Designing Nonblocking Networks With a General Topology

BEY-CHI LIN1 AND CHIN-TAU LEA2

1Department of Applied Math, National University of Tainan, Tainan 70005, Taiwan
2Department of ECE, The Hong Kong University of Science and Technology, Hong Kong

Corresponding author: Bey-Chi Lin (beychi@mail.nutn.edu.tw)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Contract MOST 110-2221-E-024-004, and in part by Hong Kong Innovation and Technology Funds (ITF) under Grant ITS-356-12 and Grant UIM-338.

ABSTRACT Conventional theory for designing strictly nonblocking networks, such as crossbars or Clos networks, assumes that these networks have a centralized topology. Many new applications, however, require networks to have a distributed topology, like 2-D mesh or torus. In this paper, we present a new theoretical framework for designing such nonblocking networks. The framework is based on a linear programming formulation originally proposed for solving the hose-model traffic routing problem. The main difference, however, is that the link bandwidths in our problem must be discrete. This makes the problem much more challenging. We show how to apply the developed theorems to tackle the problem of designing bufferless NoCs (networks-on-chip). The proposed bufferless NoCs are deadlock/livelock-free and consume significantly less power than their buffered counterparts. We also present a multi-slice technique to reduce node capacity variations. This can make the proposed NoC architecture more cost efficient in a VLSI (very large-scale integration) implementation. In addition, we offer a detailed delay/throughput performance evaluation of the proposed bufferless NoC in the paper.

INDEX TERMS Nonblocking networks, networks-on-chip (NoCs), bufferless, hose model.

I. INTRODUCTION

Conventional nonblocking networks, such as crossbars and Clos networks, are designed for a centralized topology. However, new applications require networks with a distributed topology. One example is networks-on-chip (NoCs) [1]–[3] which often use a mesh or torus topology [4]–[6]. Most conventional NoCs adopt a buffered architecture, but deadlocks and the inability to handle adversarial traffic patterns are well-known problems in buffered NoCs [7]. A buffered NoC has another major drawback: buffers consume a large amount of power and silicon real estate [8]–[11]. [8] showed that removing buffers in its proposed architecture could lower power consumption by 40% and reduce silicon area by 70%. A buffered architecture is also incompatible with emerging optical NoC technologies, such as silicon photonic rings [12], [13], which are bufferless by nature.

However, bufferless NoCs have their own problems. For example, one such architecture is based on deflecting rout-
of a nonblocking NoC with a distributed topology, such as mesh, ring, or torus. Filling that gap is the focus of this paper.

Here, we will present a theoretical framework for the design of nonblocking networks with a general topology. We use bufferless NoCs to illustrate the applications of this new type of network, as being nonblocking is essential for such NoCs. Our contributions are summarized as follows:

1. We develop the theory for designing nonblocking networks with a general topology. In other words, the entire network performs like a crossbar switch.

2. We show how to apply the theory to optical and electronic NoC design. Such bufferless NoCs are deadlock/livelock-free and congestion-free.

The rest of the paper is organized as follows. Section II presents the theoretical foundation of the paper. Section III presents an application of the theory to bufferless NoC design. Section IV presents a multi-slice technique to balance node capacities, and Section V compares the performance of the new NoC with that of a conventional buffered NoC. We conclude our discussion in Section VI.

II. THEORETICAL FOUNDATION

The theory for designing a nonblocking network with a general topology is given below. Various topologies, such as 2-D mesh [17] and torus [7], have been proposed for NoCs. Our theory is general and applies to all of these topologies.

A. HOSE MODEL NETWORKS

A network can be considered as a graph $G(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges (links). We list several additional notations used in our formulation in Table 1.

The theoretical foundation of the design of a nonblocking bufferless NoC is related to work on the hose-model traffic pattern [18], [19]. Traditionally, the traffic demand specification is given in the form of a traffic matrix $\{t_{ij}\}$, where $t_{ij}$ is the amount of traffic from node $i$ to node $j$, where $i, j \in \mathcal{V}$.

TABLE 1. Notations defined under hose model networks.

| Notations | Definitions |
|-----------|-------------|
| $T$:     | The traffic matrix of the network. $T = \{t_{ij}\}$, where $t_{ij} \in (0, 1)$. If there is a connection request from node $i$ to node $j$, $t_{ij} = 1$, and otherwise, 0, where $i, j \in \mathcal{V}$. |
| $\alpha$: | The routing parameter. If traffic from node $i$ to node $j$ is routed through link $e$, $\alpha_e = 1$, and otherwise, 0, where $i, j \in \mathcal{V}, e \in \mathcal{E}$. |
| $\beta$: | The capacity of link $e \in \mathcal{E}$. |
| $\mu_e$: | The cost-scaling constant related to link $e$. |
| $\gamma$: | The maximum node capacity, in discrete channels, of a node in the network. |

But in a hose-model network, the traffic demand constraint is given in the form of row and column sums:

$$\sum_j t_{ij} \leq \alpha_i, \quad \forall i \in \mathcal{V},$$

$$\sum_i t_{ji} \leq \beta_i, \quad \forall i \in \mathcal{V},$$

(1)

where $(\alpha_i, \beta_i)$ represent the amount of ingress and egress traffic admissible at processor $i$ (see Fig. 1). In practice, $\alpha_i$ and $\beta_i$ are usually set to the capacity, denoted by $\gamma$, of the link connecting a processor and the NoC. Note that in the model above, traffic loads are considered continuous. This obviously is not true in a nonblocking circuit network.

