SLIDER: Smart Late Injection DEflection Router for Mesh NoCs

Bhawna Nayak* nayakbhawna06@gmail.com
John Jose*† johnj@rajagiritech.ac.in
Madhu Mutyam* madhu@cse.iitm.ac.in

† Department of Computer Science and Engineering, Rajagiri School of Engineering & Technology, Kochi, India.
* Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, India.

***Abstract*—Network-on-Chip (NoC) provides a scalable communication interface for processing cores in large multicore systems. An efficient NoC router should not only minimize the average packet latency of the network but also have minimum pipeline latency, area, and power. Area and power overheads are affecting the scalability and popularity of traditional input buffered routers. In this context minimally buffered deflection routers are emerging as a cost effective alternative.**

We propose SLIDER, Smart Late Injection DEflection Router, that uses side buffers for accommodating a fraction of deflected flits. The main contributions of this work are *smart late injection* and *selective flit preemption*. In SLIDER the injection stage is kept at the end of the router pipeline. This reduces the contention in the arbitration stage, eliminates unwanted intra-router movement of flits and effectively utilizes the idle output channels. We parallelize independent operations in the router pipeline and reduce the pipeline latency by 25%. Experimental results on synthetic and real workloads show that SLIDER reduces average flit latency, channel wastage, and deflection rate, and increases throughput in the network when compared to the state-of-the-art minimally buffered deflection routers.

I. Introduction

NoC architectures are proposed to replace design specific global on-chip wiring with general purpose on-chip interconnection network in multicore systems. Scalable packet switched NoCs offer higher bandwidth compared to traditional bus based communication [1]. Router micro-architecture plays a vital role in the performance of an NoC. Though the conventional buffered routers are giving very good performance in terms of average packet latency and throughput, area and power overheads associated with buffers are affecting their scalability and popularity [2]. Approximately 30% to 40% of chip power is consumed by the NoC, out of which a significant contribution is by the router buffers [3], [4].

Buffer-less routers are proposed to address the rising power concerns associated with the traditional input buffered routers [5]. In buffer-less NoC routers, flits failing out in port contentions are handled by two approaches: deflection and dropping. The deflection approach [2], [5] is preferred to dropping approach [6], [7] due to the later's overhead in managing the retransmission requests for the dropped flits.

Buffer-less deflection router for NoC is first proposed in [8]. A detailed implementation and analysis are discussed in the design of BLESS router [5]. The sequential port prioritization mechanism that increases the critical path delay of the BLESS router pipeline is replaced by a parallel port allocation scheme with a better pipeline stage delay in CHIPPER [2]. CHIPPER experiences very high deflection rate because it guarantees productive port only for the highest priority flit. The radial deflection algorithm with sequential port allocation [9] reduces the formation of hotspots in the center of the mesh network by deflecting flits towards the edges and corners.

At higher loads performance of buffer-less routers reduces significantly. Even at medium load, in few occasions port contentions result in too many deflections, leading to longer delay in delivering flits at the destination. This has triggered the need for adding a minimal set of buffers to the deflection routers. Rather than using buffers in input ports, side buffers are introduced to accommodate a fraction of deflected flits in MinBD router [10] and DeBAR [11]. These side buffers bring down the deflection rate of flits to a significant extent as compared to the totally buffer-less CHIPPER. The buffered flits are re-injected to the router pipeline and arbitrate to get their desired port in the subsequent cycles.

Switching buffered router to buffer-less mode under low network load by power gating is explored in Automatic Flow control [12]. The Flexi-buffer design [13] employs fine grained power gating by adaptive adjustment of the count of active buffers. Both these techniques achieve dynamic power reduction, but the router area remains the same due to the presence of buffers. An exhaustive study on the congestion issues in buffer-less NoCs is carried out in [14] and in [15].

In this paper, we propose SLIDER, a new novel deflection router with a side buffer that offers significant reduction in terms of router pipeline latency and power consumption than DeBAR [11], state-of-the-art minimally buffered deflection router. The concepts of *smart late injection* and *selective flit preemption* effectively coordinate intra-router and inter-router flit movements and make SLIDER a superior design choice for deflection routers.

The rest of the paper is organized as follows. Section II discuss the motivation for this work followed by the architecture discussion of the SLIDER in Section III. Section IV describes the experimental setup followed by result analysis in Section V and we conclude the paper in Section VI.

978-1-4799-2987-0/13/$31.00 ©2013 IEEE

II. Motivation

We identify three potential problems that occur due to the positioning of various units in the router pipelines of MinBD [10] and DeBAR [11].

