Flattened Butterfly Topology for On-Chip Networks

John Kim, James Balfour, and William J. Dally
Computer Systems Laboratory
Stanford University, Stanford, CA 94305
{*jjk12, jbalfour, billd*}*@cva.stanford.edu*

***Abstract*— With the trend towards increasing number of cores in a multicore processors, the on-chip network that connects the cores needs to scale efficiently. In this work, we propose the use of high-radix networks in on-chip networks and describe how the flattened butterfly topology can be mapped to on-chip networks. By using high-radix routers to reduce the diameter of the network, the flattened butterfly offers lower latency and energy consumption than conventional on-chip topologies. In addition, by properly using bypass channels in the flattened butterfly network, non-minimal routing can be employed without increasing latency or the energy consumption.**

***Index Terms*— on-chip networks, topology, flattened butterfly, high-radix routers**

I. Introduction

Chip multiprocessors are widely used to efficiently utilize the increasing number of transistors in modern VLSI technology. As the number of cores increases, on-chip networks are being used to provide a scalable communication infrastructure. Such on-chip networks need to provide low latency and high bandwidth to increasing numbers of processors.

Most on-chip networks that have been proposed are low-radix, with most using a 2-D mesh or a torus network [4]. While the design complexity of low-radix networks is modest and the wire lengths are short, these networks suffer several disadvantages, which include a large network diameter and energy inefficiencies due to higher hop counts. Balfour and Dally [1] proposed using concentrated meshes and express channels in on-chip networks to reduce the diameter and energy of the network. This work extends their idea to a symmetric topology that further reduces latency and energy. Kumar et al. [7] proposed the use of express virtual channels (EVC) to reduce the latency of 2-D mesh on-chip network by bypassing intermediate routers. However, both EVC and non-EVC packets compete for bandwidth on their 2-D mesh network. Kim et al. [6] showed that the increasing pin bandwidth in off-chip interconnection networks can be exploited by using *high*-radix routers to lower cost and reduce latency. Although on-chip networks have different resource constraints, we show that the flattened butterfly topology [5] proposed for high-radix off-chip interconnection networks retains many of its performance and efficiency advantages when used on-chip.

In this paper, we explain how on-chip networks benefit from using high-radix routers and describe a flattened butterfly topology for on-chip networks. This topology offers lower latency than a concentrated mesh. In addition, we present a method for non-minimal routing on the flattened butterfly that does not increase latency or energy consumption. Non-minimal routing requires that packets travel non-minimal physical distance; by utilizing *bypass* channels, latency and energy consumption can be further reduced while providing high performance.

The remainder of the paper is organized as follows. In Section II, we describe the flattened butterfly topology and routing for on-chip networks. The router microarchitecture and bypass channels are described in Section III, with evaluation and comparison following in Section IV. We conclude in Section V.

II. On-Chip Flattened Butterfly

A. Topology Description

The flattened butterfly topology [5] is a cost-efficient topology for use with high-radix routers. The flattened butterfly is derived by *flattening* the routers in each row of a conventional butterfly topology while maintaining the same inter-router connections. The flattened butterfly is similar to a generalized hypercube [2]; however, by providing concentration in the routers, the flattened butterfly significantly reduces the wiring complexity of the topology, which allows it to scale more efficiently [5].

To map a 64-node on-chip network onto the flattened butterfly topology, we collapse a 3-stage radix-4 butterfly network (4-ary 3-fly) to produce the flattened butterfly shown in Figure 1(a). The resulting network has 2 dimensions and uses radix-10 routers. With four processor nodes attached to a router, the routers have a concentration factor of 4. The remaining 6 router ports are used for inter-router connections: 3 ports are used for the dimension 1 connections, and 3 ports are used for the dimension 2 connections. Routers are placed as shown in Figure 1(b) with routers connected in dimension 1 aligned horizontally and routers connected in dimension 2 aligned vertically. Thus, the routers within each row and column are fully connected.

The wire delay associated with the Manhattan distance between a packet's source and its destination provides a lower bound on latency required to traverse an on-chip network. When minimal routing is used, processors in this flattened butterfly network are separated by only 2 hops, which is a significant improvement over the hop count of a 2-D mesh topology. The flattened butterfly attempts to approach the wire delay bound by reducing the number of intermediate routers – resulting in not only lower latency but also lower energy consumption. The flattened butterfly introduces long wires in the topology. The impact of long wires is reduced by optimally inserting repeaters and inserting pipeline registers to account for the longer delays [1].