A network, if designed according to (1), will be congestion-free as long as the total amount of ingress and egress traffic of each node does not exceed the constraints specified by $(\alpha_i, \beta_i)$, regardless of the traffic demand pattern $\{t_{ij}\}$. In such a network, a new flow from node $i$ to $j$ can always be added to the network as long as the total ingress traffic at node $i$ is no more than $\alpha_i$ and the total egress traffic at node $j$ is no more than $\beta_j$. Thus, there is no need to check internal links’ statuses. In other words, a new flow will never be blocked inside the network, provided that the ingress and egress nodes have the capacity to accommodate the flow. This property is likened to the nonblocking property in switching. Such designed networks are thus called nonblocking networks in [19]–[21].

Prior works on the hose-model assume continuous bandwidths. However, this is certainly not true in a discrete bandwidth circuit switching network, where setting up a path will consume one channel from each link along the path. In the following, we therefore show how to use the hose-model concept to design a nonblocking circuit switching network.

B. INITIAL FORMULATION

As mentioned in Sec. I, our goal is to design a general-topology network that performs like a crossbar switch. So we set $\gamma = 1$ in our discussion below. The formulation for designing a minimum-cost nonblocking network is given below in (2). Note that $C_{\text{total}} = \sum_{e \in \mathcal{E}} c_e \cdot \mu_e$ is the total cost of the network, where $c_e$ is the capacity of link $e$, and $\mu_e$ is a cost-scaling constant related to link $e$. Implicitly, we assume that the silicon area, which includes nodes and links, for

FIGURE 1. In a nonblocking network, each edge node has an ingress and an egress traffic limit. As long as the traffic admitted to the network does not exceed these limits, the network will never be congested.
implementing link capacity \( c_e \) is linearly proportional to \( c_e \).

\[
\begin{align*}
\min \quad & C_{\text{total}} \\
\text{s.t.} \quad & \sum_{e \in \Gamma^+(v)} x_{ij}^e - \sum_{e \in \Gamma^-(v)} x_{ij}^e = 0, \quad \forall i, j, v \in V, \ v \neq i, j, \\
& \sum_{e \in \Gamma^+(v)} x_{ij}^e - \sum_{e \in \Gamma^-(v)} x_{ij}^e = 1, \quad \forall i, j, v \in V, \ v = i, \\
& \sum_{e \in \Gamma^+(v)} x_{ij}^e - \sum_{e \in \Gamma^-(v)} x_{ij}^e = -1, \quad \forall i, j, v \in V, \ v = j, \\
& \sum_{i,j} i \cdot x_{ij}^e \leq c_e, \quad \forall e \in E, \ \forall \{t_{ij}\} \in T \in D, \\
& \sum_{e} c_e + \sum_{e} c_e \leq M_{\text{nd}}, \\
& x_{ij}^e, t_{ij} \in \{0, 1\}, \quad c_e \geq 0, \quad \forall e \in E, \ \forall i, j \in V.
\end{align*}
\]  

For a given \( V \) and \( E \), (2b–d) are for the flow conservation constraint, and (2e) means that the total traffic routed through a link is smaller than the link capacity. In (2e), \( D \) is the set of all legal traffic matrices; that is, all traffic \( \{t_{ij}\} \) is subject to the following constraint:

\[
\begin{align*}
\sum_{j \in V} t_{ij} & \leq 1, \quad \forall i \in V. \\
\sum_{i \in V} t_{ij} & \leq 1, \quad \forall j \in V.
\end{align*}
\]

\( M_{\text{nd}} \) in (2f) is given. Although this constraint is not required, we use it to limit the maximum capacity of each node. Equivalently, this can reduce the node capacity variance. To select the value of \( M_{\text{nd}} \), we can set \( M_{\text{nd}} \) to infinity in the beginning and solve the formulation. Based on the derived maximum node capacity, we can reduce \( M_{\text{nd}} \) gradually and see its impact on the total cost and the node capacity variance. More discussion will be provided later, in Sec. IV.

The formulation of (2), however, cannot be solved directly because there are too many traffic matrices \( T \) satisfying (3), and they all need to be included in (2). In the following, we show how to tackle this problem.

**Theorem 1:** Let \( t_{ij} \in [0, 1] \) (i.e., \( t_{ij} \in \mathbb{R} \)). This relaxation will not change the result of (2).

**Proof:**

(i) Assume \( t_{ij} \in [0, 1] \). Let \( x_{ij}^e \) denote the derived routing and \( C_{\text{min}} \) the solution of (2).

(ii) Assume \( t_{ij} \in [0, 1] \). Let \( x_{ij}^e \) be the derived routing and \( C_{\text{min}} \) the solution of (2).

(iii) Assume \( t_{ij} \in [0, 1] \) and that the network uses routing \( x_{ij}^e \) defined above. Let \( C_{\text{min}}^\prime \) be the minimum total capacity required to support all traffic patterns.

We are going to prove the following three states:

(a) \( C_{\text{min}}^\prime \leq C_{\text{min}}^\prime \) is true.

(b) \( C_{\text{min}}^\prime \geq C_{\text{min}}^\prime \) must hold.

(c) \( C_{\text{min}}^\prime \leq C_{\text{min}}^\prime \) is true.

