2015 28th International Conference on VLSI Design and 2015 14th International Conference on Embedded Systems

Smart Port Allocation in Adaptive NoC Routers

Reenu James, John Jose, Jobin K. Antony
Rajagiri School of Engineering and Technology, Kochi, Kerala, India
reenujms@gmail.com, (johnj, jobinka)@rajagiritech.ac.in

***Abstract*—Network on Chip (NoC) is an emerging communication framework for multiple processing cores on a System on Chip (SoC). The router micro-architecture determines the performance of such a communication network to a great extend. Considering the cost effective performance and scalability, minimally buffered deflection routers are emerging as a popular design choice for NoC based multicore systems. In this paper, a new router architecture is proposed which has an enhanced pipeline register and a smart port allocator that significantly reduces the pipeline stage delay in the router. The proposed smart port allocator assigns output port to incoming flits dynamically based on available output ports and flit occupancy level of the enhanced pipeline register. This eliminates unwanted intra-router movement of flits. Experimental results on synthetic and real workloads show that the proposed router reduces average packet latency, output channel wastage, deflection rate of flits and increases the throughput in the network when compared to the state-of-the-art minimally buffered deflection routers.**

***Keywords*-buffer-pool, router pipeline, deflection routing.**

I. Introduction

With the advancements in VLSI technology, the number of processing cores that can be integrated on a single chip is increasing. In multi-core chips, scalable packet switched Network on Chip (NoC) architectures are proposed to mitigate the on-chip wire delay associated with the bus-based inter-core communication framework [1].

In a typical multicore design, each core consists of an out-of-order superscalar processor, a private L1 cache and a distributed shared L2 cache. Each core is attached to a router which is the connecting point with the on-chip network. L1, L2 cache misses and coherence transactions generate inter-core communication messages in the form of packets. Packets are further divided into flow control units called flits. A flit is the basic unit of data that can be forwarded from one router to its neighbor. Credit based flow control and inter router handshaking ensure smooth flow of packets between routers. Conventional input buffered NoC routers have high load handling capacity which give the network very good performance in terms of average packet latency and throughput at the cost of increase in area and power overheads. Experimental studies show that approximately 30% to 40% of chip power is consumed by NoC, out of which a significant contribution is by the flit buffers residing in the input ports of routers [2], [3].

A conventional NoC router has buffers in their input ports which are occupied by incoming flits until they get their desired output ports. Routing operation on flits identifies the desired output port to reach the next downstream router. The virtual channel allocator assigns an input buffer in the downstream router. The switch allocator chooses a flit when multiple flits are contending for the same output link. Flits that loose the switch allocation process will stay back in the input buffers to take part in the arbitration process in subsequent cycles. The flits which are assigned their desired output ports move to output links through the crossbar switch.

Studies show that input buffers in NoC routers are over provisioned for common case low-injection workloads [4]. As a cost effective alternative, bufferless deflection routing is proposed [5]. NoC with bufferless deflection routers is emerging as a better design choice for low-injection rate applications. In bufferless deflection routers, port contentions are handled by deflecting the traffic to non-productive (undesired) directions. Bufferless deflection routers suffer from high deflection rate at high injection rates due to increase in count of flits with non-productive output ports. Side-buffering technique adopted in some bufferless routers [4], [6], [7] stores the non-productively assigned flits in special storage locations called side-buffers. In input buffered routers, all the incoming flits are stored in the buffers and they stay there till they get productive ports. But in side-buffered routers, flits are stored in buffers only if they are assigned non-productive ports. The side-buffers are removed in our proposed router and the pipeline register at the end of stage-1 is facilitated with additional circuitry to form a buffer-pool for intermediate storage of flits. Depending on the occupancy level of the buffer-pool, our port allocator takes intelligent decisions whether to assign output ports productively or to deflect the flits.

The rest of the paper is organised as follows. We describe the related works in Section II. In Section III, limitations of the DeBAR design and the motivation for the proposed work are explained. The architectural details of the proposed router is given in Section IV. Implementation details and experimental results are covered in Section V and we conclude our work in Section VI.

