Predictability Enhancement in Non-preemptive NoCs using Selective Packet Splitting

Bharath Sudev and Leandro Soares Indrusiak
Department of Computer Science
The University of York, U.K. YO10 5GH
Email: [bs638, leandro.indrusiak]@york.ac.uk

Abstract— **Performance predictability enhancement in Network-on-chip routers are usually brought about by employing techniques like Time Division Multiplexing or Virtual Channels, which comes with overheads in area and energy which can be substantial. With this paper, we present an alternative approach that can emulate the effect of preemption without the major area overheads which are associated with preemption implementation. The technique employs routers capable of splitting low priority packets (rather than preempting) when a higher priority packet is encountered which is destined to use the same link. This splitting would hence allow the high priority packet to have the same effect as if the low priority packet was preempted while having lower implementation overhead as the splitting logic is simpler. We test the technique with different traffic scenarios and load levels, and analyse performance merits as well as implementation overheads.**

Keywords- Network-on-Chip, Predictability, Packet Splitting, Non-preemptive Network-on-Chip.

I. Introduction

Predictability enhancement techniques like Preemptive arbitration [1], Virtual Channels [2] and Time Division Multiplexing [3] usually comes with high hardware (and hence power) requirements primarily due to the use of buffers which are expensive in both aspects. With this paper we explore the Selective Packet Splitting (SPS) technique which aims to improve Network-On-Chip (NoC) packet predictability by emulating the effect of preemption without employing a massive array of buffers hence making the NoC simpler and less expensive. This would allow NoC routers to provide prioritisation for packets without the use of excessive hardware hence making it a viable option for embedded applications with strict resource restrictions. Using a prototype, we validate the performance enhancement brought about by the technique and estimate the resultant implementation overhead.

The paper continues with the review of literature from the field in Section II followed by Section III where the details of proposed technique can be seen. Section IV provides the implementation results followed by future work and conclusion as subsequent sections V and VI.

II. Background

Several NoC architectures have been developed and used over the years like Hermes [4], QNoC [5], AEthereal [6] and MANGO [7], each presenting a specific kind of trade-off between benefit and overhead. For example; Moraes et al designed the Hermes NoC to be extremely light weight and simple but it came with the overhead of its inability to provide latency guarantees hence making it unpredictable. On the contrary, AEthereal used TDM to ensure packet predictability guarantees thereby making the NoC almost completely predictable. Even though TDM ensured the timeliness of the packets in this case it made the NoC bulky and as TDM utilises slot tables, it limited the scalability and link utilisation.

Designers have also employed Virtual Channels to ensure packet predictability like in QNoC and MANGO where the routers would be able to transmit a higher priority packet even when the link is being used by a lower priority packet. Such NoCs utilised buffers to preempt the low priority packet so that the higher priority packet can be transmitted as soon as possible. Commercial NoCs like seen in Tilera's [8] processors and NetSpeed [9] designs also use VCs but the use of VCs bring about a heavy overhead of buffers and associated logic hardware. This effect is quite clear in the work by Mello et al in [10] where a Hermes NoC with Virtual Channels was evaluated. The paper stated that while the Hermes NoC with a single VC took 17% of the logic area of their hardware test-bed, the design with two and four VCs took 32.61% and 75.41% of the logic area respectively, which certainly cannot be neglected.

Another similar technique is Link Division Multiplexing (LDM) [11] where the physical link is multiplexed to transfer multiple packets simultaneously. Though LDM needs fewer buffers than TDM or VCs, it also results in significant hardware overhead due to its complexity.

One of the prominent characteristics shared by most of these designs is the extensive use of buffers. This can be infeasible for lightweight embedded systems as buffers cost heavily in area and consume high magnitudes of power. For example; Kundu in [12] point out that the buffers amount to 15% of the area of their basic router and it consumes 46% of the total power and in the case of the TRIPS [5] processor, 75% of the router area is dominated by input buffers alone.