With (a–c) above, it is trivial to see that \( C_{\text{min}}^\prime = C_{\text{min}} \) must hold.

We now show that all three statements are true.

(a) This statement is trivially true because under \( t_{ij} \in [0, 1] \), the optimal routing is \( x_{ij}^e \), not \( x_{ij}^e \). Thus the derived \( C_{\text{min}}^\prime \) is not optimal, which means \( C_{\text{min}}^\prime \leq C_{\text{min}}^\prime \) will hold.

(b) Every valid traffic matrix in (2e) under \( t_{ij} \in [0, 1] \) is also valid under \( t_{ij} \in [0, 1] \), which means that the constraint equations under \( t_{ij} \in [0, 1] \) are only a subset of the constraint equations under \( t_{ij} \in [0, 1] \). Thus the result derived from the formulation under \( t_{ij} \in [0, 1] \) is worse than the result from the former formulation under \( t_{ij} \in [0, 1] \), which means \( C_{\text{min}}^\prime \leq C_{\text{min}}^\prime \) must hold.

(c) Assume the network uses routing \( x_{ij}^e \) derived from (i). Note that the network is intended to be a crossbar-like nonblocking network, and is a single-path network since \( x_{ij}^e \in [0, 1] \). Let \( <p, q> \) denote the path from source node \( p \) to destination node \( q \). Let \( \Psi \) be the subset of source nodes and \( \Phi \) the set of destination nodes of all such paths \( <p, q> \). Note that if \( p \in \Psi \) and \( q \in \Phi \), it does not imply \( <p, q> \) must pass through \( e \). For example, if \( <1, 3> \) and \( <2, 4> \) are all the paths passing through \( e \), then \( \Psi = \{1, 2\} \) and \( \Phi = \{3, 4\} \). Obviously \( <2, 3> \) does not pass through \( e \).

We now construct a bipartite graph \( G' \) with \( \Psi \) and \( \Phi \). Let \( p \in \Psi \) and \( q \in \Phi \). We add an edge \( pq \) between \( p \) and \( q \) if \( <p, q> \) passes through link \( e \). Let \( z_{pq} \) be the weight assigned to \( pq \). A legal traffic pattern (i.e., a traffic matrix satisfying (3)) routed through \( e \) can be considered as a matching in the bipartite graph \( G' \), and the amount of traffic in the pattern equals the total weights of the matching. Assume \( z_{pq} \in [0, 1] \). The total weights can be calculated as

\[
\begin{align*}
\max \quad & \sum_{p, q \in V} z_{pq} \\
\text{s.t.} \quad & \sum_{q \in \Phi} z_{pq} \leq 1, \\
& \sum_{p \in \Psi} z_{pq} \leq 1, \\
& z_{pq} \in [0, 1].
\end{align*}
\]

In (iii), we assume the same routing \( x_{ij}^e \) is used, but \( z_{pq} \in [0, 1] \). Under this condition, a source node can be connected to multiple destination nodes simultaneously and vice versa. The maximum amount of traffic that can be routed through \( e \) can now be calculated through the following fractional matching formulation:

\[
\begin{align*}
\max \quad & \sum_{p, q \in V} z_{pq} \\
\text{s.t.} \quad & \sum_{q \in \Phi} z_{pq} \leq 1, \\
& \sum_{p \in \Psi} z_{pq} \leq 1, \\
& z_{pq} \in [0, 1].
\end{align*}
\]

Comparing (4a) and (4b), we can see that (4a) is an integral matching formulation and (4b) a fractional matching formulation. From graph theory [22], the maximum integral
matching equals the maximum fractional matching. Therefore, the link capacity of $e$ derived from (i) can support any traffic pattern under the assumption of (iii). Thus, $C_{\text{min}}^* \leq C_{\text{min}}$ must hold.

### C. FINAL FORMULATION

With Theorem 1, we can assume $t_{ij} \in [0, 1]$ and derive the same result. Once we assume $t_{ij} \in [0, 1]$, we can use the technique described in [23] to remove the requirement of exhaustively listing $T$ in (2).

Theorem 2: Assume $t_{ij} \in \mathbb{R}$ in (2). Routing $x^e_{ij}$ is feasible (i.e., satisfying (2e)) if and only if there exist non-negative $\pi_e(i)$ and $\lambda_e(j)$ for all $e \in E$ and $i, j \in V$ such that

\[
\sum_{i \in V} \pi_e(i) + \sum_{j \in V} \lambda_e(j) \leq c_e, \quad \forall e \in E.
\]

\[
x^e_{ij} \leq \pi_e(i) + \lambda_e(j), \quad \forall i, j \in V, \forall e \in E.
\]

**Proof:** The proof is similar to that in [23]. Thus, it is omitted here. ■

Note that in the above, $\pi_e(i)$ and $\lambda_e(j)$ are the auxiliary variables introduced by the duality transformation of the following linear programming (LP) formulation:

\[
\begin{align*}
\text{max} & \quad \sum_{i, j \in V} t_{ij} \cdot x^e_{ij}, \quad \forall e \in E \\
\text{s.t.} & \quad \sum_{j \in V} t_{ij} \leq 1, \quad \forall i \in V, \\
& \quad \sum_{i \in V} t_{ij} \leq 1, \quad \forall j \in V.
\end{align*}
\]

