Design of a Performance Enhanced and Power Reduced Dual-Crossbar Network-on-Chip (NoC) Architecture

Yixuan Zhang, Randy Morris Jr, and Avinash K. Kodi

School of Electrical Engineering and Computer Science
Ohio University, Athens, OH 45701

Abstract

The input buffers of the current packet-switched Network-on-Chip (NoC) architectures consume a significant portion of the total power of the interconnection network. Reducing the size of input buffers would result in degraded performance, while eliminating all buffers would result in increased power at high network load. In this article, we propose DXbar: an innovative dual-crossbar design. By combining the advantages of buffered and bufferless networks, we achieve at least 20% performance improvement in terms of throughput and latency, and at least 20% power saving over buffered networks with virtual-channels. Furthermore, DXbar can outperform current bufferless networks with deflecting and dropping protocols while consuming at most half of the power.

Keywords:
Dual-Crossbar, Power Reduction, Network-on-Chip

1. Introduction

The International Technology Roadmap for Semiconductors (ITRS) predicts that hundreds of cores will be deployed on a single die in future Chip-Multiprocessors (CMPs) designs. In order to resolve the problems of shared buses such as scalability issue and limited bandwidth [1], the promising packet-switched Network-on-Chip (NoC) have emerged to be a solution for designing interconnection fabric, with the objective of providing a low-latency,
low-power, high bandwidth communication medium for CMPs [2, 3, 4, 5, 6, 7]. However, researchers are continuously confronted by one of the major challenges: reducing power dissipation in the network [8]. For instance, the power consumption of the interconnection network of Intel TeraFLOPS processor exceeds 28% of the total chip power, while the target budget of power dissipation was only 10% [9]. Moreover, the interconnection network of MIT RAW chip consumes almost 40% of the total chip power [10].

The power dissipation of modern NoC architectures are mainly characterized by the power consumed in the links, crossbars, and input buffers, which are used for data transmission, switching, and storing, respectively. Amongst these components, input buffers alone could consume up to 46% percent of the total power of the whole interconnection network [11]. As a result, reducing the size of input buffers or completely eliminating input buffers is a natural approach to design low-power NoC. On the other hand, increasing the size of an input buffer will lead to more upstream packets being transmitted and buffered, thus increasing the throughput. Therefore, simply reducing the size of input buffers in each router may result in a degraded performance, as the performance of an interconnection network is primarily affected by the size of the buffers [12].

In this paper, we propose DXBar, a high-performance and power-efficient dual-crossbar design targeting both performance enhancement and power reduction. In order to take the advantage of power-efficient and low-latency bufferless networks for low network load, and cost-efficient conflict-handling capability at high network load, the design becomes a combination of a bufferless primary crossbar and a buffered secondary crossbar. At a low network load, almost all packets will only traverse through the primary crossbar and take minimal routes, experiencing similar delays of what the packets will experience in a bufferless network. Whenever there is a conflict, the losing packet will be buffered in the secondary crossbar, and wait until its desired output port is free. The introduction of the secondary crossbar enables blocked packets to be removed from the critical path of the primary crossbar, thus maintaining the continuous flow of packets and achieving low-latency. Moreover, packets which are not buffered simply traverse a switch, therefore the power of buffer read/write is saved. As packets do not take extra hops when conflicts arise, there are no extra link traversals when the network load keeps increasing, ensuring that the design is power-efficient for any network load.

The major contributions of this work are as follow:
We implement a primary crossbar with no input buffers. The bufferless network can provide simple and efficient switching function for packets, without any buffer operations. Packets traversing the primary crossbar only need one cycle to be switched. The elimination of buffers also results in a decreased power consumption and latency. Compared to the generic routers, the proposed approach reduces power by 20% for Uniform Random traffic pattern.

- We also implement a secondary crossbar with buffers. Packets do not need to take extra hops to solve conflicts because of the secondary crossbar, therefore the power consumption at either low or high network load is reduced. Compared to bufferless networks at high network load, the proposed approach reduces power considerably.

