Dynamic Resource Sharing for High-Performance 3-D Networks-on-Chip

Seyyed Hossein Seyyedaghaei Rezaei[§], Abbas Mazloumi[§], Mehdi Modarressi[§], and Pejman Lotfi-Kamran[†]
[§]Department of Electrical and Computer Engineering, College of Engineering, University of Tehran
[†]School of Computer Science, Institute for Research in Fundamental Sciences (IPM)

Abstract—3D logic-on-logic technology is a promising approach for extending the validity of Moore's law when technology scaling stops. 3D technology can also lead to a paradigm shift in on-chip communication design by providing orders of magnitude higher bandwidth and lower latency for inter-layer communication. To turn the 3D technology bandwidth and latency benefits into network latency reductions and performance improvement, we need networks-on-chip (NoCs) that are specially designed to take advantage of what 3D technology has to offer. While in parallel workloads many packets experience blocking in the network due to losing arbitration for crossbars' input/output ports, we observe that in a considerable fraction of these cases in a 3D NoC, the corresponding input and output ports of the crossbar in the above or below router are idle. Given this observation, we propose FRESH, a router microarchitecture with Fine-grained 3D REsource SHaring capability that leverages the ultra-low latency vertical links of a 3D chip to share crossbars and links at a fine granularity between vertically stacked routers. It enables packets that lose arbitration for crossbars' input/output ports to use idle resources of the above or below routers, and effectively eliminates the unnecessary packet blocking time. We will show that our proposal lowers network latency by up to 21% over the state-of-the-art 3D NoC.

Index Terms—Resource sharing, 3-D integration, network-on-chip.

1 Introduction

PARTITIONING a die into smaller parts and stacking them on top of each other (i.e., 3D integration) reduce latency and energy usage by replacing long on-die links with short vertical connections. Stacked layers can be interconnected using various technologies, but the dense and low-latency Through-Silicon Vias (TSVs) are predominant due to the high inter-layer bandwidth that they offer [12].

3D chips have the potential to significantly reduce on-chip latency and energy usage, but for maximizing the benefit, we need 3D NoCs that can fully take advantage of what 3D chips have to offer. There is a large body of work on 3D NoC design. Some of these designs use 3D links as normal NoC links that have lower power consumption and higher bandwidths [9], [11]. Such designs are simple but cannot fully realize the benefits of 3D chips. Other designs benefit from features unique to 3D chips (e.g., fast vertical links) to optimize routing and switching mechanism [4], [5], [8], [10]. This work falls in the second category.

Packet blocking is a source of major performance loss in networks-on-chip. NoCs experience both temporal and spatial variations in application traffic: a link experiences variation in traffic across time (i.e., temporal), and different links experience different loads in any given time (i.e., spatial) [2], [6]. Without dynamic adaption, routers should be statically over-provisioned at design time to tolerate peak traffic loads and avoid performance loss.

We observed that when a link is experiencing heavy load, with high probability, the corresponding link in the above or below router in a 3D chip is idle (spatial variation in traffics). Based on this observation, we present FRESH, a 3D NoC with Fine-grained REsource SHaring capability. FRESH takes advantage of low-latency inter-layer links (i.e., TSVs) to dynamically share links and crossbars between vertically adjacent routers. It allows packets that lose crossbars' input/output arbitration to use idle resources of the above or below routers (i.e., borrow unused resources of vertically adjacent routers). Consequently, routers can grow dynamically under heavy traffic load and shrink again when the load becomes moderate or low.

Through detailed cycle accurate simulation, we will show that our proposal reduces packet latency by up to 21% (16% on average) over the state-of-the-art 3D NoC.

Manuscript submitted: 18-Feb-2015. Manuscript accepted: 18-Jun-2015. Final manuscript received: 19-Jun-2015.

