

On Source Routing for Mesh Topology Network on Chip

Saad Mubeen, Shashi Kumar

*Department of Electronics and Computer Engineering
School of Engineering, Jönköping University, Sweden
{mems07musa, Shashi.Kumar}@jth.hj.se*

Abstract— Deterministic and adaptive distributed routing algorithms have been advocated in all the current NoC architectural proposals. In this paper we make a case for the use of source routing for NoCs, especially for regular topologies like mesh. The advantages of source routing include in-order packet delivery; faster and simpler router design; and possibility of mixing non-minimal paths in a mainly minimal routing. We propose a method to compute paths for various communications in such a way that traffic congestion is avoided while ensuring deadlock free routing. We also propose an efficient scheme to encode the paths.

Keywords— Network on Chip, Distributed Routing, Source Routing, Routing Algorithms, Performance Analysis

I. INTRODUCTION

Network on Chip (NoC) is being considered as the most suitable candidate for implementing interconnections in core based system on chip (SoC) design [1]. A NoC architecture provides packet switched communication among cores in the system. Topology and routing algorithm are two important features which distinguish various NoC proposals. Communication performance of a NoC depends heavily on the routing algorithm used. Routing methods can be classified into two types, namely, source routing and distributed routing. In source routing the information about the whole path from the source to the destination is pre-computed and provided in packet header. In distributed routing, the header contains destination address only and the path is computed dynamically by participation of routers on the path [2][3][4].

A very large number of routing algorithms have been proposed in literature [5][6][7]. All the proposals so far fall under distributed routing type. Source routing has not been so far considered for NoCs, due to its apparent large overhead to store path information in the header. Since the paths in source routing are pre-computed offline, therefore source routing can provide no or limited path adaptivity in the case of faults and traffic congestion. In spite of these disadvantages, we feel source routing has many advantages over distributed routing.

A. Advantages of Source Routing for Mesh NoC

Source routing is not perhaps suitable for dynamic networks where network size and topology are changing. But in a NoC with fixed size and regular topology like mesh, the path information can be efficiently encoded with small number of bits. It can be easily shown that two bits are sufficient to encode every hop in the path. Since the packet

entering a router contains the pre-computed decision about the output port, the router design is significantly simplified. Since NoCs used in embedded systems are expected to be application specific, we can have a good profile of the communication traffic in the network [5]. This allows us to analyse the traffic and compute, in offline, efficient paths giving the desired performance characteristics, like uniform link load distribution. Source routing also provides possibility of mixing minimal and non-minimal paths for this purpose. Since only a few bits of the header are used in every router, its design is not only simple but is also independent of the network size. We feel that the above advantages more than compensate its disadvantages.

B. Paper Layout

In the next section, we list and discuss the issues and tasks required to implement source routing in a mesh topology NoC. In section III, we compare the overhead and link bandwidth utilization of source routing verses distributed routing. In section IV, a method to compute paths for source routing is described by utilizing available distributed routing algorithms. In section V, we propose a method to iteratively improve the path computed in section IV. Section VI concludes by listing our plan for future work in this area.

II. STEPS FOR DESIGNING SOURCE ROUTING

Design of complete source routing scheme involves a number of steps including computation of path for each communication, encoding of the computed path and finally design of a router. These steps are briefly described in the following sub-sections.

A. Path Computation

Path computation refers to finding a complete path or route from source to destination resource. In general case, a path should be computed for each pair of resources in the network. In an application specific case, paths must be computed for only those pairs of resources which communicate. The computed paths must avoid any possibility of deadlock. Apart from that, computed paths should provide small link delay and avoid link congestion. For good performance, paths should also be computed in such a way that the load is uniformly distributed in the network.

A large number of deadlock free routing algorithms exist for mesh NoC which can be used for path computation of source routing. XY is a deterministic deadlock free routing

algorithm for mesh topology. Similarly, turn model based partially adaptive wormhole routing algorithms [6] can also be used for path computation in source routing. Some of these algorithms are West-First, North-Last, and Negative-First etc. These algorithms restrict at least half of the source-destination communications to one minimal path, while rest of the pairs can communicate with full adaptivity. Odd-Even is also a partially adaptive deadlock free routing algorithm based on restrictions of taking turns at some locations. It provides more even degree of adaptiveness comparatively [6].