- We utilize a dual-crossbar design to improve the performance. The secondary crossbar enables two flits from the same input port to go to different output ports at the same time. Cycle-accurate simulation on $8 \times 8$ 2D mesh topology shows that DXbar has at least 20% performance improvement over the generic routers with synthetic traffic patterns and real-application traces from SPLASH-2. The proposed design is also able to outperform current bufferless networks.

- We further exploit the possibility of hardware-level fault tolerance with fault-detecting units and control units deployed. The failure of either crossbar will not disable the router, and all packets can be routed through the working crossbar with appropriate modification in router architecture and switch allocation components.

The rest of the article is organized as following. Section 2 introduces the related works. Section 3 illustrates the architecture of the design and how it works in details, such as how packets are routed and so on. Section 4 is for the methodology used in performance analysis, the evaluation results, and the discussion. Section 5 will conclude the paper.

2. Related Work

Different techniques have been proposed to reduce or eliminate the size of input buffers. Recently, iDEAL (inter-router Dual-function Energy and Area-efficient Links) proposed to reduce the size of the input buffers and
utilize repeaters along inter-router channels as storage units when needed by appropriately controlling functionality of the dual-function repeaters [11]. The design could maintain similar performance with only half the number of input buffers, but with the cost of increased latency and complexity. Another approach utilizing channel buffering is the Elastic Channel Buffers (ECB), which replaces all the repeaters with flip-flops, and eliminates the input buffers [13, 14]. In this design, all packets are transmitted from one channel buffer to the next by a handshaking protocol. The clock-based nature of flip-flops results in excessive link delay even if all the buffers along the channels are completely free.

Bufferless routing is another novel and unique approach which eliminates all input buffers without utilizing channel buffering. Flit-Bless proposed a routing scheme to send all incoming packets to output ports, irrespective of the fact whether those output ports are productive [15]. The age-based priority for arbitration indicates that the oldest incoming packet is guaranteed to be routed to its productive output port, while younger packets may be deflected to their non-productive output ports and take non-minimal numbers of hops before reaching their destinations. Another design which improves upon bufferless routing is SCARAB (Single Cycle Adaptive Routing and Bufferless Network), where packets are minimally-adaptively routed [16]. If none of the productive output ports are available, the packet will be dropped, and a NACK signal will be transmitted through a dedicated circuit-switched NACK network to trigger a retransmission, saving the bandwidth of the data network. Both designs can achieve substantial power saving and latency reduction at low network load. At higher network load, conflicts are more frequent and packets experience much more link and crossbar traversals, due to the increasing rate of deflections or retransmissions. This leads to significantly higher power consumption, and could exceed the network power budget.

Other designs targeting power saving with router design have different approaches. A dynamic buffering resources allocation design named ViChaR (dynamic Virtual Channel Regulator) focuses on efficiently allocating buffers to all virtual channels, by deploying a unified buffering unit instead of a series of separated buffers, and minimizing the required size [17]. Express Virtual Channels (EVCs) enables packets to bypass buffer operations in the intermediate nodes, thus saving power which would have been consumed by buffers to write/read [18]. These two designs could significantly improve power consumption, but with increased router complexity. A low-cost router micro-
architecture has been proposed to partition the crossbar into two decoupled low-radix sub-crossbars for the two dimensions, and connect them with an intermediate buffer queue [19]. Packets traversing in their own dimensions are prioritized to encounter minimal delay, while turning packets need to wait in the intermediate buffer queues until no more in-flight packets occupy the desired output ports. Similar to bufferless designs, low network load could yield satisfactory performance on this design, while the intermediate buffer queues become the prominent bottleneck as there will be more turning packets when network load increases. RoCo, which is also a dual-crossbar design, decouples the crossbar similarly into two sub-crossbars for two dimensions [20]. The design utilizes smaller crossbars to reduce crossbar power, and the dual-crossbar design enables fault-tolerance. However, the wiring in front of the two crossbars is complex, and buffer power is not reduced. Another design which utilizes dual-crossbar is called Distributed Shared-Buffer, in which buffer slots are located between two crossbars in order to rearrange packet sequences, so that conflicts can be avoided and packets would be switched more quickly [21]. This design of dual-crossbar targets improving throughput, yet increases power consumption considerably, as each flit needs to traverse two crossbars in the same router.

