# Cbufferless: A Novel Congestion Control for Bufferless Networks On-Chip

Jili Yan, Xiaola Lin
School of Information Science and Technology
Sun Yat-sen University
Guangzho, China
E-mail: yanjili750401@sina.com
E-mail: linxl@mail.sysu.edu.cn

Guoming Lai
Department of Computer Application and Technology
Hanshan Normal University
Chaozhou, China
E-mail: laigm@hstc.edu.cn

***Abstract*—In this paper, we propose a novel distributed source-throttling congestion control mechanism for bufferless NoC, called Cbufferless. Our strategy uses a novel congestion detection and control model, computing deflection rate of routing flit and throttling message injection. The congestion information can be directly obtained inside network interface (NI), which allows the mechanism to be fully distributed without requiring any transmission of global congestion information among neighbor routers and within router. Simulation results show that the proposed mechanism improves system throughput by up to ~30%, saves energy consumption by up to ~40% and keeps low message latency under congested load compared with a baseline bufferless NoC.**

***Keywords-network-on-chip; bufferless NoC; congestion control; source-throttling***

## I. INTRODUCTION

With continued developments in VLSI technology, system on-chip (SoC) enables to integrate more and more intellectual property (IP) cores [2]. However, traditional bus-based communication and other centrally-controlled architectures are unable to meet requirements of communication of SoC, such as predictable delay, low power consumption and scalability [1]. In order to address these issues, many buffered network on-chip (NoC) or on-chip network communication architectures have been recently proposed, which present probability for meeting requirements of communication of SoC both on a variety of 2D uniform and non-uniform topologies [2, 18, 19].

Nevertheless, those architectures are typically limited by tight constraints on power consumption and die area. It has been reported that, in several buffered NoC prototypes, buffered NoCs consume a substantial portion of system power, for example, ~30% in the Intel 80-core Terascale chip [5] and ~40% in the MIT RAW chip [6]. Especially, buffers of router occupy ~75% of total on-chip network area and dissipate ~22% of the router power [3, 4]. Power consumption and silicon area have been emerging as the most dominant constraints for NoC design. Therefore, it is critical to design power- and area-efficient NoC.

As an alternative design, NoC without buffers (known as bufferless NoC) has been proposed in order to gain more efficiency [7, 22]. In bufferless NoC, router buffer is completely eliminated. Output port contention in router is resolved by dropping and retransmitting, or deflection routing, packets. This design yields significant power and die area savings with minimal performance loss under low-to-medium loads. But bufferless NoC performs much worse than buffered NoC with heavy workloads, because the deflection and self-throttling natures of bufferless NoC would reduce the saturation bandwidth. Congestion is a major factor that affects overall system performance of bufferless NoC [17], and existing centralized congestion control methods for bufferless NoC are not fit. Therefore, new congestion control technology with least additional traffic and faster reaction on network congestion should be developed to increase congestion threshold and improve system performance of bufferless NoC.

In this paper, we propose a novel distributed congestion control mechanism, Cbufferless, aiming to improve system throughput, keep low message latency and save energy consumption and silicon area in bufferless NoC. We make the following contributions:

- We analyze the relationship of deflection ratio of routing message and congestion of bufferless NoC, which indicates that the deflection ratio of routing message shows directly the level of network congestion due to the nature of deflection routing.
- We develop Cbufferless, a source throttling and distributed mechanism for bufferless NoC, which uses the novel congestion detection and control strategy.

The rest of the paper is organized as follows. In Section Ⅱ, we discuss related congestion control mechanisms for buffered and bufferless NoCs. The details of the proposed congestion control mechanism are presented in Section Ⅲ. In Section IV, we evaluate our proposed algorithm along with a baseline bufferless NoC by simulations. The conclusion is given in Section Ⅴ.

## II. RELATED WORKS

### *A. Congestion Control for Buffered NoCs*

There has been an expanding body working in providing congestion control in buffered NoCs over the years. Researchers use two main strategies to avoid network congestion: congestion prevention and congestion recovery. Congestion prevention techniques require some kinds of authorization from the message destination or other where in order to inject it. The two most known techniques are based on closed-loop control, reserving the necessary resources before injecting the message, and on limiting the number of sent messages [14].