B. Routing and Deadlock

Both minimal and non-minimal routing algorithms can be implemented on the flattened butterfly topology. Virtual chan-

Manuscript submitted: 10-May-2007. Manuscript accepted: 06-Jun-2007. Final manuscript received: 26-Jun-2007

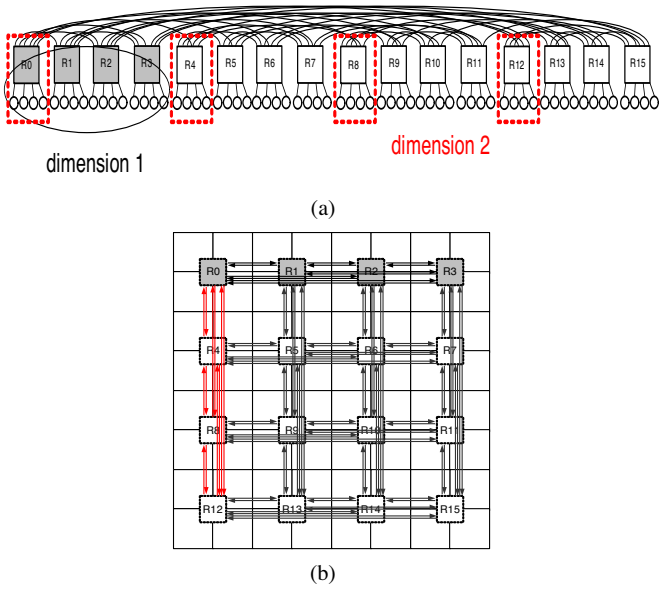

Fig. 1. (a) Block diagram of a 2-dimension flattened butterfly consisting of 64 nodes. and (b) the corresponding layout of the flattened butterfly where dimension1 routers are horizontally placed and dimension2 router are vertically placed.

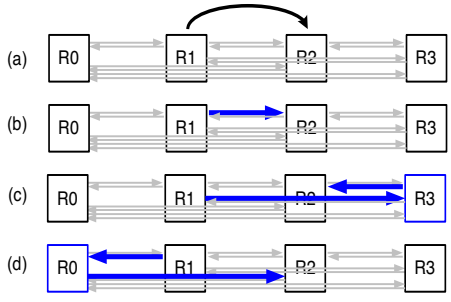

Fig. 2. Routing paths in the 2-D on-chip flattened butterfly. (a) All of the traffic from nodes attached to R1 is sent to nodes attached to R2. The minimal path routing is shown in (b) and the two non-minimal paths are shown in (c) and (d). For simplicity, the processing nodes attached to these routers are not shown and only the top row of routers is shown.

nels (VCs) [3] are needed to prevent deadlock. [1] Dimension-ordered routing (DOR) can be used when minimal routing is desired (e.g. route in dimension1, then route in dimension 2) and the routing restriction itself prevents deadlock. Non-minimal routing increases path diversity and improves load balance. For these reasons, we evaluate the UGAL [8] non-minimal global adaptive routing algorithm. UGAL routing balances load by routing minimally to an intermediate node in the first phase, and then routing minimally to the destination in the second phase. In general, a non-minimal routing algorithm requires $2d$ VCs, where d is the number of dimensions in the flattened butterfly. To reduce the number of VCs that are needed, we use DOR within each phase of UGAL routing – thus, only 2 VCs are needed.

[1]Additional VCs may be needed for other VC usage such as separating traffic into different classes or breaking protocol deadlock.

