Deflection Routing for Multi-Level FPGA Overlay NoCs

Chethan Kumar H B, Shubham Agarwal
School of Computer Science and Engineering
Nanyang Technological University
Singapore, 639798.
chethank001@e.ntu.edu.sg

Nachiket Kapre
Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, Canada.
nachiket@uwaterloo.ca

***Abstract*—**
Reducing worst case routing latencies while delivering high throughput and low energy are key design concerns in the engineering of overlay packet-switched NoCs for FPGA fabrics. Deflection routed torus NoCs are known to map particularly well to modern wire-rich FPGA substrates with fracturable LUT organizations while delivering high sustained bandwidths for various workloads and traffic patterns. However, they suffer from significantly higher worst case routing latencies due to deflections, particularly at large system sizes, when compared to classic buffered NoCs. To tackle this challenge, we design a deadlock-free hierarchical torus that (1) targets worst case latencies in deflection torus NoCs by separating deflections into two levels of the NoC, (2) delivers an FPGA-friendly design for deadlock freedom by providing physical escape channels in the lower levels, and (3) naturally supports physical layout for large multi-die FPGA chips by mapping upper level links to expensive interposer connections between dies. We generate layouts for implementing the NoC on the ML605 board (XCV6LX240T FPGA), VC707 board (XC7VX485T FPGA) and large multi-die XC7V2000T chips while delivering fast 300–500 MHz NoCs while consuming 10–15% of FPGA LUT resources. For instance with the 16×16 NoC, we reduce worst case deflection costs by 1.5–10× while simultaneously improving sustained rates by 1.5–2× and lowering energy requirements by 25% for a range of statistically-generated traffic patterns.

I. Introduction

Modern FPGAs contain millions of LUTs, DSP blocks, on-chip memories and a rich communication fabric between them. However, programming FPGAs to take advantage of the communication fabric is a challenge. To overcome these limitations, we can *overlay* the FPGA with massively-parallel soft CPUs [5], [14] interconnected with a packet-switched NoC. The overlay can directly be programmed using a higher-level programming environment and enables to developer to take advantage of on-chip communication primitives to move data rapidly between the computing elements. In fact, the Microsoft Catapult FPGA accelerator [14] uses custom free-form expression processors interconnected with a custom NoC to distribute Bing queries within the FPGA. Furthermore, it has been shown how to tile modern FPGAs with hundreds of communicating RISC-V processor cores [5] with an FPGA overlay NoC [8] to boost accelerator design productivity. These NoCs route single-flit packets that coincide with the multi-processor-style load-store style workloads that are amenable to these fabrics (80%+ traffic in PARSEC [9] is single-flit).

While it may be tempting to replicate ASIC-style router design on FPGAs, they suffer from an *impedance mismatch* with the FPGA fabric resulting in large, bloated designs. For instance, the state-of-the art CMU Connect [13] FPGA NoC router uses virtual channels, complex allocation strategies, while the Penn Split-Merge [7] router uses expensive FIFO buffers at all internal links within the switch. These designs occupy 1.2–1.5 K LUTs per 32b router which, for context, is ≈5× larger than a custom RISC-V CPU implementation that requires 300–400 LUTs per CPU [5]. This high cost can be reduced by using Hoplite [8] which is an extremely lightweight NoC (60 LUT, 100 FF, 2.9 ns per 32b router) that outperforms CMU and Penn designs by as much as 25× in area, 1.5× in clock speed, and 2–3× when considering sustained rates on synthetic NoC workloads. Hoplite is based on the idea of bufferless deflection routing [12] on a 2D folded unidirectional torus layout that fits the modern fracturable 6-LUT FPGA fabric particularly well. However, despite these advantages, Hoplite suffers from significantly higher worst case routing delays and associated energy overheads [11]. For latency-critical dataflow workloads, timing-critical systems, and energy-conscious designs this is a significant challenge. For example, Bing query acceleration must meet specific timing deadlines in order to retain end-user engagement which directly translates into worst-case latency expectations from the acceleration fabric.

In this paper, we address the challenge of reducing the worst case latency losses in Hoplite at modest extra cost. We decompose the flat 2D torus Hoplite NoC into a multi-level design that intuitively allows us to separate traffic into local and remote partitions. Local traffic deflects within a lower level network while remote traffic that must traverse longer distances in the network, competes in the upper level network. The key contributions of our paper include:

- Design of deadlock-free multi-level deflection-routed torus NoC suitable for FPGA embodiment through provisioning of physical escape channels in lower levels.
- Extensive simulations and design optimization for various system sizes under synthetic traffic patterns to evaluate

978-1-5090-5602-6/16/$31.00 ©2016 IEEE

latency and energy tradeoffs.
- Optimized, automated Xilinx FPGA layouts for multi-level NoCs (including multi-die interposer-based chips).

II. Background

FPGA-based overlay NoCs have traditionally been mere copies of ASIC NoCs with complex buffering strategies, virtual channel support, and associated flow control techniques. All these features are perfectly suited for ASIC implementations, but do not translate well to the unique wiring-logic balance available on modern FPGAs. The FPGA substrate provides a fundamentally different tradeoff than ASICs by presenting the developer with a fixed allocation of logic, memory and interconnect resources. FPGA vendors must accommodate the most complex user designs which drive them to supply an abundance of wiring at the expense of logic and memory.

A. *Hoplite Router*

Hoplite [8] demonstrates a remarkably lightweight design that emphasizes FPGA implementation economy by using bufferless, deflection routed approach with a 2D torus resulting in simpler cheaper switches. It removes expensive FPGA buffers, logic overheads for supporting virtual channels and complex flow control approaches and tilts the balance in favour of using as few LUTs as possible. Similar designs have also been explored in the context of ASIC implementations in [12], and found to be of limited utility in [11]. Hoplite is a well-matched NoC router design for FPGA implementation due to its low cost, and high performance and serves as the basis for the design in this paper.