With Theorem 2 and $\pi_e(i)$ and $\lambda_e(j)$, we derive the final mixed-integer LP formulation as follows:

\[
\begin{align*}
\min \quad & C_{\text{total}} \quad (5a) \\
\text{s.t.} & \quad \sum_{e \in \Gamma^+(v)} x^e_{ij} - \sum_{e \in \Gamma^-(v)} x^e_{ij} = 0, \quad \forall i, j, v \in V, \quad v \neq i, j, \quad (5b) \\
& \quad \sum_{e \in \Gamma^+(v)} x^e_{ij} - \sum_{e \in \Gamma^-(v)} x^e_{ij} = 1, \quad \forall i, j \in V, \quad v = i, \quad (5c) \\
& \quad \sum_{e \in \Gamma^+(v)} x^e_{ij} - \sum_{e \in \Gamma^-(v)} x^e_{ij} = -1, \quad \forall i, j \in V, \quad v = j, \quad (5d) \\
& \quad \sum_{i \in V} \pi_e(i) + \sum_{j \in V} \lambda_e(j) \leq c_e, \quad \forall e \in E, \quad (5e) \\
& \quad x^e_{ij} \leq \pi_e(i) + \lambda_e(j), \quad \forall e \in E, \quad \forall i, j \in V, \quad (5f) \\
& \quad \sum_{e \in \Gamma^+(v)} c_e + \sum_{e \in \Gamma^-(v)} c_e \leq M_{nd}, \quad (5g) \\
& \quad x^e_{ij} \in \{0, 1\}, \pi_e, \lambda_e, c_e \geq 0, \quad \forall e \in E, \quad \forall i, j \in V. \quad (5h)
\end{align*}
\]

In (5), only $x^e_{ij}$ is discrete and the other variables are real.

### III. APPLICATION OF NONBLOCKING THEORY: BUFFERLESS NoCs

In this section, we use bufferless NoCs as an example application of the nonblocking theory presented in Section II. Although the theory is for designing a network that performs like a crossbar switch, the bufferless NoC application, as shown below, is for a packet switch.

### A. ROUTER NODE DESIGN

A packet in NoCs is usually divided into flits (flow control digits), which form the basic data-transmission and buffering units at the link layer. The term “phit” (physical unit), on the other hand, represents the number of bits that can be transferred in parallel in a single cycle. As with many NoC designs, we assume that the flit size is equal to the phit size. In other words, the entire flit is transmitted in one clock cycle.

If an NoC uses a buffered approach [2], [3], different flits can be held in different routers, with the head flit holding the routing information. Packets from different input ports may be headed for the same output port, and the head flit of a packet may be blocked in a router. When that happens, flow control signals will be sent to downstream routers, and the subsequent flits may be buffered in different routers. However, the NoC proposed in this paper uses a bufferless approach. Each router only holds one flit. After one clock cycle, the entire packet moves one flit toward the destination. The entire NoC is nonblocking in the sense that, if a destination is free, a packet from a source station can definitely reach the destination node without being blocked inside the switch (more details are provided below).

The hardware implementation of the router node of our bufferless NoC is quite simple. It only comprises a crossbar and a controller. No virtual channels, flow control, deadlock prevention mechanisms, or adaptive routing are needed. Since more than 60% of NoCs adopt a mesh or torus topology, we focus on these two in our discussion below.

In a bufferless NoC, a router node of a 2-D mesh or torus topology has at most four bi-directional links plus one link connecting to the local processor. The link connecting to the local processor contains only one channel, but the other links may have multiple channels. Each channel consists of two sub-channels, DATA and ACK:

DATA: For sending data. The width of a channel is the same as the width of a flit.

ACK: A 1-bit sub-channel for sending the ACK signal back to the source. Its direction is the opposite of the DATA sub-channel.

The first bit of each flit is the VALID bit (Fig. 2b) — 1 for valid and 0 for idle flits. A change from 0 to 1 in the VALID bit indicates that the first flit of a packet arrives, and a change from 1 to 0 means the end of the current packet transmission.

There is a unique path between any two processors. The specification of the entire path is given in the head flit(s) of a packet. This style of routing is usually called source routing. It requires 2 bits to specify the routing direction of each node (north, south, east, west), and the “hop count” field indicates how many routers the packet needs to go through. The “hops passed” field indicates how many hops the packet has already passed through (this field is incremented by each router). When hop count = hops passed, the arriving packet
will be handed to the local processor. For a large NoC, the path information may be carried by more than one flit. The format for the second flit will be the same. In this case, when hop count = hops passed in each flit, the flit will be popped out by a router and the next flit will be used for routing.

A channel has two states: Idle and Busy. When a new packet arrives, the controller of the crossbar in Fig. 2 will select an Idle channel and set up the state of the crossbar accordingly. The packet is sent out after a one-cycle delay. The selected output channel is marked Busy, and the corresponding 1-bit ACK channel (in the reverse direction) will also be turned on. If no channel of the selected link is available, the packet is dropped. Because the NoC is nonblocking, packet dropping means that the destination is already occupied.

When a source processor sends out a packet, it also starts a timer. If no ACK (from the ACK channel) arrives before the timer expires, the transmission fails (blocked at the destination, not inside the network), and the source processor will stop the transmission immediately by changing the VALID bit from 1 to 0. This will set all the reserved channels along the path to the Idle state again. The penalty for a failed transmission is the time-out interval, which is quite small compared to a packet length (see discussion below). A failed transmission will be sent out a short while later. No overhead is incurred if the transmission is a success.