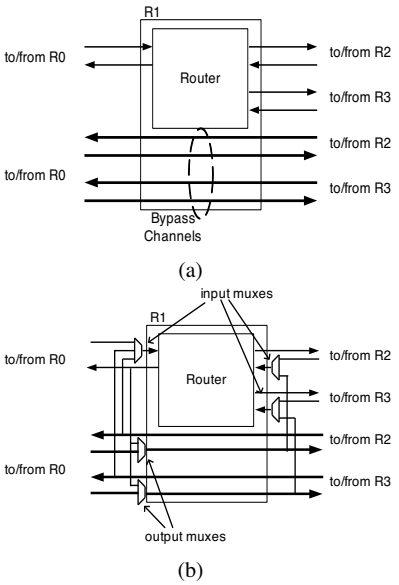

Fig. 3. Flattened butterfly router diagram with bypass channels in a (a) conventional flattened butterfly router diagram and (b) flattened butterfly with muxes to efficiently utilize the bypass channels. The router diagram is illustrated for router R1 in Figure 2 with the connections shown for only single dimension of the flattened butterfly.

III. Bypass Channels and Microarchitecture

As shown in Figure 1, the routers are fully connected along each dimension. Those channels that pass over other routers in the same row or column are *bypass channels*. Using non-minimal routing in the flattened butterfly topology can increase both packet latency and energy consumption because packets are routed to intermediate routers for load-balancing before being delivered to their destinations. The layout of an on-chip network can result in the non-minimal routes *overshooting* the destination on the way to the intermediate node selected for load-balancing, as shown in Figure 2(c). A non-minimal route may also route a packet away from its destination before it is routed back to its destination on a bypass channel that passes over the source (Figure 2(d)). To avoid the inefficiencies of routing packets on paths of non-minimal physical lengths, bypass channels are connected to the routers they pass over. These additional connections allow packets to leave the bypass channels early when doing so is beneficial. In this subsection, we explain how the router microarchitecture and the flow control are modified to connect the bypass channels directly to the router switch in order to reduce latency and power.

A. Datapath

A high-level diagram of a router in an on-chip flattened butterfly is shown in Figure 3(a). It consists of the switch and four bypass channels that connect neighboring routers. [2] One method to connect the bypass channels to the router is to add additional inputs to the switch. However, doing so would significantly increase the complexity of the switch. For example, in the flattened butterfly shown in Figure 1, the

[2]The routers on the corners of the topology do not have any bypass channels.

switch would increase from 10×10 to 18×18 in the worst case, nearly quadrupling the area consumed by the switch. In addition, the use of bypass channels is not intended to increase the bandwidth of the topology, but rather to reduce latency and energy – thus, a larger crossbar switch with additional bandwidth is not needed. To overcome the complexity, we introduce muxes in the datapath of the bypass channels as shown in Figure 3(b).

Two types of muxes are added: input muxes and output muxes. [3] The inputs to the muxes can be classified as either bypass inputs (e.g. inputs from the bypass channels) or direct ports (e.g. inputs/outputs to/from the local router where *local* router is defined as the router neighboring the bypass channels).

Input muxes are added to accept packets destined for the local router that would otherwise bypass the local router enroute to the intermediate node selected by the routing algorithm, as illustrated in Figure 2(c). Thus, the inputs to the input muxes consist of the direct inputs that the packet would have used if the non-minimal path was taken and the inputs from the bypass channels. The output muxes are used for packets that would have been routed indirectly away from the destination before being routed over the local router to its destination, as illustrated in Figure 2(d). The inputs of the output muxes consist of the direct outputs from the local router and the bypass channel inputs – the path the packet would have taken if non-minimal routing path was taken. The addition of these muxes does not eliminate the need for non-minimal routing for load-balancing purpose. Instead, the muxes reduce the distance traveled by packets to improve energy consumption and latency.

B. Mux Arbiter

The arbiters that control the bypass muxes are critical for the proper utilization of the bypass channels. Although a simple round-robin arbiter can be implemented at the muxes, this leads to a locally fair arbitration at the mux but does not guarantee global fairness. To provide global fairness, we implement a *yield* arbiter that yields to the *primary* input – i.e. the input that would have used the mux output bandwidth if the bypass channels were not connected to the local router. As a result, for the input muxes, the direct input is given priority while for the output muxes, the bypass channel is always given priority. Thus, if the primary input is idle, the arbiter grants access to the non-primary input.

In order to prevent starvation of the non-primary inputs, a control packet is sent along the non-minimal path originally selected by the routing algorithm. This control packet contains only routing information and a marker bit to identify the packet as a control packet. The control packet is routed at the intermediate node as though it were a regular packet, which results in it eventually arriving at the primary input of the muxes at the destination router. When the control packet arrives, the non-primary input is guaranteed access

[3]Using the analogy of cars and highways, the long wires introduced correspond to adding highways. The input muxes correspond to adding additional exit ramps to the highways while the outputs muxes correspond to adding entrance ramps to get on the highway.