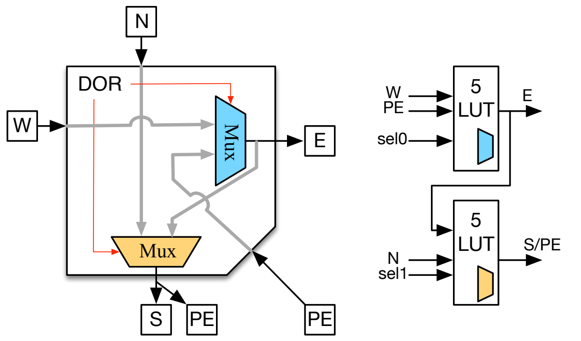


Fig. 1: High-Level diagram of Hoplite router. Mux mapped to fractured Xilinx 5-LUTs. Processing Element (PE) injects and receives network traffic.

We show a high-level diagram of the Hoplite [8] router microarchitecture in Figure 1. As we can see, it is merely a set of two multiplexers implementing the switch crossbar and some minimal logic for dimension ordered deflection routing. There are no FIFO buffers, or virtual channel logic thereby keeping the design simple. There are three key design principles that enable an FPGA-efficient high-speed implementation of this microarchitecture:

- **Topology:** Hoplite uses the unidirectional 2D torus topology thereby simplifying the switch crossbar. This makes it possible to pack each bit of the switch crossbar into two cascaded 5-LUTs (obtained by fracturing the Xilinx 6-LUT). In contrast, the designs presented in [13], [7] use 2D meshes which require 5-input crossbars that consume 2–3× more resource per bit just for the switch crossbar.
- **Bufferless, Deflection-Routing:** FIFO implementations of FPGAs make expensive use of LUT resources and are a dominant source of area requirements in [13], [7]. Hoplite uses bufferless, deflection-routing to avoid this cost entirely. This also simplifies flow control as the underlying routing algorithm is simply makes decisions based on arriving packet destinations. There is no backpressure between routers or other complex credit-based control strategies resulting in very fast FPGA logic paths. The Hoplite design even when mapped to large FPGAs with 100s of routers, still routes at 300–500 MHz.
- **FPGA-aware Mapping:** Hoplite exploits the fracturable 6-LUT architecture of the Xilinx FPGA, and abundance of wiring to economically map the router to FPGA logic. In Figure 1, the cascaded LUT connections are designed to fit both multiplexers into a single 6-LUT. To further save resources, Hoplite resource shares the South and Processing Element (PE) output ports with the same register with separate valid bits indicating the mode of use. Under this arrangement, (1) West→S turn will also block a PE→E packet, and (2) either West→PE or North→PE exits are allowed. These routability restrictions, have a minimal 2–3% reduction in sustained rates [8] but permit an economical 1 6-LUT/bit mapping.

When evaluated against statistical traffic patterns, Hoplite demonstrates a 1.5–2× advantage over conventional buffered NoCs mapped to identical FPGAs. However, this comes at the expense of longer worst case routing latencies compared to buffered routers.

III. Deadlock-free Hierarchical Torus

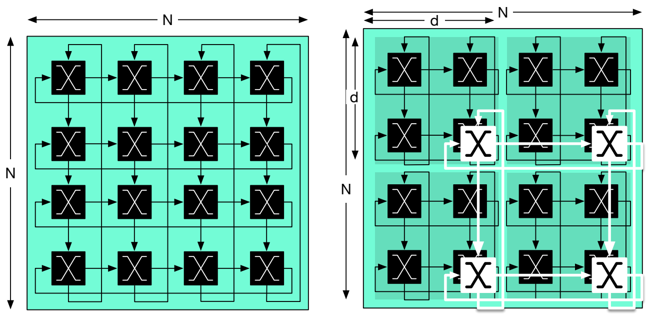


Fig. 2: Comparing Flat 2D Torus with Multi-Layer NoCs. Overall system size $N \times N$, while lower-level is $d \times d$.

In order to improve latency behaviour of the 2D torus, we restructure the network-on-chip into multiple levels such that traffic is split into two levels – a local sub-network that captures local traffic, and a global upper-level network that supports longer distance communication. Within each network we still implement 2D unidirectional bufferless deflection routing. The intuition behind this method of organization is to isolate local traffic to multiple local regions of the NoC. Thus, local traffic will not interfere with traffic in other local regions thereby reducing the overhead and penalty of deflections in the routing. Only global traffic will traverse both networks and will suffer less congestion and associated penalties in the upper level network. When implementing this NoC, we must design the inter-level interface carefully to avoid deadlock [2] as packets may endlessly await transfers into other levels.

In Figure 2, we show a representative example of a 4×4 flat 2D torus NoC and a 2-level NoC with 2×2 upper-level network and a 2×2 lower-level network. The **black** network is the local network while the **white** network manages global traffic. Our design space exploration tools can evaluate various partition sizes for splitting an arbitrary N×N network into a two-level design with d×d local networks (upper-level network of size $\frac{N}{d} \times \frac{N}{d}$). The upper levels can be configured to have at most d parallel channels to bandwidth match the bisection [2] against the original flat design. When mapped to span the entire FPGA, we are able to tightly fit a 69×21 32b-wide 340 MHz NoC utilizing 40% LUTs on a Xilinx Virtex-7 XC7VX485T (VC707 board).