For industrial or automotive embedded applications that are supposed to be minimalistic; such excessive use of resources can be quite infeasible even though there might be packet flows which are time critical. With this paper, we are investigating the use of packet splitting in NoC routers so that the effect of preemption can be emulated without the wide use of buffers as seen with the classical preemption technique. Since the packet splitting logic is relatively simple and less hardware intensive; routers employing packet splitting would be light weight and hence more suitable to be used in lightweight embedded systems.

III. Selective Packet Splitting

Consider QNoC as an example for a typical priority preemptive NoC architecture. QNoC has a 4 virtual channel design hence supporting packets with four service levels. In QNoC, preemption allows the transmission of high priority packets even if the corresponding link is being used by a lower priority packet. To enable preemption, QNoC routers have buffers for each service levels each of which would be capable of storing a few flits. Having buffer for the four service levels on each of the five input ports on each and every router can have an undesirable impact on the overall hardware requirements primarily when dealing with embedded systems where hardware and power requirements are restricted. Similarly, as the number of VCs increase, the associated overhead also multiplies many folds along with increase in power consumption.

By this paper we are exploring a low overhead technique to emulate the effect of preemption by splitting packets. SPS aims to improve packet predictability by enabling routers that encounters preemptible scenarios to split the associated low priority packet so that the high priority packet can be transmitted as soon as possible. The decision to split a packet would depend on two parameters: priority difference (PD) between competing packets and the number of remaining flits (RF) from the low priority packet. Both parameters can be assigned statically or varied dynamically according to the required magnitude of slack in predictability. These two parameters would allow the NoC to vary its effort in predictability enhancement depending on the criticality and type of data.

If a packet satisfies these two conditions, the router would split the transmission by sending a tail flit followed by the construction of a new header which would then initiate a new arbitration request at that router. This would allow the higher priority packet (which would already have requested arbitration) to secure arbitration immediately and hence reduce latency. This would hence relieve the routers from the issue of handling multiple service levels simultaneously using buffers thereby allowing routers to be simple and lightweight.

The technique was tested using a router design following a five port architecture based roughly around Hermes [4] hence employing XY-routing [13] and wormhole switching to reduce hardware requirements. The design uses a uniform mesh topology and unlike Hermes, each packet header includes a priority value which is used by the arbitration unit of the router to resolve contention between packets over output ports.

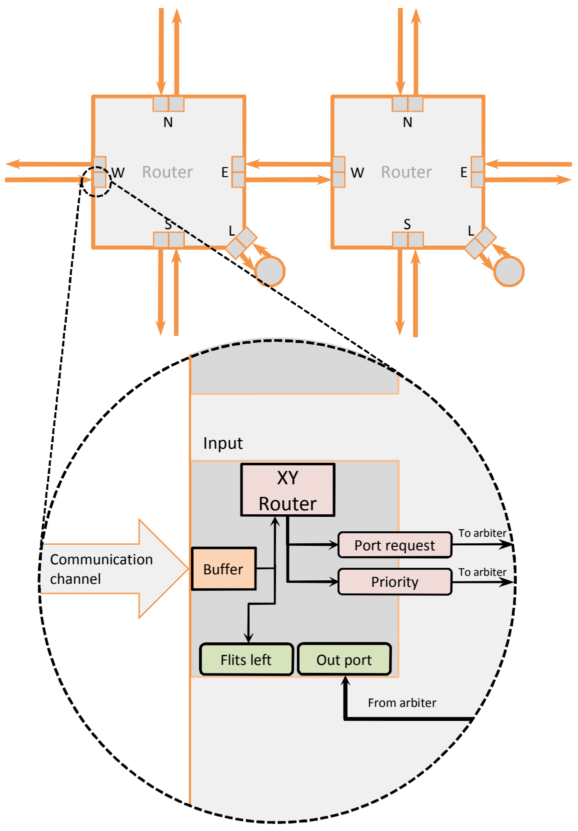


Figure 1: Basic Router Design

As shown in Figure 1, the routers have buffered input ports which on reception of a packet header employ XY-routing to set the 'port request' register and the 'priority' register in accordance with the destination and priority information carried. The arbitration unit in the router then checks 'port request' and 'priority' registers of all input ports to provide arbitration to the qualified ports.

