VP-Router: On balancing the traffic load in on-chip networks

Zicong Wang¹, Xiaowen Chen¹,²a), Junyang Zhang¹, and Yang Guo¹

¹ College of Computer, National University of Defense Technology, China
² School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Sweden
a) xwchen@nudt.edu.cn

Abstract: Along with the scaling up for network-on-chips (NoC), the network traffic grows increasingly, and generally the central region is easily to become the traffic hotspots. The problem of unbalanced traffic can lead to a part of network links becoming the bottleneck of network communication, and thus hurt the network and system performance. In this paper, we propose load-balanced link distribution method, which is intended to allocating physical channels according to the traffic load on each link. To support connecting multiple physical channels between two routers, we propose a novel concept of virtual port, and design a low-cost multi-port router called virtual port router (VP-Router). Compared to the network with traditional routers, the network with VP-Routers can effectively balance the network traffic load on links. The experiments with SPLASH² benchmarks exhibit that VP-Router performs 6.3% and 9.0% better in energy-delay-product (EDP) for 4 × 4 and 8 × 8 mesh networks respectively. As for system throughput, VP-Router improves by about 3.5% and 5.8% on average respectively.

Keywords: networks-on-chip, router, load-balanced, virtual port

Classification: Integrated circuits

References

[1] N. E. Jerger, et al.: “On-chip networks, second edition,” Synthesis Lectures on Computer Architecture, vol. 12 (2017) 1.
[2] B. S. Feero and P. P. Pande: “Networks-on-chip in a three-dimensional environment: A performance evaluation,” IEEE Trans. Comput. 58 (2009) 32 (DOI: 10.1109/TC.2008.142).
[3] Z. Wenbiao, et al.: “Link-load balance aware mapping and routing for noc,” Architecture 4 (2007) 6.
[4] S.-Y. Lin, et al.: “Traffic-balanced routing algorithm for irregular mesh-based on-chip networks,” IEEE Trans. Comput. 57 (2008) 1156 (DOI: 10.1109/TC.2008.60).
[5] P. Gratz, et al.: “Regional congestion awareness for load balance in networks-on-chip,” High Performance Computer Architecture (2008) 203 (DOI: 10.1109/HPCA.2008.4658640).
1 Introduction

Networks-on-Chip (NoC) was introduced as a promising paradigm of interconnection for modern chip multiprocessors (CMPs). Compared with the traditional bus-based interconnection, NoC can solve the performance limitations arising out of long interconnects [1,2]. In addition, NoC connects every core using packet (flit) switching scheme on a hop-by-hop basis, and thus provides a better solution for supporting the scalability of interconnection as the complexity of CMPs grows and the number of integrated cores increases.

While NoC helps to improve communication scalability, it also introduces lots of network traffic to on-chip communication, which may cause traffic load imbalance. Take an 8 × 8 mesh network as an example. In a mesh network, the central links are more easier to suffer from heavy traffic load than the peripheral links. The links can be divided into two groups according to the directions: horizontal and vertical links. For an 8 × 8 mesh network, the horizontal and vertical links can respectively have the size of 8 × 7 and 7 × 8. Fig. 1 shows the normalized traffic load across horizontal links with four typical synthetic traffic patterns (uniform-random, bit-complement, bit-rotation, and shuffle) on a heatmap scale.

We can see that in most cases the closer to the central region of network, the heavier the links are utilized.

In this paper, different from the traditional uniform interconnection between network nodes, we propose load-balanced link distribution method, which is intended to allocate physical channels according to the traffic load on each link. In our design, the links are connected by virtual port router (VP-Router), which needs no additional buffers but can support multi-port and connect multiple physical channels between two routers to widen the local bandwidth. In this way, we can balance the traffic load on each link.

---

[6] M. Ramakrishna, et al.: “Gca: Global congestion awareness for load balance in networks-on-chip,” IEEE Trans. Parallel Distrib. Syst. 27 (2016) 2022 (DOI: 10.1109/TPDS.2015.2477840).