The time-out interval is determined by the number of hops of a path. This information is available because there is a unique path to each destination processor. Each packet is delayed by 1 flit cycle per node. If the total hop count is $k$, the length of the time-out interval is around $k + \Delta$ cycles, where $\Delta$ is the time for the ACK to travel back through the ACK channel (which is an analog channel and does not contain flip flops). $\Delta$ may be larger than 1 flit cycle due to the capacitance loading of the ACK channel, but should still be small. To simplify our discussion, we assume it is 2. In a 2-D torus NoC with $n^2$ processors, the average hop count is around $n$ clock cycles. So for a 36-processor multicore system, the time-out interval is around 8 flit cycles. In other words, a processor will waste 8 flit cycles for a failed transmission. This overhead will be further reduced by another factor, $K$, in our final architecture discussed in Sec. IV, where $K$ is the number of slices used to implement the NoC. If $K = 4$, then the average path setup overhead is about 2 conventional flit cycles.

From the discussion above, it is easy to see that the bufferless architecture is deadlock-free and livelock-free because a failed transmission does not hold up any network resources. Also, several lines of a flit in conventional NoCs are used for carrying flow control signals and virtual channel (VC) numbers. This type of overhead does not exist in the proposed bufferless architecture.

**B. COST FUNCTION AND RESULTANT TOPOLOGY**

As can be seen in Fig. 2, a bufferless router has no buffers and is much simpler than in a buffered NoC. The silicon area required for a bufferless NoC is dominated by the total link capacity cost, $C_{\text{total}}$, which is expressed as

$$\text{Total link cost} = C_{\text{total}} = \omega \sum_{e \in E} c_e d_e,$$  \hspace{1cm} (5a')

where $\omega$ is the wire pitch, $c_e$ is the capacity of link $e$, and $d_e$ is the physical length of link $e$. Since $\omega$ is fixed for a given fabrication technology, it can be dropped from the formulation by setting $\omega$ to 1.

We can use (5) where (5a) is replaced with (5a') to design a bufferless NoC. Consider the conventional mesh network
shown in Fig. 3a. The formulation of (5) where (5a) is replaced with (5a′) produces many solutions, and one is given in Fig. 3b, where $c_e$ is labeled on each unidirectional link. Since this resulting topology is a tree, we can easily verify that this solution is nonblocking. Note that there are many other solutions which look very different from the one shown in Fig. 3.

One issue with the result is that the node capacity variance is quite large. Every node in a bufferless NoC is just a crossbar. To simplify the VLSI implementation, it is preferable to lay out just one node (the largest node) and use it for all nodes. But this can lead to inefficient utilization of silicon area if the NoC has a large node capacity variance. This issue is tackled in the next section.

**IV. MULTI-SLICE ARCHITECTURE**

To tackle the node capacity variance, we will present a multi-slice architecture below. As in Sec. III, we focus on torus networks, which are north-south, east-west symmetric (see Fig. 4). This symmetry is important for the multi-slice architecture presented.

**A. NODE CAPACITY VARIANCE**

Node capacity variance can lead to inefficient VLSI implementation. Let $n_{\text{max}} \times n_{\text{max}}$ be the largest node size. The efficiency, in terms of fewer crosspoints wasted, of using the layout of this node for all nodes can be represented as

$$\frac{1}{M} \sum_{i=1}^{M} \frac{n_i^2}{n_{\text{max}}^2},$$

(6)

where $M$ is the total number of nodes and $n_i^2$ represents the number of crosspoints required in a particular node. We call the formula of (6) the node capacity variance indicator: the larger the value, the smaller the variance.

**B. MULTI-SLICE ARCHITECTURES**

In a multi-slice architecture, we divide the network into $K$ independent slices, and the width of each channel in a slice is only $1/K$ the width of the channel of a single-slice architecture. We then combine $K$ independent slices into the final solution. The advantage of a multi-slice architecture is three-fold: (i) it reduces node capacity variance, (ii) it reduces the path set-up overhead, by a factor of $K$ in a $K$-slice architecture, and (iii) it reduces the impact of a link failure (a single link failure only removes $1/K$ of the total bandwidth in a $K$-slice network).

To minimize the node capacity variance in a multi-slice NoC, how to combine the slices becomes important. Let A,
B, C, and D represent the four slices of the solution shown in Fig. 5a. Each slice is a torus, although the capacities of some links are zero. We can horizontally or vertically shift or rotate each slice and then combine them. The result is still a torus network, due to the symmetry possessed by a torus network.

Denote the shift and rotation operation of a slice by a two-tuple \((O_S, R_S)\), where \(O_S\) represents the shift and \(R_S\) the rotation (in clockwise) operation. For an \(n \times n\) network, we use the two-tuple \((c, r)\) for \(0 \leq c, r \leq n-1\) to represent \(O_S\) where the network is shifted \(c\) columns (or \(r\) rows) to the right (or downward). \(R_S\) can assume four values: 0°, 90°, 180°, or 270°. An example is given in Fig. 5, where the slice of a 4 \(\times\) 4 torus network given in Fig. 5b is obtained by changing \(((2, 0), 0°)\) of the slice given in Fig. 5a. By changing the \((O_S, R_S)\) of each slice, we intend to find the locations of the four slices such that the node capacity variance is minimized (Fig. 5c). Note that the torus network given in Fig. 5c combines the four slices \(S \in \{A, B, C, D\}\), each of which is obtained by changing the \((O_S, R_S)\) of the torus given in Fig. 5a, where \((O_A, R_A) = ((0, 0), 0°), (O_B, R_B) = ((2, 0), 0°), (O_C, R_C) = ((1, 2), 0°), (O_D, R_D) = ((3, 2), 0°)\).