A. Problem of channel wastage

As per the injection policy [2] of buffer-less routers, injection of new flits from a local core (core buffer) is allowed only when there is a free channel in the router pipeline after the ejection stage. Consider a case in DeBAR when all the four input channels of a router are busy with incoming flits and none of the flits are destined to be ejected. Hence the local core cannot inject a new flit into the router. The flits in the router pipeline channel reach the Permutation Deflection Network (PDN) [2], where they are allocated with the output ports. After the PDN, if a flit does not get its productive port, it is moved out of the router pipeline to the central buffer pool. This creates an idle channel that remains unutilized as no more injections are allowed at this point. Our experiments using uniform traffic at pre-saturation load in 8×8 mesh network employing DeBAR show channel idleness for 18% of cases. This points to the existence of idle output channel even when flits are waiting in the core buffer, which is undesirable.

B. Problem of unnecessary internal flit movements

In both MinBD and DeBAR router pipelines, injection from the core buffer and re-injection from the side buffer happen before the PDN. Hence these injected and re-injected flits participate in the arbitration process for acquiring the output ports. During the arbitration if they fail to win their desired port (deflected), they become potential candidates for side buffering. Hence there is a path for the injected flits to move from core buffer to side buffer. Similarly a flit re-injected from the side buffer could eventually return back to the side buffer itself if it gets a non-productive output port. Our experiments with uniform traffic at pre-saturation load in 8×8 mesh network employing DeBAR show that in 22% of cases the flits re-injected from the central buffer pool get deflected (in PDN) and come back to the central buffer pool itself. Similarly in 11% of cases flits injected from core buffer are moved to central buffer pool due to the non-productive port allocation in PDN. These results in movement of flits from one buffer to another leading to unnecessary power consumption without any forward progress for the flits. These problems occur due to the positioning of inject stage (entry point of flit from core/side buffers to router pipeline) before the buffer eject stage (exit point of flit to side buffer).

C. Problem of older flit penalization

In both MinBD and DeBAR, since the injections from the local core are done early in the pipeline, the newly injected flits also participate in the arbitration process in PDN. MinBD uses randomized silver priority and DeBAR uses priority based on hops to destination. In both these cases, the newly injected flits can get the highest priority. Situations can arise where these high priority newly injected flits can have port conflicts with incoming older flits from the neighbors and the older flits may get deflected. In our experiments, we observe 10% of such old flit deflections. Deflections of old flits by new flits can lead to livelock problems. Figure 1 shows that the negative impact of these three problems increases with injection rate.

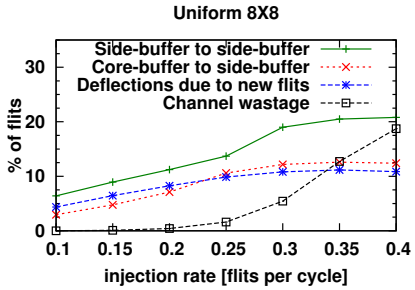

Fig. 1: Statistics of internal flit movements, deflections occurring to old flits due to new flits, and channel wastage for varying injection rates using uniform traffic in 8×8 mesh with DeBAR.

Our experimental studies on various real and synthetic workloads show that the metric that decides when and whether to buffer or deflect a flit has significant impact on the average flit latency, deflection rate, and overall performance. To mitigate these problems, we suggest an alternate design that brings in a major improvement in performance. We believe that by positioning the injection stage as the last unit of the router pipeline, the drawbacks of MinBD and DeBAR as mentioned above can be rectified. The decision of when to inject and where to inject can be coordinated with a smart intelligent injection algorithm.

III. SLIDER: Smart Late Injection Deflection Router

We propose a new deflection router architecture, SLIDER, whose main features are: (1) parallelized independent operations to reduce the pipeline latency; (2) selective flit preemption mechanism, which works in two modes: one to reduce deflection and other to reduce starvation; (3) injection and the re-injection stages kept late in the pipeline so as to minimize channel wastage and prevent intra-router flit movements. We discuss below these three features in detail.

A. Independent Operations

Since routing, ejection, and prioritization operations can work independently, we parallelize these three operations. We use XY routing, which needs the position coordinates of the current and destination nodes. Similarly we use hops to destination, the one used in DeBAR [11], as the flit priority scheme that requires only the flit's destination coordinates. As soon as a flit reaches the router, the destination coordinates are extracted and sent to the routing and prioritization units. The flit moves to the ejection unit, where ejection decision is taken based on whether the current router is the destination for the flit or not. By parallelizing these three units, that are

Algorithm 1 Selective Flit Preemption

1. No Removal: If the flits waiting in either the side buffer or the core buffer are not starving and all the flits are assigned productive ports after the PDN, the preemption unit will not remove any flit from the pipeline.
2. Forced Removal: If all the four channels are occupied by flits with productive port assignments, one among the flits is forcefully moved to the side buffer to accommodate a starving flit in either the core buffer or the side buffer.
3. Needed Removal: If at least one of the flits is assigned a non-productive port after the PDN, one among the deflected flits is selected and moved to the side buffer.