[7] A. K. Mishra, et al.: “A case for heterogeneous on-chip interconnects for CMPs,” Proc. 38th Annual International Symposium on Computer Architecture, ISCA ’11, ACM (2011) 389.

[8] D. U. Becker: “Efficient microarchitecture for network-on-chip routers,” Ph.D. dissertation, Stanford University Palo Alto (2012).

[9] N. Binkert, et al.: “The gem5 simulator,” ACM SIGARCH Computer Architecture News 39 (2011) 1 (DOI: 10.1145/2024716.2024718).

[10] N. Agarwal, et al.: “Garnet: A detailed on-chip network model inside a full-system simulator,” Performance Analysis of Systems and Software (2009) 33 (DOI: 10.1109/ISPASS.2009.4919636).
2 Related work

Unbalanced network traffic load can lead to hotspots in network, and thus cause degradation in network throughput and system performance. Zhou et al. [3] present a routing technique for the mesh-based NoC, which can keep the link-load balance by allocating routing path. Lin et al. [4] propose a traffic-balanced routing algorithm to make traffic loads evenly spread on the networks and shorten the average paths of packets. Gratz et al. [5] point out that the traditional adaptive routing can cause inherent global load balance through greedy local strategies, so they propose region congestion awareness (RCA) adaptive routing policy. Instead of relying solely on local congestion information, RCA spreads the traffic information beyond adjacent routers so that each router has a better picture of network hotspots. Ramakrishna et al. [6] propose global congestion awareness (GCA), which is a light weight, adaptive routing algorithm based on global link state and congestion information.

In [7], Mishra et al. propose a heterogeneous network, called HeteroNoC, which apportions the resources to leverage the non-uniformity in network resources demand. From the similar perspective, our proposition strives to solve the unbalanced network traffic load problem by redesigning the network link distribution and router architecture to satisfy the bandwidth requirement of each link. Compared with HeteroNoC, our work adopts a more quantitative and accurate design method to calculate the link distribution. We propose a novel concept of virtual port, and design a low-cost multi-port router called VP-Router to support routers connected by multiple physical channels. In addition, the parallel transmission of flits need not rely on the flit combination mechanism which is used by HeteroNoC, and thus the flit transmission can be more efficient which benefits from the independence of virtual ports.

3 Load-balanced link distribution

3.1 System architecture overview

The baseline system architecture overview is shown as Fig. 2. This figure depicts a 4 × 4 mesh network. In each node, a processing element (PE) is connected to the on-chip network by a router, and a PE includes a core, a shared level two (L2) cache bank, and a network interface (NI). L2 cache is the last level cache (LLC) of the system architecture. NI is the interface of connecting the core and the router. It is important to point out that the distributed L2 cache banks are organized as static non-uniform cache architecture (NUCA), which maps each cacheline (namely the
cache block) in physical memory to each bank sequentially. This block-interleaving used for the cache system can destroy the data locality when mapping from the memory to the LLC. Therefore, the traffic pattern can to some extent be uniformly distributed on each bank due to the uniform memory-to-LLC mapping scheme. As a result, the uniform traffic can be a good generalization pattern to the network traffic. We will calculate the load-balanced link distribution on the basis of uniform traffic pattern in the next subsection.

3.2 Scheme

Our basic idea is that increasing the bandwidth for those links that suffered with heavy traffic load. Therefore, it is necessary to analyze and quantify the traffic load on each network link. Considering an \( M \times N \) mesh network with XY deterministic dimension-order routing (XY-DOR). As a result, the horizontal and vertical links respectively have the size of \( M \times (N - 1) \) and \( (M - 1) \times N \). To simulate the network traffic, we suppose each node sends a packet to each one of other nodes, and such a uniform traffic can be a comprehensive simulation for calculating the load of general traffic. We use load factor to denote the traffic load of each link, which is defined as the number of packets through a link. As a result, we can count the number of packets through each link to draw out the distribution of traffic load on network links. This procedure is described as Algorithm 1.

