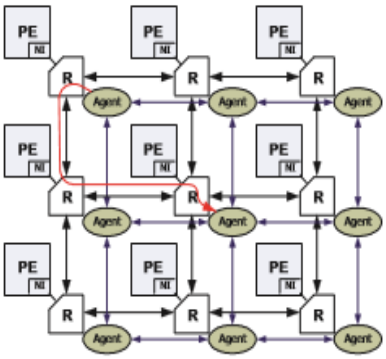

Fig. 3.2: A 3×3 NoC with agents and interconnections

C. Agent Tasks

In most previous works it is assumed that for the routing process the NoC routers are aware about the traffic occurred in the neighbor nodes in addition to their own components and interconnection links. However, in reality there should be a mechanism to inform the routers about congestion in the network. In the proposed method, this awareness is distributed as traffic information by hierarchical agents and their interconnections.

1) *Cell agent*

In the agent hierarchy shown in Fig. 3.1, all the traffic information needed for the neighbor node can be sent to the appropriate cluster agent, and then the cluster agent can distribute and send it to the appropriate cell agents. However, to minimize the impact of delay on the network performance especially just after traffic occurrence, it is better that the cell agents themselves distribute the congestion information to their neighbors because this approach is faster. This way, the congestion information essential for the routing process is distributed at the lowest level of management hierarchy through the dedicated network between the cell agents.

2) *Cluster agent*

A cluster agent informs the higher level about the congestion occurred inside the cluster and receives reconfiguration commands from the higher level about remapping of the tasks on the nodes or task migration. However, it is clear that a cluster agent itself is informed by the cell agents. In addition, for a more effective routing process, a cluster agent should be informed about traffic inside a node, because it should be able to send different types of traffic information to other cluster agents in addition to informing other nodes inside the cluster. A cluster agent can be placed in the processing element of a node as a software agent which is implemented entirely in software (SW). It can also be a hardware (HW) component similar to the proposed cell agent or can be implemented in HW/SW co-design manner. In our method a cluster agent is implemented in hardware and it includes the cell agent of the node in which it is located. For packet-based communication and to use existing resources in a node, the cell and cluster agents exploit the local port to send and receive packets. Thus, some extra logic for multiplexing is required to separate data and control packets and then to direct packets to the agents or the local core (PE) or to accept packets from them.

IV. EXPERIMENTAL RESULTS

To evaluate the efficiency of the proposed ANoC architecture, two other on-chip networks are also implemented. These on-chip networks, named NoC1 and NoC2, are formed by utilizing Dynamic XY (DyXY) and Neighbors-on-Path (NoP) approaches. A 2D-NoC simulator is implemented with VHDL to model all major components of the NoC and simulations are carried out to determine the latency-throughput characteristics of each network. For all the routers, the data width is set to 32 bits. Each input virtual channel has a buffer (FIFO) with the size of six flits. The congestion threshold value is set to four meaning that the congestion condition is considered when four out of six buffer slots are occupied. In simulations, the latency is measured by averaging the latency of the packets when each local core generates 3000 packets. As a performance metric, we use latency defined as the number of cycles between the initiation of a message transmission issued by a Processing Element (PE) and the time when the message is completely delivered to the destination PE. The request rate is defined as the ratio of the successful message injections into the network interface over the total number of injection attempts.

Each processing element (PE) generates data packets and sends them to another PE using a uniform distribution [20][21][22]. The mesh size is considered to be 3X3. As we observed from the Fig 4, ANoC leads to the lowest latency. This was expected due to the distribution of traffic over less congested areas. Because of the ANoC structure, each router can observe the congestion information of not only the neighboring routers, but also the routers residing beyond the neighboring routers. Therefore, routers in ANoC distribute traffic more efficient than the other two networks because the routing unit can determine the non-congested areas using both local and non-local congestion information.

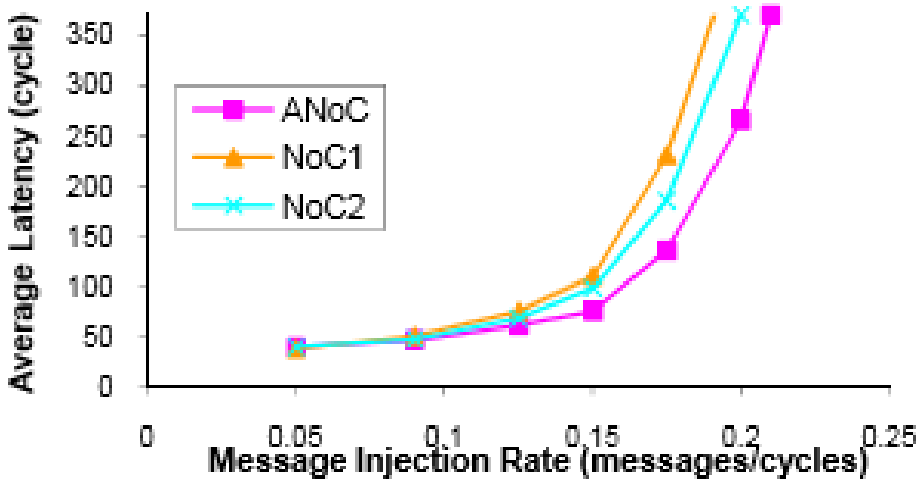

Fig. 4: latency for different networks

Table 1: Hardware Implementation Details.

