# Run-time Accelerate Channel for Communication-Aware Network-on-Chip

**Tianpeng Ai[1], Tongsen Hu[1], Tianzhou Chen[2]**

[1]Department of Computer science and technology, Zhejiang University of Technology, Zhejiang Province, China
[2]Department of Computer science and technology, Zhejiang University, Zhejiang Province, China
aitp2011@126.com, hts@zjut.edu.cn, 451783536@qq.com

**Abstract** - Network-on-Chip (NoC) is a major communication architecture in multi-core system. Reducing the average delay of network communication is a crucial problem to improve system performance. In this paper, a run-time acceleration mechanism is proposed to reduce the latency of busy traffic in a network. We first predict the pair-wise nodes who communicate with each other frequently based on application's locality. Then we propose a run-time acceleration mechanism which combines the packet-switched and the virtual-circuit switching. It can shorten the router pipeline of busy traffic we predicted, thereby achieves the goal of reducing network's average latency. We evaluate the proposed scheme on a 64-core CMP with a mesh topology, using a suite of applications from PARSEC. Our proposed scheme reduces network's average latency by 17.4% compared to a traditional packet-switched NoC.

**Index Terms** - Network-on-Chip (NoC), run-time accelerate channel (RAC), application locality, end-to-end communication

## 1. Introduction

As the number of cores increases in a multi-core system, Network-on-Chip has been a common communication architecture because of its high scalability and sufficient bandwidth. There are several ways adopted by the cores transmitting messages with each other such as circuit switching [1], virtual circuit switching [2] and packet switching [1].

The advantage of circuit switching is that transmission latency is small. But the circuit's setup time is long, it always occupies a physics link and other traffic can not pass the link before it is released. Thus, the bandwidth resource utilization is low. Virtual circuit switching is a modified circuit switching which occupies the virtual channel (VC) instead of the physics link. To some extent it raised the utilization of network's resource. When the number of nodes in the network increases, some communication nodes can not get a VC due to limited VCs. This will block some packets which originally should be transferred timely. Packet switching has a high utilization of network resource because it does not occupy the link resource, and does not set up a connection between the source and destination node. For this reason, packet switching has been widely used in the NoC. When a packet using the packet switching arrives at a router, it should go through several pipelined stages and then leave to the downstream node. Such a long pipeline causes high latency at every hop.

Previous work has attempted to reduce communication latency from different perspectives. Express virtual channels [3] try to shorten the router's pipeline. They set some privileged channel along a straight direction named EVC. The packets transmitted in the privileged channel needn't VC allocation and routing computation stage due to its destination node is the end of the channel. These packets have a high priority to uses crossbar than the normal packets and thus go through the crossbar directly. But the packet using EVC can only traverse along a straight direction. What is more, the normal packets may not get the right to use crossbar because of its low priority.

Hybrid circuit switching (HCS) [4] is a new network design which removes the circuit setup time overhead by intermingling packet-switched flits with circuit-switched flits. Often, the circuit utilization is low because a circuit will tear down when other conflicting circuits to be constructed. Circuit pinning [5] was proposed which can promote higher circuit utilization. The circuit should maintain for a period of time instead of tearing it down immediately even there is a conflict. Without considering the run-time traffic in the network, the established circuit using the alternative method mentioned above has little impact on the network average latency when the communication between the pair-wise nodes is not often.

Some studies on the behavior of application like [6] tested the application from the Splash-2 benchmark suite. They count the message number sending by a source thread to other different threads, and then they find only a few threads have a large number, the rest have little messages from the source thread. This means the application exhibits a spatial locality; there are some busy pair-wise communication nodes in the network at a time, this is also our motivation to propose the Run-time Accelerate Channel.

Application also exhibit temporal locality. By testing applications, Minseon *et al.* find that there is a high reusability of the end-to-end communication and the crossbar connections [7]. The temporal locality provide us a way to predict the run-time traffic in the network, we can use a time window to statistics the status of communication, and then predict that the status in the next time window is similar with the last window.

