

# A Distributed Multi-Point Network Interface for Low-Latency, Deadlock-Free On-Chip Interconnects

Dongkook Park, Chrysostomos Nicopoulos, Jongman Kim, N. Vijaykrishnan, Chita R. Das

Department of Computer Science and Engineering, Pennsylvania State University

University Park, PA, USA, 16802

{dpark, nicopoul, jmkim, vijay, das}@cse.psu.edu

**Abstract**—The notion of a Network-on-Chip (NoC) is rapidly gaining a foothold as the communication fabric in complex System-on-Chip (SoC) architectures. Scalability is the NoC's most valuable asset, which makes it ideal for larger designs. However, increasingly diminishing feature sizes have rendered the interconnect as the primary bottleneck in terms of both latency and power consumption in on-chip systems. It is, therefore, imperative to optimize the network infrastructure to maximize performance. Research has primarily focused on architectural improvements within the router and the development of deadlock avoidance/recovery schemes. The latter tend to rely on fairly complex algorithms, which are sometimes infeasible to implement in NoCs due to their resource-constrained nature. In this paper, we propose a new NoC topology and architecture which injects data into the network using four sub-NICs (Network Interface Controllers), rather than one NIC, per node. It is shown that this scheme achieves significant improvements in network latency and energy consumption with only negligible area overhead and complexity over existing architectures. In fact, in the case of MESH network topologies, the proposed scheme provides substantial savings in area as well, because it requires fewer routers. Cycle-accurate simulation validates our assertions. Most importantly, it is also shown that this implementation is inherently deadlock-free, thus eliminating the need to rely on specialized, resource-hungry algorithms for deadlock avoidance.

**Keywords**—Network-on-Chip; Router Design; Deadlock; Topology

## I. INTRODUCTION

With the advent of complex system-on-chip (SoC) architectures, the interconnection among several functional modules inside a chip poses several challenges. The increasing complexity of SoCs, along with reduced feature sizes in deep sub-micron technologies, have elevated the on-chip interconnect into a critical bottleneck in meeting the performance and power consumption budgets of the design. In the forthcoming 65 nm regime, up to 77% of the delay will be attributed to the interconnect [1]. Recently, the concept of packet-based on-chip communication networks, also known as network-on-chip (NoC) architectures, has been introduced [2,3] as a successor of the shared bus architecture to address the challenges of increasing interconnect complexity. However, unlike traditional macro-networks, NoCs entail several design challenges stemming from their inherent resource constraints; namely, area and power limitations. These limitations gravely affect the choice of routing algorithms and protocols.

To maximize the performance of a system, prior works have concentrated on the design of efficient routers [4,5,9,21], development of fault-tolerant [14,20] architecture, and deadlock avoidance/recovery schemes [15,16]. In this work, we approach the issue from the perspective of the network topology itself. We propose a Multiple Entry Point (MEP) topology that provides both low-latency and inherent deadlock freedom without relying on the architecture of the routers or any proprietary and complex algorithms. In fact, the proposed topology is algorithm agnostic, and can afford its benefits to any algorithm chosen. Hence, the proposed MEP implementation is not a new router design; it is a new network topology, which can be applied to existing router designs with minimal modifications to further improve the on-chip interconnect performance. Unlike other topologies

adopted in NoC architectures, which employ a Node:Router mapping of either 1:1 (MESH or k-ary n-cube) or N:1 (clustered topology, where several nodes are connected to a single router), the proposed MEP topology uses N:M mapping. This significantly reduces the overall message latency. It also works as a framework for deadlock-free interconnection networks. Furthermore, the fact that each router connects to more than one Processing Element (PE) eliminates the need for edge routers in MESH network topologies, as it will be demonstrated in this paper. For the connection among multiple PE-nodes and routers, the generic Network Interface Controller (NIC) is divided into four smaller sub-NICs which are interconnected within a PE-node. Hence, messages in a router can be forwarded to another router via these additional paths, reducing network latency and alleviating contention.

The proposed topology and NIC/router architectures were evaluated through cycle-accurate simulation and hardware synthesis implementations, and compared to those of generic topologies and architectures, in terms of latency and power consumption. Simulation results show that the proposed MEP topology and accompanying architectures can significantly reduce overall latency in both deterministic (X-Y) and minimal adaptive routing, especially at higher message injection rates. Specifically, the average latency and Energy-Delay-Product (EDP) are reduced by approximately 20% and 50%, respectively, over existing implementations. The removal of restrictions on the use of Virtual Channels (VC) for deadlock-free routing in the MEP topology model enables more efficient and versatile use of VCs, resulting in substantially reduced network contention.

This paper is organized as follows. First, a short background of existing work is given in Section II. Detailed descriptions of the proposed MEP topology, NIC/router architectures and the deadlock recovery scheme are then presented in Section III. Simulation results are shown in Section IV, and, finally, Section V concludes the paper.

## II. RELATED WORK

Low network latency can be achieved in several ways and the most common approach is to reduce the number of pipeline stages within the router [4,5,9]. These papers used next-hop routing that enabled the Routing (RT) stage to be executed in parallel with other pipeline stages, such as Virtual channel Allocation (VA) and Switch Arbitration (SA). They also used speculative SA to reduce latency, which assumed that a flit will always win the arbitration. Overhead occurs only when this speculation is wrong (mis-speculation), in which case the flit will go through a non-speculative arbitration in the next cycle. A single-stage router implementation proposed in [9] uses a doubly speculative pipeline (speculation in both VA and SA), running the RT, VA and SA stages simultaneously to minimize routing latency.

