Reducing Packet Dropping in a Bufferless NoC*

Crispín Gómez, María E. Gómez, Pedro López, and José Duato

Dept. of Computer Engineering, Universidad Politécnica de Valencia, Camino de Vera, s/n, 46071–Valencia, Spain
crigore@gap.upv.es,{megomez,plopez,jduato}@disca.upv.es

Abstract. Networks on chip (NoCs) has a strong impact on overall chip performance. Interconnection bandwidth is limited by the critical path delay. Recent works show that the critical path includes the switch input buffer control logic. As a consequence, by removing buffers, switch clock frequency can be doubled. Recently, a new switching technique for NoCs called Blind Packet Switching (BPS) has been proposed. It is based on replacing the buffers of the switch ports by simple latches. Since buffers consume a high percentage of switch power and area, BPS not only improves performance but also helps in reducing power and area. In BPS there are no buffers at the switch ports, so packets can not be stopped. If the required output port is busy, the packet will be dropped. In order to prevent packet dropping, some techniques based on resource replication has been proposed. In this paper, we propose some alternative and complementary techniques that does not rely on resource replication. By using these techniques, packet dropping and its negative effects are highly reduced. In particular, packet dropping is completely removed for a very wide network traffic range. The first dropped packet appears at a 11.6 higher traffic load. As a consequence, network throughput is increased and the packet latency is kept almost constant.

1 Introduction

Current high performance microprocessors contain several processing cores on a single chip. Recently, Intel has announced a TeraFLOP processor containing 80 cores on a single chip connected by a NoC [17]. Also, Tilera Corporation [16] has developed the Tile64$^{\text{TM}}$processor, which implements 64 general–purpose processors inside the chip. In these chip multiprocessors (CMPs), shared buses and dedicated wires can serve a limited number of processing elements, thus being unable to keep up with the rapidly growing demands for high communication performance. Packet switched NoCs provides the bandwidth required for on-chip communications. NoCs adopt concepts and design methodologies inherited from off-chip interconnection networks. However, NoCs have very different characteristics. While many designs for off–chip networks are mainly performance–driven, power and area consumption are also two important constraints in NoCs. As an example, buffer size should be limited since they consume a lot of area and power [4,5]. On the contrary, wires are an abundant available resource. These differences lead to explore new approaches for NoCs.

* This work was supported by the Spanish MEC under Grant TIN2006-15516-C04-01, by CONSOLIDER-INGENIO 2010 under Grant CSD2006-00046, and by the European Commission in the context of the SCALA integrated project #27648 (FP6).

2 Motivation

The growing number of processing elements inside a chip increases the on–chip communication demands. There are several ways to increase NoC performance, most of them inherited from the off-chip networks. Another interesting approach, inherited from the VLSI design, is to increase the network clock frequency. The network clock frequency is determined by the switch critical path delay. Some recent works [12, 13] show that as the switch buffer size is decreased, the clock frequency can be increased. In particular, by eliminating the buffers at the switch ports and replacing them by one–flit latches, the switch frequency can be increased from 1GHz up to more than 2GHz [12, 13] (a 2X increase). This is due to the fact that the critical path is reduced, since the buffering architecture at the switch ports requires some control logic whose delay adds up to data–path delay.

Other works [4, 5] show that a high percentage of both, area and power, is consumed by the switch port buffers, so reducing buffer size can mitigate or even compensate the power consumption increase due to working at higher frequencies. Therefore, reducing the buffer size is an objective in NoCs as it has a positive impact on area and power and enables improving performance.

On the other hand, the high wiring capability of NoCs has led to networks with very wide links. For instance, in [4, 15], two different NoCs are proposed with link widths of 80 and 256 bits, respectively. Moreover, 512–bit wires are considered feasible in the near future [1]. These wide links provide low latency communication, because it decreases the number of *phits* that compose a packet.

Network topology also plays a key role, since it has a high impact on the overall performance and power consumption. The current trend is to use low dimensional networks such as meshes and tori. Meshes are the preferred choice because they allow exploiting the locality of communication, they are scalable, regular and are very energy–efficient [10].

These two facts could lead to use meshes with wide links. This will benefit systems that can take advantage of wide links, such as transmitting large packets or streaming communication networks. However, there are some systems that can not take advantage of these wide links, because they generate small packets that cannot fill the link width. An example of small packets are cache coherence messages, where most of the packets are just composed of a little header and a memory address. For current systems, link widths around 64–bit should be enough.