In this paper, we will describe how to identify the busy traffic and how to establish run-time accelerate channel (RAC). We improve the Pseudo-Circuit proposed in [7] for bypassing the SA stage. Also we will discuss the earnings of RAC and give the evaluate results.

The remainder of this paper is organized as follows. In Section 2, we introduce the router pipeline of traditional NoC, and then describe the process of constructing RAC; at last we improve the mechanism of bypassing SA. In Section 3, we

propose a method to identify the busy pair-wise communication, and then we discuss the earnings of RAC. We describe evaluation methodology and present simulation results in Section 4. In Section 5, we conclude our work.

## 2. Run-time Accelerate Channel

In this section, we first introduce the base pipeline of the traditional router, then we describe how we construct the accelerate channel and show the pipeline in our scheme.

### *A. Baseline Router Pipeline*

For simplicity, we assume the NoC's topology is mesh and a general router has five input and output ports corresponding to the four neighboring directions and the local processing element (PE) port. Each port has multiple VCs for avoiding the Head-of-Line blocking. There are some other major components such as route computation logic, VC allocator, switch allocator and crossbar switch.

Fig. 1(a) shows a general router's pipeline. When a head flit arrives at an input port, it will first be written into the buffers (BW). In the next stage, the route computation logic calculates the output port for the flit (RC). The flit then arbitrates for a VC of its output port (VA). If the flit gets a VC successfully, it then arbitrates for the switch (SA). If the flit wins the competition, it will traverse the crossbar in the next stage (ST). At last, it will travel to the next node in the link traversal stage (LT). Body and tail flit follow a similar pipeline except the RC and VA stages, because they have the same output VC as the head flit.

Lookahead routing [8] was proposed which can enable flits to arbitrate for VCs immediately after the BW stage. That means the RC and BW can be done in only one stage. Fig. 1(b) shows the pipeline using this technology. In this paper, we use this pipeline as our baseline router pipeline.

### *B. Establish the Accelerate Channel*

We first divide VCs in a port into three categories: normal VCs; reserved VCs and exclusive VCs. The normal VCs can be assigned to common packets which needn't acceleration; they will be released for later use after the tail flit of the packets leaves. Reserved VC is assigned to the busy pair-wise communication when the source node sends a query packet to the destination node. When the query is successful, reserved VC will be signed as an exclusive VC. Reserved and exclusive VCs only belong to some busy communication.

| | | | | | | |
|---|---|---|---|---|---|---|
| Head | BW | RC | VA | SA | ST | LT |
| Body/Tail | | BW | SA | ST | LT | |

(a) General router pipeline

| | | | | | |
|---|---|---|---|---|---|
| Head | BW | VA | SA | ST | LT |
| Body/Tail | BW | SA | ST | LT | |

(b) Baseline router pipeline

Fig. 1 General& baseline router pipeline

In order to guarantee enough normal VCs for common packets, we institute a rule that the maximum number of VCs used for the busy communication can not exceed a certain value $n$ in every router input port.

At the end of every period of time, each node will statistics the number of communication packets which it sent to all the other nodes, and then determines whether the pair-wise communication is busy, we will minutely discuss the judging method in Section 3. If there is a busy pair-wise communication, a RAC is needed to establish. Fig. 2 is the procedure of establishing an accelerate channel. It divides into two stages.

First, when we detect that the communication from a source node S to a destination node D is busy, node S sends a query packet to look up each node on the path. If the sum of reserved VCs and exclusive VCs at a node does not reach the maximum value, the pair-wise communication can get a reserved VC from the router's input port. Then we sign the normal VC occupied by the query packet as a reserved VC. Packets from other pair-wise communication will not be able to use this VC. If there are packets from S to D at this time, they can use the reserved state VC, and they can bypass the RC and VA stages. When the query packet reserves the reserved VC successfully in all nodes along the pair-wise path, node D sends a successful confirmation to the source node S after received the query. If the sum of reserved VCs and exclusive VCs is equals to $n$ at an intermediate node Z, that means the accelerate channel is failure to establish. At this time, node Z sends back a failing confirmation to node S along the way it came. When other intermediate nodes receive this failing confirmation, it signs the reserved state VC as a normal VC. When the source node S receives this failing confirmation, it will give up constructing the accelerate channel in this time window.