Congestion recovery techniques that are more studied to avoid network congestion are based on monitoring the network and triggering some actions when congestion is detected. In this case, the solution has two steps: congestion detection and congestion control. For congestion detection, there are two different kinds of strategies. One of them is based on measuring the waiting time of blocked messages [15], while the other uses the number of busy resources in networks [8-13]. Busy resources may be the buffer count [9, 12], the number of free virtual channels [10], the length of the source buffer [11], and a combination of these [8, 13]. Message throttling [12] has been the most frequently used action to avoid network congestion.

### *B. Congestion Control for Bufferless NoCs*

For bufferless NoC, we are aware of there are a few congestion control studies proposed [16, 17]. In [16], authors utilize stress values from neighbor switches for their own switching decisions in order to bypass the congestion area. This technology delays the occurring of congestion, but cannot alleviate congestion. In [17], George Nychis et al propose a complex congestion control mechanism to improve throughput in small size NoC and scale small size to large size NoC. However, it is centrally-coordinated with application-level awareness, and needs global congestion information, which leads to high communication overhead and hardware cost. In addition, the authors evaluate only communication-locality scenario in which data requested is in range of few hops, which depends remarkably on communication modeling of applications [21] and limits extremely the application fields.

## III. THE CONGESTION CONTROL MECHANISM FOR BUFFERLESS NOC

### *A. Analysis of Deflection Routing*

Deflection routing is a routing strategy for any network topologies. Every packet in network has preferred output(s) along which it wants to leave the router, and when possible a packet is sent along it or one of these outputs. However, two or more packets may want to leave along the same output which is referred to as a contention among packets, and then only one of the packets may be sent along the link, while the others are sent along any available outputs, even though the other links are not preferred by the packets. Using deflection routing, any packet in router may be routed to any available output due to desired output contention, which shows, on the one hand, that deflection routing has the best traffic balance feature against any other routing algorithms, on the other hand, that deflection routing can lead to network congestion early and easily. So, this feature of deflection routing demonstrates that, as packet injection rate increases, the deflection ratio will rise synchronously when network become congested.

Algorithm 1 Computing Deflection rate Algorithm

```
   at node k, k=1,……,  nxn:
1  Initiation: active = false;
   at beginning of every period, W:
2     r_avg = 0;
3    D_value = 0;
4    for W cycles do
5         Receiving flit;
6         if (!active or D_value <= 0) then
7              computing FDR of the i^th flit, r_i;
8              --D_value;
9              endif
10        if (W cycles end and (!active
               or D_value <= 0)) then
11             computing ADR, r_avg;
12             if (r_avg > threshold) then
13                active = true;
14             endif
15        endif
16   endfor
```

### *B. Congestion Control Model*

In order to cope with congestion situation of bufferless NoC, we propose a novel congestion control mechanism, which identifies the congestion occurrence and controls congestion through computing Average Deflection Rate (ADR) of the all received flits and the difference of the numbers of all the sent and received flits in every period $W$. To this end, we need to add a hop-field in the head-flit of a packet, $hp$, with $\lceil \log_2 {}^{4(N-1)} \rceil$[1] bits to store the number of $2*h$ hops, where, $h$ is the maximum of the shortest distance between any two nodes in bufferless NoC. Flit Deflection Rate (FDR) of head-flit with $2*h$ hops is high enough to detect congestion. We also need to define metrics accurately FDR, ADR and *D_value*.

We give the definition of FDR as the following equation:

$$r_i = (hp_i - h_i)/h_i \quad (1)$$

where $r_i$ is flit deflection rate of the $i^{th}$ flit, $hp_i$ is the number of hops experienced by the $i^{th}$ flit between the source and destination nodes, which is less than or equal to $2*h$, and $h_i$ is the shortest distance of the source and destination nodes of the $i^{th}$ flit.

The definition of ADR of all the received flits per $W$ cycles is given by:

$$r_{avg} = \frac{1}{k}\sum_{i=1}^{k} r_i \quad , \quad k \le W \quad (2)$$

where $r_{avg}$ is mean of flit deflection rate of $k$ flits accepted by a node in an period $W$. $W = \lambda * N$, $\lambda$ is scaling factor and is defined as $\lceil 2^{N^{1/2}} \rceil$, $N$ is size of dimension of network, and the unit of $W$ is cycle.