For 8 \(\times\) 8 networks, testing all possible locations of different slices can be too time consuming. Thus, we use a coarse-resolution approach to tackle this problem. For example, the \(x\) or \(y\) coordinates range from 0 to 7 in an 8 \(\times\) 8 network. We can divide the \(x\) and \(y\) axis into four regions with boundary points at 0, 2, 4, 6, and 8. As a result, we only need to consider 16 square regions on the plane, each square containing four points of the original plane (see Fig. 6). We randomly select a point as the location of a slice in that region.

### C. COMPARISON IN TERMS OF NODE CAPACITY VARIANCE

We first consider a single-slice architecture. Table 2 presents the results of a 2-D torus network (Fig. 4), where the total link cost is computed from (5) with \(d_c = 1\), in which (5a) is replaced with (5a’) for a link between adjacent nodes and with \(d_c = n-1\) for a link between the first and the last node in an \(n \times n\) torus. We can see that the node capacity variance derived by (5) is large. The total link cost of a nonblocking bufferless NoC in this case is even smaller than that of a conventional buffered NoC. Note that the node complexity of the former is much lower and performance much better (see Sec. V).

For multi-slice architectures, the results of four-slice 4 \(\times\) 4 and 6 \(\times\) 6 NoCs are given in Tables 3 and 4, respectively, where the total link cost is computed from (5) with \(d_c = 1\), in which (5a) is replaced with (5a’). A node in a multi-slice NoC contains multiple nodes of different slices. Although they are independent, we can still combine them into one crossbar switch (with some configuration logic inside, of course). An interesting result is provided by the 4 \(\times\) 4 torus NoC. With \(M_{nd} = 24\), all nodes in the bufferless 4-slice torus NoC (Fig. 5c) and the conventional buffered torus NoC (Fig. 4) have the same capacity (Fig. 7). Furthermore, the two networks have the same total link cost.

### V. DELAY/THROUGHPUT PERFORMANCES

In addition to computing, telecom applications are gaining more importance for multi-core systems [24]. For such applications, a trace-based simulation, that is, a system simulation performed by looking at traces of program execution or system component access with the purpose of performance prediction, is meaningless. Therefore, we use traffic-pattern-based simulation to compare the performance of the proposed bufferless NoC with that of a conventional buffered NoC.

---

**TABLE 2.** Total link cost and node capacity variance of 1-slice 4 \(\times\) 4 torus Networks.

| Topology                        | Total link cost | Node capacity variance indicator |
|---------------------------------|-----------------|----------------------------------|
| Conventional buffered torus     | 96              | 100.00%                          |
| Nonblocking bufferless torus derived from (5), \(M_{nd} = 24\). | 64              | 28.99%                           |
| Nonblocking bufferless torus derived from (5), \(M_{nd} = 20\). | 80              | 43.50%                           |

**TABLE 3.** Total link cost and node capacity variance of 4-slice 4 \(\times\) 4 torus Networks.

| Topology                        | Total link cost | Node capacity variance indicator |
|---------------------------------|-----------------|----------------------------------|
| Conventional buffered torus     | 96              | 100.00%                          |
| Nonblocking torus derived from (5), \(M_{nd} = 24\). | 96              | 100.00%                          |

**TABLE 4.** Total link cost and node capacity variance of 4-slice 6 \(\times\) 6 torus NoCs.

| Topology                        | Total link cost | Node capacity variance indicator |
|---------------------------------|-----------------|----------------------------------|
| Conventional buffered torus     | 240             | 100.00%                          |
| Nonblocking torus derived from (5), \(M_{nd} = 54\). | 360             | 70.52%                           |
In the simulation, we use a 4-slice architecture. A node processor of such an NoC can transmit 4 packets simultaneously. Although the transmission time of each packet is 4 times that of the prior packet, the packet for the 1-slice architecture, transmission time, the overall latency, as shown below, will be dominated by the queuing delay in the network. The routing for the NoC has already been described in Sec. II.

Many sophisticated adaptive routing schemes and congestion avoidance and deadlock prevention algorithms have been proposed for buffered NoCs. But many are very complicated. We choose a simple XY routing for a buffered network and compare its performance with our bufferless network. This XY routing is the simplest among all deadlock-free routing schemes. The simulation details of the buffered NoC are the following:

1. It uses XY routing by first routing a packet in the x dimension and then in the y dimension.
2. Credit-based flow control is deployed between neighboring nodes.
3. Each link contains four virtual channels, and 16 flit buffers are dedicated to each virtual channel.

To avoid head of line (HOL) blocking, virtual output queues (VOQs) using a round robin are implemented in each processor. Each VOQ holds the packets for the same destination, while a processor serves its VOQs in a round-robin fashion. Since the bufferless NoC is nonblocking, a packet from a processor of such an NoC can transmit 4 packets simultaneously.