Secondly, when node S receives the successful confirmation, it notifies the entire nodes on the path to sign their reserved VCs as exclusive VCs. Meanwhile, the exclusive VCs binding with the crossbar and record this binding information in a register of the port. In this time window, the packets sending to D from S will go through this RAC with bypassing VA stage.

When S detected that the pair-wise communication is not busy any more, it can notify all the nodes on the path to sign their exclusive VCs as normal VCs. So, there is almost no cost to release the RAC.

### *C. Improve the Bypass of SA*

Pseudo-Circuit is a crossbar connection created by a flit traversal within a single router. It is recorded in a register and remains connected for future uses and thus enable the flit to be sent directly to the downstream router bypassing the SA stage. The pseudo-circuit is terminated when another flit from different VC claims either the input port or the output port. Multiple VCs in a router caused low reusability of the Pseudo-Circuit.

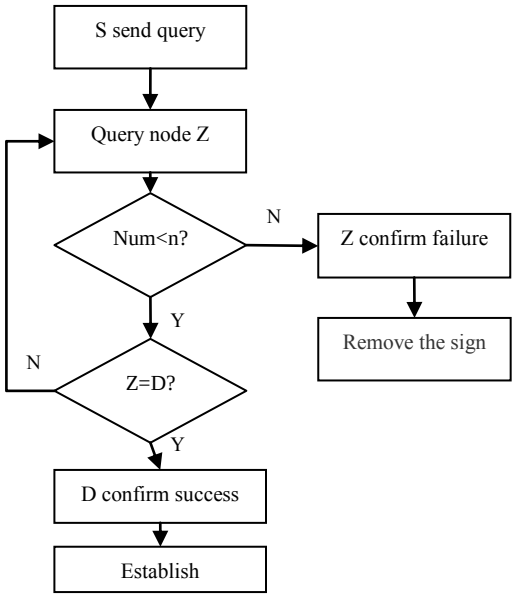


Fig. 2 the procedure of establishing RAC

In order to raise the reusability of the crossbar connection, an advanced mechanism is proposed. It includes three rules: first, only the flits from the exclusive VC can create a crossbar connection and record the information in a register; second, normal packets recover the crossbar connection according to the information recorded in the register after they go though the crossbar; third, other flits from a different exclusive VC will tear down the old connection and create a new one, at the same time, they should update the information of the register.

There are two reasons lead to significant increasing of reusability. On the one hand the number of VCs who can tear down the old connection decreased, on the other hand flits traverse the exclusive state VCs always come from busy end-to-end communication. When flits from exclusive VCs arrive at a router, they first detect whether the crossbar is free. If the crossbar is free and the connection information recorded in the register is same to their own, they can advance to the ST stage with bypassing BW and SA. The router pipeline using RAC and this advanced mechanism is show in Fig. 3.

## 3. Quantification of Network Communication

### *A. The determination of busy communication.*

A distributed method is used to statistic the status in a time window. It means that each node only use its own statistical result to identify the pair-wise communication's status, busy or not, without knowing the global communication information.

In this paper, we suppose that $N$ is the total number of VCs in a port. In a time window, the proportions of communication between the source node S and all the other nodes is sorted. For example, $\{r1, r2, r3 \ldots\}$. The order to establish a RAC is started from the node in the front of the sequence which has a greater proportion. In order to ensure a higher utilization of VC resource, we defined three constrains.

First, the source node of a busy pair-wise communication should send a certain number of packets in a time window. The number is a threshold for busy traffic, it equals to the size of time window divided by the average interval of sending two consecutive packets.