The arbiter then establishes connection by setting the 'out port' and 'flits left' registers on the input port which permits the input port to send flits to the allocated output port so that the flits could be transferred away through the communication links. As the flits are being transferred, the input port also decrements the value in the 'flits left' register so that when the value reaches zero, the connection can be closed by re-setting the 'out port' register value to zero.

As shown in Figure 2, the state machine that manages the different stages of communication (arbitration request, arbitration, data transfer and close connection) has an extra state to perform packet splitting depending on the parameters set. So, once the conditions are satisfied for packet splitting, the input port concludes the active low priority communication with a tail flit and then releases the associated output port by setting the respective 'out port' register to zero. Simultaneously with the termination of the communication the input port also issues a new arbitration request for the splitted packet so that once the high priority communication is completed, the low priority communication can be resumed. To enable this, the most significant bit of every flit is used to denote a tail flit.

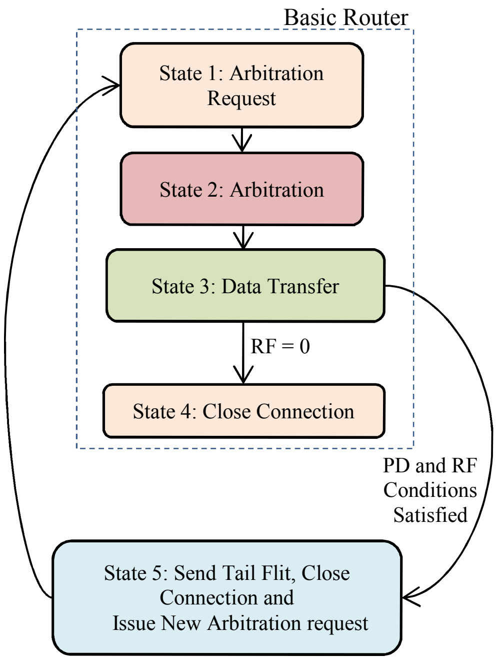

Basic Router
State 1: Arbitration Request
State 2: Arbitration
State 3: Data Transfer
RF = 0
State 4: Close Connection
PD and RF Conditions Satisfied
State 5: Send Tail Flit, Close Connection and Issue New Arbitration request

Figure 2: Input port operation

As SPS does not employ buffers to deal with each service level, the NoC can be scaled easily and the technique would be able to function irrespective of the number of service levels.

IV. Implementation Result

The evaluation prototype for the technique was designed in Bluespec System Verilog [14] and consists of the router design enveloped in a parameterisable testbench which replicates and interconnects the routers according to a 2D-mesh topology. The local port of each router was connected to packet-generators which could be pre-set with parameters like start time, period, packet size, priority and destination. The packet-generator configuration is auto-generated as Bluespec source code using a custom built code generator which could either configure the generators randomly or in accordance with a series of algorithms to generate specific configuration patterns. Apart from conditioning the data to the required flit format and injecting into the NoC, the packet-generators also receive packets from the NoC and export evaluation figures to an external file. This external data file is then analysed by a custom built macro code running inside the spread sheet software to generate latency statistics and graphs.

A. Performance

We use average load per link (V) as the measure of the load on the NoC (Eq. 1) and a 4x4 NoC was analysed with random traffic to extract performance statistics.

$$V = \left\{\sum_{x=0\ to\ W-1} \sum^{y=0\ to\ H-1} \left(\frac{C_{x,y}}{P_{x,y}}\right)\right\} / L_{(W \times H)} \qquad \text{Eq. 1}$$

(W- NoC Width, H- NoC Height, C- No Load Latency, H- Hops, P- Period, L- Number of links)

The latency performance of the technique is interpreted as boxplots with priority of the packet on the X-axis and latency on the Y-axis. In box plots the whiskers shows the extreme cases of latency and the boxes indicate the upper and lower quartile of latency with the middle line depicting the median. So, shorter box and whiskers show lower variability in latency and lower box and whiskers show lower magnitude of latency.