In Figure 3, we show the path taken by a packet from location (0,0) to (3,3). The packet first navigates its way to the exit switch to enter the upper level network. It then moves to the correct region of the destination and descends to the lower level network and routes to the intended destination in that region. Thus, the packet spends time in the lower-level networks of the source and destinations and the upper-level network. It entirely avoids the lower-level networks in other regions.

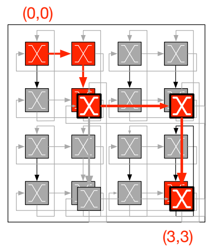


Fig. 3: Packet from (0,0) to (3,3).

A. Deadlock-Free interface design

While it is relatively straightforward to generate multi-level topologies from the Hoplite design, there is a possibility of deadlock at the inter-level transfer interfaces. To understand how a deadlock occurs, you can see the cycle in Figure 4.

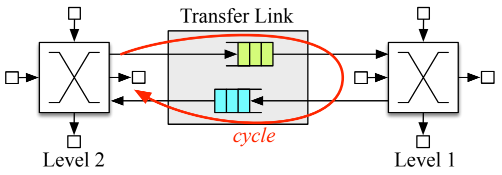


Fig. 4: Deadlock at inter-level interface.

While the networks at each level are simply deflection routed Hoplite designs, there is a need to provide some form of flow control for transferring packets across levels as the networks may be busy and unable to accept packets. We support the transfer of packets across levels through FIFO connections with associated flow control (valid/backpressure) in a manner similar to [2]. This behaviour is unlike processor sink interfaces (where packets exit the NoC) that always accept incoming packets without backpressure. This makes the deadlock-free crossing of packets across levels somewhat tricky. Packets wishing to cross levels either enter the transfer FIFOs or simply deflect back within the level when the FIFO is full. A cycle is introduced when both networks are saturated and make simultaneous requests to transfer packets across levels. In this scenario, the both networks will be unable to provide an empty slot for performing the crossing. Even with the presence of a dedicated `swap` logic will be insufficient as a similar deadlock case may still arise if only one transfer FIFO is full and there is no empty slot available for a transfer along the other direction.

In [2], the authors identify a similar deadlock at the transfer points of a hierarchical ring. In that instance, they resolve the deadlock through a combination of *injection guarantee* and *transfer guarantee* mechanisms. The injection guarantee forces the NoC clients to restrict traffic injection and the *transfer guarantee* tracks packet deflections in the upper level networks to prioritize older traffic. Both these mechanisms impose awkward restrictions on the network clients as the design of the transfer logic. Instead, we identify the fundamental cause of the deadlock (presence of cyclic waits) and directly resolve that through the provisioning on escape channels in the lower levels. This strategy of provisioning extra wiring in the lower levels is particularly well-suited to the FPGA fabric due to relative abundance of routing resources compared to logic. Particularly, when considering multi-die FPGA chips, we have substantially more intra-die bandwidth to accommodate this extra traffic while restricting the use of inter-die connections that traverse the interposer links for upper level traffic. On ASICs, and other wiring constrained substrates, a virtual-channel based approach that conserves expensive wires is more appropriate.

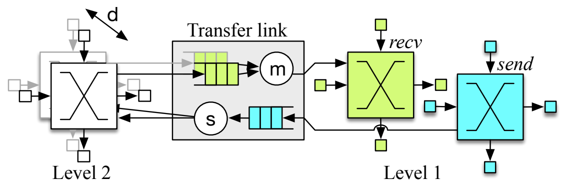


Fig. 5: Deadlock-Free design of the inter-level interface

As shown in Figure 5, we provision two separate physical channels (`SEND` and `RECEIVE`) in the lower levels of the network. Network clients are only allowed to inject new packets into the `SEND` network. Traffic that wants to descend from the upper level to the lower level is allocated its own `RECEIVE` channel (the escape channel for breaking deadlock). This splitting of channels in the lower levels guarantees that the `RECEIVE` network eventually consumes a descending packet. Since no clients are injecting traffic into the `RECEIVE` network, there is assurance that the network will eventually flush all packets that enter. This breaks the cycle and resolves the transfer deadlock.

We optimize the design of the interface logic by providing bypass paths for the FIFOs to enable single-cycle inter-level

TABLE I: Design-space Exploration of NoC parameters.

Parameter	**Range**
N	4×4, 8×8, 12×12, 16×16
d	2×2, 3×3, 4×4
Pattern	`LOCAL`, `RANDOM`, `BITREV`, `TRANSPOSE`, `TORNADO`
Sigma (`LOCAL`)	1,2,4,8,16
Injection Rate	1%, 5%, 7%, 10%, 15%, 20%, 50%, 100%

transfers in absence of any congestion. This is important to avoid wait time in the FIFO that only increases latency in the network. Furthermore, we design our FIFOs to be 32-deep to exploit low-cost SRL-based FIFO designs that are possible on the Xilinx FPGAs. Furthermore, the upper stages of the multi-level NoC will require d parallel channels to match bisection bandwidths with the flat NoC.

IV. Methodology

We perform simulations under various system sizes, synthetic workloads, injection rates and hierarchical arrangements along with physical mapping experiments on modern Xilinx FPGAs. This lets us evaluate metrics such as achieved bandwidths, worst case latencies, physical resource requirements and operating frequencies.

A. Simulation

We directly run RTL[1] simulations of the various network configurations. Our performance and energy data is derived directly from synthesizable RTL implementation of the switch. We implement packet injection logic to support various patterns while also correctly modeling source queueing delays. We also develop automated analysis tools that post-process simulation logs to confirm correctness of the NoC operation as well as permit a detailed analysis of performance. As the design space of possible combinations is large, we use the open-source `iverilog` tool and cheap Google Compute Engine resources to run 16–32 parallel instances of the simulation on one machine. We explore various combinations of N, d, synthetic pattern [1], and injection rates as listed in Table I. We sample the lower injection rates more as performance generally saturates above 50% injection rates and most realistic multi-processor workloads offer 5–10% injection rates [4]. For the `LOCAL` traffic pattern, we set the locality diameter to various sizes to generate traffic with configurable locality.