Algorithm 2 Smart Late Injection

1. Restricted Injection: If the number of flits in the core/side buffer is less than or equal to 2,
(a) choose a flit F from core/side buffer that has productive direction same as one of the empty channels E.
(b) inject F into E.
2. Non-restricted Injection: If the number of flits in the core/side buffer is more than 2,
(a) identify an empty channel E.
(b) choose a flit F whose channel on productive port is E. If no such F exists, choose a random F.
(c) inject F into E.

sequentially arranged in DeBAR, we effectively reduce the critical path, thereby reducing the cycle time.

B. Selective Flit Preemption

Similar to DeBAR, we use PDN in SLIDER for output port allocation. After PDN, based on the number of non-empty channels and the nature of port assigned to flits (whether productive port or not), we use a selective flit preemption logic. Preemption is the process of preventing a flit from moving out through its assigned output port. In SLIDER we carefully select a flit and preempt it to a side buffer. This preemption is done (1) to prevent a flit from moving out through a non-productive port or (2) to make space for a starving flit waiting in the router buffers (core buffer/side buffer). We call the former one *Needed Removal* and the later one *Forced Removal*.

Needed Removal reduces the deflection rate of flits and hence reduces the flit movement through non-profitable paths. *Forced Removal* of a flit reduces the waiting time of another flit waiting in the router buffers. This ensures forward progress and livelock-free movement of flits to the respective destinations. The preempted flits get re-injected into the router pipeline by a smart late injection operation. Algorithm 1 describes the selective flit preemption in brief.

C. Smart Late Injection

In a conventional deflection router without input buffers, flits can be injected to the router pipeline only when there is a free channel after the ejection stage [2]. But if we move injection to the end of the router pipeline, it gets more flexibility. In SLIDER the injections from the side buffer and the core buffer are done at the end of the router pipeline. The concept of smart late injection effectively utilizes the idle output channels already existing in the router pipeline or created by flit preemption.

We call it smart injection because we take an adaptive decision on when to inject a flit and to which channel we should inject. When there are two or more empty channels in the router pipeline after the flit preemption stage, we can inject from both the side buffer and the core buffer. When there is only one empty channel, we give preference to injections from the core buffer in the odd cycles and to injections from the side buffer in the even cycles like the *Dual Injection* technique used in DEBAR [11]. Injection can be in two modes based on the occupany level of the buffer. If the buffer from which injection takes place is more than half full, we call it as operating in *Restricted Injection*, otherwise it is in *Non-restricted Injection*. In *Restricted Injection*, we choose a flit for injection only if the available channel is productive for it. In *Non-restricted Injection*, a flit is injected into the available channel irrespective of whether it is productive or not. The smart late injection is outlined in Algorithm 2.

D. SLIDER Pipeline

SLIDER has a two stage pipeline. The architecture of SLIDER is shown in Figure 2. Operations of various units in the SLIDER pipeline are explained below in detail. A, B, and C shown in Figure 2 are the pipeline registers. Flits from the neighboring routers reach the router pipeline register A. From register A the destination coordinates of the flits are extracted and given to both the *Routing* and *Prioritization Unit* and the flits move to the *Eject Unit*.

Routing Unit: It does the route computation for all the incoming flits irrespective of whether they will move to the PDN or get ejected in the current router. We can use either XY routing or the quadrant routing mechanism [11] used in DeBAR for route computation. All our results we present in this paper are based on XY routing.

Eject Unit: By checking the coordinates of current router and the flit destination, the *Eject Unit* identifies the flits to be ejected. Even though DeBAR and MinBD offer dual ejection scheme, we stick on to single ejection stage as dual ejection is needed only in less than 12% cases for uniform traffic at saturation load. In case of multiple flits to be ejected in the current router, one among them is selected randomly for ejection. Other flits destined for ejection continue to move in the router pipeline, get deflected to neighbors and eventually return to the current router in subsequent cycles. MinBD has dual ejection channel that doubles the router-core channel wiring where as DeBAR use special control circuitry to handle the dual ejection. We eliminate both these overheads and stick to single ejection channel and compromise on negligible performance loss due to this re-routing of locally destined flits.