![Algorithm 1](image)

**Algorithm 1** Calculate the traffic load distributed on each link.

**Input:**

The size of the mesh network, \( M \times N \);

**Output:**

The load factor distribution for horizontal links, \( H_I = [h_{i,j}]_{M \times (N-1)} \);

The load factor distribution for vertical links, \( V_I = [v_{i,j}]_{(M-1) \times N} \);

1: for \((i, j) \in (0,0) \rightarrow (M-1,N-2)\) do
2: \( h_{i,j} \leftarrow 0 \)

2The minmax\((x, y)\) function that appears in the algorithm returns a vector of minimum and maximum values between the inputs \( x \) and \( y \). It is used to locate the range of links.
Without loss of generality, we can get the distributions of load factor (i.e., $H_l$ and $V_l$) for an $8 \times 8$ mesh network, which are revealed in Equation 1:

$$H_l = V_l^T = \begin{bmatrix}
112 & 192 & 240 & 256 & 240 & 192 & 112 \\
112 & 192 & 240 & 256 & 240 & 192 & 112 \\
112 & 192 & 240 & 256 & 240 & 192 & 112 \\
112 & 192 & 240 & 256 & 240 & 192 & 112 \\
112 & 192 & 240 & 256 & 240 & 192 & 112 \\
112 & 192 & 240 & 256 & 240 & 192 & 112 \\
112 & 192 & 240 & 256 & 240 & 192 & 112 \\
112 & 192 & 240 & 256 & 240 & 192 & 112 
\end{bmatrix}$$

Note that $H_l$ and $V_l$ have the same distribution pattern (i.e., $H_l = V_l^T$), except the different dimensions. In addition, observing the load factor distributions, we can find that the closer to the central region of the network, the heavier the traffic load of link. The distributions explicitly reveal that the central links have to suffer from more network traffics than the peripheral links.

To relieve the traffic load for those links that suffering from heavy traffic, we should increase their communication bandwidth. As a result, we allocate additional physical channels to these links, and the number of physical channels for each link is proportional to the load factor. The minimum load factor within all the links is $\ell$:

$$\ell = \min\{h_{i,j}\} \cup \{v_{i,j}\}$$

To reasonably allocate physical channels, the number of physical channels allocated to each link equals to the link’s load factor divided by the minimum load factor (i.e., $\ell$), and then we can get the horizontal and vertical link distributions, i.e., $H_c$ and $V_c$:
\begin{align*}
H_c &= \frac{H_l}{\ell} = \left[ \text{round}\left( \frac{h_{i,j}}{\ell} \right) \right]_{M \times (N-1)}
\end{align*}

\begin{align*}
V_c &= \frac{V_l}{\ell} = \left[ \text{round}\left( \frac{v_{i,j}}{\ell} \right) \right]_{(M-1) \times N}
\end{align*}

Based on the load factor distributions $H_l$ and $V_l$, we can get the link distribution according to Equation 3, as shown in Fig. 3a. For $4 \times 4$ mesh network, similar to the calculation of link distribution for $8 \times 8$ mesh network, the load-balanced distribution can be calculated according to Algorithm 1 and Equation 3. The load-balanced link distribution for $4 \times 4$ mesh network is illustrated as Fig. 3b.

![Fig. 3.](image)

(a) $8 \times 8$ (b) $4 \times 4$

**Fig. 3.** The link distribution for $8 \times 8$ and $4 \times 4$ mesh networks. The number attached on a link denotes the number of physical channels allocated.