1063-9667/15 $31.00 © 2015 IEEE
DOI 10.1109/VLSID.2015.86

II. Related Work

BLESS router contains a crossbar that uses a sequential port allocator and age based priority for allocating output ports to the incoming flits [5]. Age based priority in BLESS leads to livelock situation. The critical path delay and hence the router latency is very high in BLESS which is reduced by a parallel port allocator in CHIPPER [8]. CHIPPER gives a golden priority to certain flits, ensuring their delivery to destination without deflections. This design increases the deflection rate as the golden priority approach ensures starvation freedom only for golden flits leaving the progress of other flits unattended.

Deflection increases the average latency of flits due to the usage of non-minimal paths. MinBD [4], DeBAR [6] and SLIDER [7] are two stage pipeline routers that employ side-buffers after the port allocator phase that can accommodate one among the deflected flits per cycle. This minimal buffering approach is a promising solution which reduces the latency and deflection rate of flits. The side-buffered flits get re-injected into the router pipeline during the subsequent cycles and participate in the arbitration process to get their desired output ports. SLIDER uses a late injection to exploit the idle channels occuring due to pre-emption of flits with non-productive ports. AFC [9] and Flexibuffer [10] uses a conventional input-buffered router with a provision to switch to bufferless mode under low network load by using the power gating technique. A minimally buffered single cycle deflection router architecture is proposed in MinBSD [11] which reduces the router delay by reducing the critical path latency of the router.

In this paper we introduce a novel two cycle deflection router that employs a smart port allocation for adaptive routing. Our router eliminates side-buffers and enhances the pipeline register at the end of stage-1 of the router pipeline to form a buffer-pool for storing the flits that do not get a productive port. Significant reduction in router pipeline latency as well as deflection rate is acheived with respect to the state-of-the-art technique DeBAR. The proposed router switches between two modes; waiting mode (flits will wait in buffer-pool till they get their desired port) and deflection mode (all the outgoing ports will be alloted with flits, out of which a few flits may get undesired ports). This intelligent switching of mode ensures that buffer-pool always have sufficient space for incoming and newly injected flits.

III. Motivation

We identify a few limitations in the state-of-the-art deflection router DeBAR design and propose a cost effective solution to address these limitations.

DeBAR [6] is a two stage deflection router with A, B, C as the various pipeline registers as shown in Figure 1. Units kept between A and B form the first stage of the pipeline whereas units between B and C form the second stage. At the beginning of a cycle, flits from four neighbors N, E, W and S reach the pipeline register A. Flits arriving at register A will move through Hybrid Ejection Unit (HEU). HEU takes care of atmost two ejections with the help of a single port. Flits residing in the core-buffer (newly generated flits from processing core) are injected into the router pipeline by Dual Inject Unit (DIU). Computation of flit priority and output port is done in the second stage by Priority Fixer Unit (PFU) and Quadrant Routing Unit (QRU), respectively. Based on the priority and route obtained, output port allocation takes place in the Permutation Deflection Network (PDN). Buffer Eject Unit (BEU) forwards one among the flits that is assigned a non-productive port to the side-buffer. The other flits move to the respective output ports through pipeline register C. The side-buffered flits will get re-injected to the router pipeline through DIU. If all input links are busy and a flit is starving for injection in any of the buffers (core-buffer or side-buffer), then a flit is forcefully moved to the side-buffer and a link is made free by the Flit Preemption Unit (FPU). We identify the following limitations in DeBAR design.

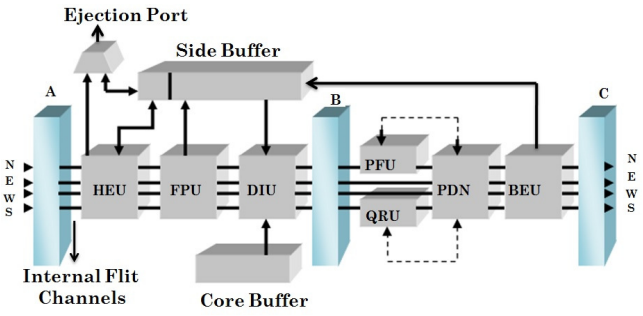

Figure 1. DeBAR Architecture.