B. Physical Layout

We use Xilinx ISE 14.7 and Vivado 2015.4 to target the ML605 (Virtex-6) and VC707 (Virtex-7) parts respectively. We provide UCF and XDC constraints for precisely layout the NoCs on the FPGA. We use XPower (ISE/Virtex-6) and Vivado Power Analysis/Estimation (Virtex-7) tools coupled with activity traces extracted from Verilog simulations to calculate power usage of the chips. When using folded layout for the 2D torus, we consistently achieve clock frequency of 320–450 MHz for various NoC configurations. This frequency

[1]Verilog source adapted from *www.fpga.org/hoplite* for our multi level design.

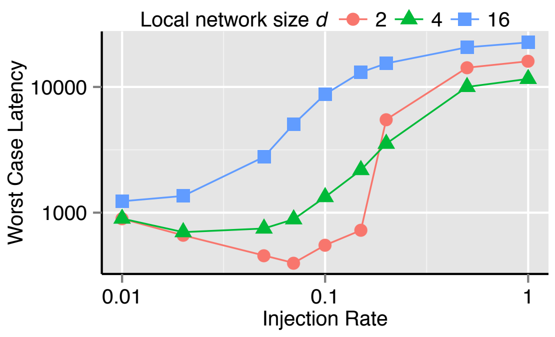


Fig. 6: Worst-case latency trends for 16×16 system for `LOCAL` traffic and varying d.

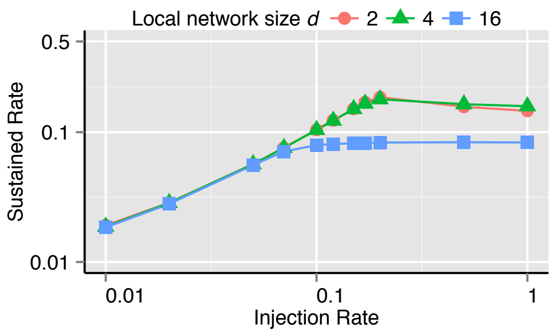


Fig. 7: Sustained rate trends for 16×16 system for `LOCAL` traffic and varying d.

is close to the typical operating frequency of the logic and interconnect fabric of the FPGA. For the large FPGA devices such as the Virtex-7 2000T, the chip is composed of four separate dies, called SLRs *super logic regions* in the FPGA CAD flow, connected by interposers. For these designs, crossing dies is slower (capped at 400 MHz with aggressive pipelining) and consumes more power. Layouts and NoC sizing for large FPGAs must minimize inter-SLR crossing to NoC links. For the XC7V2000T quad-die FPGA, we have ≈13K interposer connections that support larger datawidths up to 512b which are identical to the 64B cache-line flits [9] in PARSEC. We are able to fit 16×16 NoCs with a 4×4 subnetwork within the die constraints of the XC7V2000T chip (see layout in Figure 15 later).

V. Results

We now show latency and throughput results for the various design configurations explored in this paper. We also perform resource (area utilization), energy (deflection costs) and FPGA floorplanning experiments to further understand the benefits and limits of our design. We perform sensitivity analysis to the synthetic traffic patterns as well as system size.

A. Worst-Case Latency and Sustained Rate (`LOCAL` traffic)

Since minimization of worst case latency is the primary target in this work, we show the impact of synthetic traffic generated for the `LOCAL` pattern on latency in Figure 6 (other patterns shown later in Figure 9). Here, we observe that the use of two-level networks greatly reduces the worst-case latency particularly for modest injection rates <10–15%. For a 2×2

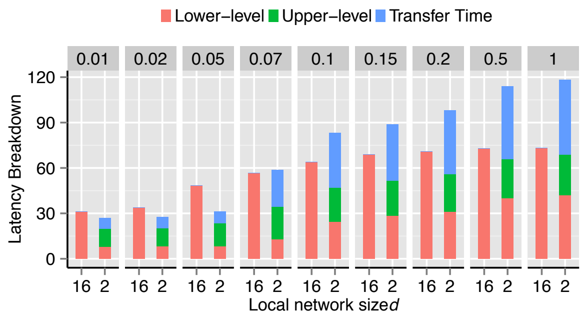


Fig. 8: Showing latency breakdown for NoC traversal in 16×16 NoC with d=2 and `LOCAL` pattern. (source queueing delay excluded for clarity).

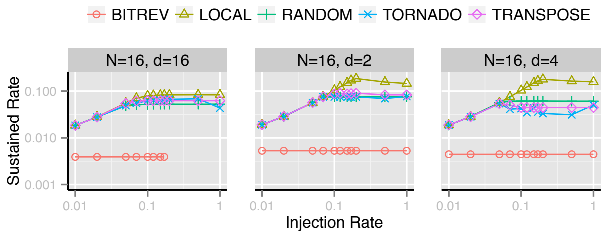
lower-level network, the localization of traffic significantly benefits the design at low injection rates. At 100% offered rates, the gap between the flat design and the multi-level design is still ≈1.5×. We observe a slight reduction in worst case latency when increasing injection rates from 1% to 7%. In this instance, the improvement is a result of the particular synthetic trace that is used in the simulation.