Nevertheless, there are other options to take advantage of the huge wiring capabilities of NoCs. For instance, splitting the wide links into narrower links that better fit packet size, allowing in this way multiple parallel channels in the same direction [7], known as Space Division Multiplexing (SDM).

In [8], we took advantage of these ideas to propose a new switching technique for NoCs called BPS, based on replacing the switch port buffers by simple latches, with the aim of increasing the system performance and, at the same time saving area and power. The main drawback of BPS is that some packets may be dropped when contention arises. These dropped packets are reinjected later from their source node. In [8], some techniques based on resource replication were proposed to reduce the number of dropped packets, since it affects to the network performance.

In this paper, we are going to propose some complementary techniques to reduce the number of dropped packets that are not based on resource replication. For this purpose, we will summarize BPS in Section 4 to make this paper as self–contained as possible. Following, we will present the proposed techniques in Section 4.1. These techniques are going to be applied to BPS, but they could be used to prevent packet dropping in any other NoC that allows packet dropping. They will be evaluated in Section 5. Finally, some conclusions and future work will be drawn in Section 6.

3 Related Work

In NoCs, performance and saving power and area are equally important in most cases [2, 5]. Among all the traditional switching techniques, wormhole is the most appropriate for NoCs [3, 4], because it relies on small size buffers [10]. Some works [9] even propose to return to previous switching techniques such as circuit switching to completely remove buffers. The problem is that it wastes network resources due to the need of resource reservation. We believe that a new switching technique that fits into the limitations of NoCs must be developed, efficiently taking advantage of their singular characteristics. There are some recent works that progress in this direction. For instance, in [3] the authors propose to use the link repeaters in order to provide additional buffering resources, also defining a new flow control mechanism. In [14], the authors propose to retake the *hot potato routing*, or deflection routing, with the aim of using small buffer networks [11], that also work at higher frequencies. However, deflection limits are not guaranteed to prevent livelocks, and the huge wiring capability of NoCs is not exploited.

4 Blind Packet Switching

As buffers are the most consuming part in terms of power and area, BPS remove them from switch ports. Additionally, this allows to achieve a higher network performance, since it is possible to increase the network frequency. Figure 1.(a) shows a switch with input and output buffers. Only one input and one output port of the switch are shown. Buffers are composed by multiple latches serially connected plus some control logic. Every latch can store a flit. Figure 1.(b) represents our switch. Input and output queues have been reduced to just one flit latch per port and, therefore, buffer control logic has been removed. These latches actually work as repeaters, they do not store the flits while packets are blocked. They only store a flit during a clock cycle. As stated in Section 2 and according to [12, 13], this switch can work approximately at least at twice the frequency of the buffered switch.

As there are no buffers, when a packet can not get the resources it requires to go on through the network, it is dropped. The switch that drops the packet sends a Negative Acknowledgment (nack) to the corresponding source node, which will reinject the packet. Dropped packets introduce additional traffic load due to the nack packet and the reinjection of the packet, and also increase the packet latency. To avoid packet dropping, packets must advance through the network like a never-stopping flow. In [8], we proposed several solutions to reduce packet dropping.

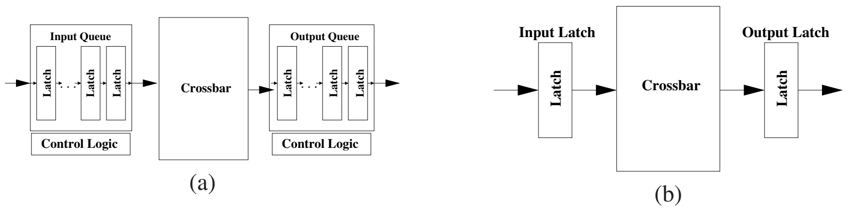

Fig. 1. (a) A traditional buffered switch. (b) A BPS switch.

We proposed to replicate the conflicting resources such as network links, ejection channels and routing circuits.

Concerning network links, we proposed to use fully adaptive routing to increase the number of routing options. To increase even more the routing options, we use multiple parallel links in the same direction (SDM). In the same way, we propose to provide multiple ejection channels between switches and nodes to relax the ejection bottleneck due to SDM. Routing availability can be easily increased by replicating the routing circuit, or by using a single routing circuit that can handle multiple packets at once.