2 Preliminaries and Motivation

In a wormhole-switched router, arriving header flits advance through pipeline stages and allocate virtual channels (VCs), buffer slots, crossbars, and links. Ideally, all allocations succeed and flits proceed down the pipeline one stage per clock cycle. If any of these allocations fails, this ideal behavior is interrupted by stalls. Unsuccessful switch input and output allocation is an important source of flit stalls.

Switch input allocation stalls happen when all VCs of an input port are multiplexed to a single crossbar input port (rather than spreading over multiple crossbar inputs). In such cases, flits at the input virtual channels cannot perform switch traversal at the same time, even if their assigned switch outputs are idle. Switch output allocation stalls happen when multiple inputs compete for the same output port. In such cases, all but one requester will fail to allocate the requested switch output.

* Copyright © 2015 IEEE. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in IEEE Computer Architecture Letters, vol. 15, no. 1, pp. 5–8, Jan.–June 2016 (DOI: 10.1109/LCA.2015.2448532).

TABLE 1
Switch allocation failure (%) and the recovered fraction.

Traffic Load (injection rate)	SA Failures (%)		Recovered Fraction (%)	
	XYZ	Adaptive	XYZ	Adaptive
Low (0.005)	16.31	11.53	72.18	75.68
Heavy (0.4)	49.74	46.60	41.48	43.05

In this paper, we attempt to address flit stalls related to switch input and output ports in a 3D chip. Our proposal allows flits that are blocked in routers to borrow idle switches and links of vertically adjacent routers and proceed.

Table 1 shows the fraction of unsuccessful input and output switch allocation in a 4×4×3 3D-mesh NoC with wormhole switching under uniform traffic for both deterministic and adaptive routing algorithms. The deterministic routing uses a dimension-order XYZ algorithm. The adaptive routing measures the congestion (i.e., a linear combination of average number of requests for the output port and the number of busy VCs) and always forwards packets to less congested output ports. The table also shows the fraction of blocked requests that can be resolved if routers are allowed to borrow idle resources from their adjacent routers in the third dimension. When an input virtual channel at input port P fails to allocate the connection to output port Q through the switch (either due to input or output conflict), the blocking is *resolvable* if both P and Q are idle in at least one of the vertically adjacent routers. Table 1 suggests that resource sharing across the third dimension can improve NoC performance because a considerable fraction of flits can bypass busy resources, even in heavy traffic.

3 Proposed Router Architecture

In this section, we describe FRESH which enables vertical resource sharing in 3D NoCs. While FRESH can resolve both switch input and output conflicts, in this section, we only discuss how FRESH resolves switch output conflicts. Switch input conflicts are resolved in a similar way using the same TSVs, but the details are omitted due to space limitation.

3.1 Vertical Resource Sharing Mechanism

Global view. Figure 1 depicts a logical view of the two stacked layers that are participating in resource sharing. For each router, only one input port is shown for clarity. The figure shows vertical connections (TSV1 and TSV2) and multiplexers (MUX1 and MUX2) that allow flits to switch between adjacent layers.

With FRESH, a router sends remote switch allocation to the vertically adjacent routers whenever it determines that some flits cannot allocate the requested switch output. The switch input and output ports and the link of a neighboring router will then be borrowed and allocated to the failed flit, provided that they are idle.

3D resource sharing. FRESH requires minor modifications to the switch allocator of the baseline router; each switch allocator is connected to allocators of the same output port at adjacent routers in the upper and lower layers (using TSV3). Every cycle, an allocator with more than one request for an output port (where the arbitration will definitely have losers) sends remote allocation requests up and down to the

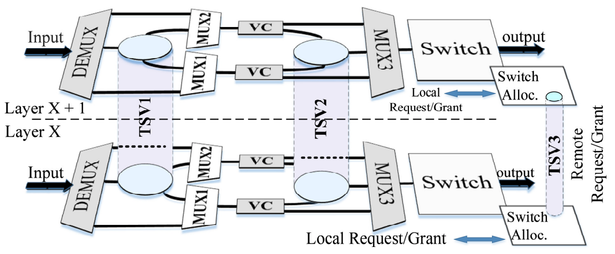

Fig. 1. The proposed architecture for 3D resource sharing.

allocators of the corresponding output ports at the upper and lower layers. In parallel, it arbitrates among its requests and grants the output to one of them. If the allocator receives grants from adjacent allocators (which happens most of the times according to Table 1), the next highest-priority flits will be selected to be sent through the remote crossbars and links. Thus, up to three flits can be sent to a single downstream router every cycle.