When evaluating sustained rates for the same traffic conditions, in Figure 7, we again see the hierarchical network is able to improve throughputs by 2× over the flat design at 10–15% injection rates and 1.5× at larger rates. And again, the benefits are highest at the modest injection rate scenario of <10–15%. For multi-processor workloads, we expect injection rates to be 5–6% for SPEC CPU2000 benchmarks [4]. At very low injection rates, there are insufficient deflections to cause a significant impact to sustained rates. However, even at those low rates, the worst case latency still improves as well saw earlier in Figure 6.

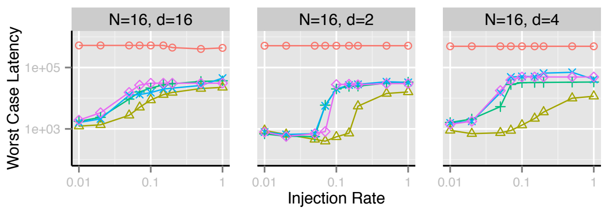
To understand how latency is distributed across various portions of the NoC, we provide a breakdown of time spent in the lower, upper and transfer stages in Figure 8. Source queueing is counted but not shown here as that dominates bulk of the time. It is clear that a roughly equal portion of time is spent in the transfer queues waiting to switch the network levels. When compared to a flat network, even when ignoring source queueing, the multi-level network beats the flat NoC for injection rates below 7%. For larger injection rates, if we strictly count the cost of NoC traversal itself without transfer queueing time, the multi-level NoC shows reduced impact of deflections. When counting all delays, the multi-level NoC still outperforms the flat NoC.

B. Sensitivity Analysis

We analyze the impact of different synthetic traffic patterns on sustained rate and worst case latency in Figure 9 for a size of 16×16. We clearly see that the multi-level network increases sustained rates and reduces worst case latencies particularly for `LOCAL` traffic. The `BITREV` pattern performs poorly in all scenarios and the other adversarial patterns such as `TORNADO` and `TRANSPOSE` suffer slight degradation in sustained rates at d=4. Local network size of d=2 delivers the



(a) Sustained Rate



(b) Worst Latency

Fig. 9: Impact of Traffic Pattern on Sustained Rate and Worst Case Latency for size of 16 × 16.

best results in this comparison and is very effective at lower injection rates <10–15%.

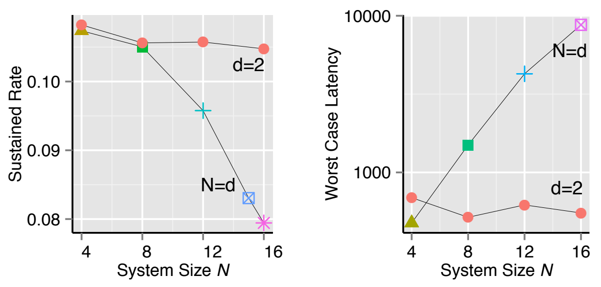


Fig. 10: Sustained rate and latency trends for various system sizes N for `LOCAL` traffic and injection rate of 10%. Considering $N = d$ (single-level network) and d=2 (multi-level network).

We also consider the impact of system size N on both sustained rate and worst case latency in Figure 10 for an injection rate of 10% and `LOCAL` traffic pattern. Here, we observe that the benefits of a multi-level network increases with larger system size when long distance packets are able to effectively exploit the multi level network architecture. Thus, as we scale to chip-spanning larger scale NoCs, we need to resort to the use of multi-level designs to help improve NoC performance at those large sizes.

C. Understanding Deflections

Next, we attempt to understand the mechanism of these improvements shown for the multi-level networks. To do this, we measure the distribution of packet latencies and time spent by packets in the two levels of the NoC. This has a direct impact on performance as well energy required for routing the workload. In Figure 11, we observe a large reduction in

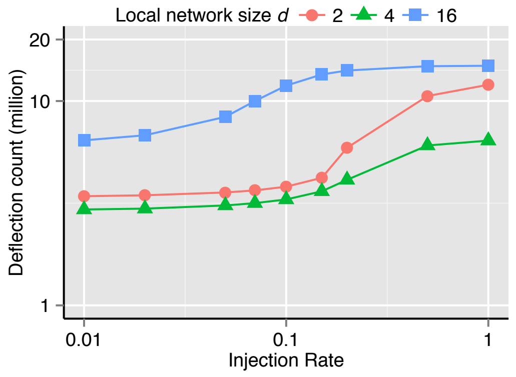


Fig. 11: Deflection Counts for varying sizes of d for different injection rates, (`LOCAL` pattern, N=16).

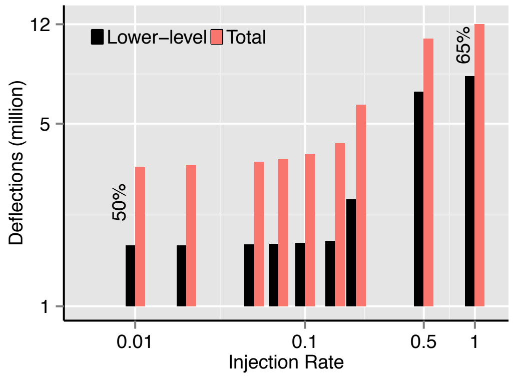


Fig. 12: Deflections in lower level network along with Total Deflections (`LOCAL` pattern, N=16, d=2).

the number of deflections suffered by packets when using a multi level NoC over a flat design. At low injection rates, the gap is large but narrows down as we increase injection rates (particularly for d=2). We further dissect these deflections to understand their relative proportion in the two levels. In Figure 12, we see that 50–65% of the reduced deflections are in fact localized to the lower level network. This means that majority of the packets are deflected within the $d \times d$ network with lower penalties per deflection. This means that the packets travel shorter distances (on average) over the multi-level NoC.