B. Path Encoding

Once path from source to destination is computed, it should be encoded in such a way that the overhead of routing information is minimized. Moreover, path encoding should be such that it is easy to decode in the routers on the way to the destination. In order to minimize the route information overhead in the head flit, we have used 2-bit clockwise port address encoding scheme. As we are using minimal routing, so a flit cannot be forwarded to the same channel from where it is coming. Therefore, a flit can only be forwarded to any of four out of five output ports in a router. Thus address of each output port of a router is encoded with two bits. Accordingly, "00", "01", "10" and "11" represent the addressees of one, two, three and four steps away output ports respectively in the clockwise direction with respect to the input port where flit was received.

C. Router Design

Router design for source routing will be much simpler than the router design to handle a distributed routing algorithm. It does not need to compute a routing function to select the output port for an incoming packet. This pre-decided information is available in the header flit. It decodes the 2-bit clockwise port address and forwards the head flit to the desired port. Other flits of the same packet follow the head flit if wormhole switching is used. But this router still needs to implement the other functions like packet buffering, credit based flow control and arbitration to resolve port conflicts when two or more packets contend for the same output port. The simplicity of the source router makes it relatively smaller and faster. Schematic diagram of a NoC router implementing source routing is depicted in Fig. 1.

III. COMPARISON BETWEEN DISTRIBUTED ROUTING AND SOURCE ROUTING

A. Overhead

As complete route information should be stored along with the payload in case of source routing, large underutilization is expected. But analytical comparison between source and distributed routing eliminates this fear of underutilization. Fig. 2 shows a graph of the number of bits required for routing against different size of square mesh NoC for both source and distributed routing. It is clear that with source routing, number of routing bits increases proportional to the diameter of the network (that is square-root of the size of the network). But routing bits required with a distributed routing increase logarithmically with the network size. The gap between the

two graphs keeps on increasing with the increase in size of NoC and thus apparently making source routing look unusable in practice. For both types of routing, the bits required to route data in a $N \times N$ mesh NoC is given by the following formulae. Number of routing bits in Source Routing = $2(2N-1)$

Number of routing bits in Distributed Routing = $2 * \lceil \log_2(N) \rceil$ But if the overhead is measured in terms of extra flits to be communicated, the difference is zero or very small. We analyse this in the next sub-section.

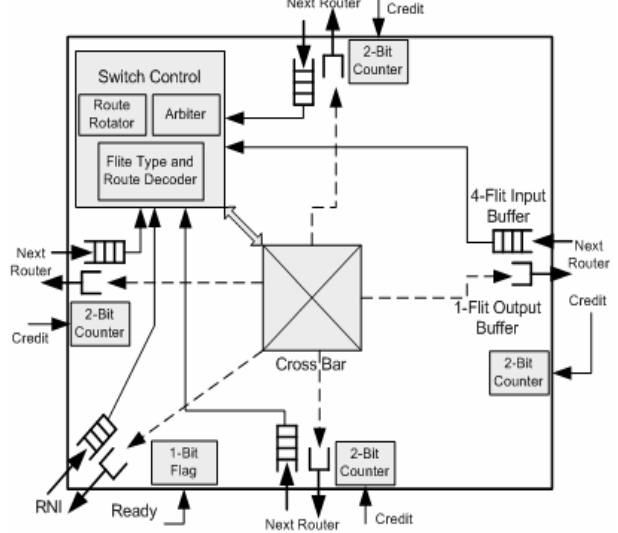


Fig. 1 Schematic diagram of a NoC Router Implementing Source routing

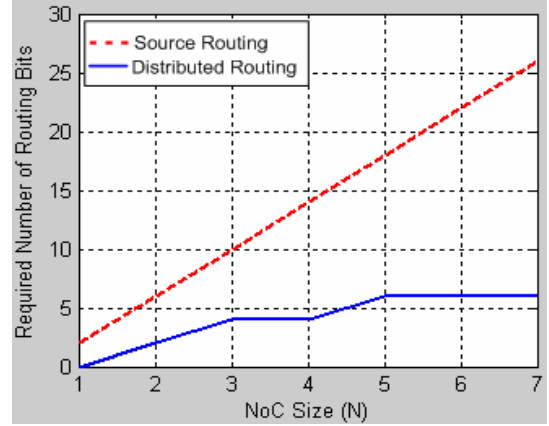


Fig. 2 Routing bits requirement for $N \times N$ sized NoC for Source and Distributed routings

