DyXY - A Proximity Congestion-Aware Deadlock-Free Dynamic Routing Method for Network on Chip

Ming Li, Qing-An Zeng, Wen-Ben Jone
Dept. of Electrical & Computer Engineering and Computer Science, University of Cincinnati
Cincinnati, OH, USA
lim0@ececs.uc.edu, qzeng@ececs.uc.edu, wjone@ececs.uc.edu

ABSTRACT

A novel routing algorithm, namely dynamic XY (DyXY) routing, is proposed for NoCs to provide adaptive routing and ensure deadlock-free and livelock-free routing at the same time. A new router architecture is developed to support the routing algorithm. Analytical models based on queuing theory are developed for DyXY routing for a two-dimensional mesh NoC architecture, and analytical results match very well with the simulation results. It is observed that DyXY routing can achieve better performance compared with static XY routing and odd-even routing.

Categories and Subject Descriptors: B.4.3 [Hardware]: Input/Output and Data Communication - Interconnections.

General Terms: Algorithms, Performance, Design.

Keywords: Network-on-Chip, Packet Routing, Queuing Theory.

1. INTRODUCTION

A layered architecture called Network on Chip (NoC) [1] [2] has been proposed for global communication in complex SoCs to meet the performance requirements. In NoCs, routing algorithms are used to determine the path of a packet traversing from the source to the destination. Routing algorithms can be generally classified as *deterministic* routing and *adaptive* routing. The former benefits from its simplicity in router design; however, it is likely to suffer from throughput degradation when the packet injection rate increases. The later determines routing paths based on the congestion conditions in the network. The adaptiveness reduces the chance for packets to enter hot-spots or faulty components, and hence reduces the blocking probability of packets. Adaptiveness is an important factors for message routing, and the other important requirement of a routing algorithm is the freedom from *deadlock* and *livelock*.

Many routing algorithms dealing with networks with the mesh architecture have been proposed for deadlock-free and adaptiveness recently. In [3]-[5], *virtual channels* are introduced to assist the design of nonadaptive and adaptive routing algorithms for a variety of network architectures. In [6]-[9], routing algorithms that require no virtual channels have been proposed for networks with the mesh architecture. A static XY routing algorithm for two-dimensional meshes has been presented in [6]. With static XY routing, a packet first traverses along the x dimension and then along the y dimension. This algorithm is deadlock-free but provides no adaptiveness. The work in [7] proposed another algorithm called the *turn* model, which is a partially adaptive routing algorithm without virtual channels. In [8], a routing algorithm called *odd-even turn* was proposed based on the turn model. It restricts some locations where turns can be taken so that deadlock can be avoided. A routing scheme called DyAD was proposed in [9]. This algorithm is the combination of a deterministic routing algorithm called *oe-fix*, and an adaptive routing algorithm called odd-even as proposed in [8]. The router can switch between these two routing modes based on the network's congestion conditions.

In this paper, we propose a novel routing algorithm, namely dynamic XY (DyXY) routing, which provides adaptive routing based on congestion conditions in the proximity, and ensures deadlock-free and livelock-free routing at the same time. The adaptiveness lies in making routing decisions by monitoring congestion status in the proximity, and the deadlock-free and livelock-free features are incorporated by limiting a packet to traverse the network only following one of the shortest paths between the source and the destination. The DyXY routing method can be supported by a router architecture efficiently. Analytical models based on queuing theory are developed for both XY routing (called static XY in the following part of this paper) and DyXY routing to evaluate their performance for a two-dimensional mesh NoC architecture. Extensive simulation is done to validate the analytical models, and it is observed that the simulation results match very well with the analytical results. To further evaluate the performance of DyXY, we compare it with both static XY routing and odd-even routing under different traffic patterns, and it is shown that DyXY routing can achieve the best performance.

2. DYXY ROUTING AND ROUTER ARCHITECTURE

With the DyXY routing algorithm, each packet only travels along a shortest path between the source and the destination (this guarantees the deadlock-free feature of the routing algorithm). If there are multiple shortest paths available, the routers will help the packet to choose one of them based on the congestion condition of the network. The detailed routing algorithm can be summarized as follows:

- Read the destination of an incoming packet.
- Compare addresses of the destination and the current router.
  - If the destination is the local core of the current router, send the packet to the local core;
  - Else
    - If the destination has the same x (or y) address as the current router, send the packet to the neighboring router on the y-axis (or x-axis) towards the destination;
    - Else, check the stress values of current router's neighbors towards the destination, and send the packet to the neighbor with the smallest stress value.