Nevertheless, there are still some scenarios where a packet still may be dropped. In order to prevent packet losing, nack packets are used. When a packet is dropped in a switch, that switch is responsible of forwarding the corresponding nack packet to the source node. Once the nack has reached the source of the packet, it is reinjected into the network. Nack packets are given priority over data packets and can not be dropped. Therefore, if there is not an available output port to send the nack, they are ejected from the network and are stored in a small queue (unique per switch).

Notice that in BPS, deadlocks can not occur since packets are never stopped, so virtual channels are not required [6] even for adaptive routing. On the contrary, in a buffered wormhole network, at least two virtual channels per direction are required to prevent deadlocks. Hence, with BPS all the hardware related to virtual channel multiplexing is removed and the crossbar complexity is reduced to the quarter. Furthermore, no link level flow control mechanism is required. Packets move blindly as switches do not use buffer availability to route packets. As there is neither a link level flow control mechanism, nor a deadlock avoidance mechanism, the switch control logic is expected to be less complex.

In [8], we showed that BPS can outperform wormhole in most of the cases. However, the number of parallel links has a great impact on BPS throughput. In particular, throughput was strongly increased with 4 and 8 parallel links (a 8×8 mesh achieved a 50% higher throughput than wormhole). Finally, BPS latency did not depend on traffic for low and medium network loads and was approximately half the latency of the corresponding buffered network. The percentage of reinjected packets was negligible at low and medium traffic loads for 4 and 8 parallel links. Moreover, for high traffic loads before saturation point, the highest percentage of reinjected packets for a 8×8 mesh was close to a 5%. Nack queue size did not exceed 4 packets for the aforementioned networks.

However, dropping packets and reinjecting them from the source nodes not only requires the burden of nacks but also it may introduce out of order delivery of packets.

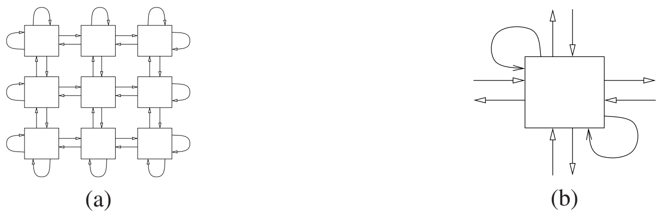

Fig. 2. (a) A closed 3×3 mesh, external output ports are connected to their corresponding input port. (b) A switch with two extra loopback channels.

Therefore, some additional techniques to reduce the percentage of reinjected packets are required. What is actually needed is that there will not be any reinjected packet for an extended range of network traffic. In this way, a technique that expands this traffic range is welcomed.

4.1 Reducing Packet Dropping by Misrouting and Loopback Channels

In this section, we propose some techniques to reduce packet dropping and its consequences. All the proposed techniques provide a second chance to packets that should be dropped. We propose using misrouting, external misrouting and loopback channels. Misrouting consists of forwarding the packet through any free output port that provides a non-minimal path towards its destination and is well–known in interconnection networks [6, 11]. We also propose two variants of misrouting specially designed for NoCs. All these techniques can be used on their own or combined, and can be applied to any network that has a reduced amount of buffers like [11] or that allows packet dropping, like BPS.

External Misrouting. External misrouting is an extension of the common misrouting. When misrouting a packet, its latency is increased. However, when compared with dropping the packet, the increase of packet latency is clearly smaller than the reinjection time.

The novelty of external misrouting comes from the following idea. External links of the mesh–border switches are not used. We propose to connect these unconnected switch output ports to their corresponding input port, forming a loopback channel (Figure 2.(a)). This loopback channel will act as a false buffer to store one flit during one clock cycle, providing an additional chance to the packet. The switch provides an extra buffer at almost no cost (a very short link).

When a packet should be dropped in a switch that belongs to the border of the mesh, external misrouting is first tried, misrouting the packet through the loopback channels instead of misrouting it through other network links, thus avoiding the use of network links. If the loopback link is busy, then traditional misrouting is used.