Latency savings can also be achieved by distributing traffic as evenly as possible over the whole network. Since network behavior is not always predictable, adaptive routing algorithms, rather than deterministic, are more suitable for this purpose. But adaptive routing algorithms usually suffer from deadlocks and, thus, additional efforts to provide either deadlock recovery or avoidance are necessary. Even deterministic routing algorithms suffer from deadlock, depending on the topology. For example, the X-Y routing algorithm cannot avoid deadlock in a TORUS topology and [6] uses virtual channels to overcome this problem. For deadlock recovery, they specified the use of VCs such that cyclic dependency cannot occur. But all these solutions rely entirely on complex routing algorithms which make the

\* This research was supported in part by NSF grants CCR-0208734, EIA-0202007, CCF-0429631, CNS-0509251, CRI-0454123, CAREER 0093085, and SRC grant 00541.

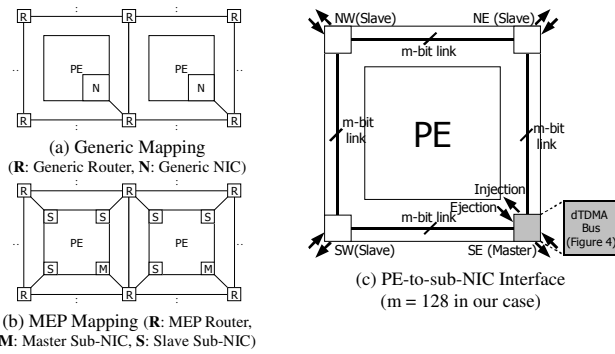


Figure 1. PE-Router Mapping & PE Interface

router implementation more complicated, and, thus, area and power hungry. A software-based deadlock recovery scheme is proposed in [7], but it requires a software layer to temporarily absorb deadlocked messages out of the network. Providing a software layer may not be feasible in NoCs due to the increased implementation complexity. Our proposed topology/architecture offers deadlock freedom with only minimal requirements from the routing algorithm. This will enable the routing algorithms to concentrate on traffic distribution rather than deadlock freedom, and will provide much more flexibility in algorithm design. Also, as our topology/architecture can adopt most on-chip router designs currently available (with only minimal modifications), we can expect greater latency savings when coupled with advanced router designs.

### III. PROPOSED TOPOLOGY & ROUTER MODEL

In this section, we describe the details of our proposed network topology, NIC architecture and router architecture design, respectively, in the following subsections. Alongside the NIC design, we also provide an interface between a PE (core) and an intra-PE network (sub-NIC-to-sub-NIC connections) that surrounds the PE, so as to enable the use of the proposed multiple entry topology with any general core/PE without any modification.

To facilitate a detailed insight of the proposed topology's area and power budgets, and how they fare against a generic implementation, both architectures were implemented in structural Register-Transfer Level (RTL) Verilog and then synthesized in Synopsys Design Compiler using a TSMC 90 nm cell library. The resulting designs operate at a supply voltage of 1 V and a clock speed of 500 MHz. Both dynamic and leakage power estimates were extracted from the synthesized implementations. A flit width of 128 bits was used for all implementations.

#### A. Topology

Traditional NoC topologies such as MESH (MESH-GEN, GEN stands for Generic) and TORUS (TORUS-GEN) assume that a PE is connected to a router as shown in Figure 1 (a). In our proposed topology model (MESH-MEP and TORUS-MEP), a PE is connected to four adjacent routers at the same time, as shown in Figure 1 (b).

In an  $n \times n$  network, both TORUS and MESH topologies require  $n \times n$  routers. In our proposed architecture, the TORUS topology, i.e. TORUS-MEP, still requires  $n \times n$  routers. However, in the MESH topology, i.e. MESH-MEP,  $(n-1) \times (n-1)$  routers are required instead, as shown in Figure 2 (b). This is a direct result of the advantageous N:M

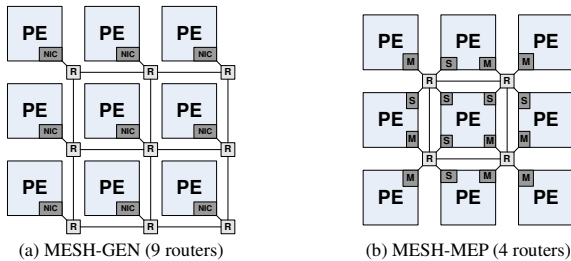


Figure 2. Number of routers needed in MESH topologies

mapping of the MEP topology. This mapping allows more than one PE to be connected to a single router (Figure 1 (b)). Hence, the edge routers in a MESH topology are no longer required with the MEP mapping. On the contrary, a generic MESH requires one router per PE (Figure 1 (a)). The proposed architecture, therefore, requires  $(2n-1)$  fewer routers in an  $n \times n$  MESH topology. This result is illustrated in Figure 2, which compares the two topologies using a  $3 \times 3$  MESH network. The reduction in number of routers amounts to considerable area and energy savings, with no sacrifice in performance. It is important to note that in the MESH-MEP topology, the size of the NICs should be adjusted based on the location of each node. This will be discussed in the following section.