Prioritization Unit: We prioritize flits based on the number of hops remaining to reach the destination. The flits with less

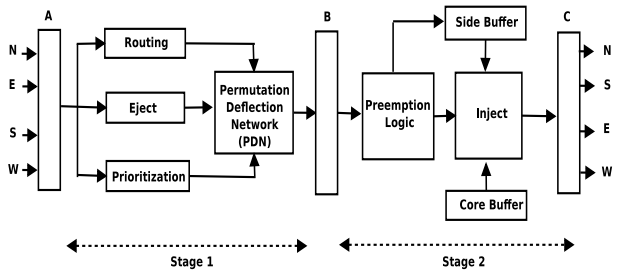

Fig. 2: SLIDER architecture.

number of hops to destination (other than destined locally) get the highest priority and this priority is communicated to the PDN for resolving port conflicts [11]. This priority scheme ensures that flits closer to destination are never deflected or side buffered.

Permutation Deflection Network (PDN): It consists of two stages with two 2×2 arbiters [2]. It allocates the output ports to each of the incoming flits based on the information obtained from *Routing Unit* and *Prioritization Unit*. When flits reach the *PDN*, the highest priority flit always gets the desired output port. Other flits may get deflected or assigned productive ports based on port conflicts in each stage of the arbiter.

Preemption Logic: As mentioned earlier, the *Preemption Logic* performs the task of side buffering certain flits. After the *PDN*, we check whether flits are assigned productive ports or not. Non-productive port allocation results in flits moving away from the destination, leading to increase in the flit latency. To tackle this, we accommodate one such flit per cycle in the side buffer so that the flit can be prevented from moving away from its destination. This is termed as *Needed Removal*, mentioned in Algorithm 1. This results in substantial reduction in the deflection rate.

We need to preempt flits in certain cases to ensure starvation free injection. After the *PDN*, if all the four channels are assigned to flits in their productive directions, then injections from neither the side buffer nor the core buffer are allowed due to lack of free channel. We define *starvation threshold* as the maximum number of cycles for which the injection is blocked due to lack of free channels. Once the *starvation threshold* is reached, we forcefully remove one of the flits from the router pipeline to make space for starving flits waiting in the core buffer or the side buffer to inject.

Side Buffer and Core Buffer: There are two categories of buffers in SLIDER. Side buffer stores preempted flits and core buffer holds the flits injected by the local core. Since all the flits in these buffers are potential candidates for injection, the core buffer and side buffer are not FIFO queues. To implement this non-FIFO buffer scheme, we associate a 2-bit flag for each flit in the core buffer and the side buffer to represent the desired port of the flit. Similarly we maintain a busy signal, for each of the pipeline channel, that is reset when the channel is empty. During the injection, we look for an ideal channel-flit mapping by comparing this 2-bit flag and the busy signal. We consider 4-flit buffers each for both core buffer and side buffer for a fair comparison with respect to MinBD and DeBAR.

Inject Unit: It performs the task of injecting flits from the router buffers. Depending on the availability of free channels and the occupancy level of flits in buffers, the *Inject Unit* performs a smart injection. In *Restricted Injection* as mentioned in Section III-C, we choose the right flit whose productive output port is same as the available free channel. This results in proper utilization of that channel. For example, consider a case where north channel becomes free after preemption logic, and the side buffer is starving with three flits f_1, f_2, f_3, requiring south, east and north ports, respectively. Now the injection logic injects flit f_3 (demanding north port) in the north channel.

Since *Restricted Injection* is selective (inject only if productive channel is available), it may slowly increase the number of flits waiting in the respective buffers. This is because preemption logic adds new flits in side buffer and local processing core adds new flits in core buffer. But lack of availability of productive channel may prevent flit injection from these buffers. If this situation prevails, core buffer will become full, leading to throttling of running applications in the core. When the side buffer is full it leads to increased deflection rate. To make sure that such a situation do not arise, we switch to *Non-Restricted Injection* when the buffer occupancy of core buffer exceed a threshold. We fix the threshold to 2 (half of the buffer capacity of 4 flits). In *Non-Restricted Injection*, we inject flits into the available channel irrespective of whether they have productive port or not. We dynamically switch between these two modes of injection depending on buffer occupancy level. Our synthesis results using Verilog modeling show that the smart late injection and preemption logic will not increase the router pipeline latency.

IV. Experimental Methodology

We modify the cycle accurate input buffered NoC simulator Booksim [1] to model the two-cycle deflection router micro architecture mentioned in CHIPPER [2]. Since deflection routers route each flit independently, every flit contains necessary control information needed for routing. We use necessary reassembly mechanism for handling out-of-order delivery of flits. The flit channel is 140-bit wide: 128-bit data field and 12-bit control field. On this baseline deflection router, we model MinBD [10], DeBAR [11] and SLIDER, and conduct experimental analysis.