Notice that when using external misrouting, the packet is kept at the same switch. Furthermore, when a packet is misrouted though a loopback channel, its route is increased by only one hop, whereas traditional misrouted packets suffer a packet route length increase of two hops, as they are misrouted out of their minimal path (one hop) and then back to their minimal path (another hop). Nevertheless, packets may still be dropped if there is not any available output port, or if the number of misroutings has reached its maximum value, since the number of misroutings must be limited to avoid livelocks.

BPS preserves in-order packet delivery for all the packets that are not dropped. Non-dropped packets use minimal path routing, and there is no packet blocking, therefore all the paths between a source and a destination take the same time. Dropped packets can be delivered out-of-order because a new packet can be injected into the network from the same source to the same destination in the elapsed time between the first injection of the dropped packet and its reinjection. Misrouting steps up packet in-order delivery, since it strongly reduces this elapsed time, as the reinjection takes longer than the misrouting.

Extra Loopback Channels. We propose extending the loopback channels of the mesh border switches to all network switches, thus allowing additional buffering resources at each switch to prevent packet dropping. In Figure 2.(b), we show a central switch of the mesh with two extra loopback channels. Switch degree is increased, but the number of wires out of the switch is the same, as loopbacks are a switch internal connection.

The extra loopback channels can be added by using new wires or not. By using new wires, the total number of switch ports will be increased. This approach provides the best performance. On the other hand, if the number of switch ports can not be increased, we can sacrifice some ports used to connect to other switches to implement the loopback channels. In this case, it is expected that network throughput will be reduced. An analysis of the tradeoff between loopback and network links will be performed in the next section.

5 Evaluation

5.1 Simulation Environment

A detailed event-driven simulator that implements BPS has been developed. The simulator models a mesh with multiple parallel point-to-point bidirectional links in the same dimension. Packets are adaptively routed. Additionally, processing nodes have as injection and ejection channels as parallel links in the network. Routing, link and crossbar delay are supposed to be equal to one clock cycle. We assume that the packet generation rate is constant and the same for all the nodes. The destination of a packet is randomly chosen. Packet sizes and percentage of short packets are set to model a cache coherence system and were used in [1, 7, 8] to simulate a CMP system. Short packet size is 32 bits and correspond to command packets such as requests and invalidations. Large packet size is 544 bits and correspond to response packets that contain cache lines. The percentage of short packets is 60%. A 32–bit header is attached to all the packets. Link width will be fixed to 64 bits, because that is the total size of short packets. Packet latency is the difference between the first packet injection and its ejection from the network at its destination, including all the packet reinjections.

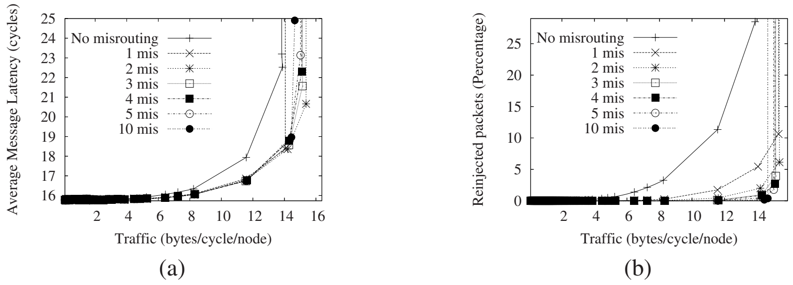

Fig. 3. Impact of misrouting on performance. 4×4 mesh with 4 parallel links per dimension.

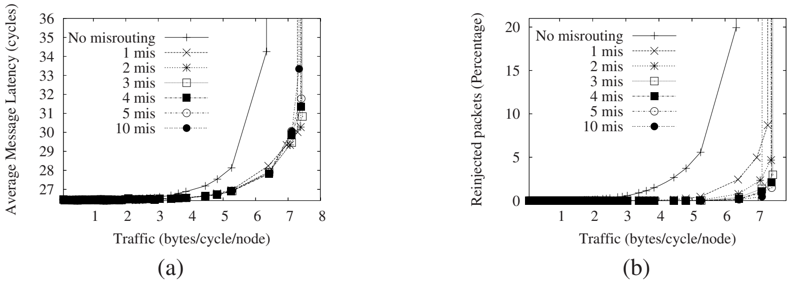

Fig. 4. Impact of misrouting on performance. 8×8 mesh with 4 parallel links per dimension.