The stress value is a parameter representing the congestion condition of a router. Here, we use the 'instant queue length' of each router (i.e., the number of occupied cells in all input buffers) as the stress value, since it achieves the best results among all kinds of average or flow-control types we have attempted. Each router stores instant stress values for all neighbors, and each stress value is updated based on an event-driven mechanism.

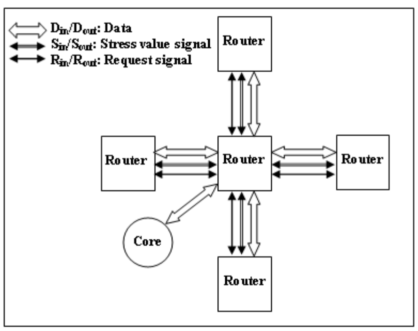

Figure 1: NoC interconnections under DyXY routing.

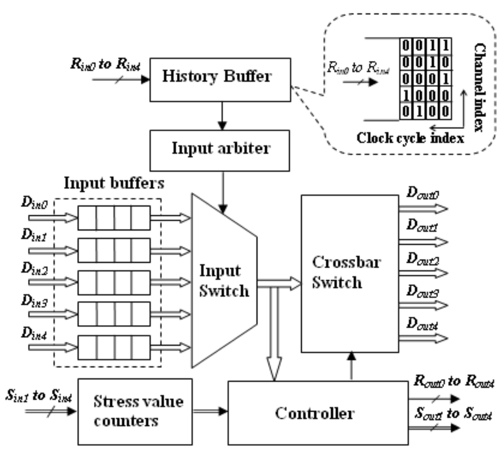

Figure 2: Router Architecture for DyXY routing.

The NoC system interconnection under DyXY routing is shown in Figure 1, and the router architecture is shown in Figure 2. Each router contains a set of first-in first-out (FIFO) input buffers, an input arbiter, a history buffer, a crossbar switch circuit, a controller, and four stress value counters. The size of each input buffer is a design parameter. In Figure 2, D_{in0}/D_{out0} to D_{in4}/D_{out4} represent the data lines between a router and its local core, right router, up router, left router and down router, respectively. R_{in0}/R_{out0} to R_{in4}/R_{out4} represent the request signal lines between a router to its local core and all neighbor routers. S_{in1}/S_{out1} to S_{in4}/S_{out4} represent the input/output signal lines to update stress value between the local router and its neighbors.

At each clock cycle, the history buffer records the channels that have input requests. The input arbiter selects a request from input buffers to process based on the FIFO mechanism referring to records in the history buffer. The main task of the controller is to determine the routing path for incoming packets, based on the routing algorithm described above. Besides this, the controller also needs to send signals to its neighbors for updating its stress value. When there are new incoming packets from neighbors or the local core, the controller will inform neighbors to increase its stress value. When the outgoing direction for a packet is determined, the controller will set a request signal to the local core or the corresponding neighbor router, and inform all neighbors to decrease its stress value.

3. MODELING AND ANALYSIS

A NoC system can be modeled as a queuing network. The cores generate packets and inject them into the routing network. Each packet is queued in the input buffer of the first router, and then transmitted to the next router until it reaches its destination.

3.1 Router Modeling and Analysis

One of the best indicators for a router's performance is its mean response time. In our analysis, we model each buffer in a single router as a non-preemptive infinite buffer. Although each channel has a separate input buffer, the sequence to process all requests is based on the FIFO mechanism. Hence, all input buffers of a router can be modeled as a single FIFO queue. Using the infinite buffer model, we can estimate the mean waiting time of a packet in each router, and thus can use this information to estimate the required buffer size of each router for a specific traffic load. To model and analyze the mean response time of each router, we firstly analyze the traffic load of each router.

A. Traffic load

Assume that the NoC network is a two-dimensional network with $U \times V$ routers (cores). Router i (core i) has a network address (i_x, i_y) which indicates its x and y coordinates, respectively. A packet enters router i due to one of the following three reasons: 1) The packet from core i has to be sent out from its local router (fixed regardless of the routing algorithm); 2) The packet whose destination is core i has to go through its local router(varied with different network communication patterns); 3) The packet needs to go through router i to be passed to other routers (affected by both the network communication pattern and the routing algorithm).