We use Orion 2.0 [16] to evaluate the dynamic power consumption of the network and wiring overhead. We implement a Verilog model for all the above routers and synthesize using Synopsys Design Compiler with 65nm CMOS library to compute the pipeline latency, area and power consumption.

We evaluate the performance of MinBD, DeBAR and SLIDER in 8×8 mesh network using four synthetic traffic patterns: *uniform, transpose, tornado* and *bit-complement*. We capture the average flit latency, overall throughput, and deflection rate by varying the injection rate from zero load till saturation and analyze the results. We also evaluate the proposed router using multiprogrammed workloads consisting of SPEC2006 CPU benchmark mixes.

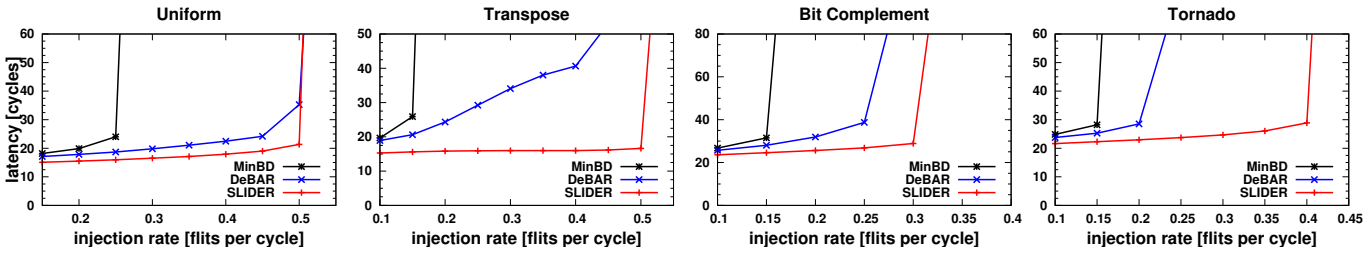

Fig. 3: Average flit latency comparison under various synthetic traffic patterns in 8×8 mesh network.

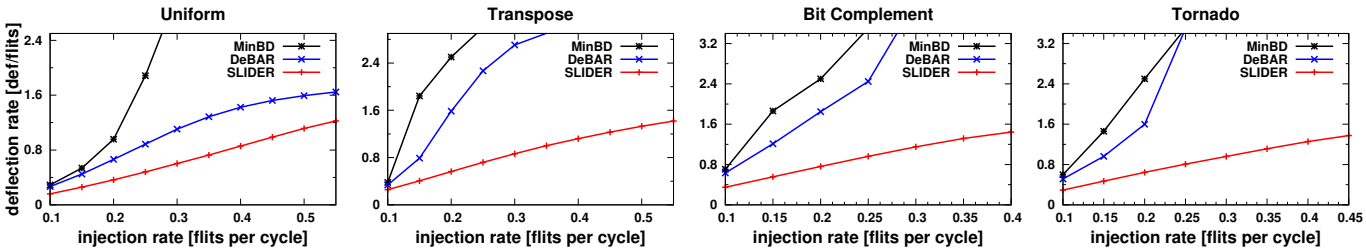

Fig. 4: Deflection rate comparison under various synthetic traffic patterns in 8×8 mesh network.

V. Experimental Results And Analysis

The relative performance of SLIDER with respect to MinBD and DeBAR is analyzed to understand the impact on various network parameters.

A. Effect on Average Flit Latency

Latency of a flit is defined as the number of cycles needed for the flit to travel from its source to destination. Figure 3 shows the plots of injection rate vs flit latency for various synthetic traffic patterns in 8×8 mesh network. The average latency increases with increase in the injection rate. There exists a point in the injection rate (saturation limit) beyond which latency increases exponentially. Proper flit management in terms of buffering, deflection and injection at the right time extends the saturation point further, which gives the network more load handling capacity.

We observe that for all synthetic traffic patterns, the saturation point is extended in SLIDER as compared to MinBD and DeBAR. This indicates that SLIDER is capable of supporting higher injection rates. We can see that SLIDER has the minimum average flit latency as compared to MinBD and DeBAR during pre-saturation load. The late injection of the new flits avoids port conflicts with old flits, which helps the old flits to reach their destination without much deflection there by reducing the flit latency.

B. Effect on Deflection Rate

Deflection rate is defined as the average number of deflections per flit. Reduction in deflection rate causes reduction in network activity and hence reduction in dynamic power. Injection rate vs deflection rate plot for various synthetic traffic patterns is shown in Figure 4. When the load on the network is less (at lower injection rates), the deflection rate is low as there are not many conflicts occuring between the flits for the outgoing ports. As injection rate increases, deflection rate also increases due to high port contention. Figure 4 shows that the deflection rate is much lower in SLIDER than MinBD and DeBAR. In SLIDER, the newly injected flits are not part of the port allocation and hence the deflection rate of older flits is reduced. Moreover at low injection rates, SLIDER works in *Restricted Injection* that forwards flits only to productive ports leading to the low deflection rate in SLIDER. Dynamic link power analysis using Orion2.0 shows that SLIDER reduces link power by 16% with respect to DeBAR.