5.2 Evaluation Results

In this section, we will analyze the impact of the proposed techniques in the reduction of the number of dropped (and reinjected) packets. As the number of dropped packets is reduced, the throughput is expected to be increased and the packet latency reduced. First, we study the impact of misrouting combined with external misrouting on BPS. Next, we analyze the impact of having extra loopback channels at any switch, both by adding new switch ports and by sacrificing network links to implement loopbacks. Finally, we analyze the impact of using both techniques simultaneously.

Misrouting Plus External Misrouting. In Figures 3 and 4, we show results for a 4×4 mesh and a 8×8 mesh with 4 parallel links per dimension varying the number of maximum misroutes[1]. Misrouting increases the network throughput by 1.5% in the small network and by 15% in the large network (Figures 3.(a) and 4.(a), respectively). The throughput increase is proportional to the network size.

Only the first misroute seems to be significant in both networks. The influence of the additional misroutes seems negligible. As we can see in Figures 3.(b) and 4.(b), the first misroute reduces the percentage of dropped packets from 28% to 5.2%, in the small network, and from 20.1% to 3%, in the large network. In the original BPS the first dropped packet appears when the network throughput is 0.27 bytes/cycle/node. By

[1] Eventhough more network sizes were studied, as the results are similar to the ones presented in here, we do not show them due to lack of space.

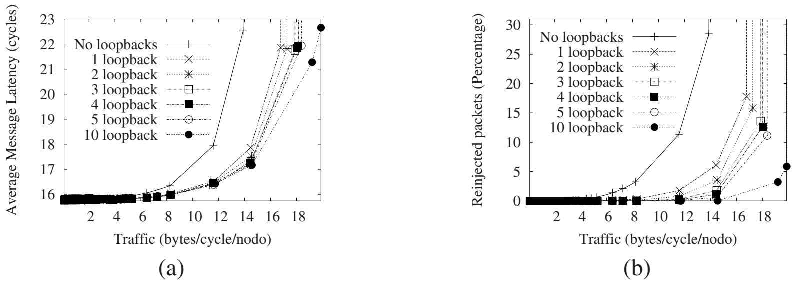

Fig. 5. Impact of loopbacks on performance. 8×8 mesh with 4 parallel links per dimension.

using one misrouting, the first dropped packet appears at 0.86 bytes/cycle/node. The network has to drop the first packet at a 2.2 higher network traffic.

Extra Loopback Channels. In Figures 5 and 6, we show results for the same networks, varying the number of extra loopback channels without using misrouting. In Figures 5.(a) and 6.(a), we can see that just by adding one extra loopback channel per switch, the network throughput is increased by 21% and by 13% for the small and the large network, respectively. On the contrary to misroute, each additional loopback significantly increases the network throughput. For instance, it is increased by 18.4% and by 34% for the small and the large network, when the number of loopbacks is ten instead of one. As the network size is increased, the influence of each additional loopback is also increased. A single loopback provides the packet with another opportunity before dropping it, but the packet is kept in the same switch that was going to drop it. If this switch was suffering a high network traffic load, it is highly probable that the packet can not find a suitable output port in only one clock cycle. It is well-known that the network congestion tends to increase with the network size in a mesh. Hence, as the network size is increased, the percentage of packets that can be saved from dropping is increased by each additional opportunity.

In Figures 5.(b) and 6.(b), we can see that the percentage of dropped packets is strongly reduced as we add loopback channels. Notice that it is negligible till the network saturation when the large network is using ten loopback channels. The first dropped packet appears when the network traffic is 0.58, 1.14, 2.89 and 6.41 bytes/cycle/node when the switches have 1, 2, 5 and 10 extra loopback channels, respectively.