In Figure 13, we show the latency profile of packets for a 16×16 system with `LOCAL` traffic pattern and an injection rate of 10%. For multi level networks with d=2 and d=4, we observe that not only is the worst case latency better (right-most edge of the curves), but bulk of the packets have now shifted to the left of the plot resulting in lower latency per packet. The reduction in deflections for multi level networks observed earlier in Figure 11 are explained by this trend.

D. Physical Implementation

We quantify the impact of area (LUTs) on the sustained rate and worst case latency trends of our NoC for an injection rate of 10% and `LOCAL` traffic pattern in Figure 14. We also include a naïve multi-channel design that simply duplicates the network for higher bandwidth and the expense of more LUTs and wires. We see that the extra cost of the multi-channel NoC is not justified as the d=4 design is able to outperform the improved performance (marginally higher sustained rates and worst case latencies) at 20–30% less cost. It is important to observe that the multi-channel design will not only increase LUT use, but also impact interconnect requirements. Our multi-level design in contrast does not impose the proportional increase in interconnect needs as the upper level is bandwidth matched to the flat single-channel design. For d=4, our multi-level design has lower LUT and wiring needs than the multi-channel NoC.

We also generate folded floorplans for FPGA devices that span complete chip as shown in Figure 15 for the large multi-die XC7V2000T FPGA. We took particular care to ensure that

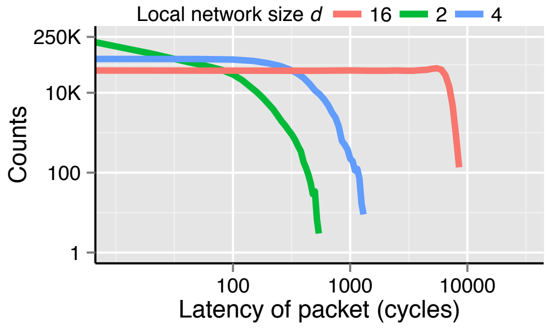


Fig. 13: Histogram of packet latencies for various multi-level configurations (`LOCAL` pattern, N=16, injection rate=0.1).

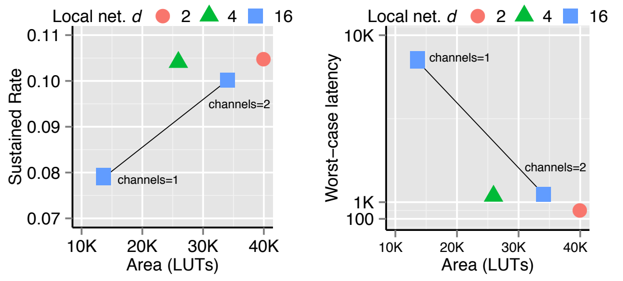


Fig. 14: Impact of resources (LUTs) on Sustained rate and latency trends for 16×16 system and d of 2 and 4 (injection rate=10%, `LOCAL` pattern).

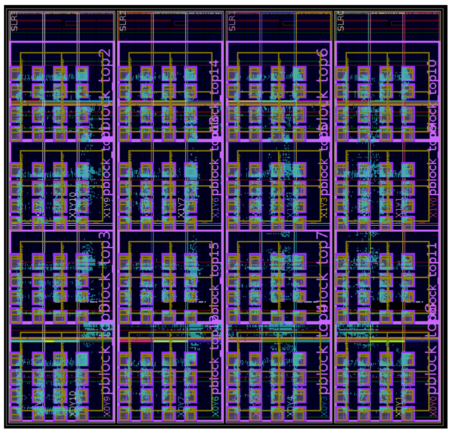


Fig. 15: 16×16 (4×4 lower level) 320 MHz chip-spanning 32b NoC on the Xilinx XC7V2000T (multi-die FPGA).

TABLE II: FPGA Mapping Data for VC707 board.

System	**Lower Lvl.**	**FPGA Area**			**Clock**	**Power**[a]
$N \times N$	$d \times d$	FFs	LUTs	(%)	MHz	W
4×4	-	0.8K	1.6K	0.3	340	0.4
8×8	-	3.6K	6.7K	1	430	0.8
16×16	-	15.6K	28K	10	490	2.3
4×4	2×2	2.7K	3.9K	1	450	0.5
8×8	2×2	10.8K	14.8K	4	340	1.1
8×8	4×4	8.1K	12.8K	2.6	410	0.9
16×16	2×2	46K	62K	15	360	4.1
16×16	4×4	33.2K	50.8K	11	320	2.9

[a]FPGA chip power modeled in Vivado 2015.4 for injection rate of 50%. Power measurements on VC707 board confirm this data.

TABLE III: Full-System Power Measurement VC707 board.

System	**Lower Lvl.**	**Board Power (W)**[b]		
$N \times N$	$d \times d$	Reset	Active	Δ
4×4	-	14.6	15.7	1.1
8×8	-	14.6	16	1.4
16×16	-	14.6	17.9	3.3
4×4	2×2	14.6	15.1	0.5
8×8	2×2	14.6	15.9	1.3
8×8	4×4	14.6	16	1.4
16×16	2×2	14.6	18.2	3.6
16×16	4×4	14.6	18	3.4

[b]Actual Measurement

Hoplite routers did not split across SLR boundaries to preserve desirable timing properties of the layout. Even 16×16 NoCs consume less than 15% of the chip resources for the NoC infrastructure leaving most of them for user logic. We tabulate the cost and performance of various design configurations in Table II. It is evident that with careful folded layout we are able to achieve 490 MHz speeds for 16×16 NoCs while exceeding 300 MHz+ frequencies for all configurations. Typical FPGA designs run at around 250-300 MHz implying the regular layout of the NoC is unlikely to affect overall system critical path delay and fast speeds of the flat single-level NoCs offer no particular advantage. Furthermore, is it clear that the addition of hierarchy increases cost only marginally 1–5% of chip area.