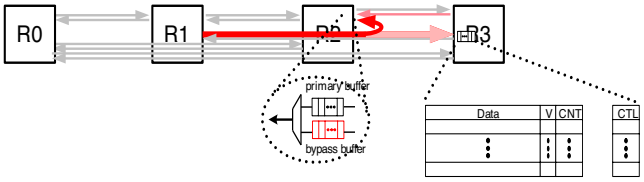

Fig. 4. Modification to the buffers introduced into the flow control with the utilization of bypass channels. The additional bits of the buffers correspond to V : valid bit, CNT : count of control packet, and CTL corresponds to control packet content which contains a destination.

to the mux output bandwidth. In the worst-case (i.e. highly congested) environment, the latency of the non-minimal routed packets will be identical to the flattened butterfly that does not directly utilize the bypass channels. However, there will still be an energy savings because data flits do not traverse the non-minimal physical distance; instead, only the control flit physically travels the non-minimal route.

C. Switch

With minimal routing, the crossbar switch can be simplified because it need not be fully connected (e.g. flits arriving on ports from dimension 1 routers will not be routed to another dimension 1 router). Non-minimal routing increases the complexity of the switch because some packets might need to be routed twice within a dimension, which requires more connections within the crossbar. However, by using the bypass channels efficiently, non-minimal routing can be implemented using a switch of lesser complexity – one that approaches the complexity of a flattened butterfly that only supports minimal routing. If the bypass channels are utilized, non-minimal routing does not require sending full packets through intermediate routers, and as a result, the connections within the switch themselves approaches that of the switch that only supports minimal routing.

D. Flow Control

For proper flow control, bypass buffers need to be added to non-primary inputs of the muxes that are added (Figure 4). Thus with non-minimal routing, credits for the bypass buffers are needed before packet can be routed. In addition, the existing input buffers need to be slightly modified to handle the control packets. The modification introduces a small overhead because of the relatively large width of the datapath in on-chip networks.

IV. Evaluation

We compare the performance of the following different topologies in this section:

1) conventional 2-D mesh (MESH)
2) concentrated mesh with express channels [1] (CMESH)
3) flattened butterfly (FBFLY)
    a) flattened butterfly with minimal routing only (FBFLY-MIN)
    b) flattened butterfly with non-minimal routing (FBFLY-NONMIN)
    c) flattened butterfly with non-minimal routing and use of bypass channels (FBFLY-BYP)

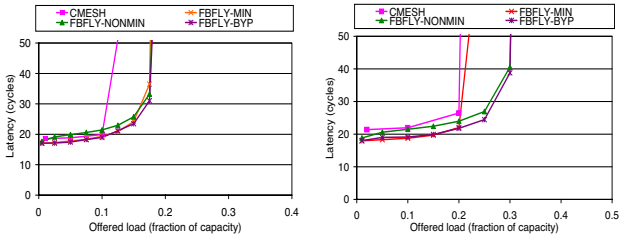

Fig. 5. Throughput comparison of CMESH and FBFLY for (a) tornado and (b) bit complement traffic pattern.

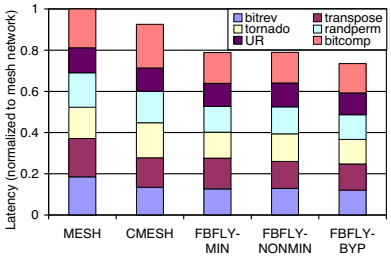

Fig. 6. Latency comparison of alternative topologies across different synthetic traffic pattern.

The topologies were evaluated using a cycle accurate network simulator. We compare the networks' performance and power consumption. The power consumption is based on the model used in [1]. The bisection bandwidth of the different topologies as well as the total amount of buffering across the different topologies are held constant to provide a fair comparison. The router delay is assumed to be 2 cycles for MESH and 3 cycles for CMESH and FBFLY.