Packets arrive in a Poisson stream and the lengths follow a negative exponential distribution. The mean value of the packet lengths obviously depends on the application, and there is no consensus on this issue (for example, it is assumed to be 8 flits in [25] and between 128 and 1024 flits in [26]). For our simulation below, we assume the mean of the packet lengths is 8 flits [25]. The performance of the bufferless NoC presented below will be better than an NoC whose mean packet length is larger than 8 flits.

In Fig. 8, we compare the delay/throughput performances of the 4 × 4 torus NoCs (4-slice) under uniform and non-uniform traffic loads. For the latter, we use the four non-uniform traffic patterns discussed in [27], and their definitions are given in Table 5. If we use the knee points in these curves as the reference points for comparing the maximum throughput, Fig. 8a shows that the throughput gain of the proposed bufferless NoC over the conventional buffered NoC is about 21% (0.75 versus 0.62) under a uniform traffic load, and can be as high as 100% under a non-uniform load (Fig. 8b–d). These results show that the total link capacity required for a bufferless network (see Sec. III) is about 1.8 times that of the conventional buffered NoC (4-slice). Finally, it should be pointed out that in a buffered NoC, several lines of each flit must be used to carry flow control signals and VC numbers. This overhead is ignored in our comparison.

Fig. 9 shows the delay/throughput performances of the 8 × 8 torus NoCs (4-slice). It shows that the throughput of the bufferless NoC is about 1.4 times that of the conventional buffered NoC when traffic is uniform (see Fig. 9a). But it should also be pointed out that the total link capacity required for a bufferless network (see Sec. III) is about 1.8 times that of a buffered NoC (this does not consider node complexity). For non-uniform loads, Fig. 9b and Fig. 9c show that the capacity gain of the new bufferless NoC can be as high as 400% that of the conventional buffered NoC. Considering the simplicity

| Name           | Description                                                                 |
|----------------|-----------------------------------------------------------------------------|
| Uniform Random (UR) | Each packet is sent from the source node to a randomly selected node.    |
| Bit Complement (BC) | Node (x, y) sends to (n-x-1, n-y-1).                                       |
| Transpose (TP)    | Node (x, y) sends to (y, x).                                              |
| Tornado (TOR)     | Node (x, y) sends to (x+n/2-1, y).                                         |
FIGURE 8. Delay performance of a 4 × 4 bufferless torus NoC (4-slice) and of a buffered torus NoC with XY routing: (a) uniform traffic, (b) bit complement traffic, (c) tornado traffic, and (d) transpose traffic. The delay unit is the average packet transmission time in a buffered NoC.

FIGURE 9. Delay performance of 8 × 8 bufferless torus NoC (4-slice) and of the conventional 8 × 8 buffered torus NoC with XY routing: (a) uniform traffic, (b) bit complement traffic, (c) tornado traffic, and (d) transpose traffic. The delay unit is the average packet transmission time in a buffered NoC.
of a bufferless node, these figures clearly demonstrate the advantage of the new approach.

VI. CONCLUSION

In this paper, we have presented the theory for the construction of nonblocking networks with a general topology. We use bufferless NoCs to demonstrate an application of the new theory. The presented bufferless NoC is nonblocking, deadlock-free, livelock-free, and congestion-free. It has a simple node design and a throughput close to 100%. Its performance is traffic pattern independent, which greatly simplifies task mapping for a multi-core system. The bufferless nature of the new architecture can achieve great savings in terms of power consumption and silicon area.

The developed theorems can also be applied to a wavelength selective switch (WSS)-based optical network, which also adopts a distributed topology. A workstation can use a length selective switch (WSS)-based optical network, which greatly simplified. Blocking WSS-based network in which the RWA problem is often a blocking network, and its routing and wavelength assignment (RWA) problem is NP-hard. Using the theorems developed in the paper, we can derive a non-blocking WSS-based network in which the RWA problem is greatly simplified.

ACKNOWLEDGMENT

The author Bey-Chi Lin would like to thank Prof. Zhe XuanYuan of UIUC and Dr. Vincent Wing-Hei Luk for giving suggestions for the programming.

REFERENCES

[1] W. J. Dally and B. Towles, “Route packets, not wires: On-chip interconnection networks,” in Proc. Design Autom., New York, NY, USA, 2001, pp. 684–689.

[2] T. Bjørregaard and S. Mahadevan, “A survey of research and practices of network-on-chip,” ACM Comput. Surv., vol. 38, no. 1, pp. 1–51, 2006.

[3] S. Hesham, J. Rettkowski, D. Goehringer, and M. A. A. El Ghany, “Survey on real-time networks-on-chip,” IEEE Trans. Parallel Distrib. Syst., vol. 28, no. 5, pp. 1500–1517, May 2017.

[4] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani, “A network on chip architecture and design methodology,” in Proc. IEEE Comput. Soc. Annu. Symp. VLSI. New Paradigms, Oct. 2002, pp. 117–124.

[5] P. Guerrier and A. Greiner, “A generic architecture for on-chip packet-switched interconnections,” in Proc. Design, Autom. Test Eur. Conf. Exhib., 2000, pp. 250–256.

[6] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, “Performance evaluation and design trade-offs for network-on-chip interconnect architectures,” J. VLSI Syst. Comput. Archit., vol. 41, no. 1, pp. 83–109, May 2005.