### 4 VP-Router implementation

Since we allocate multiple physical channels between two routers, it is necessary to support connecting multiple physical channels. The most straightforward implementation fashion is using a multi-port router to instead the single-port router. In a multi-port router, multiple ports can exist in the same direction. However, the extra ports will bring about the additional buffers which can be a big overhead for the router implementation. Therefore, we propose *virtual port router* (VP-Router), a router architecture which supports attaching multiple ports in the same direction without additional buffers. Instead, each port does not have their own buffers but share the same buffers in the same direction. As a result, we use *virtual port* to denote the multiple ports in the same direction, because these ports do not have their own buffers but act as a real port to perform the functions of transferring packet.

Our basic idea is that we can widen the flit transferring path in the same direction but keep the number of buffers unchanged and shared between the virtual ports in the direction. To achieve this goal, it requires some modifications to the router architecture.

To support simultaneous transmission of multiple flits, the traditional credit based flow control scheme has to be improved by allowing multiple credits transmission. This is easy to implemented and just needs additional credit signals between the upstream and downstream router. In addition, the crossbar and switch
allocator (SA) should be redesigned to adapt to the simultaneous transmission of multiple flits. Here we will talk about the design details of crossbar and SA in VP-Router.

### 4.1 Crossbar design

When multiple physical channels are connected to a router’s port, the path toward the port in the crossbar must be widen. Such a case is depicted in Fig. 4, where we show the difference between the crossbar design of the baseline and VP-Router. In Fig. 4a, the baseline crossbar can allow that one flit in input port S goes to output port E as well as one flit in input port W goes to output port S. One output port can only be mapped to one input port, and vice versa. Fig. 4b shows the crossbar design in VP-Router. We assume that two physical channels are connected to port S. As a result, the datapath width along the input and output port S is doubled. Hence, two flits from different input ports W_in and L_in can be received together by the virtual output ports S_vout1 and S_vout2 (W_in + L_in → S_out). Unlike the crossbar design in HeteroNoC, VP-Router need not rely on flit combination to realize parallel transmission of flits. Therefore, two flits within a single input port but from different virtual input ports (e.g., S_vin1 and S_vin2) requesting different output ports can be sent together (S_in → N_out + E_out). Such a case benefits from the independence of each virtual port in VP-Router.

![Crossbar Design](image)

**Fig. 4.** The crossbar organization in the traditional router and VP-Router.

### 4.2 Switch allocator design

Fig. 5a shows the switch allocator of the baseline router. Suppose that the baseline router has V virtual channels, P input/output ports. r_i^v represents VC request signals from i-th input port. The allocation includes two stages, i.e., input and output arbitrations. The input arbitrations are performed in parallel by P input arbiters to select one input VC of each arbiter. g_i^v represents VC grant signals from i-th input port to indicate which VC is granted. Next, the decoder unit translates the granted VC request into the corresponding output port request, which is propagated to output arbiters. Then the output arbitrations are performed in parallel by the P output arbiters to select one input port for each arbiter. g_i^p represents the input port...
grant signals to \(i\)th output port to indicate which input port is granted. By combining \(g_v^i\) and \(g_p^i\) together, the switch allocator accomplishes the matching of the input ports to output ports.

![Fig. 5. The SA organization in the traditional router and VP-Router.](image_url)

Fig. 5b depicts a VP-Router which is connected with two physical channels in the 1st port and thus has two virtual input/output ports in the 1st port. Compared with the baseline design, the \(V:1\) arbiter in the 1st input port is divided into two separable \(V/2:1\) arbiters which are responsible for the arbitration between a half virtual channels. As for the output arbiters, the \(P:1\) arbiter in the 1st output port is divided into two separable \(P_t/2:1\) arbiters (\(P_t\) denotes the total number of ports, including virtual ports) which are responsible for the arbitration between a half input ports. For other output ports, the size of the arbiter is changed to \(P_t:1\), because it needs performing arbitration between all the ports including the virtual ports. Fig. 6 shows the detailed implementation of SA in VP-Router. Compared with the SA organization in HeteroNoC, VP-Router ensures the independence of arbitration for each virtual port to avoid duplicate arbitration, because each virtual input/output port has its own arbiter to perform arbitration as an individual physical port.