3. Design of DXbar

3.1. Architecture

In this section, we demonstrate the architecture of the DXbar router. The Figure 1(a) illustrates the router architecture of the state-of-the-art
packet-switched network. Each router has five input ports, and five output ports, from and to 4 cardinal directions, north, south, east, and west, plus the processing element (PE). Each input port has 16 buffer slots with four virtual-channels (VCs), and a pair of multiplexer and de-multiplexer. The four VCs are sandwiched between the de-multiplexer connected to the input port, and the multiplexer connected to the crossbar. Each input unit can communicate with router, virtual-channel allocator, and switch allocator, which are responsible for Routing Computation (RC), Virtual-Channel Allocation (VA), and Switch Allocation (SA), respectively. The crossbar is controlled by the switch allocator for correctly connecting input ports to output ports [12, 22].

Figure 1(b) illustrates the router architecture of DXbar, which is quite different from that of the baseline VC routers. There are two crossbars within each router, with the primary crossbar having four input ports, the secondary crossbar having five input ports, and both of them having five output ports. The four input links are connected to both crossbars via de-multiplexers, and the injection port of the PE is connected to the last input port of the secondary crossbar. All of the five output ports of those two crossbars are connected via multiplexers, which are then connected to the output links of the four directions and the receiving port of the PE. Four buffers are located in between the de-multiplexer and the first four input ports of the secondary crossbar. There are no buffers deployed between the injection port of the PE and the secondary crossbar. The buffer slots are connected serially, thus eliminating VCs and the corresponding virtual-channel allocator. Look-ahead signal, which carries the routing information of the incoming flit, is connected directly to routing computation, which is then connected to the switch allocator. Switch allocator is modified to control the de-multiplexers, the crossbars, and the multiplexers to maintain the correct packet flow in both crossbars.

3.2. Router Implementation

Figure 2(a) shows the 5-stage router pipeline of the baseline design. The stages of router pipeline are as follows: 1) buffer write (BW) and routing computation (RC) for head flits, 2) virtual-channel allocation (VA) and switch allocation (SA) which are performed in parallel, 3) switch traversal (ST). An extra link traversal (LT) stage is needed to transmit a flit to the downstream router. Figure 2(b) shows the reduced router pipeline of the baseline design using look-ahead routing. RC is done at the previous router, and the result
is written to the flit and transmitted to the current router so that no RC is needed for switching the flit to the next router, reducing the number of stages from 5 to 4. Figure 2(c) shows the further reduced router pipeline of the baseline design via speculation. Speculative SA is performed in order to finish both VA and SA in the same cycle. This reduces the number of stages from 4 to 3, and this pipeline is used as the baseline design in this article. Figure 2(d) shows the pipeline of DXbar router. By using look-ahead signal, the routing information can traverse the link one cycle ahead of the flit, and routing decision can be made before the flit actually arrives. As a result, other than the source node, RC stage does not occupy a dedicated cycle. Look-ahead routing has been proposed previously [23, 24, 12]. Moreover, the elimination of VCs and buffers enables the removal of BW and VA stages. Therefore, SA/ST is the only stage in the router pipeline of DXbar. The simplification of router pipeline, which is possible by eliminating buffers in the primary crossbar, reduces the number of cycles to switch a flit from three to only one. This proposed pipeline is similar to those used in Flit-Bless [15] and SCARAB [16]. These two bufferless designs both use only ST stage to switch packets, and use another LT stage to transmit packets.

![Diagram](image)

Figure 2: The illustration of (a) 5-stage basic pipeline (b) 4-stage pipeline with look-ahead routing (c) 3-stage pipeline with speculation (d) proposed DXbar pipeline. The stages are buffer write (BW), routing computation (RC), virtual-channel allocation (VA), switch allocation (SA), switch traversal (ST), and link traversal (LT).