The definition of difference of the number of the send and received flits per $W$ cycles is given by:

$$D\_value = total\ sent\ flits - total\ received\ flits \quad (3)$$

where the t*otal sent flits* is total flits sent by a node, and

Algorithm 2 Injection Throttling Algorithm

```
   at node k, k=1,……,  nxn;
1  if (active && D_value > 0) then
2     block injection for a period, W;
3     when W cycles end,active = false;
4  else
5     allow injection for a period, W;
6     once a flit is injected,++D_value;
7  endif
```

[1] ⌈ ⌉ is ceiling operator.

the *total received flits* is total flits received by the same node in a period, *W*. The maximum value of *D_value* is equal to the number of cycles of an interval period, *W*.

Fig. 1 shows the logic overview of the congestion control mechanism for Cbufferless NoC in NI of a 3×3 mesh network. There are three modules, Flit Deflection Rate and Average Deflection Rate Computing Unit (FDR/ADR CU), *D_value* Computing Unit (D_value CU) and Sending and Receiving Management Unit (S/R MU). The FDR/ADR CU calculates the FDR of each received flit in period *W* and ADR of all received flits at end of period *W*, after that, this module informs the congestion status to S/R MU. *D_value* CU module computes the difference of the numbers of the sent and received flits in every period *W* and sends it back to S/R MU. According to the information, S/R MU module estimates the congestion level and determines whether or not the node should be throttled in next period *W*.

### C. Congestion Detection

In our method, we use the novel and effective congestion detection metric, ADR. Each node computes FDR of each flit received during a fixed time interval *W*. After the end of period *W*, ADR is computed and transferred to S/R MU to compare with a predefined congestion threshold *threshold*. If ADR is smaller than *threshold*, the network is in the healthy state. Otherwise, congestion is detected (details seen in algorithm 1), and if *D_value* > 0 at the same time, the control strategy will be invoked in the next period *W* (details seen in algorithm 2).

### D. Congestion Control

In our method, we use a distributed algorithm to throttle message injection of nodes contributing congestion. The every node decides independently whether to throttle message injection or not in Cbufferless NoC. Once any of nodes has detected network congestion and *D_value* > 0 at the same time, the node will block the injection of flits in the next period, *W*. Otherwise, the node can still inject flits into network (pseudo-code described as algorithm 2). In the throttling period, all nodes contributing network congestion stop flit injection, while after a throttling period, the throttled nodes can resume injecting flits into network. Finally, network can run under congestion point through this dynamic throttling mechanism.

## IV. Performance Evaluation

### A. Evaluation Methodology

We evaluate the effectiveness of our congestion control mechanism by using *noxim* [20], which is a flit-accurate simulator based on *systemC*. Within each simulation for a synthetic traffic, there is a warm-up period of 1,000 cycles. Thereafter, the volume of 100,000 flits has been injected or 100,000 cycles have been expired. As for other parameters, the scaling factor, $\lambda$, is $\lceil 2^{N^{1/2}} \rceil$, and the *threshold*, is $\lceil 1/N^{1/2} \rceil$. In our simulation experiments, baseline bufferless refers to the FLIT-BLESS [7] and Cbufferless is FLIT-BLESS with our proposed congestion control mechanism.

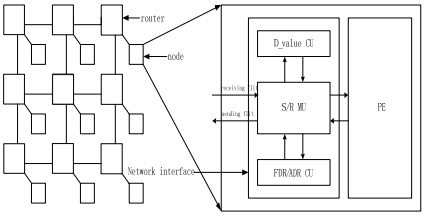


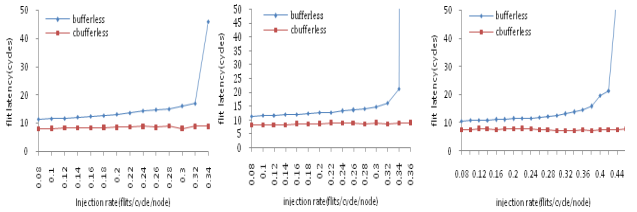

Fig. 1 logical view of Congestion Control Module

We evaluate our proposed strategy with three standard synthetic traffic patterns: uniform random (UR), transpose (TP) and bit-reverse (BR). All synthetic traffic patterns employ a uniform random injection process.