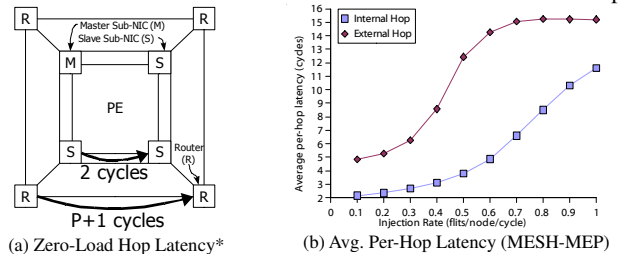
The fundamental differentiator between the MEP topology and generic implementations is the existence of two, instead of one, types of network hops in MEP. While traditional topologies transfer messages through inter-router hops, MEP employs both inter-router hops (called external hops) and sub-NIC-to-sub-NIC hops (called internal hops). This characteristic is illustrated in Figure 3 (a). Even though the total number of hops from a source to a destination is the same in both the MEP and generic architectures, the former requires fewer external hops. For example, in a generic mapping, if we assume that a destination node is on a north-west direction, while the injection point is in the south-east corner, as shown in Figure 1 (a), a message should traverse two more external (inter-router) hops than in the MEP mapping case. External hops induce more network latency than internal hops, because of increased contention and router complexity. Since most routers use pipelined architectures, a single external hop includes router pipeline delay plus link traversal delay, which sum up to a total of  $P+1$  cycles, where  $P$  is the number of pipeline stages in the router. Internal hops, on the other hand, require two clock cycles, and suffer from significantly reduced contention, as will be explained in Section III-B. Consequently, reducing the number of external hops equates to reduced latency in the network. This assertion is validated in Figure 3 (b), which shows the average per-hop loaded latencies (i.e. including contention) for both internal and external links against injection rate. Clearly, internal hops are significantly faster than external hops. Hence, the key advantage of the MEP topology is that the overall network latency can be markedly decreased by reducing the average number of external hops from a source node to a destination node through the use of more efficient internal hops.

However, since most generic PE cores are not able to inject messages in four different directions, and modifying the PEs for this purpose would be extremely costly, we need to provide an interface between a traditional, single-point injection PE, and the surrounding routers. This will allow multiple entry points to the network without any PE modifications. The proposed interface will be described in the following section.

Another advantage of the MEP topology is its automatic deadlock recovery, which inherently places minimum requirements on the routing algorithms employed. Deadlock is avoided regardless of the routing algorithm chosen, as long as the algorithm follows a set of minimal requirements which will be presented in Section III-D in detail.

#### B. NIC Design

An interface between a PE and the attached NoC router is required to facilitate communication between the PE and the on-chip



(a) Zero-Load Hop Latency\*

\* Zero-Load Hop Latency:

(1) Internal Hop (Sub-NIC-to-Sub-NIC):

= 1 (dTDM bus arbitration) + 1 (link traversal) = 2 cycles

(2) External Hop (Inter-Router):

=  $P$  (# router pipeline stages) + 1 (link traversal) =  $P+1$  cycles

Figure 3. Average Per-Hop Latency ( $P=3$  in our case)

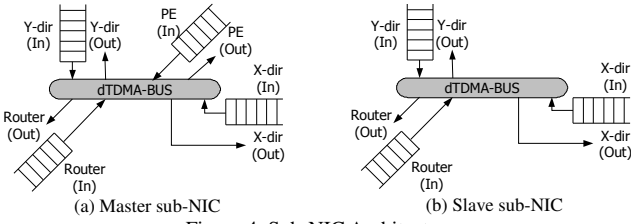


Figure 4. Sub-NIC Architecture

communication fabric. This interface is provided by the Network Interface Controller (NIC), as shown in Figure 5 (a). The NIC is responsible for the packetization and de-packetization of data, and may also include an error detection/correction module [17,18]. Most of its area, though, is consumed by data buffers for packets injected into and ejected from the network.

To provide multiple entry points to a PE core, we need to provide as many NICs in a single node, so that each NIC communicates with a corresponding router. To that extent, a generic NIC is divided into four sub-NICs which are then connected via internal links, as shown in Figure 1 (c). A sub-NIC can be either a ‘Master’ or a ‘Slave’, depending on its location. The sub-NIC located at the PE injection/ejection point is the master sub-NIC, while the remaining three are slaves. Note that the proposed architecture simply divides one NIC into four smaller ones by redistributing existing resources; the additional overhead is kept to a minimum. The packetization and de-packetization modules are located in the Master sub-NIC, and the buffers are equally distributed to all sub-NICs.

A PE injects messages into the master sub-NIC, and then, depending on their destinations, messages are either injected to the router connected to the master sub-NIC, or forwarded to appropriate slave sub-NICs through the forwarding path around the PE. Similarly, all the ejecting messages coming into each sub-NIC are forwarded to the master sub-NIC, where they are ejected to the PE.

The architecture of the master/slave sub-NICs is shown in Figure 4. Each of the three slave sub-NICs has three input/output ports (Router, Y-dir, X-dir), while the master sub-NIC has four ports (Router, Y-dir, X-dir, PE). Each of these ports has an associated input buffer, as shown in Figure 4. The output ports are directly connected to the input buffer of the ports of the adjacent sub-NIC. For communication among these ports within a sub-NIC, we adopted the transaction-less dTDMA (dynamic Time Division Multiple Access) bus architecture [8]. In dTDMA, a bus arbiter dynamically grows and shrinks the number of timeslots to match the number of active clients. When a client needs to transmit data, it requests a timeslot from the arbiter. The arbiter then modifies the timeslot configuration of the system before the next clock cycle. When a client is finished communicating, the arbiter de-allocates a timeslot in a similar manner. This method of dynamic allocation always produces the most efficient timeslot configuration, making the dTDMA bus nearly 100% bandwidth efficient [8]. Furthermore, its single-hop communication nature and lack of transaction-style arbitration, allows for low and predictable latencies. Each client requires a compact transceiver module to interface with the bus.