We also show the results of live experiments with power measurements on the VC707 board in Table III. We use Energenie power meter to record steady-state values for total board-level power draw after 2 minutes of sustained NoC packet activity. For this configuration we power up the VC707 directly with the supplied power adapter without hooking up the PCIe or Ethernet interfaces. We supply pin constraints to route the NoC edge connections the FMC pins to ensure that FPGA logic is retained by the compilation process and configure the PLLs to supply configurable 300–500 MHz clock to our NoC. We also synthesize-in dummy packet generation logic that injects `RANDOM` traffic at a rate of 50%. When considering *relative* values of power measurements with reference to the 4×4 NoC as baseline, the numbers in both Table II and Table III show remarkable agreement.

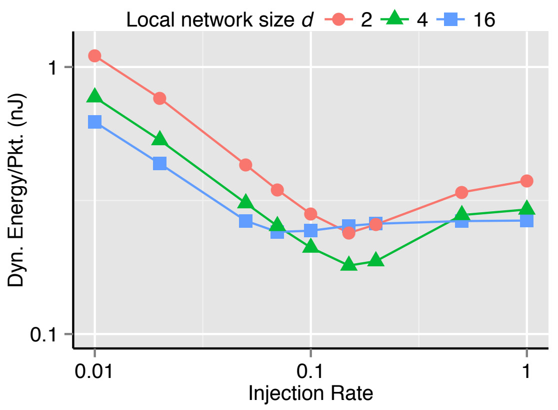


Fig. 16: Average energy/packet for varying injection rates for 16×16 NoC and `LOCAL` traffic pattern.

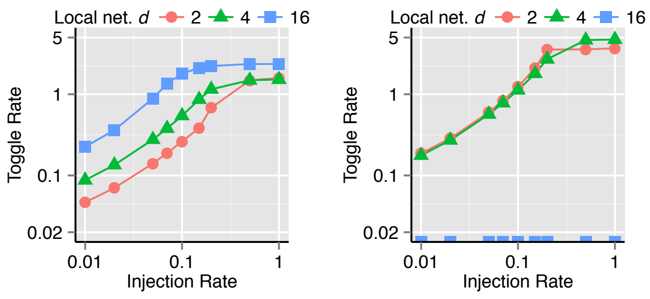
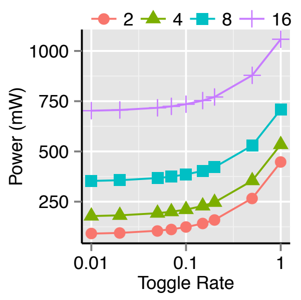


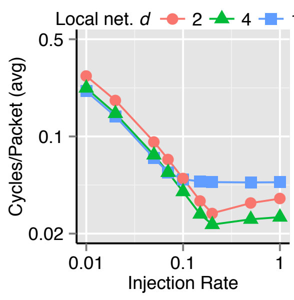
Fig. 17: Toggle rates in the lower and upper level networks for `LOCAL` pattern, 10% injection rate, and 16×16 size.

E. Energy Analysis

We also perform energy analysis of the deflections to understand the advantages of multi-level NoCs over flat single-level NoCs. In Figure 16, we plot the average energy per packet to route the `LOCAL` traffic pattern on a 16×16 NoC under varying injection rate. We see a tradeoff curve where the multi-level NoCs are up to 25% more energy-efficient at modest injection rates that are neither too low (<5%) nor too high (>50%). This trend can be explained by examining several factors (1) toggle rates in Figure 17, (2) power scaling in Figure 18a, and (3) latency/packet shown in Figure 18b. First, we observe a levelling of toggle rates at injections beyond 20%. Furthermore, there are no toggles in the upper level networks for the flat 16×16 design as expected. The toggle rates in the upper level networks are slightly more than those in the lower stages as traffic is forced to move in a smaller $\frac{N}{d} \times \frac{N}{d}$ network. When considering the power model shown in Figure 18a, we notice a large dynamic power requirement even at 1% FF toggle rates. This model is built from Vivado power characterization of the complete NoC netlist. This trend is expected as the dynamic energy calculations account for various other sources from the FPGA outside our design. Thus, we expect dynamic power calculations that use the power models to lookup the associated power draw in the multi-level network to closely track the *product* of Figure 17 and Figure 18a. Finally, to calculate energy per packet, we look at the average cycles counts per packet as shown in Figure 18b. It



(a) Power Model



(b) Cycle Trends

Fig. 18: Power model for various N×N systems, and cycles for `LOCAL` pattern with 10% injection rate, and 16×16 size.

is clear that at lower injection rates, we observe a high average packet latency dominated primarily by the end-to-end latency of traversing the network. At higher injection rates this drops and then levels off due to congestion in the network. For the multi-level networks, we observe performance improvements in average latencies above 10% when the flat NoC saturates. Thus, the energy trends in Figure 16 can be explained due to a combination of high average latencies per packet for low injection rates (high energy use for low activity) and increase in power requirements at high injection rates due to larger toggle rates in the upper level networks. Under realistic loading conditions of 5–50% injection rates, the multi-level NoCs still deliver superior energy properties.