C. Effect on Real Workloads

We also evaluate the effectiveness of our technique using multiprogrammed workloads. We use Multi2sim [17] simulator to model a 64-core CMP setup with CPU cores, cache hierarchy, and coherence protocols in sufficient detail and accuracy. Each core consists of an out-of-order x86 processing unit with a 64KB, dual ported, unified, private L1 cache. We use a shared distributed L2 cache with 512KB per core. The L1 cache is 4-way associative and L2 is 16-way associative. The block size of L1 and L2 caches are 32B and 64B, respectively.

Each core runs a SPEC CPU2006 benchmark application. Based on the *misses per kilo instructions* (MPKI) on a 64KB L1 cache, we classify the benchmarks into *Low* (MPKI less than 5), *Medium* (MPKI between 5 and 25) and *High* (MPKI greater than 25). In our experiments, we use *calculix, gobmk, gromacs,* and *h264ref* in the *Low* MPKI group, *bwaves, bzip2, gamess,* and *gcc* in the *Medium* MPKI group, and *hmmer.nph3, lbm, mcf,* and *leslie3d* for the *High* MPKI group.

We construct 35 multiprogrammed workloads, each with 64 applications selected from the SPEC CPU2006 benchmark suite. We categorize these workloads into 7 mixes (M1 to M7) based on the proportion of the network injection intensity *(Low / Medium / High)* of the constituent benchmarks. Details of the mixes are given in Table I. After sufficient fast forwarding, we capture the L1 cache misses that generate network traffic and feed it to the modified Booksim model to simulate the network operations and capture the statistics.

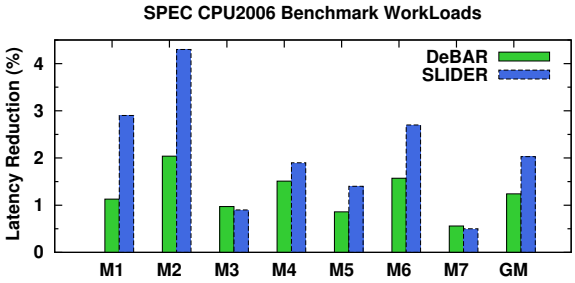

Fig. 5: Reduction in average flit latency rate for DeBAR and SLIDER with respect to MinBD for multiprogrammed workloads.

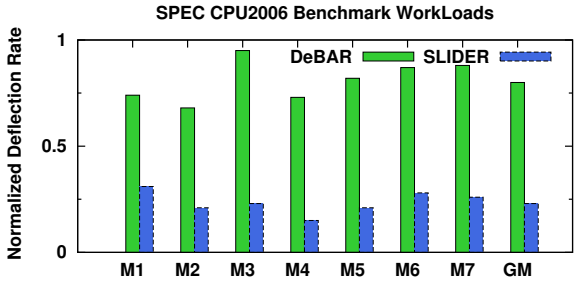

Fig. 6: Normalized deflection rate for DeBAR and SLIDER with respect to MinBD for multiprogrammed workloads.

Benchmark Mix	M1	M2	M3	M4	M5	M6	M7
% of Low	100	0	0	50	0	50	31
% of Medium	0	100	0	0	50	50	31
% of High	0	0	100	50	50	0	38

TABLE I: Percentage of different network injection intensity applications in various benchmark mixes.

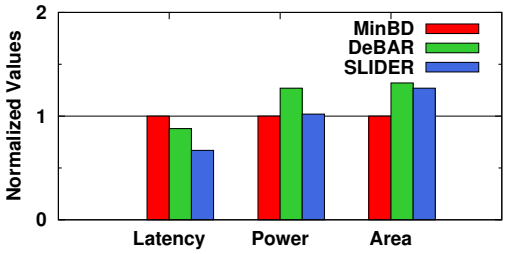

Fig. 7: Router pipeline latency, static power and area comparison of DeBAR and SLIDER normalized with respect to MinBD values.