This parallel transfer would not violate correctness of communication as flits enter separate virtual channels at the downstream router. All requesting flits at this stage have completed the routing and virtual channel allocation stages, and there is at least one empty slot in their VC buffer.

Allocators prioritize requests from their local ports to remote requests received from adjacent routers. As such, allocators ignore remote requests if they have at least one local request, and otherwise, grant remote requests. If an allocator with no local request at a cycle receives two remote requests for an output port from the upper and lower routers, it grants one of them based on round-robin policy.

Remote input allocation. If a flit f at input port x in router r_n (in layer n) fails to allocate output y, it can move to the router r_{n+1} (or r_{n-1}) in layer $n+1$ (or $n-1$) only if the same ports (both x and y) are idle at that router. The availability of the output port (y) is ensured by receiving a remote grant signal from $r_{n\pm1}$. For an input port x in $r_{n\pm1}$, the availability status will be known when the allocation completes at $r_{n\pm1}$ (i.e., it might have a flit to send). If flit f receives a remote grant from $r_{n\pm1}$ for the output port y, it then checks the remote input port status. If the remote port has no flit to transfer or the flit has not been granted the requested output port, input x in $r_{n\pm1}$ is left idle and the $x-y$ connection at that router can be allocated to f. Otherwise, the output time slot that was reserved by the switch allocator is wasted. By giving higher priority to local requests in switch allocation, we guarantee that a switch output that could be used by a local request will never go idle because it was granted to a remote request that later failed to use it. This guarantees that link sharing does not hurt performance.

Inter-layer transfer. The bypass path for a flit in layer x in Figure 1 is set up by setting the control line of MUX3 at the remote router (i.e., layer $x+1$) to connect TSV2 to its crossbar (i.e., accept remote flit). The demultiplexer at the downstream router at layer $x+1$ should also be configured to connect the input link to TSV1. This configuration returns the flit to its original layer, and then to its assigned VC through MUX1 or MUX2. The path at the downstream router is set by the same mechanism that enables a conventional router to direct flits to the appropriate VC buffer.

3.2 Area Overhead and TSV Reduction

In our design, in addition to the regular vertical links along the third dimension, each router requires extra connections to the upper and lower routers for resource sharing. For 64-bit NoCs, every pair of vertically adjacent routers in the baseline architecture are connected by eight bundles of 64 TSVs (two bundles of 64 TSVs per port). Consequently, a router at the middle layer requires 16 bundles of 64 TSVs to connect to the upper and lower layers. In addition, each allocator requires four pairs of TSVs to send and receive remote requests (one pair per each output port) and another four pairs of TSVs to send/receive input availability status. Like data TSVs, the number of control TSVs is doubled for routers in middle layers.

We use two techniques to reduce the number of required TSVs by a factor of four. The first technique relies on an observation that TSV1 and TSV2 of a router are rarely simultaneously used in the same cycle. Our experiments shows that under uniform traffic, the probability of simultaneously using both TSV1 and TSV2 in a single cycle is 0.6% in low traffic and 8.4% in heavy traffic (near saturation). Thus, we can eliminate TSV2 and reuse TSV1 for situations where TSV2 is needed, thereby reducing the number of TSVs by a factor of two. Like prior work [10], TSV1 is used in either outgoing or incoming direction. In case of a conflict, the outgoing direction is always prioritized; the TSV path is set up for the local router and a signal informs the upstream router of the situation.