The flits arriving at the de-multiplexers are called incoming flits. When
a incoming flit arrives, it would compete with other incoming flits for the output ports, using the routing decisions made by the router one cycle ahead. If the flit wins in the arbitration, it could traverse the primary crossbar and go to its designated output port. Otherwise, it would be sent to the secondary crossbar and be buffered. Figure 3 shows the timing of an incoming flit which wins in arbitration and traverses the primary crossbar. Flits waiting at the head of buffers and in the injection port are called buffered flits. Those flits could compete in arbitration as well, but they have lower priority than incoming flits. Once a flit wins in arbitration, the look-ahead signal leaves the router and is transmitted to the downstream router, while the flit is traversing the crossbar. The de-multiplexers and multiplexers are set to select the primary crossbar inputs/outputs by default. Switch allocator uses the result provided by the arbitration to not only set the two crossbars to connect input ports and output ports correctly, but also control all of the multiplexers and de-multiplexers to link each output port to the correct crossbar, and to change the direction of incoming flits between the primary and the secondary crossbar, respectively.
At the beginning of each cycle, one list is maintained to record the routing decisions of the incoming flits of the cycle, and another one is maintained for the buffered flits. The arbitration is based on the age of the flits. Therefore, flits are ranked based on their ages in both lists. Furthermore, incoming flits have higher priority than buffered flits, which helps to reduce the chance for flits to be buffered. For example, there are two incoming flits A and B, and three buffered flit C, D, and E. A is 24 cycles old, B is 10 cycles old, C is 20 cycles old, D is 18 cycles old, and E is 28 cycles old. C, D, and E are older than B, but are ranked lower than B, as an incoming flit has a higher rank than a buffered flit. Therefore, the final rank is A, B, E, C, and D. The allocation starts with the highest ranked flit to assign to each flit the desired output port, which has not been assigned to other higher-ranked flits. Using the same example, the desired output ports of A, B, C, D, and E are x+, x+, x-, y+, and x-, respectively. The switch allocator assigns x+ to A, x- to E, and y+ to D. As x+ and x- have been assigned to other flits, B and C lose in arbitration. Figure 4 illustrates the allocation process within the tables. The age-based arbitration ensures that flits would be buffered for finite time, as older flits in the network have greater possibility to win in arbitration, thus using the primary crossbar without being buffered. Credit-based buffer management is adopted in this design. When a flit departs from the output, the number of credits is decreased by one. This would happen for all of the flits leaving the router, as it is not certain that whether those flits would be
buffered. When a flit wins in the arbitration at the downstream router, a credit is sent back and the number is increased by one. As another buffered flit, coming from the same direction, could be sent to the secondary crossbar simultaneously, it is possible that two credits are received within a single cycle.

DXbar uses buffered secondary crossbar to enable bufferless routing in the primary crossbar, without using misrouting or dropping. More importantly, the buffered flits could use the secondary crossbar to go to any occupied output ports. Being different from simple buffer bypassing, this dual-crossbar design does not have contention between bufferless routes and buffer slots for the input ports of the crossbar. As a result, buffered flits increase the probability to win in arbitration. The removal of VCs could potentially decrease the performance, and prevents the network from providing traffic classes, or utilizing some of the adaptive routing algorithms. However, the purpose of deploying buffer slots is to handle conflicts in a bufferless network without using misrouting or dropping to save power. Therefore, less complex buffer organizations are preferred. Moreover, the extra crossbar increases the speed of progressing buffered flits. In conclusion, removing VCs is suitable for the proposed dual-crossbar design. Another issue is that each flit of a packet needs to be a head flit, as it is possible to have out-of-order receiving of flits. If in-order transmission is used, all of the flits of a packet which loses in arbitration should be buffered, thus consuming all of the buffers, and stopping the transmission from upstream. This is the reason for each packet to have only one flit. As a result, the power is slightly increased due to extra computations, and extra receiving buffers.

3.3. Walkthrough Examples

Figure 5 shows several possible scenarios to illustrate how flits are switched in DXbar design.