We plot the latency results using multiprogrammed workloads in Figure 5. The plot shows the percentage reduction in latency with respect to MinBD. We observe that the performance of SLIDER is better than that of DeBAR for all the workloads except M3 and M7. The packet injection behavior in M3 and M7 contains relatively heavy injections from all the cores. So routers operating in the *Restricted Injection* refrain from injecting packets due to lack of productive ports. This increases the buffer occupancy of flits in the core buffer, leading to increase in flit latency. Some of the benchmarks in M3 and M7 occasionally create bursty packet injections that make SLIDER to operate in *Non-Restricted Injection*. In *Non-Restricted Injection*, flits are deflected from the source itself and the impact of this is severe if the deflection is just opposite to that of their productive ports. But we can see that on an average (GM), SLIDER performs better than MinBD and DeBAR on SPEC CPU2006 benchmark mixes. Figure 6 contains the deflection rate for DeBAR and SLIDER normalized to MinBD for multiprogrammed workloads. We can observe that SLIDER achieves significant reduction in deflection rate as compared to DeBAR for all the mixes. In more than 76% of cases, the packet injection behavior of SPEC CPU2006 benchmarks keeps SLIDER in *Restricted Injection* mode. This results in reduction in deflection rate. The remaining 24% cases only account for the deflection.

D. Effect on Router Pipeline Latency, Static Power and Area

Figure 7 compares the router pipeline latency, static power, and area of DeBAR and SLIDER normalized with respect to MinBD. The timing analysis of the three router designs reveals that the router pipeline latency reduces by 17% in DeBAR and 32% in SLIDER as compared to that of MinBD. This reduction results from the parallelization of ejection, routing and prioritization operations. All the results with respect to average flit latency (Figures 3 and 5) are taken by assuming that MinBD, DeBAR, and SLIDER operates at the same clock frequency. But after the Verilog synthesis, we confirm that an NoC with SLIDER can operate 25% faster than one with DeBAR. There is a reduction of 19% in static power consumption for SLIDER compared to DeBAR. This is due to the reduction in the number of units in the pipeline structure. SLIDER incurs 21% area overhead with respect to MinBD because of the control circuitry added for smart late injection and selective flit preemption. But SLIDER consumes 3.5% less area than that of DeBAR.

E. Effect on Throughput

Figure 8 shows the plot of throughput vs injection rate for various synthetic traffic patterns. We compute the throughput as the number of flits ejected per router per cycle. Higher throughput indicates that a given load (a pre-defined count of flits) can be delivered in less time. In *uniform* traffic pattern, the throughput for SLIDER is almost same as that for DeBAR. For all other traffic patterns, we find that the throughput improves for SLIDER compared to both MinBD and DeBAR. The smart late injection helps in faster delivery of flits by making use of available channels effectively. Throughput graph has a sudden dip in MinBD and DeBAR due to their early saturation. Since SLIDER extends the saturation limit of the network, the throughput graph does not display any dip at the same injection window. When we run millions of instructions in real workload, we do not find any difference in throughput for SLIDER, MinBD and DeBAR.

F. Other Experimental Statistics

Due to the absence of age based priorities, occasionally long flow flits (flits whose minimal hop distance between source

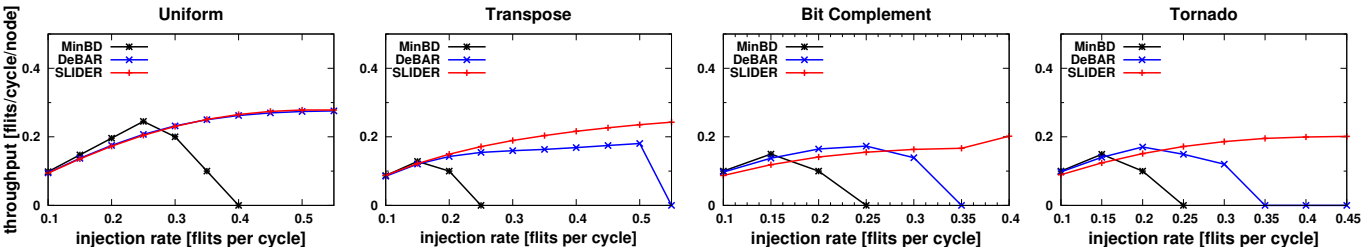

Fig. 8: Throughput comparison under various synthetic traffic patterns in 8×8 mesh network.

Traffic	Restr. Inject(%)	Non-Restr Inject(%)	Needed Removal(%)	Forced Removal(%)
Uniform	59.38	40.62	93.16	6.84
transpose	88.52	11.48	97.10	2.90
Bit-Comp	85.32	14.68	92.80	7.20
Tornado	68.44	31.55	94.71	5.29
Average	75.42	24.48	94.44	5.56

TABLE II: Percentage of time SLIDER is operating in *Restricted Injection*, *Non-Restricted Injection*, *Needed Removal* and *Forced Removal* at saturation load conditions under various synthetic traffic in 8×8 mesh network.

and destination is large) get penalized. We observe that in uniform traffic at pre-saturation load, 0.57% flits are having their flit latency more than three times the average flit latency of the network. The percentage moves upto 3.8% at saturation load. This shows that even without using age based priority we can bring down the average flit latency.