To further reduce the number of TSVs, we exploit the ultra-low latency operation of TSVs, which enables them to operate at higher frequencies than regular links. This way, we further halve the number of TSVs from 64 to 32, and keep the bandwidth intact by doubling the TSV frequency. The viability of applying high frequency to TSVs is demonstrated in [10].

With 32-bit TSVs, half of a 64-bit flit is transferred in each TSV cycle; thereby the entire flit is transferred in a single cycle from the router's perspective. This technique is also applied to halve the number of control TSVs. As a result, each pair of routers is connected by 32 TSVs per port, and hence each port at a middle layer needs two bundles of 32 TSVs. Their allocators are also connected by 8 TSVs.

To evaluate the area overhead, we use the TSV area estimation model presented in [3] for the 45 nm technology node. The block out area of the TSVs for our design is estimated to be 26400 μm^2 per router at a middle layer (12800 μm^2 for the first and last layers), which imposes 18.3% overhead to the total router area. The router area is estimated to be 0.144 mm^2 by the area model developed in [6] for a NoC with parameters described in Section 4. While the area blocked by TSVs may complicate the place-and-route process, techniques like [11] can be used to mitigate the problem.

Moreover, on average, FRESH increases the NoC (and not the entire chip) power consumption by less than 7% over the baseline network-on-chip. Although the power overhead is small (especially when we compare it against the power consumption of a multi-core chip), it might create local temperature hotspots as it increases the power density as compared to a planar 2D design. To mitigate the potential thermal challenges, techniques like thermal-aware routing [1] or topology reconfiguration [7] can be used.

3.3 FRESH Timing Issues

Arbitration and data transfer must be completed in a single cycle for FRESH to function properly. We modeled the datapath elements of our router using SPICE in 45 nm to obtain the latency of each element. The latency of TSVs in 45 nm (i.e., 1.3 ps) is taken from [3].

Arbitration. If an arbiter has some requests from its own router, it ignores remote requests, and otherwise, grants one of the remote requests. As there are at most two remote requests from the upper and lower layers, the remote arbitration is done using a 2-to-1 round-robin arbiter. Sending remote requests does not change the speed of arbiter as sending requests to remote arbiters happens in parallel with local arbitration. Upon receiving a grant from a remote router, the next highest-priority request is granted.

At each cycle, a 6-to-1 regular arbitration in a local router is done in parallel with a 2-to-1 remote arbitration plus two vertical link traversals. As the latency of local arbitration (134 ps) is higher than that of the remote arbitration (estimated as 39.6 ps), this process does not affect the arbitration latency. However, once the arbitration is completed, each input port propagates its availability to upper and lower layers to prevent input conflicts (see "Remote input allocation"). This process increases the arbitration latency by the latency of a vertical link traversal. Given the final arbitration latency (135.3 ps), this stage can be completed in a single cycle for frequencies up to 7 GHz.

Crossbar and link traversal. In this stage, a flit is read from the buffer and then traverses the crossbar and link. For crossbar traversal in FRESH, the flit passes over one TSV (TSV2 in Figure 1) and one multiplexer (MUM3 in Figure 1). Note that MUX3 is used in the baseline router, but FRESH adds one extra input to it. The new data-paths increase the base latency of 51 ps by 17.2 ps. For link traversal in FRESH, the flit passes through one demultiplexer, one multiplexer (MUX1 or MUX2 in Figure 1) and a TSV (TSV1 in Figure 1). Note that the demultiplexer is also in the baseline router: FRESH just adds an extra output to it. Using SPICE simulation, the latency of link traversal (1 mm) for the baseline router and FRESH is 110 ps and 128.3 ps, respectively. Therefore, crossbar and link traversal in a FRESH router (i.e., 196.5 ps) can be completed in a single cycle for frequencies up to 5 Ghz.