- Figure 3.3(a) shows the scenario when there is no conflict among packets. Flits from the x+ and the x- want to go to the x- output port and x+ output port, respectively. At the same time, flits from the y+ and the y- want to go to the y- output port and y+ output port, respectively. All of the four incoming flits are switched simultaneously, and the network operates similarly to bufferless networks, because no packets are buffered and latency is minimized. This is the best scenario.
Figure 5: The illustration of dual xbar: (a) No conflict appears, the network operates as bufferless network. (b) Conflict appears, one flit is sent to the secondary crossbar and buffered first. (c) The flit coming after the buffered flit sees no delay and can proceed without being buffered, thus ensuring no instant back pressure. The injection port can send a flit whenever the desired output port is not occupied. (d) Even though there is still a flit coming in through the same input port, the buffered flit can proceed when the desired output port is not occupied by flits from the input ports. The incoming flit can proceed simultaneously, encountering no delay.

- Figure 3.3(b) shows the scenario when conflict appears. A flit from the y- input port and another flit from the x+ input port compete for the x- output port. Assume the flit from x+ is older, then that flit is switched from x+ input port to the x- output port through the primary crossbar. The flit from the y- input port loses in arbitration, and the y- input de-multiplexer receives a control signal from switch allocator, routing the flit from the y- input port to the secondary crossbar and buffering it.
- Figure 3.3(c) shows the scenario when another flit comes from the y-input port in the next cycle. Because the flit coming from the y-input port one cycle ago is currently buffered in front of the secondary crossbar, the path from the y-input port to the primary crossbar is unoccupied. Therefore, the incoming flit in the current cycle can be switched to the desired output port. Otherwise, it needs to wait in buffer until the previous packet is switched. The figure also illustrates how the injection ports work. Since the injection ports have the same level of priority as buffers, a flit can be injected whenever the desired output port is not occupied.

- Figure 3.3(d) shows the scenario when no more flits from the input ports need the x-output port. The buffered flit can then traverse the secondary crossbar and be routed downstream. At the same time, a flit from the y-input port can traverse the primary crossbar to a different output port. This is not feasible in other designs, because there can be only one flit coming from one particular input port at any single time.

3.4. Fairness Maintenance

Because the arbitration is age-based, flits coming from the nodes on the edges of the mesh network will always have higher priority when they pass through the nodes in the center. As a result, the flits injected by center nodes are more likely to lose in arbitration and traverse the secondary crossbar. More importantly, this scenario could limit the injection of flits of center nodes, as injection ports and buffers have lower priority than those of input ports. A fairness issue thereby arises, since flits can be stalled in the injection ports or buffers for long cycles, when the desired output ports are taken by other flits from the input ports.

In order to maintain fairness between the primary crossbar and the secondary crossbar, a fairness counter is maintained in each router to count the number of times flits from the primary crossbar win consecutively in arbitration. The counter works only when there are flits waiting in the buffers or in the injection port, and it is reset every time a waiting flit wins. Once the number exceeds a threshold, the priority flips so that flits from the secondary crossbar and the injection port have higher priority, so that they will be assigned output ports ahead of the flits from the input ports. Although the continuous flow of flits on the primary crossbar is interrupted, the nodes
in the center are able to inject new flits even if the network is heavily saturated, and the buffered flits are guaranteed to leave in finite number of cycles. Setting the threshold too small can lead to difficulty covering the round-trip delay of credits, while setting the number too large does not help to solve the fairness issue. After testing with different traffic patterns, the threshold is set to four to bring the overall best performance.

3.5. Routing Algorithms

In this design, two different routing algorithms are tested for the Routing Computation component, which are Dimensional-Order Routing (DOR) and West-First adaptive routing (WF). DOR algorithm lets packets travel on the x-dimension first, and then turn to the y-dimension when the current position of the x-dimension equals to the position of the x-dimension of the destination. Because packets always finish going along the x-dimension first, using DOR means that no cyclic dependency can happen, thus ensuring deadlock-free. WF routes packets along the west direction first, then turns those packets either north or south. The adaptation happens only when the starting positions of x-dimension of the packets are to the west of the positions of x-dimension of their destinations, as they can travel freely between going north and going east, or going south and going east. As turning from going north to going west is prohibited, cyclic dependency is not possible.