A. Channel wastage

Injection from the local core occurs when there is a free link in the internal flit channel after the ejection stage. When a flit is waiting for injection from the local core (core-buffer) and if none of the links are available, the flit has to stay back in the core-buffer itself. After the port allocation stage (PDN), one among the flits which does not get its productive port is stored in the side-buffer. This creates a free output link which remain unutilised as further injection is not possible at this stage. The presence of an idle output channel in a router when a flit is waiting for injection to that idle channel is undesirable. In this situation we say the channel is wasted. We analysed the behaviour of flits residing in core buffer and found that in an 8x8 network running on uniform traffic, at saturation load there is channel wastage in 18% of clock cycles.

B. Unnecessary flit movements

In DeBAR, injection from core-buffer as well as re-injection from side-buffer occur before PDN. As a result the

flits that are injected from these buffers through DIU reach the arbitration stage and take part in the arbitration process. In the PDN stage, if they fail to get their productive port, flits will be moved to the side-buffer again. Hence flits injected from core-buffer as well as re-injected from the side-buffer can get stored in the side-buffer if they do not get their desired ports. This creates internal flit movements leading to unnecessary power consumption without any forward progress for the flits. Our experimental results show that for uniform traffic in 8x8 mesh network at pre-saturation load, during 22% and 11% of clock cycles there is a side-buffer to side-buffer movement and core-buffer to side-buffer movement, respectively.

C. Non productive operations for side-buffer entry

At low injection rate, FPU will not pre-empt any flits as the possibility of all the four internal flit channels getting busy and possibility of starvation of flits in core-buffer and side-buffer is very rare. Similarly, due to less number of port conflicts in low injection rate, majority of the flits will get productive port in PDN. This is validated by the low occupancy of side-buffer in DeBAR at low injection rates. At high-injection rate, flits are removed from the router pipeline by FPU (to avoid starvation of flits in side-buffer and core-buffer) and from BEU (to prevent flits from taking non-minimal paths). Both these units will remove a flit from router pipeline to side-buffer. We feel this multiple units for flit removal to side-buffer can be replaced with a single unit like in SLIDER [7]. Multiple units to side-buffer entry causes unnecessary logic complexity in the critical path and adds to router pipeline delay.

D. Limitation in injection efficiency

In the DeBAR architecture, injection from the core-buffer occurs in alternate cycles. If the processing core generates flits during consecutive cycles, injecting them into the network is impossible. Such a case can cause the core-buffer to be overloaded. We experience these situations in real traffic when a core reply with a cache packet in the form of multiple flits injected in adjacent cycles.

Experimental studies on real and synthetic traffic show that the above mentioned limitations of DeBAR is a critical performance bottleneck. We try to address all these limitations in our proposed design.

IV. The Proposed Work

The router pipeline of the proposed architecture contains two stages as shown in Figure 2. In the first stage of the router pipeline, routing, port prioritisation, inject and eject units are present. Out of this, eject and inject units are operating parallely. Second stage consists of smart port allocation unit which is the heart of our design. The buffer-pool plays a major role in this architecture by storing the flits from neighboring routers (that have passed stage-1) and the newly injected flits. Smart port allocator adaptively picks flits from the buffer-pool and allots the output ports. Based on the availability of their required output ports and occupancy level of the buffer-pool, the flits either wait in the buffer-pool or move out from it through their respective output ports.

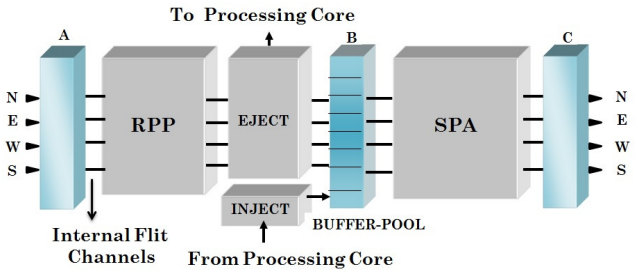

Figure 2. Proposed Architecture. Stage 1: RPP (Routing & Port Prioritisation), Eject & Inject units. Stage 2: SPA (Smart Port Allocation). A, B, C are pipeline registers.