4 Evaluation Results

Simulation environment. We use a cycle-accurate NoC simulator to compare FRESH with the packet-switched 3D NoC with full 3D crossbar [5] and Hi-Rise 3D NoC [4]. We also compare the results with a NoC that combines both Hi-Rise and FRESH. Evaluations are done using both synthetic and real traffic patterns. For real workloads, we use SPLASH-2 traffic.

The topology is a 4×4×3 3D mesh with 64-bit links and wormhole-switched routers. Each input port has two VCs with the buffer size of eight flits. The simulator is configured to have a per-hop latency of three cycles ((1) VC

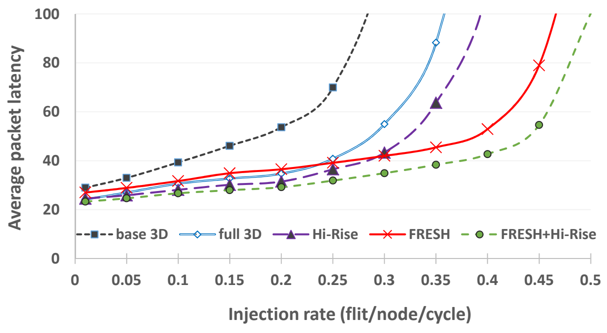

Fig. 2. Average packet latency (cycles) under uniform traffic.

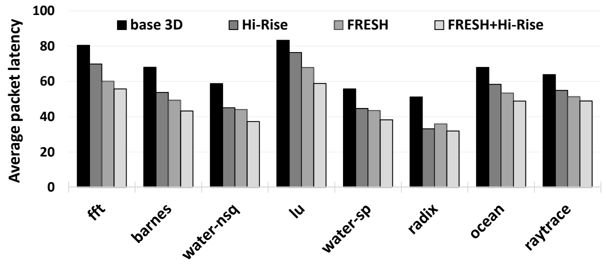

Fig. 3. Average packet latency (cycles) under SPLASH.

allocation+lookahead routing, (2) switch allocation, and (3) crossbar and link traversal). Packet size is set to five flits for uniform traffic. In SPLASH evaluation, data and control packets are five- and one-flit long, respectively.

Synthetic traffic. Figure 2 plots the average packet latency for uniform random traffic as a function of packet injection rate. The 3D NoC with full 3D crossbar and Hi-Rise offer lower latency (5% for Full 3D and 9% for Hi-Rise) near the zero-load region, as they offer smaller average hop count. However, as the injection rate increases, Hi-Rise and full 3D crossbar's performance starts to degrade as compared to FRESH. When injection rate increases, the contention latency begins to dominate, as more flits fail to obtain passage through the crossbar. FRESH, however, can still forward a considerable fraction of blocked flits through adjacent layers (even at high injection rates), and hence reduces contention latency (whose contribution to packet latency increases with traffic) significantly. As the figure shows, FRESH also offers more than 30% higher network throughput. Please note that FRESH requires smaller crossbar and simpler routing algorithm (i.e., the baseline deadlock-free dimension-order) than the NoC with full 3D crossbar and Hi-Rise. Finally, combining FRESH with Hi-Rise gives the best of both. Figure 2 shows that the combined NoC inherits the lower latency of Hi-Rise in low, and FRESH in heavy traffic.

SPLASH traffic. Figure 3 shows the average packet latency of the four considered 3D NoCs (baseline, Hi-Rise, FRESH, FRESH+Hi-Rise) under the SPLASH traffic. FRESH outperforms other alternative designs across all benchmarks and reduces latency by as much as 27% over the baseline and 11% over Hi-Rise. The combination of FRESH and Hi-Rise gives the best performance, reducing packet latency by up to 21% (16% on average) over the state-of-the-art (Hi-Rise).

Area-normalized comparison. We investigate how much performance would be obtained if we invest FRESH area overhead to increase router buffering or bit-width. The area of a 64-bit FRESH router with two VCs and 8-flit buffers is the same as the area of (1) a 74-bit conventional router with the same buffering, and (2) a 64-bit conventional router with two VCs each 10 flits deep. Figure 4 shows that FRESH outperforms both NoCs under uniform traffic.