Fig. 3 the router pipeline using RAC

Second, the busy pair-wise is selected according to the sorted sequence. We traversal the sequence from first to end until it meets:

$$\sum_{i=1}^{k} r_i \geq n / N \tag{1}$$

When the communication is very uneven, (1) is quickly satisfied when $k$ value is little. Conversely $k$ value will be greater.

Third, if several communications are all very busy in a node; this means that they all have urgent demand to establish a RAC. For example, if the sum of $r1$ and $r2$ meets (1), and $r3$ is greater than $1/N$, then $r3$ can be selected to establish an accelerate channel without wasting the VC resources.

### *B. The earnings of speedup channel*

In this section, we give several formulas:

$$E = Es - Ln \tag{2}$$

$$Es = T \times (t1 + p1 \times (t2 + t3)) \times n \tag{3}$$

$$Ln = (1 - T) \times p2 \times t4 \tag{4}$$

$$t4 = c / (\mathrm{N} - \mathrm{n})^2 \tag{5}$$

$E$ is the earnings of our scheme; $Es$ is the earnings of the busy traffic traverse the RAC; $Ln$ is the loss of the normal traffic; $T$ is the ratio of busy traffic in network's total traffic; $(1-T)$ is the ratio of normal traffic in network's total traffic. $t1$ is the time cost in the VA stage; $t2$ is the time cost in the SA stage; $t3$ is the time cost in the BW stage; $p1$ is the reusability of the crossbar connection created by the flit from exclusive VCs. $p2$ is the probability of normal packets blocking due to the decreasing number normal state VCs; $c$ is a parameter determine by application in the NoC. $t4$ is the time cost of blocking.

The $p1$ value becomes smaller along with the $n$ value gets larger; that means the probability of successfully bypass the VA stage becomes smaller. Because of the application's temporal locality, the decrease is slower than linear. Therefore, the earnings of the busy traffic $Es$ is increasing along with the $n$ value. When the $n$ value continue to get bigger, the $p2$ value increases linearly, but the $t4$ value is increasing

more quickly. That means $Ln$ is increasing more quickly with the $n$ value growth. Assigning an appropriate value before use $n$ is helpful for the RAC to obtain better results.

## 4. Experiment

We evaluate system's performance using Gem5 [9], a cycle-accurate on-chip network simulator implementing pipelined routers. The network simulator is configured with 4-flit buffer per each VC and 8 VCs per each input port. We assume that the bandwidth of a link is 128 bits with additional error correction bits. We extract traces from PARSEC [10], multi-threaded benchmarks. In this experiment, we use dimension order routing (DOR) [11] algorithms. We set the time window to be 16000cycles, the average interval of sending two consecutive packets is 500 cycles, so the threshold discussed previously is 32 packets per window.

We first test the network's average latency using different $n$ values; the result is show in Fig. 4. For each application, the average latency of network is lower when the $n$ value is 2 than $n$ is equals to 1. This is because $Es$ increases more quickly than $Ln$ when the $n$ value is small. When $n$ value is 7, the latency is high because of many normal packets blocked.

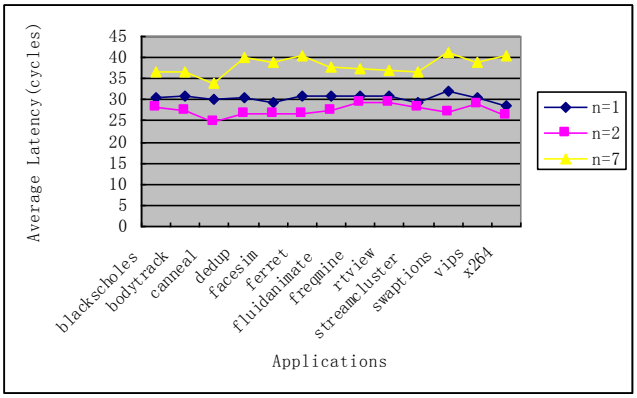


Fig. 4 Average Latency for different n Values

We compared the RAC when $n$ value is 2 with the traditional packet-switched NoC. The normalized average message latency is show in Fig. 5. We can find that the average latency reduction is 17.4%. Several applications which exhibit good locality such as *canneal, dedup, ferret and swaptions* can achieve over 20% latency reduction compared to the baseline NoC.