The flit injection switches between *Restricted* and *Non-Restricted* mode based on buffer occupancy threshold. We fix the threshold to 2 flits. This means that when the number of flits in the core buffer (total capacity is 4 flits) is less than or equal to 2, we operate in the *Restricted Injection* mode. If we move the threshold to 1 flit or 3 flits, the network is saturating early. This trend is visible across all traffic patterns.

SLIDER uses run-time flow behaviour of flits and take dynamic decisions to switch between the *Restricted Injection*, *Non-Restricted Injection*, *Needed Removal* and *Forced Removal*. We observe that this dynamic switching is contributing to a great extent in the performance of SLIDER. Table II gives the fractional split up of these various modes of operation under various traffic patterns at pre-saturation loads. We can observe that acrosss all traffic patterns on an average, around 75% of the time SLIDER operates under the *Restricted Injection* and only less than 6% cases the router operates in the *Forced Removal* mode. The high percentage of *Restricted Injection* shows that majority of the newly generated flits are getting productive ports. The low percentage of *Forced Removal* shows that very few flits are penalized to give way to starving flits. At the same time application throttling is avoided by switching to the *Non-Restricted Injection* (in 24% of cases).

One of the main motivations behind the proposal of SLIDER is to utilize the idle output channels in the DeBAR design. The channel wastage which is around 18% for DeBAR at saturation load (Refer Figure 1 at injection rate=0.4 flits/cycle/node) is reduced to 6% in SLIDER under the same conditions.

VI. Conclusion

In this work, we analyzed the existing state-of-the-art deflection routers and identified few drawbacks occuring due to the positioning of various units in their pipeline. We provided a cost effective alternative in terms of SLIDER that prevents intra-router flit movements completely and reduces deflections and average flit latency to a great extent. Results also showed that the channel wastage is reduced significantly. We further showed that NoCs using SLIDER can operate at a higher frequency and achieve considerable improvement in the network level performance.

VII. Acknowledgment

This work is supported in part by grant from DRDO under IITM-DRDO MoC.

References

[1] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. USA: Morgan Kaufmann Publishers Inc., 2003.
[2] C. Fallin et al., "CHIPPER: A low complexity bufferless deflection router," in *HPCA*, 2011, pp. 144–155.
[3] Y. Hoskote et al., "A 5-GHz mesh interconnect for a teraflops processor," *IEEE Micro*, vol. 27, no. 5, pp. 51–61, 2007.
[4] M. B. Taylor et al., "Evaluation of the raw microprocessor: An exposed-wire-delay architecture for ILP and streams," in *ISCA*, 2004.
[5] T. Moscibroda and O. Mutlu, "A case for bufferless routing in on-chip networks," in *ISCA*, 2009, pp. 196–207.
[6] C. Gomez et al., "An efficient switching technique for NoCs with reduced buffer requirements," in *ICPADS*, 2008, pp. 713–720.
[7] M. Hayenga et al., "SCARAB: A single cycle adaptive routing and bufferless network," in *MICRO*, 2009, pp. 244–254.
[8] E. Nilsson et al., "Load distribution with the proximity congestion awareness in a network-on-chip," in *DATE*, 2003, pp. 1126–1127.
[9] G. Oxman et al., "Streamlined network-on-chip for multicore embedded architectures," in *ARCS*, 2012, pp. 238–249.
[10] C. Fallin et al., "MinBD: Minimally-buffered deflection routing for energy-efficient interconnect," in *NOCS*, 2012, pp. 1–10.
[11] J. Jose et al., "DeBAR: Deflection based adaptive router with minimal buffering," in *DATE*, 2013, pp. 1583–1588.
[12] S. A. R. Jafri et al., "Adaptive flow control for robust performance and energy," in *MICRO*, 2010, pp. 433–444.
[13] G. Kim et al., "Flexibuffer : Reducing leakage power in on-chip network routers," in *DAC*, 2011, pp. 936–941.
[14] G. Nychis et al., "Next generation on-chip networks : What kind of congestion control do we need?" in *Hotnets*, 2010, pp. 1–6.
[15] Z. Lu et al., "Evaluation of on-chip networks using deflection routing," in *GLSVLSI*, 2006, pp. 296–301.
[16] A. B. Kahng et al., "Orion 2.0: A fast and accurate NoC power and area model for early stage design space exploration." in *DATE*, 2009, pp. 423–429.
[17] R. Ubal et al., "Multi2sim: A simulation framework to evaluate multicore-multithreaded processors," in *SBAC-PAD*, 2007, pp. 62–68.