3.6. Fault Tolerance

The duplication of crossbar enables hardware-level fault tolerance. The router can avoid being completely disabled, even in case of a permanent crossbar failure. If such problem occurs, the routers in DXbar design can abandon the bufferless routes, and switch from normal operation to operating similarly to a buffered network. Because both the primary and the secondary crossbar may fail, there should be a mechanism to alter the connection between the crossbars and the buffers. Figure 6 depicts the revised router architecture of DXbar. A set of $2 \times 2$ small crossbars are deployed between the buffers and the main crossbars, and the links connecting the input ports to the primary crossbar are cut into two parts. Controlled by the signals from the switch allocator, these small crossbars can connect the buffers to the primary crossbar, thus altering the outputs of those buffers. Under normal operation, the small crossbars are set so that buffered packets will traverse the secondary crossbar. If the primary crossbar fails, all of the incoming packets will be switched through the buffers and the secondary crossbar, and the routers can
still be functional. If the secondary crossbar fails, all of the incoming packets will also be buffered first. The difference is that this time the small crossbars will alter the connection under the control of the switch allocator, and buffers will be connected to the primary crossbar. Either scenario makes the buffer operations inevitable, which will degrade performance and consume more power. Nevertheless, the routers can still operate while one of the two crossbars is failed.

4. Performance Evaluation

In this section, we evaluate the DXbar design in term of area overhead, power dissipation, and overall network performance. We compare DXbar with 3-stage pipelined generic routers. The setting of 4 buffer slot per VC and 1VC per input port is called buffered 4, and the other setting of 4 buffer slots per VC and 2 VCs per input port is called buffered 8. These two settings are chosen in order to provide a impartial comparison, as DXbar only has 4 buffer slots per input port. We also compare DXbar with Flit-Bless [15] and SCARAB [16], because they are closely related. All of the designs are implemented in a 8 × 8 2D mesh topology. Two different routing algorithm are considered for DXbar, dimensional-order routing (DOR) and west-first adaptive (WF) routing, which are called DXbar DOR and DXbar WF, respectively. The power model provided by Orion 2.0 simulator is used
for the area and power estimation, with the setting of 65 nm technology, 1Ghz router at 1.0V_{dd}, and 128 bits flit size.

4.1. Methodology

We use a cycle-accurate NoC simulator to perform the detailed evaluation of DXbar, as well as other related designs. For synthetic traffic patterns, each packet consists of 1 flit. The network load varies from 0.1 to 0.9 of the network capacity, and packets are injected according to the Bernoulli process based on the given network load. All the designs are simulated to run nine different synthetic traffic traces, which are Uniform Random (UR), Non-Uniform Random (NUR), Bit Reversal (BR), Butterfly (BF), Complement (CP), Matrix Transpose (MT), Perfect Shuffle (PS), Neighbor (NB), and Tornado (TOR). UR and NUR are random traffic patterns, meaning that each node randomly chooses the destination to send a packet. The difference is that each node when running NUR has a possibility of 25% to choose destinations within certain nodes. As a result, NUR creates hot-spots where most of the packets are transmitted to, and causes heavier congestion at certain locations than UR does. The rest of the traffic patterns are permutation patterns, meaning that each node chooses a fixed destination based on the given permutation for all packets.
Table 2: Cache and memory parameters used for Splash-2 suite simulation.

<table>
<thead>
<tr>
<th>L2</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caches</td>
<td>16</td>
</tr>
<tr>
<td>Cache size</td>
<td>1 MB</td>
</tr>
<tr>
<td>Cache associativity</td>
<td>16-way</td>
</tr>
<tr>
<td>Cache access latencies</td>
<td>4 cycles</td>
</tr>
<tr>
<td>Cache write-back policy</td>
<td>Write-back</td>
</tr>
<tr>
<td>Cache block size</td>
<td>64B</td>
</tr>
<tr>
<td>MSHR entries</td>
<td>16</td>
</tr>
<tr>
<td>Coherence Protocol</td>
<td>MESI</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Main Memory</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory controllers</td>
<td>16</td>
</tr>
<tr>
<td>Memory size</td>
<td>4GB</td>
</tr>
<tr>
<td>Memory latency</td>
<td>160 cycles</td>
</tr>
<tr>
<td>Directory latency</td>
<td>80 cycles</td>
</tr>
</tbody>
</table>