### B. Simulation results

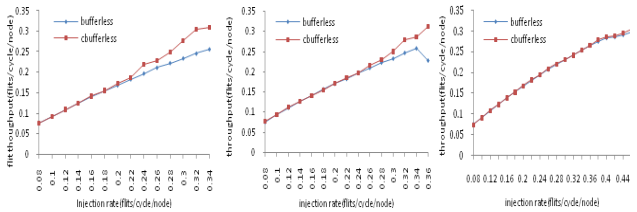
In this subsection, we compare Cbufferless with the FLIT-BLESS [7], baseline bufferless, in terms of average flit latency, average throughput and energy consumption with a 4×4 2D mesh bufferless NoC.

Fig. 2 gives the results of flit latency with different traffic patterns. From the figure, we can observe that proposed mechanism can keep lower average latency (cycles) under different traffic patterns. This gain is achieved because our mechanism monitors the deflection of flits in network, which can active congestion control to throttle flit injection as soon as average deflection rate of flits exceeds the threshold. While injection rate increases, average latency of baseline bufferless increases accordingly.




a) UR traffic pattern b) TP traffic pattern c) BR traffic pattern

Fig. 2 average latency

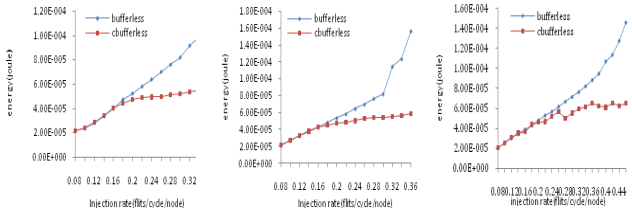



a) UR traffic pattern b) TP traffic patternc) BR traffic pattern

Fig. 3 flit throughput

Fig. 3 depicts system throughput of baseline and Cbufferless under different traffic patterns in the 4x4 network. As injection rate increases, throughput of Cbufferless compared with that of baseline bufferless improves up to ~30%, before network becomes heavy congested. With injection rate increasing, network is becoming increasingly congested, then the number of deflection of flits in network increases, which leads throughput to decrease. While Cbufferless can alleviate network congestion through monitoring deflection of flits

and blocking flit injection timely, and consequently maintain higher throughput.




a) UR traffic pattern b) TP traffic pattern c) BR traffic pattern

Fig. 4 energy consumption

It is showed in Fig. 4 that energy consumption is function of injection rate under different traffic patterns in bufferless NoC. Fig. 4 shows energy consumption of Cbufferless compared with the baseline bufferless NoC. The maximum energy saving (up to ~40%) is achieved before network becomes heavy congested. with injection rate increase, in baseline NoC, network becomes more and more congested and then average number of hops experienced by flits in network rises, which leads to more energy consumption. On the contrary, Cbufferless NoC can keep lower average latency, thus energy is saved.

### *C. Implementation Issues*

Implementation of Cbufferless scheme incurs quite low overhead and our congestion control strategy can be implemented totally in network interface, while the network node and the router of Bufferless NoC [7] need not to be changed. In network interface, operations of sending and accepting flits, and our congestion control strategy can work in parallel. For hardware, the implement of Cbufferless need only a few of hardware according to analysis in section 3. In future work, we will make accurate hardware cost and implementation evaluation.

## V. CONCLUSIONS

In this paper, we have proposed a novel congestion control mechanism for bufferless NoC which detects network congestion by monitoring deflection information of routing flits and handles it by throttling dynamically the injection rate of flits. Extensive simulations show that the proposed mechanism can improve system throughput up to ~30%, and reduces energy consumption up to ~40% as well as keeping low flit latency. Employing baseline FLIT-BLESS NoC as benchmark, only a few extra hardware cost and bandwidth overhead is brought by Cbufferless NoC.

## REFERENCES


[1] M. Horowitz, R. Ho, and K. Mai, "The future of wires," Proc. IEEE,vol. 89, no. 4, pp. 490–504, April 2001.

[2] W.J. Dally, B. Towles, "Route packets, not wires: on-chip interconnection networks," In Proceedings of Design Automation Conference, pp. 684–689, Jun. 2001

[3] P. Gratz, C. Kim, R. McDonald, S. W. Keckler, and D. Burger, "Implementation and evaluation of on-chip network architectures," In International Conference on Computer Design, pp. 477–484 , Oct. 2006.