![Fig. 6. The detailed implementation of SA in VP-Router.](image_url)

### 4.3 Hardware cost

VP-Router need no additional buffers, and the hardware overhead is primarily due to the larger crossbar and the extra arbiters. The implementation is based on the
open-source NoC router RTL code [8]. The VP-Router design is synthesized in Synopsys® Design Compiler with SMIC® 130 nm process. Table I lists the logic synthesis results. Depending on the number of ports with virtual ports, VP-Router can be classified into five types (a VP-Router without virtual port is the baseline router). According to the distribution shown in Fig. 3, we can count the number of each router type, which is also given in Table I. As a result, the total area costs are 4.482 mm² and 17.931 mm² for 4 × 4 and 8 × 8 mesh networks respectively. Compared to that, the networks with VP-Routers consume 4.547 mm² and 18.763 mm² area, which are only increased by 1.4% and 4.6% for 4 × 4 and 8 × 8 mesh networks respectively. As for the overhead of power, the total power costs are increased by 1.3% and 2.9% for 4 × 4 and 8 × 8 mesh networks respectively. Although the network power is slightly increased, VP-Router can perform better than the baseline in energy-delay-product (the detailed experiment results can be seen in subsection 5.3). In addition, the baseline network can run at 500 MHz (2 ns), and the network with VP-Router runs at 476 MHz (2.1 ns), which is slightly decreased compared with the baseline network. The corresponding frequencies are operated by the network of the baseline and VP-Router in the simulation experiments.

| VP-Router type | 0 (baseline) | 1 | 2 | 3 | 4 |
|----------------|--------------|---|---|---|---|
| Area (mm²)     | 0.2801       | 0.2834 | 0.2896 | 0.2966 | 0.3021 |
| Power (mW)     | 702.928      | 713.266 | 719.375 | 726.586 | 736.238 |
| Count          | 4 × 4        | 8 | 4 | 0 | 0 |
|                | 8 × 8        | 4 | 8 | 20 | 16 | 16 |

5 Performance evaluation

5.1 Simulation methodology

We quantify the performance of VP-Router by comparing with the traditional single port router. All simulations are carried out using gem5 [9], which is a popular cycle-accurate full-system simulator with Garnet2.0 network model [10]. The evaluation is performed on mesh networks which have 1-cycle routers and 1-cycle links. In addition, both synthetic and real network traffic is run over a directory coherence protocol that has three virtual networks (LD, ST, and IFETCH) to avoid protocol deadlocks. To evaluate the effectiveness of the design with VP-Router, the performance simulations include two parts. First, we switch gem5 to network-only mode for performing network simulation with synthetic traffics, and evaluate the network performance under 4 × 4 and 8 × 8 mesh network scales. Second, we switch gem5 to full-system mode for performing full simulation with real benchmarks when running different programs under 4 × 4 and 8 × 8 mesh networks.

5.2 Synthetic traffic

Simulation with synthetic traffic is performed under the network-only mode of gem5. In network-only mode, all the simulated cores do not execute real instruc-
tions and just act as the traffic injectors to inject packets into the network. Therefore, the system is ISA-agnostic, and the memory subsystem does not work. This mode focuses on evaluating the network performance of the simulated system. The baseline network simulation configuration is illustrated by the “Network parameters” part in Table II (seen in subsection 5.3).

Fig. 7 reveals the average packet latency under $4 \times 4$ (16 cores) and $8 \times 8$ (64 cores) mesh networks with different traffic patterns. We can observe that the simulated system with VP-Router provides 13%, 58%, 38%, and 40% improvement in saturation throughput for $4 \times 4$ mesh network with uniform-random, bit-complement, bit-rotation, and shuffle traffic patterns respectively. As for $8 \times 8$ network scale, the saturation throughput is 56%, 67%, 80%, and 70% higher compared to the baseline network. The network performance improvement benefits from the alleviation of network traffics for those heavy-load links near the central region of the network. The heavier pressure of one link suffers, the more possible that it will become the bottleneck of the network, because a heavy-load link can exacerbate the network congestion and restrict the network performance.