In the plots, the red box and whiskers (first one of each pair) depict the performance of a priority non-premptive NoC while the green box and whiskers (second one of each pair) show the performance with SPS logic engaged under the same scenario.

a) Variation with Traffic patterns

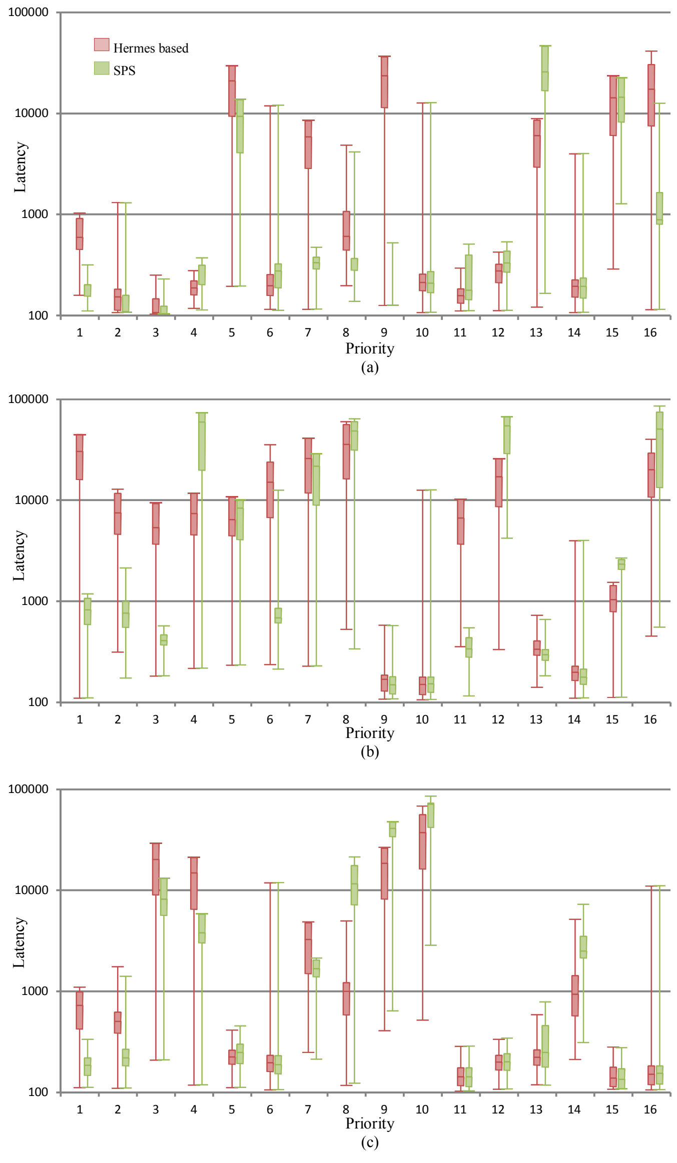


Figure 3: Performance with Random Traffic

(a) Random Traffic 1 (b)Random Traffic 2 (c) Random Traffic 3

To verify the performance enhancement brought about by the technique with varying traffic patterns, three 4x4 NoCs with different traffic flow mappings were tested and the latency statistics obtained from the simulation were interpreted. We used a moderate load of V=0.8 for the three traffic patterns and the resulting boxplots are added as Figure 3.