[7] W. Dally and B. Towles, Principles and Practices of Interconnection Networks. Burlington, MA, USA: Morgan Kaufmann, 2004.

[8] T. Moscibroda and O. Mutlu, “A case for bufferless routing in on-chip networks,” in Proc. ISCA, Austin, TX, USA, 2009, pp. 196–207.

[9] G. Nychis, C. Fallin, T. Moscibroda, S. Seshan, and O. Mutlu, “Congestion control for scalability in bufferless on-chip networks,” SAFARI, Macworld U.K., Tech. Rep. 003–2011, Jul. 2011.

[10] A. Kaki, A. Louri, and J. Wang, “Design of energy-efficient channel buffers with router bypassing for network-on-chips (NoC),” in Proc. 19th Int. Symp. Qual. Electron. Design, 2009, pp. 826–832.

[11] E. Ofori-Atta and M. A. Oyegama, “A survey of recent contributions on low power NoC architectures,” in Proc. Comput. Conf., London, U.K., 2017, pp. 1086–1090.

[12] C.-T. Lea and B.-C. Lin, “A new approach to the wavelength nonuniformity problem of silicon photonic microrings,” J. Lightw. Technol., vol. 29, no. 17, pp. 2552–2559, Sep. 14, 2011.

[13] P. Guo, W. Hou, L. Guo, Q. Yang, Y. Ge, and H. Liang, “Low insertion loss and non-blocking microring-based optical router for 3D optical network-on-chip,” IEEE Photon. J., vol. 10, no. 2, Apr. 2018, Art. no. 7901010.

[14] P. T. Wolkotte, G. J. M. Smit, G. K. Rauwerda, and L. T. Smit, “An energy-efficient reconfigurable circuit-switched network-on-chip,” in Proc. 19th IEEE Int. Parallel Distrib. Process. Symp., Apr. 2005, pp. 1–8.

[15] N. Enright Jerger, M. Lipasti, and L.-S. Peh, “Circuit-switched coherence,” IEEE Comput. Archit. Lett., vol. 6, no. 1, pp. 5–8, Jan. 2007.

[16] K. Goossens, J. Dielissen, and A. Radulescu, “A ethereal network on chip: Concepts, architectures, and implementations,” IEEE Des. Test. Comput., vol. 22, no. 5, pp. 414–421, Sep/Oct. 2005.

[17] E. Salminen, A. Kulmala, and T. Hamalainen, “Survey of network-on-chip proposals,” in Proc. OCP-IP, 2008, pp. 1–13.

[18] R. S. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, and J. E. V. der Merwe, “A flexible model for resource management in virtual private networks,” in Proc. ACM SIGCOMM, San Diego, CA, USA, Aug. 1999, pp. 95–108.

[19] J. Chu and C.-T. Lea, “New architecture and algorithms for hose-model VPN construction,” IEEE Trans. Netw., vol. 16, no. 3, pp. 670–679, 2008.

[20] Y. Xuan and C.-T. Lea, “Network-coding multicast networks with QoS guarantees,” IEEE/ACM Trans. Netw., vol. 19, no. 1, pp. 265–274, Feb. 2011.

[21] J. A. Fingerhut, S. Suri, and J. S. Turner, “Designing least-cost non-blocking broadband networks,” J. Algorithms, vol. 24, no. 2, pp. 287–309, Aug. 1997.

[22] A. Schrijver, Combinatorial Optimization: Polyhedra and Efficiency. Berlin, Germany: Springer, 2003.

[23] D. Applegate and E. Cohen, “Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental trade-offs,” in Proc. ACM SIGCOMM, Karlsruhe, Germany, 2003, pp. 313–324.

[24] Proceeding of 2013 Data Center Conference. Accessed: Feb. 5, 2003. [Online]. Available: http://www.linleygroup.com/events/event.php?num=21

[25] J. Lee, C. Nicopoulos, S. J. Park, M. Swaminathan, and J. Kim, “Do we need wide flits in networks-on-chip?” in Proc. IEEE Comput. Soc. Annu. Symp., Aug. 2013, pp. 1–7.

[26] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, “Cost considerations in network on chip,” Integration, vol. 38, no. 1, pp. 19–42, Oct. 2004.

[27] A. Singh, W. J. Dally, A. K. Gupta, and B. Towles, “GOAL: A load-balanced adaptive routing algorithm for torus networks,” ACM SIGARCH Comput. Archit. News, vol. 32, no. 2, pp. 194–205, 2003.

BEY-CHI LIN received the B.S. degree in mathematics education from National Taichung Teachers College, Taiwan, in 1999, and the Ph.D. degree in applied mathematics from National Chiao Tung University, Taiwan, in 2005. She was a Research Assistant with the Department ECE, The Hong Kong University of Science and Technology, from 2005 to 2008. She is currently a Professor with the Department Applied Math, National Tainan University, Taiwan. Her research interests include switching network analysis and combinatorial mathematics.

CHIN-TAU LEA received the B.S. and M.S. degrees in electrical engineering from National Taiwan University, in 1976 and 1978, respectively, and the Ph.D. degree in electrical engineering from the University of Washington, Seattle, in 1982. He is a Professor Emeritus with The Hong Kong University of Science and Technology, in 1996. Prior to that, he was with AT&T Bell Labs, from 1982 to 1985 and with the Georgia Institute of Technology, from 1985 to 1995. His research interests include general area of switching and networking.