Assume that each core generates packets following Poisson distribution with mean rate λ (λ is also called the average packet injection rate for the NoC). The service time of each router for all packets follows exponential distribution with mean rate μ. Let λ_i be the mean packet arrival rate of router i, λ_{s_d} be the mean rate of packets from core s to core d and $P_{s\_d\_i}$ be the probability of a packet from core s to core d via router i. The mean packet arrival rate of router i can be calculated by

$$\lambda_i = \lambda + \sum_{s=1}^{U\times V} \sum_{d=1}^{U\times V} \lambda_{s\_d} P_{s\_d\_i}, \text{ for } s \neq d. \quad (1)$$

The mean rate λ_{s_d} of packets from core s to core d is determined by the network communication pattern, and the probability of a packet from core s to core d via router i is determined by both the network communication pattern and the routing algorithm. Here, we use an uniform network communication pattern to model the traffic load of each router with both static XY and DyXY routing algorithms. With the uniform network communication pattern, λ_{s_d} can be calculated as

$$\lambda_{s\_d} = \frac{\lambda}{U \times V - 1}, \text{ for } \left\{ \begin{array}{l} 1 \leq s \leq U \times V \\ 1 \leq d \leq U \times V \end{array} \right., \text{ and } s \neq d. \tag{2}$$

For static XY routing, the probability of a packet from core s to core d via router i is given by

$$P_{s\_d\_i} = \begin{cases} 1, & \text{if } i_y = s_y \text{ and } i_x \in [s_x, d_x] \text{ (or } [d_x, s_x]) \\ & \text{or } i_x = d_x \text{ and } i_y \in [s_y, d_y] \text{ (or } ([d_y, s_y]), \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where $z \in [l, h]$ denotes that z is a value between l and h. Hence, the mean arrival rate of each router with static XY routing can be calculated using Equations (1), (2) and (3).

For DyXY routing, the probability of a packet from core s to core d via router i is given by

$$P_{s\_d\_i} = \begin{cases} 1, & \text{if } i = s \text{ or } i = d, \\ 0, & \text{if } i_y \notin [s_y, d_y] \text{ (or } [d_y, s_y]) \\ & \text{or } i_x \notin [s_x, d_x] \text{ (or } [d_x, s_x]), \\ \sum_{j \in \psi} P_{s\_d\_j} P_{j\_i}, & \text{otherwise,} \end{cases} \tag{4}$$

where ψ is the set of router i's neighbors, which is located in a packet's possible routing paths from core s to core d, immediately before router i. Further, P_{j_i} is the probability that router j forwards a packet to its neighbor router i with destination core d, and it can be calculated as follows:

$$P_{j\_i} = \begin{cases} 0, & \text{if } i_y \notin (j_y, d_y] \text{ (or } [d_y, j_y)) \\ & \text{or } i_x \notin (j_x, d_x] \text{ (or } [d_x, j_x)), \\ 1, & \text{if } i_x = j_x = d_x, i_y \in (j_y, d_y] \text{ (or } [d_y, j_y)) \\ & \text{or } i_y = j_y = d_y, i_x \in (j_x, d_x] \text{ (or } [d_x, j_x)), \\ p, & \text{otherwise,} \end{cases} \tag{5}$$

where p is a variable depending on congestion conditions of the network. For a packet in router j whose destination is a core in the right-up direction, the packet can be forwarded to either router i ($i_x = j_x$, $i_y = j_y+1$) or router k ($k_x = j_x+1$, $k_y = j_y$). If the probability to forward this packet to router i is p, the probability to forward this packet to router k is $1-p$. Since the DyXY routing algorithm chooses a path based on each possible router's stress value, the probability can be estimated by $p = W_k/(W_k + W_i)$, where W_k (or W_i) is the mean waiting time of router k (or router i). Fortunately, W_k can be approximated using M/M/1 queue mean waiting time equation $W_k = \lambda_k/(\mu-\lambda_k)$. Combining these two equations with Equations (1), (2), (4) and (5), we can calculate the value of p, and thus the mean arrival rate of each router under DyXY routing can be calculated.