Therefore, by using dTDMA buses in the four sub-NICs, an internal hop (i.e. sub-NIC-to-sub-NIC) requires only two clock cycles, assuming no contention: one for bus arbitration within the sub-NIC, and one for internal link traversal. Moreover, contention is significantly lower in the internal links compared to the external ones, since the former are used primarily by traffic destined to, or originating from, the local PE. On the other hand, external links are used by all network traffic.

As four sub-NICs are connected in a circular manner, deadlocks can occur within a PE’s forwarding path network. This necessitates the use of X-Y routing to prevent intra-PE deadlocks.

The dTDMA bus arbiter should know where to forward each message; this can be done by padding 3 control bits to the message that indicate the desired direction. Table 1 shows the details of these control bits. For example, assuming the sub-NIC placement of Figure 1 (c), a message to be injected through the NW slave sub-NIC will have control bits ‘011’ (set by master sub-NIC), indicating that it should be forwarded to both X and Y direction and then it should be sent to the router. Likewise, a message to be ejected coming through a SW slave

Table 1. Message Control Bits

Bit	Value 0	Value 1
0 (LSB)	No forwarding to X-dir	Forward to X-dir
1	No forwarding to Y-dir	Forward to Y-dir
2 (MSB)	To Router (Injection into network)	To PE (Ejection from network)

sub-NIC will have control bits ‘101’ (set by ejecting router), indicating that it should be forwarded to the X direction and then ejected to the PE.

One of the most powerful attributes of this topology is that this forwarding path can also serve as an alternate path for messages that are blocked for a long time either by network congestion or by a deadlock. A blocked message is simply absorbed into the intra-PE network through a sub-NIC, then forwarded to an appropriate sub-NIC, and finally injected again in the network just like a new message is injected from a node. In most cases, this alternate path is much faster than normal inter-router links in terms of latency, especially if the message is forwarded to both X and Y directions, since forwarding between two sub-NICs requires fewer cycles than routing between two routers, as explained in Section III-A. This will alleviate traffic delays in highly congested areas significantly.

To minimize the amount of area and power overhead incurred by the MEP architecture, the sum of all sub-NIC buffer sizes was chosen to be equal to the size of a single generic NIC buffer. The area and power budget of a NIC is dominated by the buffers – the area and power consumption of the control logic is negligible compared to the buffers. Hence, since the buffer size of the MEP architecture is the same as a generic NIC, the overall area and power overhead of the proposed architecture is minimal, as shown in Table 2 (both the generic NIC and the combined MEP NIC have a total buffer space of 128 flits). As previously mentioned, the NIC buffer size affects the queuing latency in the network. We experimented with various NIC buffer sizes and concluded that the relative trends in the total average latencies for both the MEP and generic architectures remained unchanged. Therefore, due to space limitations, we only show results with a NIC buffer size of 128 flits for both the MEP and generic implementations. The key point is that both architectures have the same NIC buffer size to ensure fairness. In the MEP case, the 128 flits are divided into the four smaller sub-NICs.

The NIC overhead in the MEP architecture (shown in Table 2) includes the dTDMA buses. In fact, the slight increase in area and power over the generic NIC is due to the dTDMA buses. However, since the NIC area in both the MEP and generic architectures is dominated by the buffers (hence the large area in Table 2), this small dTDMA overhead is negligible; 1.17% and 1.87% of the entire NIC area and power budgets, respectively.

The proposed architecture inflicts additional wiring complexity to the system due to the internal links. However, wiring occupies a separate layer in our VLSI implementation of this architecture. Since the area of the PE core is much larger compared to the network infrastructure, there is sufficient space in the wiring layer to accommodate the extra wires of the MEP topology on top of the PEs.

In the MESH-MEP topology, the number of sub-NICs in the edge nodes is smaller than that of internal nodes, as shown in Figure 2 (b). The nodes in the four corners of the network will always inject (eject) messages to (from) only one direction and, thus, need only one master sub-NIC with a buffer size equal to that of a generic NIC. Similarly, other edge nodes need two sub-NICs (each with a buffer half the size of a generic NIC buffer), since they can inject (eject) messages to (from) two directions. This ensures that the total NIC buffer size of edge nodes is still equal to the buffer size of a generic NIC.

Table 2. Overhead of the MEP Architecture

Architecture		Area	Power Dynamic/Leakage
Router	Generic	374,765.54 $\mu\text{m}^2$	119.52 mW/25.16 $\mu\text{W}$
	MEP	376,350.92 $\mu\text{m}^2$	120.26 mW/25.33 $\mu\text{W}$
	MEP Overhead	+ 1,585.38 $\mu\text{m}^2$ + 0.42 %	+ 0.74 mW/+ 0.17 $\mu\text{W}$ + 0.62 %/+ 0.68 %
NIC	Generic	542,361.86 $\mu\text{m}^2$	170.24 mW/37.37 $\mu\text{W}$
	MEP (includes dTDMA Buses)	548,703.21 $\mu\text{m}^2$	173.43 mW/37.95 $\mu\text{W}$
	MEP Overhead	+ 6,431.35 $\mu\text{m}^2$ + 1.17 %	+ 3.19 mW/+ 0.58 $\mu\text{W}$ + 1.87 %/+ 1.55 %