B. Bandwidth Utilization

We get a different view of the comparison between source and distributed routing in NoC when bandwidth utilization is considered. Utilization is defined as the ratio of the payload in bytes to be transmitted and the actual number of bytes to be sent carrying this payload. In Fig. 3, utilization is plotted against the payload in bytes to be routed for both source and distributed routing for a 7×7 NoC. It can be seen that with less number of bytes in payload, both source and distributed routing show very less utilization and relatively bigger gap

between the two. By increasing the number of bytes in payload, utilization of both routings increases while the gap between them becoming smaller. There exists a correlated trend of increasing utilization with increasing payload in both the routings. For larger payloads, utilization of both routings becomes very close to each and near to 1 thus making source routing as good as distributed routing. This quality of source routing was not evident from the analysis of Fig. 2.

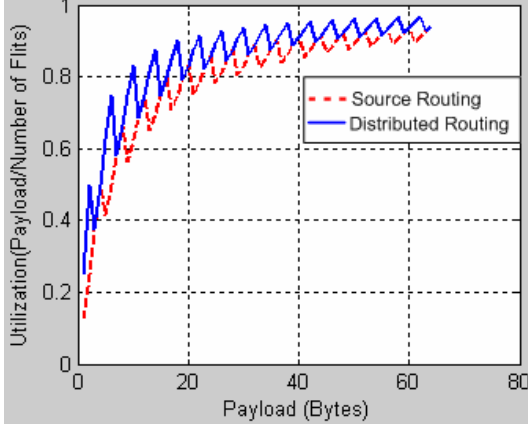


Fig. 3 Bandwidth utilization for Source and Distributed routing in NoC

C. Router Delay

Router logic delay " T_{RL} " is a very important factor to be considered when source and distributed routings are compared analytically. T_{RL} is the time required to transfer a flit from the input buffer to the output buffer of a router. It depends upon the time required to decode flit type and routing information, selection of output port and switching activity inside the router. As opposed to source routing, routing tables are consulted in each router in distributed routing and which creates an extra delay of at least one router clock. Routing information is already present in the **header** flit in case of source routing, so comparatively routing logic is simpler, faster and smaller.

IV. PATH COMPUTATION FOR SOURCE ROUTING

A. Using Available Routing Algorithms

One can use any existing distributed routing algorithm for computing the path for source routing. We have evaluated both deterministic and partially adaptive routing algorithms for this purpose. The algorithms were selected on the basis of certain routing properties such as deadlock freedom and adaptivity. Adaptivity of a routing algorithm can be used for uniform distribution of the link load in the network. Accordingly, we chose six routing algorithms namely XY, Odd-Even, West First, Negative First, East First and North Last for evaluation purpose.

Computed routing paths will be stored in a table in the core or in the Network Interface. Since only one path will be stored for each communicating pair, we exploit adaptivity by using equal probability to the selection of available routes to the destination. We use only minimal routing, so adaptivity of routing algorithms can provide a maximum choice of two

different paths in routers on the path. Thus we set 50% probability for the selection of each of the available output port of the router.

B. Comparison of Routing Algorithms for Different Traffic: Results

We choose different types of traffic in order to demonstrate that the above mentioned routing algorithms show different performance for different traffic. Link load distribution is chosen as a performance measure. A routing algorithm, which uniformly distributes the traffic, shows better performance. A number of experiments were performed using a 7x7 NoC with mesh topology. In order to get more realistic results, link bandwidth volume of each communication was randomly selected between 1 and 10 units. Similarly, communication density is also randomly selected such that each core can communicate with 2 to 5 other cores. Local biased traffic is used in the application specific case when we consider east, south and west dominated traffic.