B. Mean response time

For static XY routing, the total traffic arrival process follows Poisson distribution, and hence a router can be modeled as a M/M/1 queue. The mean response time of router i can be calculated using

$$E[Tr_i] = 1/(\mu - \lambda_i). \tag{6}$$

where λ_i can be calculated using Equations (1), (2), and (3). For the DyXY routing algorithm, since the traffic of each router changes dynamically with network congestion conditions, the real traffic distribution is not a Poisson distribution. The mean router response time in this case can be estimated using a pair of upper bound and lower bound. The real traffic distribution is an interrupted Poisson distribution, which is actually an optimization based on network congestion conditions, therefore, the real mean response time should be smaller than that calculated using the mean arrival rate (λ_i) and the Poisson distribution model. Hence, the later one can be used as an upper bound of the real mean response time. The lower bound can be estimated by the mean response time with the minimum traffic at each router. The minimum traffic of each router occurs when p is set to 0 in Equation (5). In this case, the total traffic arrival process for each router follows Poisson distribution, and the mean arrival rate of each router can be calculated using Equations (1), (2), (4) and (5).

After calculating the mean response time of each router, we can derive the mean waiting time of a packet in each router by $E[Tr_i] - 1/\mu$. The mean buffer size required for each router can be calculated using $E[W_i] \times \lambda_i$ by Little's law [10]. The assignment of the buffer size to each channel of a router can be determined based on the traffic load at each different direction of the router. The average mean response time of all routers, $E[Tr]$, can be calculated by

$$E[Tr] = \frac{1}{U \times V} \sum_{i=1}^{U \times V} E[Tr_i]. \tag{7}$$

The performance of a router with finite buffer size α can also be analyzed similarly with one more performance indicator (the blocking probability of packets) into consideration. The details are not presented here due to the space limit.

3.2 System Modeling and Analysis

The performance of the entire system can be evaluated by the average packet latency $E[Latency]$, which equals to $E[Tr] \times N$, where N is the average packet path length (i.e., average routing path length). $E[Tr]$ can be derived directly by Equation (7), and N depends on the specific communication pattern and routing algorithm employed.

With static XY routing, the length of a path traveled by packets for a given pair of source and destination is a constant, which equals the shortest path length. For DyXY routing, although the routing path is not static, it is always a shortest path and hence the length is still the shortest path length. Therefore, the average packet path length is only affected by the communication pattern. Without losing the generality, we consider both uniform and non-uniform communication patterns in this paper, and we choose Poisson distribution for the non-uniform communication pattern since it is a widely used distribution for statistical analysis, and can reflect real situations of a system. Once the communication pattern is fixed, N can be easily derived. Due to the space limit, the details are not presented here.

4. EXPERIMENTAL RESULTS

To evaluate the performance of the DyXY routing algorithm and verify the correctness of our analytical models, we developed an event-driven simulator using C++ and designed three sets of experiments. In all these experiments, the buffer depths were set to infinite. To refelct the difference in packet lengths, the service time (not including the waiting time) of each router for each packet was set to a variable which follows a Poisson distribution (with mean service rate μ equal to 1). More than 140,000 packets have been injected into the network in each simulation, and the NoC was warmed up for 20,000 packets before measuring latencies.

The first set of experiments is based on NoCs (size varied from 3×3 to 9×9) with average packet injection rate λ increasing from 0.1 to 0.3 under both the DyXY and static XY routing algorithms. We have found that the simulation results precisely match with the analytical results for both routing algorithms. Further, the DyXY routing algorithm achieves better balance in load distribution (for routers in the center, edge, and corner) compared with the static XY routing algorithm, and thus it can relieve the hot-spot problem when the network traffic is high. Results for a 3×3 NoC are shown in Figure 3. As we can see, the analytical model for static XY routing can precisely evaluate the average mean response time for all routers. For DyXY routing, the average mean response time for all routers can be effectively estimated using the analytical lower bound and upper bound.

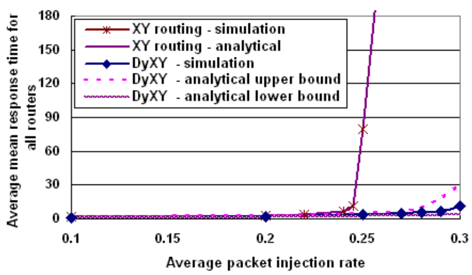

Figure 3: Average mean response time for all routers in 3x3 NoC.