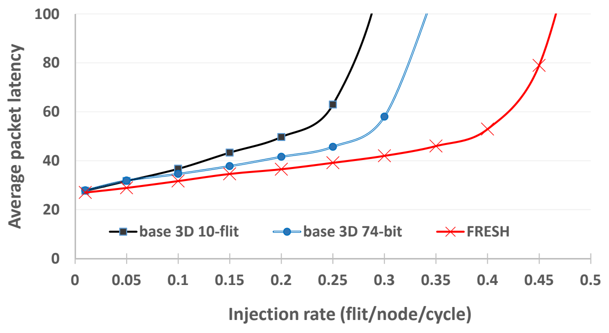

Fig. 4. Area-normalized evaluation of FRESH under uniform traffic.

5 Conclusion

We described a router microarchitecture for 3D NoCs that enables fine-grained resource sharing across layers. The proposed NoC leverages the ultra-fast vertical links to forward a blocked flit to vertically adjacent routers, provided that the corresponding link and crossbar ports are idle. The flit then traverses the borrowed crossbar and link, and returns to the original layer. Evaluation results show up to 21% (16% on average) lower packet latency as compared to the state-of-the-art 3D NoC.

References

[1] C.-H. Chao *et al.*, "Traffic- and Thermal-Aware Run-Time Thermal Management Scheme for 3D NoC Systems," in *Proceedings of NoCS*, May 2010, pp. 223–230.
[2] R. Hesse *et al.*, "Fine-Grained Bandwidth Adaptivity in Networks-on-Chip Using Bidirectional Channels," in *Proceedings of NoCS*, May 2012, pp. 132–141.
[3] H. Homayoun *et al.*, "Dynamically Heterogeneous Cores Through 3D Resource Pooling," in *Proceedings of HPCA*, Feb. 2012, pp. 1–12.
[4] S. Jeloka *et al.*, "Hi-Rise: A High-Radix Switch for 3D Integration with Single-Cycle Arbitration," in *Proceedings of MICRO*, Dec. 2014, pp. 471–483.
[5] J. Kim *et al.*, "A Novel Dimensionally-Decomposed Router for On-chip Communication in 3D Architectures," in *Proceedings of ISCA*, Nov. 2007, pp. 138–149.
[6] M. Modarressi *et al.*, "Application-Aware Topology Reconfiguration for On-Chip Networks," *IEEE Transactions on VLSI*, vol. 19, no. 11, pp. 2010–2022, Nov. 2011.
[7] R. Parikh *et al.*, "Power-Aware NoCs Through Routing and Topology Reconfiguration," in *Proceedings of DAC*, Jun. 2014, pp. 162:1–162:6.
[8] D. Park *et al.*, "MIRA: A Multi-Layered On-Chip Interconnect Router Architecture," in *Proceedings of ISCA*, Nov. 2008, pp. 251–261.
[9] V. F. Pavlidis *et al.*, "3-D Topologies for Networks-on-Chip," *IEEE Transactions on VLSI*, vol. 15, no. 10, pp. 1081–1090, Oct. 2007.
[10] A.-M. Rahmani *et al.*, "Developing a Power-Efficient and Low-Cost 3D NoC Using Smart GALS-based Vertical Channels," *Journal of Computer and System Sciences*, vol. 79, no. 4, pp. 440–456, Jun. 2013.
[11] C. Seiculescu *et al.*, "SunFloor 3D: A Tool for Networks On Chip Topology Synthesis for 3D Systems on Chips," in *Proceedings of DATE*, Apr. 2009, pp. 9–14.
[12] D. H. Woo *et al.*, "An Optimized 3D-Stacked Memory Architecture by Exploiting Excessive, High-Density TSV Bandwidth," in *Proceedings of HPCA*, Jan. 2010, pp. 1–12.