Figure 7 shows the obtained results when the loopback channels are implemented at the expense of network channels. There are two ways of doing this: taking out one of the 4 data links, resulting in three data links and one loopback channel per direction, or reducing the width of every network link to free the number of switch outwards wires required to form a loopback channel, having four narrower data links and one loopback channel per direction. Results show that, when loopback channels are implemented by sacrificing network links, the network performance is similar to the one obtained without loopback channels. Although the use of a loopback channel does not improve the throughput, the number of wires between the switches has been reduced by a 25%, which helps in simplifying the physical wiring and reducing the number of required link

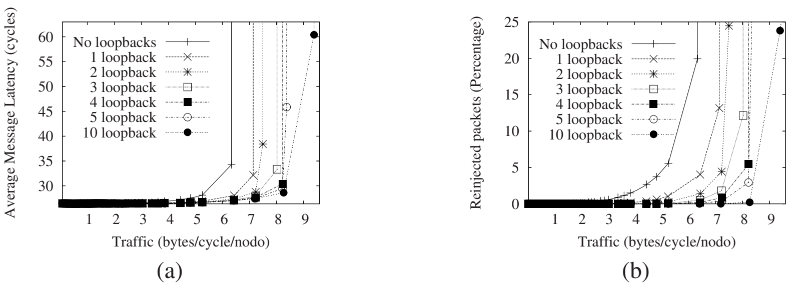

Fig. 6. mpact of loopbacks on performance. 8 × 8 mesh with 4 parallel links per dimension.

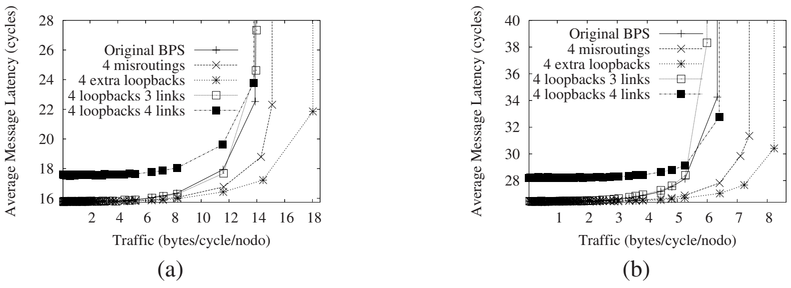

Fig. 7. Comparison of misrouting and loopbacks. 4 parallel links per dimension. (a) 4 × 4 mesh. (b) 8 × 8 mesh.

repeaters. Notice that using narrower links is not a good approach because it increases the packet latency.

Misrouting Combined with Loopback Channels. Finally, we evaluate both proposed techniques working together. Figure 8 show the obtained results. In Figure 8.(a), we can see the performance for one maximum misroute, one extra loopback channel, and one maximum misroute and one extra loopback. By combining both techniques, the throughput is increased by 10.6% over one misrouting, and by 12.8% over only one loopback. The first dropped packet appears when the network traffic is 2.51 bytes/cycle/node, a 8.3 higher traffic load.

In Figure 8.(b), we evaluate four different configurations in order to identify the most interesting technique to replicate when combining them. For this purpose, we compare the following combinations: two maximum misroutes, two loopback channels, two misroutes and one loopback, and one misroute and two loopbacks. As it was expected, the best performance is obtained for one misroute and two loopback channels, since, as it was shown, only the first misroute actually improves network performance. So, it is more interesting to have additional loopback channels than increasing the maximum number of misroutings. In this case, with 1 misrouting and 2 loopback channels, the first dropped packet appears at 3.40 bytes/cycle/node. The network traffic when the first packet is dropped is 11.6 times higher.

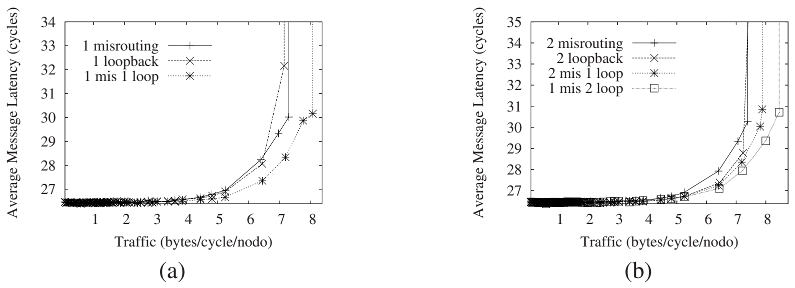

Fig. 8. Impact of combining misrouting and loopback channels on performance. 8×8 mesh with 4 parallel links per dimension.