A. Routing and Port Prioritisation (RPP)

At the beginning of a clock cycle, flits from four input links reach register A. Routing and port prioritisation unit extract the destination address field from the flits. Each of these flits are routed based on quadrant routing policy used in DeBAR [6]. We prioritise the incoming flits based on the number of hops to destination [6].

B. Eject and Inject

Ejection of flits occur when a flit finally reaches its destination. Upon comparison of destination address of a flit with current router address, if they are same, then the flit gets ejected out to the local processing core.

In DeBAR, injection is possible only if one among the internal flit channels is empty after the ejection stage. But in our proposed design, whenever a core needs to inject, it can inject directly to the buffer-pool. This eliminates the problem of starvation for flits waiting for injection in the core-buffer when all the internal flit channels are busy. In our design, eject and inject are two independent operations which take place parallely.

C. Buffer-Pool

Buffer-pool stores the incoming flits from the input links that have passed stage-1. All the newly injected flits from local core as well as flits which failed to get their productive port in the previous cycle are also stored in buffer-pool. The buffer-pool can accomodate 8 flits. Based on the priority of hops to destination and buffer stay time, flits are picked from the buffer-pool by the smart port allocator. The flits which are not able to move out of buffer-pool in the current cycle are assigned higher priority so that they can get their productive ports in the subsequent cycles.

This makes sure that flits will not starve to get a productive port by waiting in the buffer-pool.

D. Smart Port Allocation (SPA)

Smart port allocator allots ports based on the quadrant vector [6] and the priority value [6] obtained during the first stage of the router pipeline. It works in two modes based on the buffer-pool occupancy level: Mode 1- waiting and Mode 2- deflection.

1) Mode 1- Waiting: At low-injection rates, the buffer-pool occupancy will be less. In such cases, SPA picks only those flits from the buffer-pool that can be assigned their desired output ports. When more than one flit conflicts for the same output port, one with a higher priority will be given its desired output port while the other stays back in the buffer-pool. These waiting flits will be given higher priority in the next cycle for port allocation. If the number of waiting flits increases, it may lead to saturation of the buffer-pool. To avoid this scenario, when the buffer-pool occupancy reaches 50% of its original capacity, SPA switches to the deflection mode.

2) Mode 2- Deflection: Deflection mode operates during high-injection rates. When the buffer-pool occupancy is more than 50% of its capacity, all the four output links will be assigned with flits by SPA. The higher priority flits will be assigned with their desired output ports. When the desired ports are already occupied, the SPA assigns the the remaining output ports to the lower priority flits which may lead to deflection. In this mode, four flits are moved out to the output links whether they get their desired output ports or not.

Thus the SPA takes intelligent decisions of whether to retain a flit in the buffer-pool or assign a productive port to the flit or permit a flit to take a non-productive port. These decisions are dynamically taken based on the current traffic flow and port conflicts in each individual router. Switching between the two modes (waiting/deflection) is also a local decision within a router. SPA hence eliminates the need for an FPU and BEU used in DeBAR design.

V. Implementation and Experimental Results

The architecture of the proposed router is modelled in Verilog and synthesised using Synopsys Design Compiler [14] with 65nm library to obtain the router pipeline latency for each unit in the pipeline stage. The router pipeline latency obtained from Synopsys is used in Booksim [12] to get the average packet latency in nanoseconds. For a fair comparison, we also consider single flit packets like in DeBAR and MinBD designs.

Figure 3 shows the pipeline latency of the proposed architecture and DeBAR obtained from the Verilog synthesis using Synopsys Design Compiler. The comparison shows that the proposed design has a lower pipeline delay in both the stages. In DeBAR, stage-1 is determining the critical path where as in the proposed router, stage-2 is determining the critical path. The pipeline stage delay is 2 ns for DeBAR and 1.69 ns for the proposed technique. This is mainly because of the lower number of functional units in router pipeline and the parallel eject and inject processes. When NoC with DeBAR operates at 500 MHz, NoC with the proposed router can operate at 591 MHz thereby acheiving a network operating speedup of 18.2%.