Besides, SPLASH-2 traffic traces are also used in the simulation. The SPLASH-2 traffic traces were collected using the full system simulator Simics [25] with the GEMS memory module enabled for accurate cache coherence and memory access latencies [26]. The processor model used in our simulations is based on the Ultra SPARC architecture given in [27]. Table 1 gives important processor parameters, and Table 2 gives important L2 cache and memory parameters. From Table 1, each processor can issue two threads and has its own 64 KB private L1 instruction and L1 data cache. From Table 2, each processor has its own private 1 MB L2 cache and uses the MESI protocol to maintain cache coherence. After collecting the SPLASH-2 traffic traces, they run on the cycle-accurate NoC simulator. The following benchmark and input parameters are used for the select SPLASH-2 applications: FFT (16 K), LU (512 × 512), radiosity (largeroom), Ocean (258 × 258), Raytrace (Teapot), Radix (1 M), Water (512), FMM (16K) and Barnes (16 K).

4.2. Area and Power Estimation

Table 3 shows the area and power estimations of each design. Flit-Bless and SCARAB have the least area overhead, while DXbar has the most. This is due to the fact that the crossbar occupies the largest area in a router, and the secondary crossbar in DXbar router makes the area overhead exceeds those of buffered 4 and buffered 8 designs. In fact, as the technology scales,
Table 3: Area and power estimation for different designs in a 8×8 2D mesh network. The crossbar power is 159pJ/flit, and the link power is 89pJ/flit.

<table>
<thead>
<tr>
<th>Design</th>
<th>Area (mm²)</th>
<th>Buffer Power (pJ/flit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flit-Bless</td>
<td>0.302</td>
<td>0</td>
</tr>
<tr>
<td>SCARAB</td>
<td>0.302</td>
<td>0</td>
</tr>
<tr>
<td>Buffered 4</td>
<td>0.346</td>
<td>58.143</td>
</tr>
<tr>
<td>Buffered 8</td>
<td>0.389</td>
<td>160.163</td>
</tr>
<tr>
<td>DXbar</td>
<td>0.648</td>
<td>58.143</td>
</tr>
</tbody>
</table>

the size of crossbar will shrink to lessen the negative impact on area brought by the secondary crossbar. Besides, the secondary crossbar provides much more important power benefit, which will be illustrated in the next section. The leakage power for each crossbar is 5.17mW, therefore adding the second crossbar does not impose a negative impact on overall power due to leakage power.

4.3. Synthetic Traffics

Figure 7 shows the throughput plot for UR. DXbar is significantly better than bufferless designs and buffered networks. DXbar DOR brings the best performance in terms of throughput and latency, and has a saturation point at over 0.4. This has a 20% improvement over buffered 8, and a 40% improve-

Figure 7: Throughput of Uniform Random traffic pattern.
ment over buffered 4, Flit-Bless, and SCARAB. DXbar WF is slightly worse, but it is still able to completely outperform all other designs, and it has a saturation point at around 0.38. This equals a 15% improvement over buffered 8, and a 38% improvement over buffered 4, Flit-Bless, and SCARAB. Even though the buffer size is twice as many as DXbar, buffered 8 still cannot outperform. On the other hand, Flit-Bless and SCARAB perform similarly, but their saturation points are below 0.3, which are far behind that of DXbar.