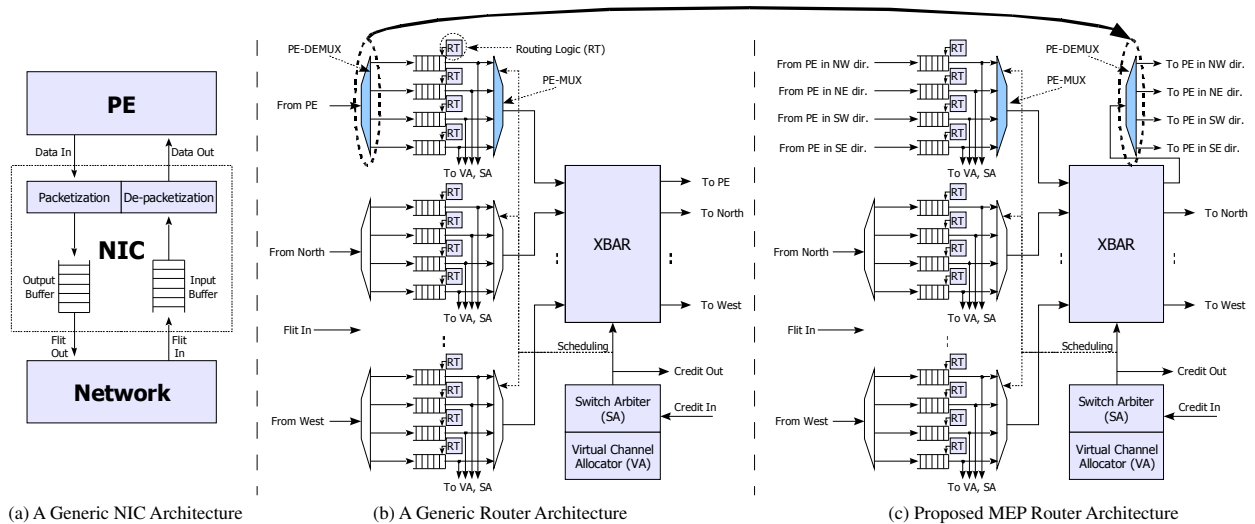


Figure 5. Two Router Architectures and a Generic NIC Architecture

### C. Router Design

The router itself should also be modified to fit into this architecture, but the required modifications are minimal, as it will be demonstrated below. Figure 5 shows the internal layouts of both a generic (Figure 5 (b)) [19] and an MEP router (Figure 5 (c)). ‘RT’, ‘VA’ and ‘SA’ represent routing logic, virtual channel allocator and switch arbiter, respectively. Both architectures adopt input buffering and both routers have five Physical Channels (PC), each with four virtual channels. The only difference between them is the injection/ejection port which, in the case of the proposed MEP router, is divided into four different injection/ejection ports for each attached PE (Northwest, Northeast, Southwest, and Southeast), as shown at the top of Figure 5 (b) and Figure 5 (c). The DEMUX at the output of the crossbar in the MEP router (marked as PE-DEMUX at the top of Figure 5 (c)) is not an overhead; it is used in the generic router architecture as well. The DEMUX simply moves from the PE input side of the injection/ejection channel in the generic router to the output of the crossbar in the MEP router, as indicated by the arrow. It is used to DEMUX the output from the crossbar to each sub-NIC during flit ejection.

Similar to the MEP NIC implementation, additional overhead in the MEP router is minimal, since existing resources are merely re-distributed in the space. Hence, the overhead is in additional control logic, which is negligible, as shown in Table 2. Both the generic router and the MEP router have a total buffer space of 80 flits (5 PCs  $\times$  4 VCs  $\times$  4 flits/VC).

In the generic router, the four virtual channels from a local PE contend for the crossbar through the MUX (marked as PE-MUX in Figure 5 (b)). The same contention exists in the MEP case between the four attached PEs (through PE-MUX in Figure 5 (c)). Furthermore, in the generic case, the local PE can only inject into one virtual channel at a time through the input DEMUX (marked as PE-DEMUX in Figure 5 (b)). However, in the MEP case, all four attached PEs can inject at the same time into their respective virtual channels, helping to alleviate blocking delay in the PE NIC buffers.

Overall, the results of Table 2 indicate that the area overhead of the proposed architecture over a generic one is minimal; both the NIC (four sub-NICs combined) and router are only slightly larger (1.16% and 0.42%, respectively). Power consumption follows the same trend; MEP consumes on an average 1.17% more power (in NIC and router). However, when we consider the  $(2n-1)$  routers saved by the MEP implementation in an  $n \times n$  MESH, then the total area and power budgets of the on-chip network actually decrease.

### D. Deadlock Recovery

The proposed topology model inherently supports deadlock recovery regardless of the routing algorithm used, as long as the routing algorithm follows the requirements below:

- If a message is blocked in a router buffer more than  $T_{thres}$  cycles, it should be temporarily absorbed into one of two nearby nodes closer to the desired direction. Then, it will be forwarded to a sub-NIC within that node, which is closest to its final destination, and, finally, it will be re-injected into the network again.
- When messages are injected into the network (including both new messages and temporarily absorbed and forwarded messages), the router should send them to one of the two directions away from the injecting sub-NIC’s PE. For example, if the message is re-injected through a SE sub-NIC into the router, as shown in the dotted line in Figure 6 (b), the message should be sent to either the south or east direction from that router, i.e. away from the injecting PE. This will avoid inter-dimensional deadlocks.

**Theorem:** The MEP topology ensures deadlock freedom if the above two requirements are met.

**(Proof)** There are two possible deadlock situations in the MEP topology. A proof is provided for both cases.

#### Case 1: Circular deadlock (in both MESH and TORUS)

For a circular, or inter-dimensional, deadlock, as shown in Figure 6 (a), to occur, all the messages involved in the deadlock configuration should be unable to be absorbed into a node even after the preset threshold value of  $T_{thres}$  cycles has elapsed. However, messages can always be absorbed into a node eventually, as long as the forwarding path is not blocked forever. The only situation where the forwarding path might be blocked forever occurs when the router connected to that sub-NIC cannot send the message to any of its two intended inter-router links due to permanent blocking of the links. The permanent blocking of these links will occur only when they are also involved in another deadlock configuration already (temporary blocking of those links is not enough to cause deadlock, since blocking will eventually disappear).