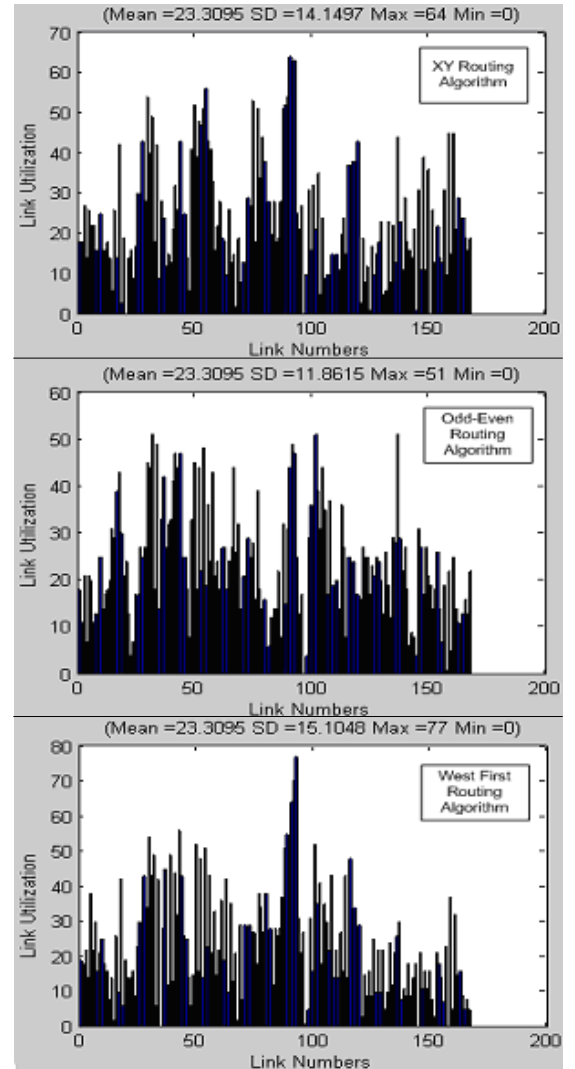


Fig. 4 Link load distribution in a 7x7 NoC for Hot Spot traffic using XY, Odd Even and West First routing algorithms respectively

Due to lack of space, only one result is shown in Fig. 4, demonstrating the link load distribution for hot spot traffic when XY, Odd-Even and West First routing algorithms are used. In this case, Odd-Even routing algorithm shows best performance among the six routing algorithms considered in this paper, with least maximum link bandwidth, mean and variance of the link load distribution. This result also matches with the results shown in [6].

C. Choosing Routing Algorithm for Application Specific Traffic

The experiments, which we performed, show that XY routing algorithm gives best performance for random traffic while Odd-Even algorithm outperforms other routing algorithms in the case of hot spot traffic. Similarly, West First, North Last and East First show best performance when east, south and west dominated traffics are used respectively. Hence, we propose a solution, shown in Fig. 5, for selecting a routing algorithm for the type of traffic in the case of application specific communication. From the communication graph of application, first the type of traffic should be analysed. Accordingly, West First, North Last, East first and Odd-Even algorithms should be selected for east, south, and west dominated and hot spot traffic respectively. Similarly, for any other traffic including random traffic, XY routing algorithm should be considered. Further more, Odd-Even algorithm should be preferred if a traffic is hot spot as well as of any other type. After selection of routing algorithm, paths form source to destination for all communicating cores can be computed for source routing.