Figure 8 shows the power plot for UR. Even though buffered 4 has the same buffer organization as DXbar, it uses buffer every time and consumes more power than DXbar. Buffered 8 has a buffer organization which is more power consuming. Therefore the power consumption of buffered 8 is higher than buffered 4. Flit-Bless and SCARAB use as little power as DXbar does at zero load. However, they consume tremendously higher power when the offered network load goes near or beyond their saturation points. SCARAB has an increase of about 2X, and Flit-Bless even has an increase of about 3X. This indicates that at high offered network load, the way bufferless networks handle conflicts sacrifices power consumption, and they actually consume more power than the generic buffered router does when offered network load increases. Nevertheless, the power consumption for DXbar hardly changes when the offered network load increases, and the only little difference comes from the fact that a portion of packets are buffered, but only when they

Figure 8: Power of Uniform Random traffic pattern.
are relatively young. The result from simulation shows that the chance for the packets to be buffered while traversing through a router is only $1/6$ after saturation point. This gives DXbar a huge advantage on power saving, compared to not only bufferless networks, but also generic routers, which put packets in buffers every time.

The figure 9 and 10 shows the throughput and power for all synthetic traffic patterns at an offered load $= 0.5$. It is clear that for UR, NUR, CP, and TOR, DXbar DOR performs the best. For BR, BT, MT, and PS, which all favors adaptive routing algorithms, DXbar WF is very competitive. DXbar uses the least power, while Flit-Bless uses the most, SCARAB the second, and the generic routers lie in between.
4.4. Real-Application Traces

SPLASH-2 traces are used in this part of the simulation. Before the simulation of each trace, each node generates a list of requests by dividing the whole trace. The simulation starts with each node sending 1-flit request packets to destinations according to their own lists. Upon receiving a request, the node responds by temporarily halting sending requests, and replies with a 5-flit data packet to the requesting node. The simulation could be considered as a heavy load test for all the designs. Rather than generating synthetic traffic patterns, it forms patterns according to the traces. The performance is measured by the time to finish all the transactions, indicating that the least time required implies the best performance.

Figure 11 and 12 shows the normalized execution time and power for
all SPLASH-2 traces. The results show that under the current simulation circumstance, DXbar DOR performs better than DXbar WF. For most of the traces, DXbar can achieve the best performance. In this simulation, Flit-Bless and SCARAB can really keep up with DXbar for some of the traces, and can even do better for FFT. The reason is explained by the difference in power consumption between DXbar and bufferless networks. The way this simulation is performed somehow gives Flit-Bless and SCARAB advantage, because of their adaptiveness. This is reflected by the exceedingly higher power consumption of Flit-Bless and SCARAB for all the traces, which is at least 16X and 2X higher than the power consumption of DXbar, respectively. In order to let excessive packets proceed, Flit-Bless deflects for an average of almost 50 times per packet transmitted, and SCARAB drops packets at a high rate. This results in tremendously higher power consumption, but also relieves congestion at some locations. This concludes that DXbar design is able to provide superior performance and decrease power consumption simultaneously.

5. Conclusion & Future Work

In order to decrease the power consumption of the NoCs under any condition, and to enhance the performance, we propose DXbar design which combines an secondary crossbar and additional buffers, with the bufferless primary crossbar. The simulation results show that by using the buffers and the secondary crossbar to handle excessive packets when network load is high, DXbar not only is able to easily beat the baseline design with the same buffer size per input port, but it is also better than the baseline with doubled buffer size, as well as the currently proposed bufferless networks. More importantly, DXbar achieves 15% power saving over the baseline design, and 60% and 40% over Flit-Bless and SCARAB, respectively. In general, DXbar design achieves the goal to improve performance and save power, with the tradeoff of area, which is easier to compensate with technology scaling.

While DXbar achieves higher throughput and saves power with dual crossbar and the combination of bufferless and buffered networks, the proposed design also has two important shortcomings. One, is the elimination of VCs which prevent the ability of the network to provide different service guarantees, inability to use adaptive networks and provide different virtual networks to avoid protocol (cache coherence) deadlocks. This can be addressed by providing elastic channel buffers on the bufferless link and creating two virtual
networks. The bufferless link can be adapted with storage when needed and will be addressed in the future. Second, is the area overhead of using dual crossbars. This can be addressed by reducing the number of crossbar input or output ports, segmenting the wires connecting the input to the output or using a single crossbar with dual inputs. We will address these issues in the future work involving NoC architectures.

Acknowledgement: This work was supported in part by the National Science Foundation (NSF) grants CCF-0915418, CCF-0538945, CCF-0915537 and ECCS-0725765.

References