In other words, for a deadlock to occur, there should be other “prior” deadlocks that have already blocked all the re-injection directions such that temporary absorption of messages becomes impossible. In turn, these prior deadlocks also need other “prior-prior” deadlocks, following the same reasoning. This dependency will propagate throughout the network until all deadlock configurations

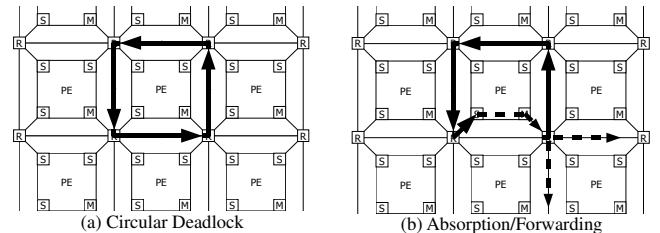


Figure 6. Deadlock Recovery - Case 1

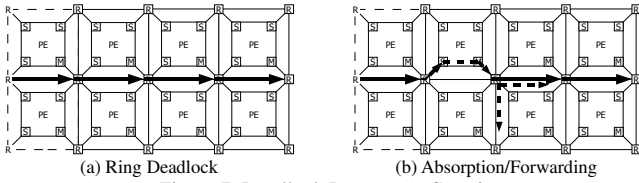


Figure 7. Deadlock Recovery - Case 2

block each other completely.

However, since the network is initially deadlock-free, deadlocks can only occur in this network if the whole network is involved in deadlocks simultaneously. All these deadlocks must occur at the same time and satisfy all the inter-dependencies among them. However, the formation of a deadlock requires message movement toward the deadlock configuration, leaving available buffer spaces behind that can be used by other messages. In turn, these available buffer spaces ensure that the whole network cannot be deadlocked at the same time. Therefore, absorption, forwarding and re-injection cycles can always break deadlocks in the proposed topology.

### Case 2: Ring deadlock (in TORUS only)

In a TORUS topology, deadlock can also occur along a row or a column (also known as intra-dimensional deadlock), as shown in Figure 7 (a). As in case 1, as long as one of the messages involved in the deadlock configuration can be forwarded as shown in Figure 7 (b) the deadlock will be broken. Once again, the only reason that such a forwarding path might not exist is if there are prior deadlocks. Following the same reasoning as in Case 1, the MEP topology ensures deadlock-free operation.

## IV. PERFORMANCE EVALUATION

To evaluate the performance of our proposed topology, we performed simulations using a cycle-accurate NoC simulator. A network with 64 nodes (8x8) was tested using the four different topologies – MESH-GEN, TORUS-GEN, MESH-MEP and TORUS-MEP – discussed in the previous sections. The simulator keeps injecting messages until 300,000 messages (including 100,000 warm-up messages) are ejected. Each message consists of four flits. For all results discussed, a uniform traffic pattern is used for message injection, where a node injects messages into the network at regular intervals. In our experiment, source and destination nodes are randomly selected.

A 4-flit buffer was assigned to each VC within the router and the number of VCs per Physical Channel (PC) (i.e. North, South, East, West and PE) was set to four. The number of VCs can vary, but for deadlock recovery in generic topologies at least 2 (TORUS-GEN with deterministic routing and MESH-GEN with adaptive routing) or 3 (TORUS-GEN with adaptive routing) VCs are required, as described below. For fair comparison, the total buffer size of a generic NIC was set to be the same as the sum of all MEP sub-NIC buffers in a node, namely 128 flits. In the generic case, these are divided into a 64-flit input NIC buffer and a 64-flit output NIC buffer. In the MEP case, the 128-flit buffer space is distributed equally to the four sub-NICs.

For deterministic (DT) routing, dimension-ordered (X-Y) routing was used. In the MESH-GEN topology, this routing is deadlock-free since cyclic dependency cannot occur in the network. In the TORUS-GEN topology, however, a ring deadlock may arise, as described in the previous section. To solve this problem, [6] proposed a scheme that classified available VCs into two sets. If the source node is to the left (bottom) of the destination node when the flit is traversing the X (Y) direction, it uses VC set 0. Otherwise, VC set 1 is used. This prevents the occurrence of a ring deadlock.

For adaptive (AD) routing, we used a minimal adaptive routing algorithm, where flits are sent to one of two directions (based on available buffer space) that reduce the distance to the destination. However, this algorithm suffers from deadlock (both Cases 1 and 2 of the previous section) under both MESH-GEN and TORUS-GEN topologies. To tackle this, we used the techniques proposed in [6] that reserve one VC (in MESH-GEN) or two VCs (in TORUS-GEN) as escape channel(s) to break possible deadlocks. When a flit using an adaptive VC has been blocked for more than a specified number of cycles, the router assigns a deterministic VC to this flit. Since

Table 3. Deadlock Recovery/Avoidance Schemes

	GENERIC		MEP	
	MESH	TORUS	MESH	TORUS
DT	None	VC classification	Temporary absorption into intra-PE network ( $T_{thres}=35$ cycles)	
AD	Reserve <b>one</b> VC for escape path that uses simple X-Y routing	Reserve <b>two</b> VCs for escape paths that use VC classification scheme		

deterministic VCs are deadlock-free, the deadlock configuration is broken. In the TORUS-GEN topology, two VCs are required for the VC classification scheme described above.