In Figure 5, we compare the latency vs. offered load on two adversarial traffic patterns for CMESH and FBFLY – two topologies that utilize concentration to reduce the cost of the network. By exploiting non-minimal routing and the smaller diameter of the topology, FBFLY can provide up to 50% increase in throughput compared to CMESH while providing lower zero-load latency. Although the MESH can provide higher throughput for some traffic pattern, it has been previously shown that CMESH results in a more cost- and energy-efficient topology compared to the MESH [1].

In addition to the throughput measurement, we use a batch experiment to model the memory coherence traffic of a shared memory multiprocessor. Figure 6 shows the performance comparison for the batch experiment and we normalize the latency to the mesh network. CMESH reduces latency, compared to the MESH, by 10% but the flattened butterfly reduces the latency further. With FBFLY-NONMIN, latency increases because of the longer paths employed. FBFLY-BYP provides the load balancing of non-minimal routing while at the same time reducing latency because all packets traverse minimum physical paths. This topology reduces latency by 28% compared to MESH.

The power consumption comparison is shown in Figure 7. The flattened butterfly provides additional power saving, compared to the CMESH. With the reduction in the width of the datapath, the power consumption of the crossbar is also reduced – thus, achieving approximately 38% power reduction compared to the mesh network.

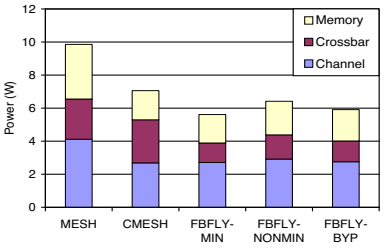

Fig. 7. Power consumption comparison of alternative topologies on UR traffic.

As radix increases, the control logic of the router, such as the allocators, consumes area proportional to the radix of the router. However, they represents only a small fraction of the total area and the buffers and switch dominate the router area. The buffer area can be kept constant as the radix increases by reducing the buffer space allocated per input port. The total switch area can be approximated as $n(bk)^2$ where n is the number of routers, b is the bandwidth per port, and k is the router radix. As k increases, b decreases because the bisection bandwidth is held constant, and n also decreases, because each router services more processors. Consequently, we expect high radix on-chip network will consume less area. We estimate that the flattened butterfly provides an area reduction of approximately 4x compared to the conventional mesh network and a reduction of 2.5x compared to the concentrated mesh.

V. Conclusion

In this paper, we described how high-radix routers can be utilized in on-chip networks to realize reduction in latency and power. By mapping the flattened butterfly topology to on-chip networks and using high-radix routers, the number of routers is reduced, leading to a more efficient network. In addition, we described the utilization of bypass channels to utilize non-minimal routing with minimal increase in power while further reducing latency in the on-chip network. We show that the flattened butterfly increases throughput by up to 50% compared to CMESH, reduces latency by 28% compared to MESH, and reduces power consumption by 38% compared to MESH.

References

[1] J. Balfour and W. J. Dally, "Design tradeoffs for tiled CMP on-chip networks," in *ICS '06: Proceedings of the 20th annual international conference on Supercomputing*, 2006, pp. 187–198.

[2] L. N. Bhuyan and D. P. Agrawal, "Generalized Hypercube and Hyperbus Structures for a Computer Network." *IEEE Trans. Computers*, vol. 33, no. 4, pp. 323–333, 1984.

[3] W. J. Dally, "Virtual-channel Flow Control," *IEEE Transactions on Parallel and Distributed Systems*, vol. 3, no. 2, pp. 194–205, 1992.

[4] W. J. Dally and B. Towles, "Route Packets, Not Wires: On-chip Interconnection Networks," in *Proc. of the 38th conference on Design Automation (DAC)*, 2001, pp. 684–689.

[5] J. Kim, W. J. Dally, and D. Abts, "Flattened Butterfly : A Cost-Efficient Topology for High-Radix Networks," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, San Diego, CA, June 2007.

[6] J. Kim, W. J. Dally, B. Towles, and A. K. Gupta, "Microarchitecture of a High-Radix Router," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, Madison, WI, 2005, pp. 420–431.

[7] A. Kumar, L.-S. Peh, P. Kundu, and N. K. Jhay, "Express Virtual Channels: Towards the Ideal Interconnection Fabric," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, San Diego, CA, June 2007.

[8] A. Singh, "Load-balanced routing in interconnection networks," Ph.D. dissertation, Stanford University, 2005.