[4] S.R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, "An80-tilesub-100-WTeraFLOPS processorin65-nm CMOS," Solid-State Circuits, *IEEE Journal of* 43 (1) , pp. 29–41, 2008. doi:10.1109/JSSC.2007.910957

[5] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar. "A 5ghz mesh interconnect for a teraflops processor," IEEE Micro, pp. 51-61, Sept-Oct. 2007.

[6] M.B. Taylor, W. Lee, J. Miller, D. Wentzlaff, I. Bratt, B. Greenwald, "Evaluation of the Raw microprocessor: An exposed-wire-delay architecture for ILP and streams," in Proc. of the 31th annual International Symposium On Computer Architecture., June 2004.

[7] T. Moscibroda and O. Mutlu, "A case for bufferless routing in on-chip networks," in Proc. of the 36th annual International Symposium On Computer Architecture, 2009.

[8] P. Gratz, B. Grot and S. W. Keckler, "Regional congestion awareness for load balance in Networks-on-Chip," In Proc. of 14th International Symposium on HighPerformance Computer Architecture, pp.203-214. Feb. 2008.

[9] Hu, J, & Marculescu, R, "DyAD smart routing for networks-on-chip, "In Proceedings of the 41st design automation conference, pp. 260–263. Jun. 2004.

[10] Baydal, E., López, P., & é Duato, "A family of mechanisms for congestion control in wormhole networks," J IEEE Transactions on Parallel and Distributed Systems, 16 (9), 772–784. 2005.

[11] J. W. van den Brand, C. Ciordas, K. Goossens and T. Basten, "Congestion controlled best effort communication for networks on chip," In Proc. of the Conf. on Design, Automation and Test in Europe , pp: 948–953, 2007.

[12] M. Thottethodi, A. Lebeck, and S. Mukherjee, "Self-tuned congestion control for multiprocessor networks," In International Symposium on High-Performance Computer Architecture, pp: 107–118, 2001.

[13] L. P. Tedesco, T. Rosa, F. Clermidy, N. Calazans, F. G. Moraes, "Implementation and evaluation of a congestion aware routing algorithm for networks-on-chip," In Proc of the 23rd symposium on Integrated circuits and system design. pp: 91-96, Sept. 2010.

[14] U. Y. Ogras and R. Marculescu, "Prediction-based flow control for network-on-chip traffic," . In International Conference on Design Automation, pp. 839–844, July 2006.

[15] [15] Nan Jiang, Daniel U. Becker, George Michelogiannakis, and William J. Dally, "Network congestion avoidance through Speculative Reservation," In International Symposium on High-Performance Computer Architecture, pp. 25-29 Feb. 2012.

[16] E. Nilsson, M. Millberg, J. Oberg, and A. Jantsch, "Load distribution with the proximity congestion awareness in a network-on-chip," In Proc. of the Conf. on Design, Automation and Test in Europe, pp. 1126–1127, Mar. 2003.

[17] G. Nychis, C. Fallin, T. Moscibroda, S. Seshan, and O. Mutlu, "Congestion Control for Scalability in Bufferless On-Chip Networks," SAFARI Technical Report 2011-003, July 2011.

[18] J. Kim, J. Balfour, and W. J. Dally, "Flattened butterfly topology for on-chip networks," In International Symposium on Microarchitecture, pp.172–182, Dec. 2007.

[19] Y. Kao, N. Alfaraj, M. Yang, H. Jonathan Chao, "Design of High-Radix Clos Network-on-Chip," in Proc. the 4th ACM/IEEE International Symposium on Networks-on-Chip, pp.181-188, May 2010.

[20] Noxim: http://sourceforge.net/projects/noxim.

[21] K. L. Johnson, "The Impact of Communication Locality on LargeScale Multiprocessor Performance," Proceedings of the 19th annual international symposium on Computer architecture. pp. 392-402, 1992.

[22] C. Fallin, C. Craik, and O. Mutlu, "CHIPPER: A low- complexity bufferless deflection router," In Proceedings of the 17th International Symposium on High Performance Computer Architecture, pp. 144–155. February 2011.