Networks	Area (mm^2)	Power (W)
NoC1	6.670	2.34
NoC2	6.843	2.61
ANoC	6.771	2.44

From the above table Comparing the area cost of the proposed platform with NoC1 and NoC2 indicates that the NoC2 platform consumes more power and has a higher area overhead and lower performance gain. On the other side, the ANoC platform imposes only 1% hardware overhead and 4% higher power consumption compared to NoC1. The performance of the ANoC platform is around 22% higher than that of NoC1.

V. CONCLUSION

In this paper, we presented a Hierarchical agent-based network-on-chip to determine the congested areas. On top of that, an efficient selection method was presented for adaptive routing algorithms to choose the appropriate channel which avoids packets to be routed in congested areas. The results revealed that the proposed agent-based strategy improve the performance significantly with minimal overhead in terms of area occupation and power consumption.

REFERENCE

[1] O. Cesariow, L. Lyonnard, G. Nicolescu, et al., "Multiprocessor SoC platforms: a component-based design approach", Proc. Int. Conf. IEEE Design and Test of Computers, pp. 52–63, France, 2002.
[2] D. Bertsekas and R. Gallager, Data Networks, Prentice Hall, 1992.
[3] W. J. Dally and B. Towles, "Principles and Practices of Interconnection Networks", Morgan Kaufmann, 2004. [4]
[4] J. Duato, S. Yalamanchili, L. Ni, "Interconnection Networks: An Engineering Approach", Morgan Kaufmann, 2002.
[5] G. Ascia, V. Catania, M. Palesi, "Implementation and Analysis of a New Selection Strategy for Adaptive Routing in Networks-on-Chip", IEEE Transaction on Computers, v.57, I.6, pp. 809 – 820, 2008.
[6] H. G. Badr, S. Podar, "An optimal shortest-path routing policy for network computers with regular mesh-connected topologies", v.38, I.10, pp.1362-1371, 1989.
[7] W. Feng and K. G. Shin, "Impact of Selection Functions on Routing Algorithm Performance in Multicomputer Networks", In Int. Conf. on Supercomputing, pp. 132–139, 1997.
[8] M. Li, Q. Zeng, W. Jone, "DyXY - a proximity congestion-aware deadlock-free dynamic routing method for network on chip", Proc. DAC, pp. 849-852, 2006.
[9] M. Daneshtalab, M. Ebrahimi, T. C. Xu, P. Liljeberg, and H. Tenhunen, "A Generic Adaptive path-based routing method for MPSoCs," Journal of Systems Architecture (JSA-elsevier), Vol. 57, No. 1, pp. 109-120, 2011.
[10] P. Lotfi-Kamran, M. Daneshtalab, Z. Navabi, and C. Lucas, "BARP- A Dynamic Routing Protocol for Balanced Distribution of Traffic in NoCs to Avoid Congestion," in Proceedings of 11th ACM/IEEE Design, Automation, and Test in Europe Conference (DATE), pp. 1408-1413, Mar 2008, Germany.
[11] A. Singh, W. J. Dally, A. K. Gupta, et al., "GOAL: A Load-Balanced Adaptive Routing Algorithm for Torus Networks", In Int. Symp. on Computer Architecture, pp. 194–205, 2003.
[12] A. Singh, W. J. Dally, B. Towles, et al., "Globally Adaptive Load- Balanced Routing on Tori", IEEE Computer Architecture Letters, v.3, I.1, pp.2-6, 2004.
[13] L.P. Tedesco, T. Rosa, F. Clermidy, et al., "Implementation and evaluation of a congestion aware routing algorithm for networks-on- chip", Proc. of the 23rd symposium on Integrated circuits and system design.
[14] P. Lotfi-Kamran, A. M. Rahmani, M. Daneshtalab, et al., "EDXY - A low cost congestion-aware routing algorithm for network-on-chips", Journal of Systems Architecture, v.56, I.7, 2010.
[15] L. M. Ni and P. K. McKinley, "A survey of wormhole routing techniques in direct networks", IEEE Tran. On Computers, v.26, pp. 62-76, Feb, 1993.
[16] M. Daneshtalab, M. Ebrahimi, P. Liljeberg, J. Plosila, and H. Tenhunen, "A Low-Latency and Memory-Efficient On-Chip Network," in Proceedings of 4th ACM/IEEE International Symposium on Networks- on-Chip (NOCS), pp. 99-106, May 2010, France. 288
[17] N. E. Jerger, L. S. Peh, and M. H. Lipasti, " Virtual Circuit Tree Multicasting: A Case for On-Chip Hardware Multicast Support", In Proc. of the Int. Symp. on Computer Architecture (ISCA), Beijing China, June 2008.
[18] G. M. Chiu, "The odd-even turn model for adaptive routing", IEEE Trans. on Parallel and Distributed Systems, 11:729 – 738, July 2000.
[19] A. Kahng, B. Li, L. Peh and K. Samadi, "ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration", In Proc. of DATE, France, April 2009.
[20] R. V. Boppana, S. Chalasani, "A Comparison of Adaptive Wormhole Routing Algorithms", Proc. Int. Symp. Computer Architecture, pp. 351– 360, May 1993.
[21] M. L Flugham, L. Snyder, "Performance of Chaos and Oblivious Routers under Non-Uniform Traffic", Technical Report UW-CSE-93-06-01, July 1993.
[22] C. J. Glass, L. M. Ni, "The Turn Model for Adaptive Routing", Proc. Symp. Computer Architecture, pp. 278-287, May 1992.