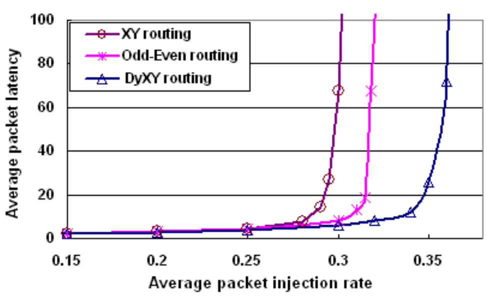

Figure 4: Average packet latency for 3x3 NoC with Poisson distributed network communication pattern.

Since DyXY routing can balance the load distribution among all routers much better than static XY routing, the average mean response time for all routers is smaller with DyXY routing than that with static XY routing. Further, since the average packet path length is the same for both routing algorithms, the average packet latency with DyXY routing is also smaller than that with static XY routing. To verify this, we conducted experiments by simulation for both Poisson and uniform distribution network communication patterns, and also compared the results with odd-even routing. It can be observed that DyXY routing achieves the best performance in average packet latency in both cases. Results for one set of experiments are shown in Figure 4 (Poisson). The performance of a network with the Poisson distribution based communication pattern is not sensitive to the network size. However, the performance under the uniform network communication pattern is affected by the network size. Therefore, our last set of experiments changed the size of NoC from 3×3 to 9×9 with λ fixed to 0.15. The results are shown in Figure 5 (uniform). Obviously, it can be seen that the system performs best with DyXY routing.

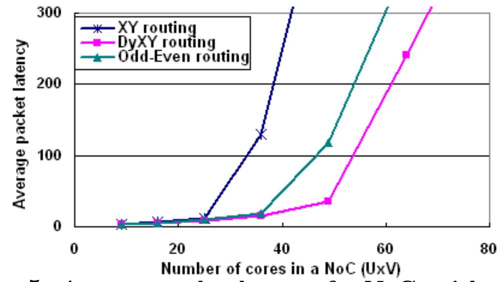

Figure 5: Average packet latency for NoCs with uniform network communication pattern.

5. CONCLUSIONS

In this paper, we have proposed a novel routing algorithm, namely dynamic XY (DyXY) routing, which provides adaptive routing based on congestion conditions in the proximity, and ensures deadlock-free and livelock-free routing at the same time. The DyXY routing method can be supported by a router architecture efficiently. Analytical models based on queuing theory were developed for both static XY routing and DyXY routing to evaluate their performance for two-dimensional mesh NoC architectures. The accuracy of the analytical models has been verified by extensive simulations. It has been observed that DyXY routing can achieve better performance than static XY routing and odd-even routing.

6. REFERENCES

[1] L. Benini and G. D. Micheli. Networks on chips: a new SOC paradigm. *IEEE computer*, 35:70–78, Jan 2002.

[2] W. J. Dally and B. Towles. Route Packets, Not Wires: On-Chip Interconnection Networks. In *Proc. Design Automation Conf.*, pages 684–689, 2001.

[3] W. J. Dally. Virtual-channel flow control. *IEEE Trans. Parallel and Distributed Systems*, 3:194–205, Mar 1992.

[4] Y. M. Boura and C. R. Das. Efficient fully adaptive wormhole routing in n-dimensional meshes. In *Proc. Int'l Conf. Distributed Computing Systems*, pages 589–596, 1994.

[5] C. J. Glass and L. M. Ni. Maximally fully adaptive routing in 2d meshes. In *Proc. 1992 Int'l Conf. Parallel Processing*, pages 101–104, 1992.

[6] I. Corporation. A touchstone delta system description. In *Intel Advanced Information*, 1991.

[7] C. J. Glass and L. M. Ni. The turn model for adaptive routing. *Journal of the ACM*, 41:874–902, Sept 1994.

[8] G. M. Chiu. The odd-even turn model for adaptive routing. *IEEE Trans. on Parallel and Distributed Systems*, 11:729 – 738, July 2000.

[9] J. C. Hu and R. Marculescu. DyAD - smart routing for networks-on-chip. In *Proc. Design Automation Conference*, pages 260 – 263, 2004.

[10] L. Kleinrock. *Queuing Systems*. John Wiley & Sons Inc., New York, 1976.