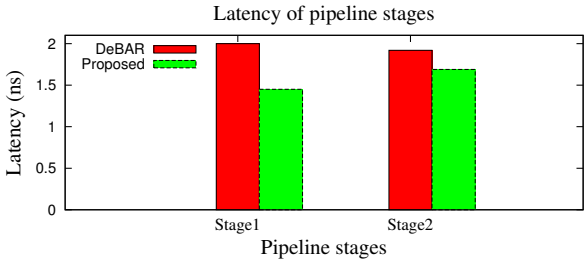

Figure 3. Router pipeline delay analysis using Verilog synthesis.

Booksim is a cycle accurate simulator that models all microarchitecture details of traditional NoC with accuracy and precision. We customize the simulator as per our proposed architecture and run for 8x8 mesh network using various standard synthetic traffic patterns.

Latency of a packet is defined as the number of cycles needed for the packet to travel from its source to destination. The average latency of packets in cycles is obtained from the network simulator Booksim, while the cycle time in nanoseconds is obtained using Synopsys Design Compiler. Combining the results, the average latency in nanoseconds is plotted for uniform, bit-complement and tornado traffics as shown in Figure 4. At saturation load, the proposed design achieves a latency reduction of 55.6%, 75% and 65.6% with respect to DeBAR for uniform, bit-complement and tornado traffic patterns, respectively. Across all synthetic traffic patterns, NoC with our proposed router saturates at a higher injection load when compared to DeBAR.

Deflection rate is defined as the average number of deflections per flit. The analysis of the deflection rate versus injection rate is shown in Figure 5. From the analysis we can see that the deflection rate of our proposed design is close to zero as it uses the smart port allocation logic. During low injection rates the deflection rate is approximately zero. This is because at low injection rates, the router always operates in Mode-1, thereby permitting only productive port assignments. In very less number of cases during high injection rates (port conflicts at Mode-2), we get non-productive port assignment to a few flits which account for deflection. The significant reduction in deflection rate substantially reduces dynamic power dissipation in the network since it reduces unnecessary movement of flits in non-productive directions.

Overall throughput is the number of packets ejected per router per cycle. As the deflection rate has reduced,

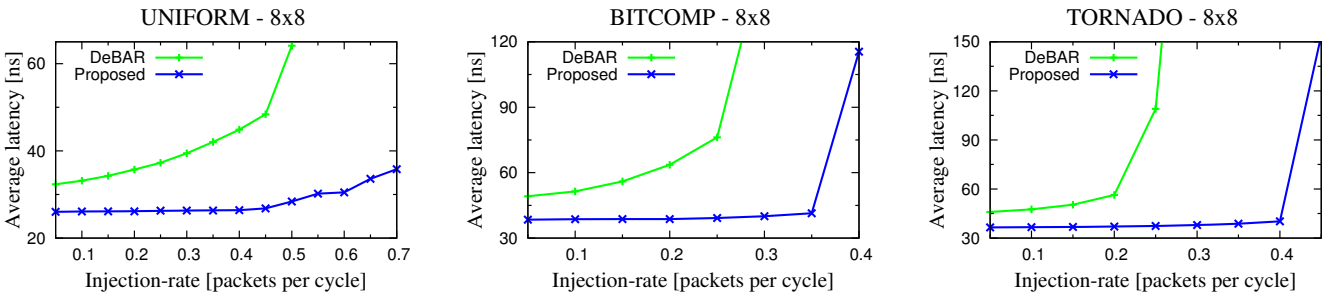

Figure 4. Comparitive analysis of average packet latency versus injection rate for various synthetic traffic patterns in 8x8 mesh network.