On the other hand, for MEP topologies, simple routing algorithms – X-Y deterministic routing and minimal adaptive routing – that do not include deadlock avoidance/recovery schemes were used due to the inherent ability of the proposed topology to avoid deadlocks through the intra-PE network. A  $T_{thres}$  (see Section III-D) of 35 clock cycles was used to initiate temporary absorption. Table 3 summarizes the deadlock recovery/avoidance schemes used during the simulations.

Figure 8 (a) and Figure 8 (b) show the overall latency vs. message injection rate with both Deterministic (DT) and Adaptive (AD) algorithms for MESH (ME) and TORUS (TR) topologies. In most cases, our MEP-based topologies showed lower latencies, especially at higher injection rates, where the networks tend to saturate. Even though MESH-GEN with adaptive routing performs better than MESH-MEP at injection rates higher than 0.7, it saturates at much lower injection rates, resulting in significantly higher latency, as shown in Figure 8 (b). In deterministic routing, the MEP topology lowers latency by 19% and 40% (on average) in TORUS and MESH implementations, respectively. In adaptive routing, the latency reduces by 23% and 20% (on average) in TORUS and MESH, respectively.

The finite buffer size of the NICs implies that the actual injection rate into the network might be less than intended, due to blocking. This achieved injection rate (AIR) is examined in Figure 8 (c) and Figure 8 (d). AIR increases linearly (following the requested injection rate) before the network saturates, but remains flat afterwards. As already mentioned, AIR depends on the size of the NIC input buffer, since PEs cannot inject messages if the NIC input buffers are full. Even though our proposed architecture has a smaller (16 flits) master sub-NIC input buffer (PE-In buffer in Figure 4(a)) than the NIC input buffer in the generic topology (16 flits  $\times$  4 VCs = 64 flits), it still provides better AIR at all times. The MEP architecture has a smaller master sub-NIC input buffer, because the remaining buffers are distributed to all other sub-NICs. However, MEP achieves higher AIR because of the removal of the input DEMUX from the injection ports, as described in Section III-C.

AIR directly affects the Time-to-Completion, i.e. the time it takes to eject a fixed number of messages from the network (Figure 8 (g) and Figure 8 (h)). Therefore, despite the fact that MESH-MEP with adaptive routing exhibits slightly higher latency at injection rates greater than 0.7, its higher AIR causes its Time-to-Completion to be almost half that of MESH-GEN. In other words, in MESH-GEN fewer messages are injected every cycle due to excessive NIC-buffer blocking.

The Energy-Delay-Product (EDP) is shown in Figure 8 (e) and Figure 8 (f). EDP is a metric combining both energy efficiency and latency. A lower EDP value signifies a more energy efficient system. In all cases, EDP increases with injection rate; the higher the injection rate, the higher the activity in the network, which implies increased dynamic energy consumption. As the network begins to saturate at high injection rates, blocking begins to dominate in the routers. This, in turn, translates into saturated latency and energy consumption; hence, the EDP curves are fairly flat at high injection rates. Clearly, the proposed MEP topology significantly reduces the EDP, a consequence of the reduced number of external hops and message forwarding paths, which consume less energy than inter-router paths which involve router pipelining. This result is crucial, because it illustrates the fact that despite the slight power overhead in the control logic of the MEP topology, the overall energy consumption is lowered due to the aforementioned two reasons. Specifically, in deterministic routing, EDP is lowered by 47% and 63% (on average) in TORUS and MESH, respectively. In adaptive routing, MEP lowers EDP by 50%, on an average, in both TORUS and MESH.

Self-similar and multimedia traffic patterns were also investigated, with the results exhibiting similar trends (we only show latency results for Self-Similar traffic in Figure 9, due to space limitation).

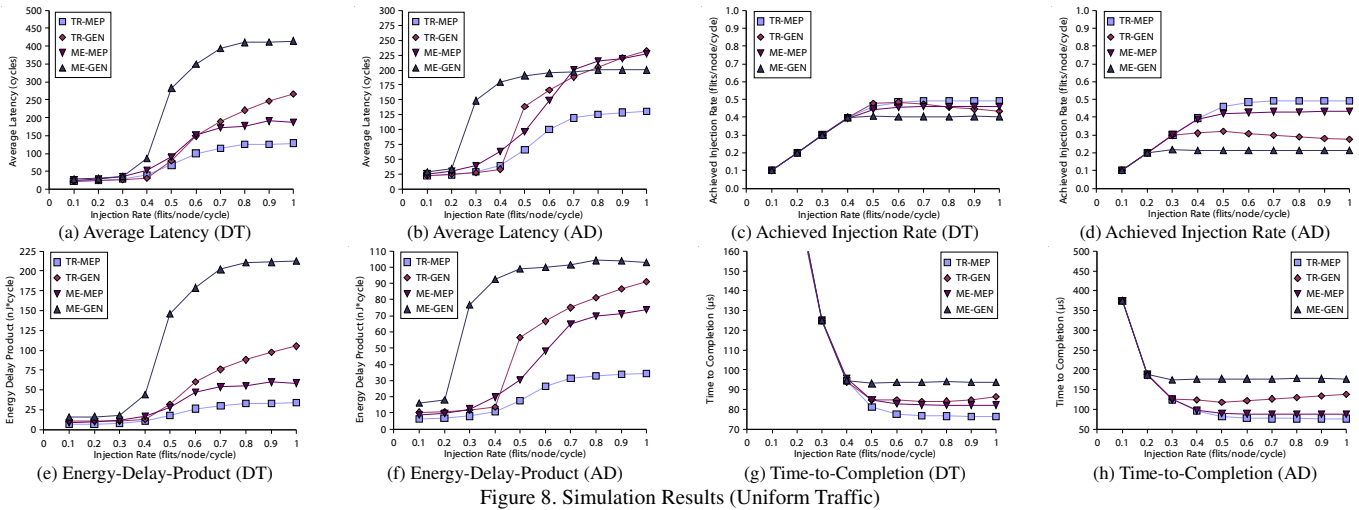


Figure 8. Simulation Results (Uniform Traffic)