## 5. Conclusions

It is crucial to design a low-latency Network-on-Chip for higher performance. The proposed run-time accelerate channel target at reducing the latency of heavy traffic which mainly caused the network latency. This scheme enables the pair-wise nodes to bypass several stages and reduce the latency at every hop. We also quantified the network communication and discussed several parameters affected the RAC.

In this paper, we used the fixed parameters for different applications to establish the RAC. The network communication exist some differences in the behaviour due to different application. We think that in our future work, the parameter adjustment will be beneficial to RAC.

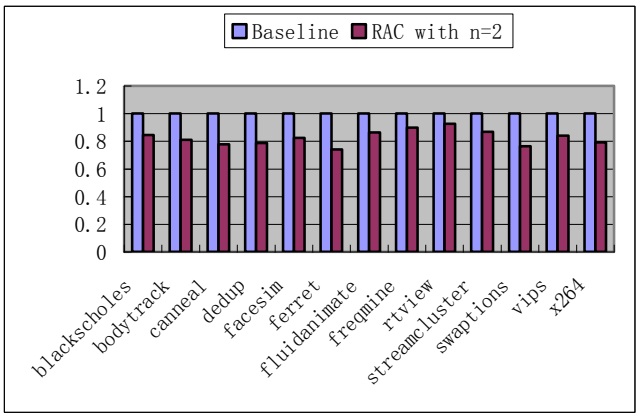


Fig. 5 Normalized average message latency

## References


[1] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*, Morgan Kaufmann, 2003.

[2] Patrick T. Gaughan and Sudhakar Yalamanchili. "Pipelined circuit-switching: a fault-tolerant variant of wormhole routing." In *Proceedings of the Symposium on Parallel and Distributed Processing*, pages148–155, December 1992.

[3] A. Kumar, L. -S. Peh, P. Kundu, and N. K. Jha. "Express virtual channels: Towards the ideal interconnection fabric," In *ISCA*, 2007, pp.150–161.

[4] N. D. E. Jerger, L.-S.Peh, and M. H. Lipasti, "Circuit-switched coherence," In *NOCS*, 2008, pp.193–202.

[5] A. Abousamra, R. Melhem, and A. Jones, "Winning with pinning in NoC," In *High Performance Interconnects,* 2009.

[6] A. Banerjee and S. W. Moore. "Flow-aware allocation for on-chip networks." In *Proceedings of the 2009 3rd ACM/IEEE International Symposium on Networks-on-Chip.*

[7] Minseon Ahn, Eun Jung Kim, "Pseudo-Circuit: Accelerating Communication for On-Chip Interconnection Networks," In *Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture*, p.399-408, 2010.

[8] M. Galles, "Scalable pipelined interconnect for distributed endpoint routing: The SGI SPIDER chip." In *Proc. Hot Interconnects 4*, Aug. 1996, pp. 141-146.

[9] Nathan Binkert, Bradford Beckmann, Gabriel Black et al. "The gem5 Simulator" In ACM *SIGARCH Computer Architecture News*, Volume 39 Issue 2, May 2011, Pages 1-7.

[10] C. Bienia, S. Kumar, Singh J.P., and Li K., "The parsec benchmark suite: Characterization and architectural implications," In *Proceedings of the 17th international conference on Parallel architectures and compilation techniques*, October 2008:72-81.

[11] H. Sullivan and T. R. Bashkow, "A Large Scale, Homogeneous, Fully Distributed Parallel Machine, I," *SIGARCH Comput. Archit. News*, vol. 5, pp. 105-117, 1977.