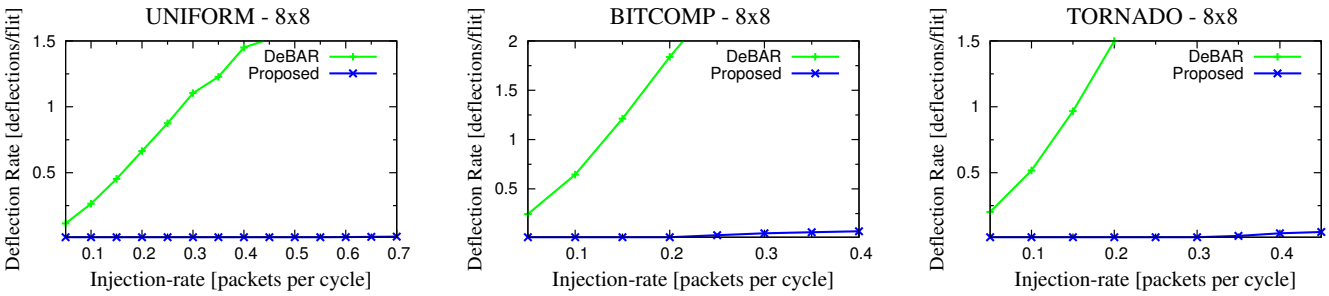

Figure 5. Comparitive analysis of deflection rate versus injection rate for various synthetic traffic patterns in 8x8 mesh network.

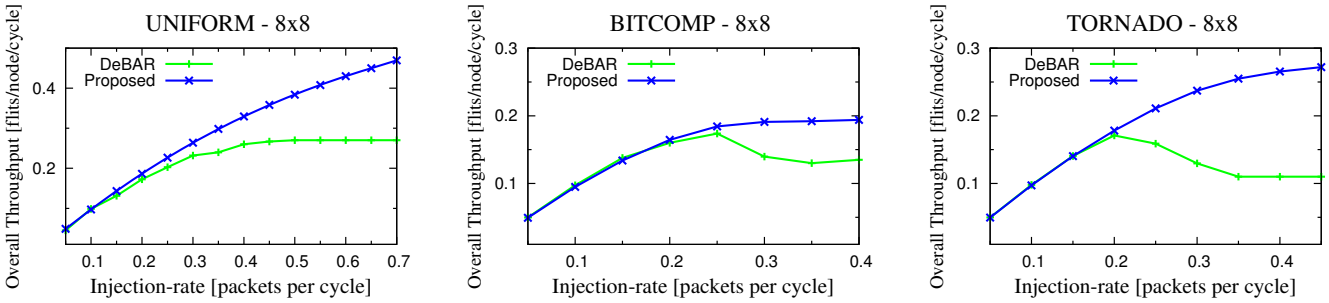

Figure 6. Comparitive analysis of overall throughput versus injection rate for various synthetic traffic patterns in 8x8 mesh network.

the flits reach their desired destination and get ejected out at a faster rate. The proposed design shows a marginal improvement in the throughput for various traffic patterns as shown in Figure 6. We can also observe that a significant change in throughput is there at relatively higher injection rates. This is due to the significant reduction of deflection scenarios, thereby making speedy progress of flits towards its destinations.

Apart from synthetic traffic, we evaluate the performance of our proposed system using traces of multiprogrammed workloads also. We use Multi2sim [13] simulator to model a 64-core CMP set up 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, 4-way set-associative, 32 byte block, private L1 cache and a 512KB, 16-way set associative, 64 byte block, shared distributed L2 cache. Each core is assigned with a SPEC 2006 CPU benchmark application for running on it. Based on the *misses per kilo instructions* (MPKI) values calculated on a 64KB L1 cache, we classify the benchmarks into Low (i.e., MPKI less than 5), Medium (i.e., MPKI between 5 and 25) and High (i.e., 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 40 multiprogrammed workloads, each with 64 single threaded benchmark instances. We categorize these workloads into 4 mixes (M1 to M4) 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. For example, M3 consist of High MPKI applications in all the 64 cores. 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.

We also compute the percentage reduction in average

Table I
PERCENTAGE OF DIFFERENT NETWORK INJECTION INTENSITY APPLICATIONS IN VARIOUS BENCHMARK MIXES.