VI. Discussion

Broadly, the idea of hierarchical NoCs has been explored extensively in NoC literature in [3], [10], [6], [12]. Most of these designs are classic packet-switched NoCs with virtual channels and ASIC implementations unlike our deflection routed FPGA overlay NoC presented in this work. In [6], the authors discuss a mechanism for evaluating deadlock freedom of hierarchical NoCs but do not consider the implications of deflection routing in their analysis. The hierarchical ring design presented in [2] requires injection rates to be throttled and deflections to be tracked for victimized packets. Our solution exploits the wire-rich FPGA substrate to deliver an improved design with physical escape paths with less complex requirements. In contrast to the original Hoplite [8] design, we perform area-time tradeoffs to help deliver substantially improved worst case latencies for realistic traffic conditions (as high at 10× for 10% injection rates on 16×16 NoCs with `LOCAL` traffic). We have previously discussed the significant area and performance advantages of Hoplite-based NoCs over contemporary FPGA overlay NoCs [13], [7] in Section II.

VII. Conclusions

With the emergence of FPGAs as energy-efficient accelerators in the data-center [14], it is important that we design lightweight, scalable FPGA-based overlay NoCs to enable rapid composition of accelerator kernels. In this paper, we show how to design a hierarchical, unidirectional torus NoC that is particularly amenable for implementation and layout on large FPGAs using physical escape channels to avoid deadlock. Using this approach, we can reduce worst case latency by as much as 10× (at <15% injection rate) and by 1.5× (heavily-loaded 100% injection rate) for `LOCAL` traffic pattern while also boosting sustained rates by 1.5–2×. This is achieved with at an affordable expense of 10–15% of FPGA chip area (5% increase over flat designs) for 16×16 NoCs. Furthermore, we also show up to 25% reduction in dynamic energy when considering injection rates between 5-50%.

Acknowledgements: The authors would like to thank Jan Gray for providing access to Hoplite RTL source code.

References

[1] P. Abad, P. Prieto, L. G. Menezo, A. Colaso, V. Puente, and J. A. Gregorio. TOPAZ: An Open-Source Interconnection Network Simulator for Chip Multiprocessors and Supercomputers. *Networks on Chip (NoCS), 2012 Sixth IEEE/ACM International Symposium on*, pages 99–106, 2012.

[2] R. Ausavarungnirun, C. Fallin, X. Yu, K. K.-W. Chang, G. Nazario, R. Das, G. H. Loh, and O. Mutlu. Design and evaluation of hierarchical rings with deflection routing. In *Computer Architecture and High Performance Computing (SBAC-PAD), 2014 IEEE 26th International Symposium on*, pages 230–237. IEEE, 2014.

[3] R. Das, S. Eachempati, A. Mishra, V. Narayanan, and C. Das. Design and evaluation of a hierarchical on-chip interconnect for next-generation cmps. In *High Performance Computer Architecture, 2009. HPCA 2009. IEEE 15th International Symposium on*, pages 175–186, Feb 2009.

[4] P. Gratz and S. W. Keclker. Realistic Workload Characterization and Analysis for Networks-on-Chip Design. In *The 4th Workshop on Chip Multiprocessor Memory Systems and Interconnects (CMP-MSI)*, 2010.

[5] J. Gray. GRVI-Phalanx: A Massively Parallel RISC-V FPGA Accelerator Accelerator. 3rd RISC-V Workshop, 1 2016.

[6] R. Holsmark, S. Kumar, M. Palesi, and A. Mejia. Hira: A methodology for deadlock free routing in hierarchical networks on chip. In *Networks-on-Chip, 2009. NoCS 2009. 3rd ACM/IEEE International Symposium on*, pages 2–11, May 2009.

[7] Y. Huan and A. DeHon. FPGA optimized packet-switched NoC using split and merge primitives. In *Field-Programmable Technology (FPT), 2012 International Conference on*, pages 47–52, Dec. 2012.

[8] N. Kapre and J. Gray. Hoplite: Building austere overlay NoCs for FPGAs. In *Field Programmable Logic and Applications (FPL), 2015 25th International Conference on*, pages 1–8, Sept 2015.

[9] S. Ma, N. E. Jerger, and Z. Wang. Whole packet forwarding: Efficient design of fully adaptive routing algorithms for networks-on-chip. In *Proceedings of the 2012 IEEE 18th International Symposium on High-Performance Computer Architecture*, HPCA '12, pages 1–12, Washington, DC, USA, 2012. IEEE Computer Society.

[10] R. Manevich, I. Cidon, and A. Kolodny. Dynamic traffic distribution among hierarchy levels in hierarchical networks-on-chip (nocs). In *Networks on Chip (NoCS), 2013 Seventh IEEE/ACM International Symposium on*, pages 1–8, April 2013.

[11] G. Michelogiannakis, D. Sanchez, W. J. Dally, and C. Kozyrakis. Evaluating Bufferless Flow Control for On-chip Networks. *Networks-on-Chip (NOCS), 2010 Fourth ACM/IEEE International Symposium on*, pages 9–16, 2010.

[12] T. Moscibroda, O. Mutlu, T. Moscibroda, and O. Mutlu. *A case for bufferless routing in on-chip networks*, volume 37. ACM, New York, New York, USA, June 2009.

[13] M. K. Papamichael and J. C. Hoe. CONNECT: re-examining conventional wisdom for designing nocs in the context of FPGAs. In *the ACM/SIGDA international symposium*, page 37, New York, New York, USA, 2012. ACM Press.

[14] A. Putnam, A. Caulfield, E. Chung, D. Chiou, K. Constantinides, J. Demme, H. Esmaeilzadeh, J. Fowers, G. Gopal, J. Gray, M. Haselman, S. Hauck, S. Heil, A. Hormati, J.-Y. Kim, S. Lanka, J. Larus, E. Peterson, S. Pope, A. Smith, J. Thong, P. Xiao, and D. Burger. A reconfigurable fabric for accelerating large-scale datacenter services. In *Computer Architecture (ISCA), 2014 ACM/IEEE 41st International Symposium on*, pages 13–24, June 2014.