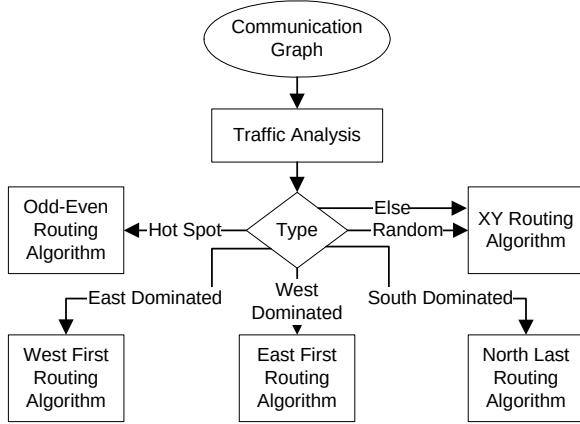


Fig. 5 Algorithm for traffic analysis and selection of routing algorithm for application specific traffic

V. PATH IMPROVEMENT ALGORITHM

The path computed in section IV may not be the best in terms of link load distribution. We propose an iterative algorithm to improve the computed path for application specific source routing. After analysing and selecting routing algorithm in section IV, initial paths are computed for all communications. In the next step link load distribution i.e. link load variance is estimated. If it is acceptably small then the paths which were computed in the first step are used for source routing. On the other hand, if link load distribution is

not acceptable then the most congested link is identified. One communication using this link is rerouted on an alternative path using the adaptivity of the routing algorithm. The above process is iteratively repeated until link load distribution becomes acceptable or it does not show any further improvement.

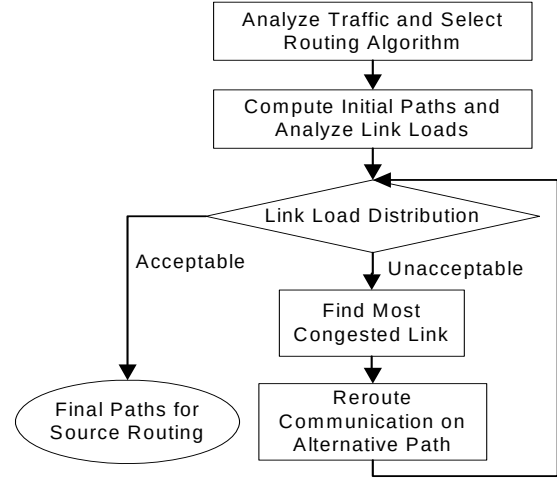


Fig. 6 Path improvement algorithm

VI. CONCLUSION AND FUTURE WORK

In this paper we have made a case for the use of source routing in mesh topology NoC. Because of the small fixed sizes of practical NoCs, the overheads and disadvantages are easily compensated by a large number of advantages, including lower router cost and higher communication speed. We have proposed an efficient method to compute and encode paths for source routing. Currently we are developing a simulator for more detailed analysis of our source routing approach. We also plan to develop a router design to support our source routing ideas.

REFERENCES

- [1] Axel Jantsch and Hannu Tenhunen. *Networks on Chip*. Kluwer Academic Publishers, 2003. ISBN: 1-4020-7392-5.
- [2] William James Dally and Brian Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers an Imprint of Elsevier Inc. 2004. ISBN: 0-12-200751-4.
- [3] J. Flich, P. Lopez, M. P. Malumbres and J. Duato. *Improving the Performance of Regular Networks with Source Routing*. Proceeding of the IEEE International Conference on Parallel Processing, 21-24 Aug. 2000. Page(s) 353-361.
- [4] Y. Aydogan, C.B. Stunkel, C. Aykanat, B. Abali. *Adaptive source routing in multistage interconnection networks*. Proceedings of IPPS '96, the 10th International Parallel Processing Symposium. 15-19 April 1996. Page(s): 258 – 267.
- [5] Maurizio Palesi, Rickard Holmsmark, Shashi Kumar and Vincenzo Catania. *Application Specific Routing Algorithms for Networks on Chip*. IEEE Transactions on Parallel and Distributed Systems, Vol. 20, No. 3, March 2009.
- [6] Ge-Ming Chiu. *The Odd-Even Turn Model for Adaptive Routing*. IEEE Transactions on Parallel and Distributed Systems, Vol. 11, No. 7, July 2000.
- [7] A. Patooghy, H. Sarbazi-Azad. *Performance comparison of partially adaptive routing algorithms*. 20th International Conference on Advanced Information Networking and Applications, 2006. AINA 2006. Volume 2, 18-20 April 2006.