6 Conclusions and Future Work

We have proposed some techniques to improve the performance of a bufferless NoC with packet dropping. These techniques are focused on reducing the number of dropped packets by using misrouting and loopback channels. The results show that both techniques are able to strongly reduce the number of dropped packets, thus allowing a higher network performance, and reducing the negative effects of packet dropping: nack-related issues and out-of-order delivery of packets. In particular, when combining all the techniques, the first dropped packet appears at a 11.6 times higher traffic; up to this point the network does neither need any nack queue, nor memory in the end nodes to reinject the packets.

Only one misrouting is required to improve performance. Concerning loopback channels, the performance is improved as loopback channels are added, especially in large networks. Furthermore, we have shown that loopbacks can help in reducing wiring requirements in the network. The best option is to combine both techniques, providing one misroute and as many loopback channels as possible.

For future work, we want to make a power and area model of the loopbacks channels in order to evaluate their influence over the total NoC consumption.

References

1. Balfour, J., Dally, W.J.: Design Tradeoffs for Tiled CMP On-chip Networks. In: 20th Annual International Conference on Supercomputing, pp. 187–198 (2006)
2. Benini, L., De Micheli, G.: Networks on Chips: A New SoC Paradigm. Computer 35(1), 70–78 (2002)
3. Bertozzi, D., Benini, L.: Xpipes: a Network-on-chip Architecture for Gigascale Systems-on-chip. IEEE Circuits and Systems Magazine 4(2) (2004)
4. Dally, W.J., Towles, B.: Route Packets, Not Wires: On-chip Interconnection Networks. In: 38th Conf. on Design Automation, pp. 684–689 (2001)
5. De Michelli, G., Benini, L.: Network on Chips. Morgan Kaufmann, San Francisco (2006)
6. Duato, J., Yalamanchili, S., Ni, L.: Interconnection Networks. An Engineering Approach. Morgan Kaufmann, San Francisco (2004)
7. Gómez, C., Gómez, M.E., López, P., Duato, J.: Exploiting Wiring Resources on Interconnection Network: Increasing Path Diversity. In: 16th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, pp. 20–29 (2008)

8. Gómez, C., Gómez, M.E., López, P., Duato, J.: An Efficient Switching Technique for NoCs with Reduced Buffer Requirements. In: International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2008) (submitted, 2008)
9. Goossens, K., Dielissen, J., van Meerbergen, J., Poplavko, P., Radulescu, A., Rijpkema, E., Waterlander, E., Wielage, P.: Guaranteeing the Quality of Services in Networks on Chip. In: Nurmi, J., Tenhunen, H., Isoaho, J., Jantsch, A. (eds.) Networks on Chip, ch. 4, Kluwer, Dordrecht (2003)
10. Kavaldjiev, N.K., Smit, G.J.M.: A Survey of Efficient On-Chip Communications for SoC. In: PROGRESS Symp. on Embedded Systems (2003)
11. Lu, Z., Zhong, M., Jantsch, A.: Evaluation of On-chip Networks Using Deflection Routing. In: 16th ACM Great Lakes Symposium on VLSI, pp. 296–301 (2006)
12. Martini, F., Bertozzi, D., Benini, L.: Assessing the Impact of Flow Control and Switching Techniques on Switch Performance for Low Latency NoC Design. In: 1st Workshop on Interconnection Network Architectures: On-Chip, Multi-Chip (2007)
13. Medardoni, S., Bertozzi, D., Benini, L., Macii, E.: Control and Datapath Decoupling in the Design of a NoC Switch: Area, Power and Performance Implications. In: International Symposium on System-on-Chip, pp. 1–4 (2007)
14. Millberg, M., Nilsson, E., Thid, R., Jantsch, A.: Guaranteed Bandwidth Using Looped Containers in Temporally Disjoint Networks within the Nostrum Network on Chip. In: Conf. on Design, Automation and Test in Europe (2004)
15. Mullins, R., West, A., Moore, S.: The Design and Implementation of a Low-latency On-chip Network. In: Asia and South Pacific Design Automation Conference, pp. 164–169 (2006)
16. Tilera Corporation, `http://www.tilera.com/`
17. Vangal, S., Howard, J., Ruhl, G., Dighe, S., Wilson, H., Tschanz, J., Finan, D., Iyer, P., Singh, A., Jacob, T., Jain, S., Venkataraman, S., Hoskote, Y., Borkar, N.: An 80-tile 1.28TFLOPS network-on-chip in 65nm CMOS. In: International Solid-State Circuits Conference (2007)