Benchmark Mix	M1	M2	M3	M4
% of Low	100	0	0	31
% of Medium	0	100	0	31
% of High	0	0	100	38

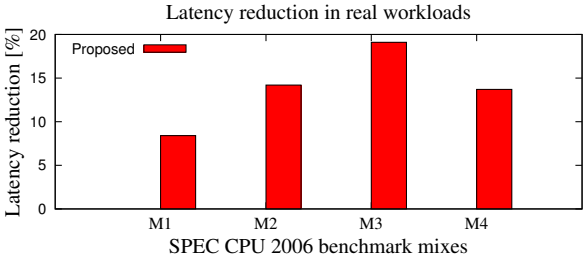

Figure 7. Percentage reduction in average packet latency with respect to DeBAR using real workloads in a 64-core CMP.

packet latency by using the SPEC 2006 CPU benchmark mixes (M1-M4) and plot them in Figure 7. The figure shows the percentage reduction in average packet latency using proposed logic with respect to DeBAR for various mixes taken during a window of five lakh clock cycles.

In mix M1, where the number of packets flowing through the network is relatively very low due to low MPKI applications running on the cores (hence few cache miss packets in the network), we observe that the latency reduction is marginal. But in mixes M2 and M4 we are able to observe more reduction in latency. We can see that the reduction is even more in mix M3 where there is heavy packet injection. These observations in real work load traffic show that our proposed design operates efficiently for various injection rate applications. Our area and power analysis show that due to reduction in functional units in the pipeline, our proposed design reduces router area by 20% and static power consumption by 18% with respect to DeBAR.

VI. CONCLUSION

A new NoC router architecture with a smart port allocator is implemented and the results obtained are compared with the existing techniques. We used a buffer-pool at the end of pipeline stage-1 that manages the packets in an efficient way. Results showed that router pipeline latency is less for the proposed design and hence the on-chip network which uses our routing logic can be operated at a higher frequency. Hence we conclude that our proposed design will be a good design alternative to future NoCs.

REFERENCES

[1] W. Dally and B. Towles *Principles and Practices of Interconnection Networks* USA: Morgan Kaufmann Publishers Inc., 2003.

[2] W. Dally and B. Towles *Route packets, not wires: on-chip interconnection networks* in proceedings of the Design Automation Conference, 2001, pp. 684-689.

[3] J. Henkely et al., *On-chip networks: A scalable, communication-centric embedded system design paradigm* in proceedings of the International Conference on VLSI Design, 2004, pp. 845-851.

[4] C. Fallin et al., *MinBD: Minimally-buffered deflection routing for energy-efficient interconnect* in proceedings of International Symposium on Networks-on-Chip, 2012, pp. 1-10.

[5] T. Moscibroda and O. Mutlu, *A case for bufferless routing in on-chip networks* in proceedings of International Symposium on Computer Architecture, 2009, pp. 196-207.

[6] J. Jose et al., *DeBAR: Deflection based adaptive router with minimal buffering* in proceedings of International Conference on Design, Automation and Test in Europe, 2013, pp. 1583-1588.

[7] B. Nayak et al., *SLIDER: Smart Late Injection Deflection Router for Mesh NoCs* in proceedings of International Conference on Intelligent Computing, Communication and Devices, 2013, pp. 377-383.

[8] C. Fallin et al., *CHIPPER: A low complexity bufferless deflection router* in proceedings of IEEE International Symposium on High Performance Computer Architecture, 2011, pp. 144-155.

[9] S. A. R. Jafri et al., *Adaptive flow control for robust performance and energy* in proceedings of International Symposium on Microarchitecture, 2010, pp. 433-444.

[10] G. Kim et al., *Flexibuffer : Reducing leakage power in on-chip network routers* in proceedings of Design Automation Conference, 2011, pp. 936-941.

[11] G. R. Jonna et al., *Minimally Buffered Single-Cycle Deflection Router* in proceedings of International Conference on Design, Automation and Test in Europe, 2014, pp. 1-4.

[12] N. Jiang et al., *A Detailed and Flexible Cycle-Accurate Network-on-Chip Simulator* in proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software, 2013, pp. 86-96.

[13] R Ubal et al., *Multi2sim: A simulation framework to evaluate multicore-multithreaded processors* in proceedings of International Symposium on Computer Architecture and High Performance Computing, 2007, pp. 62-68.

[14] www.synopsys.com