Fig. 8 shows the normalized traffic load (per channel within each link on average) across horizontal and vertical links with uniform-random and bit-complement traffic patterns in an $8 \times 8$ mesh network on a heatmap scale for the baseline and VP-Router. All the network packets are collected under the saturation point (0.09/0.14) in 10,000 cycles. Compared with the baseline network, the traffic load on the network with VP-Routers is much more uniformly distributed. Note that the traffic load is not uniformly distributed on each link perfectly, because the load-balanced method is statically designed to promote a more balanced traffic load distribution for the majority of network traffics. In order to quantify the effect of balancing traffic load, Fig. 9 illustrates the standard deviation of traffic load across all links normalized to the baseline with different traffic patterns. We can see that the standard deviation is lower than the baseline network with all four traffic patterns, which reveals that the traffic load is more uniformly distributed on each link.

### 5.3 System performance

This subsection demonstrates the impact of VP-Router on system performance when running real benchmarks, and also observe the average network latency to quantify the network performance. We configure gem5 to be in full-system mode in this experiment. The detailed simulation configuration is illustrated as Table II.
We choose ten programs from SPLASH2 benchmarks to perform our experiments. We mainly concern three metrics, i.e., average packet latency, network energy-delay-product (EDP) and instruction per cycle (IPC). Fig. 10a demonstrates the average packet latency reduction over baseline in each program’s simulation.

Compared with the baseline, VP-Router performs 7.7% and 12.5% better in network latency on average for $4\times4$ and $8\times8$ mesh networks, and the maximum reduction can be up to 8.8% ($water$-$spatial$) and 20.3% ($water$-$spatial$) respectively.

In addition, VP-Router consistently shows lower EDP over the baseline as shown in Fig. 10b. Overall, VP-Router performs 6.3% and 9.0% better over the baseline for $4\times4$ and $8\times8$ mesh networks respectively, which reveals VP-Router can provide better network performance at very little energy consumption. Fig. 10c illustrates the IPC improvement over baseline. We can see that VP-Router improves

### Table II. The baseline simulation configuration.

| System parameters | Network parameters |
|-------------------|--------------------|
| ISA               | Alpha              |
| Number of cores   | 16/64              |
| Coherency protocol| MESI               |
| Cacheline size    | 64 bytes           |
| L1D/L1 cache size | 32 kB              |
| L2 bank size      | 256 kB             |
|                   | Network topology   | 2D mesh            |
|                   | Number of VNs      | 3                   |
|                   | Number of VCs      | 4                   |
|                   | Router/Link latency| 1 cycle             |
|                   | Routing algorithm  | XY-DOR              |
|                   | Flit size          | 64 bits             |
by about 3.5% and 5.8% on average for 4 × 4 and 8 × 8 mesh networks compared with the baseline in terms of IPC, and the maximum improvement can be up to 6.9% (raytrace) and 11.3% (radiosity). The improvement benefits from weakening the restriction of the heavy-load links, and thus promote the efficiency of network communication.

Fig. 10. The performance comparison with SPLASH2 benchmarks between the baseline and VP-Router.

6 Conclusion and future work

Along with the scaling up for the on-chip network, the network traffics are not evenly distributed on each link, and the central links are heavily utilized than the peripheral links. In this paper, we have exploited the load-balanced link distribution method and its implementation for mesh network. We propose a novel concept of virtual port, which solves the problem of connecting multiple physical channels between two routers to widen the local bandwidth and thus realize the load-balanced link distribution. Compared to the network with traditional routers, the network with VP-Routers effectively balances the traffics on network links, and achieves better network and system performance at low overhead.

In the anticipated future work, we plan to expand our approach to other network topologies.

Acknowledgments

The research is partially supported by the National Natural Science Foundation of China (No. 61502508 and No. 61572025).