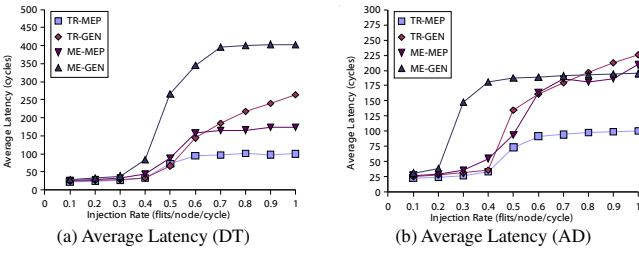


Figure 9. Simulation Results (Self-Similar Traffic)

## V. CONCLUSIONS & FUTURE WORK

The burgeoning adoption of complex SoC architectures, coupled with diminishing feature sizes, has heightened the need for very efficient on-chip interconnects. NoCs have become the dominant choice to fulfill this need. However, their resource-constrained nature has posed several challenges to their optimization in terms of area, power and performance. In this work, we propose a new network topology which injects messages into the on-chip network from four, rather than one, points per node. It is shown that this topology requires minimal modifications to existing NIC and router architectures; the changes involve dividing existing NIC resources and rearranging them into four smaller sub-NICS. Thus, the proposed Multiple Entry Point (MEP) topology can be retrofitted to existing router architectures to boost network performance. MEP incurs negligible area overhead, while providing significant improvements in both latency and energy consumption. Cycle-accurate simulations indicate that the average latency and Energy-Delay-Product (EDP) are reduced by approximately 20% and 50%, respectively, over existing implementations. In the case of MESH network implementations, the proposed MEP scheme also provides significant area savings, because it requires fewer routers than a generic MESH. This flexibility stems from the fact that multiple PEs can connect to a single router. More importantly, the new topology inherently avoids deadlock through its intra-PE network, eliminating the need for resource-hungry deadlock-avoidance algorithms.

The additional links of the proposed MEP topology are expected to substantially improve the fault-tolerance of the communication fabric as well. This facet of the new topology is currently being investigated by the authors. Future research on this topic will concentrate on the reliability of this topology as compared to existing implementations.

## VI. REFERENCES

[1] P. Rickert, Problems or opportunities? Beyond the 90nm frontier, 2004. ICCAD - Keynote Address.

[2] P. Guerrier and A. Grenier, A Generic Architecture for On-Chip Packet-Switched Interconnections, In Proc. of the IEEE DATE, 2000.

[3] L. Benini and G. D. Micheli, Networks on Chips: A New SoC Paradigm, IEEE Computer, 35(1):70–78, 2002.

[4] L.-S. Peh and W. J. Dally, A Delay Model and Speculative Architecture for Pipelined Routers, In Proc. of the HPCA, 2001

[5] J. Kim et al., A Low Latency Router Supporting Adaptivity for On-Chip Interconnects, In Proc. of the DAC, 2005.

[6] J. Duato, A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks, IEEE TPDS, vol. 4, no. 12, Dec. 1993.

[7] Martínez-Rubio et al., A Cost-Effective Approach to Deadlock Handling in Wormhole Networks, IEEE TPDS, 2001

[8] T. D. Richardson et al., A Hybrid SoC Interconnect with Dynamic TDMA-Based Transaction-Less Buses and On-Chip Networks, in Proc. of the VLSI Design, 2006

[9] R. Mullins et al., Low-latency virtual-channel routers for on-chip networks, In Proc. of the ISCA, 2004.

[10] The nostrum backbone project. <http://www.imit.kth.se/info/FOFU/Nostrum/>.

[11] Stanford-bologna netchip project. <http://akebono.stanford.edu/users/nanni/research/net/netchip/>.

[12] J. Dielissen et al., Concepts and implementation of the Philips network-on-chip, In Proc. of the IP-Based SOC Design, Nov. 2003.

[13] A. Jalabert et al., xpipes compiler: a tool for instantiating application specific networks on chip, In Proc. of the DATE, 2004.

[14] J. Kim et al., Design and Analysis of an NoC Architecture from Performance, Reliability and Energy Perspective, In Proc. of the ANCS, 2005

[15] J. H. Kim et al., Compressionless routing: a framework for adaptive and fault-tolerant routing, In Proc. of the ISCA, 1994.

[16] Anjan K. V. and T. M. Pinkston, An efficient fully adaptive deadlock recovery scheme: DISHA, In Proc. of the ISCA, 1995.

[17] P. Pratim Pande et al., Design, Synthesis, and Test of Networks on Chips, IEEE Design and Test of Computers, vol. 22, no. 5, Sep./Oct., 2005.

[18] A. Radulescu et al., An Efficient On-Chip Network Interface Offering Guaranteed Services, Shared-Memory Abstraction, and Flexible Network Configuration, In Proc. of the DATE, 2004.

[19] W. J. Dally and B. Towles, Principles and practices of interconnection networks, Morgan Kaufmann, 2003

[20] R. Marculescu, Networks-On-Chip: The Quest for on-chip fault tolerant Communication, In Proc. of the IEEE Symp. on VLSI, Feb, 2003.

[21] J. Kim et al., A Gracefully Degrading and Energy-Efficient Modular Router Architecture for On-Chip Networks, In Proc. of the ISCA, 2006.