In the box plots in Figure 3, it can be observed that irrespective of the traffic scenarios used, the box and whiskers for the high priority packets (like 1,2,3 and 4) were mostly shorter and lower with the employment of packet splitting showing lower variability and magnitude of latency. As a characteristic of some specific traffic scenarios some packets can have decreased performance (like packet 4 in Figure 3b) caused by higher priority packets than those and this is the limitation brought about by the traffic pattern than the technique.

b) *Varying load by changing overall payload flit count*

For verifying the effect of the increase in load on the NoC due to the increase in payload flits, we carried out experiments with a NoC increasing the packet sizes.

Figure 4a shows the latency comparison of a NoC at an intermediate load of V =0.7 and the same NoC with increased load levels of V=0.9 and V=1.3 are interpreted as Figure 4b and Figure 4c.

It can be observed that as the load on the NoC was increased it had an adverse effect on the latency performance of the packets on the basic NoC hence resulting in longer and higher box and whiskers. This can be quite acceptable with low priority packets but the increased load would have similar effect on high priority packets as seen in the plots. With SPS logic engaged, the adverse effect on the latency of high priority packets (like 1 to 5) can be seen less hence lower and shorter box and whiskers are observed.

c) *Varying load by changing overall header flit count*

To increase the number of header flits in the NoC and hence the load, the NoC was tested by decreasing the period of tasks and the resulting performance is interpreted as Figure 5a, Figure 5b and Figure 5c.

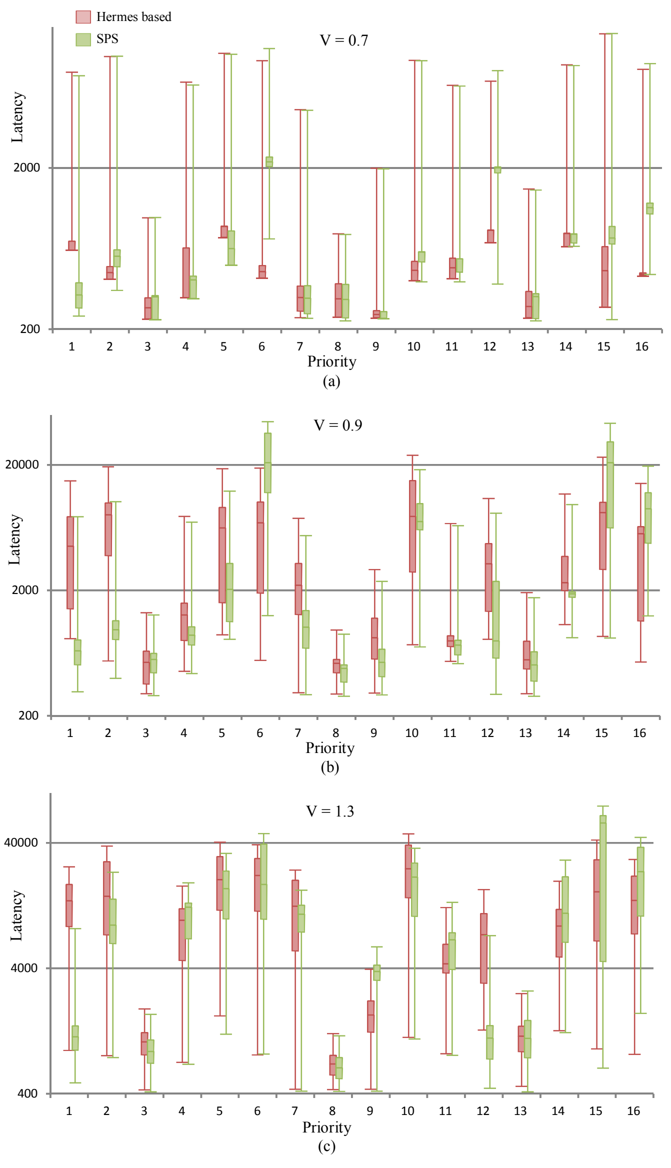


Figure 4: Performance with increased payload flit count

(a) V=0.7 (b) V=0.9 (c) V=1.3

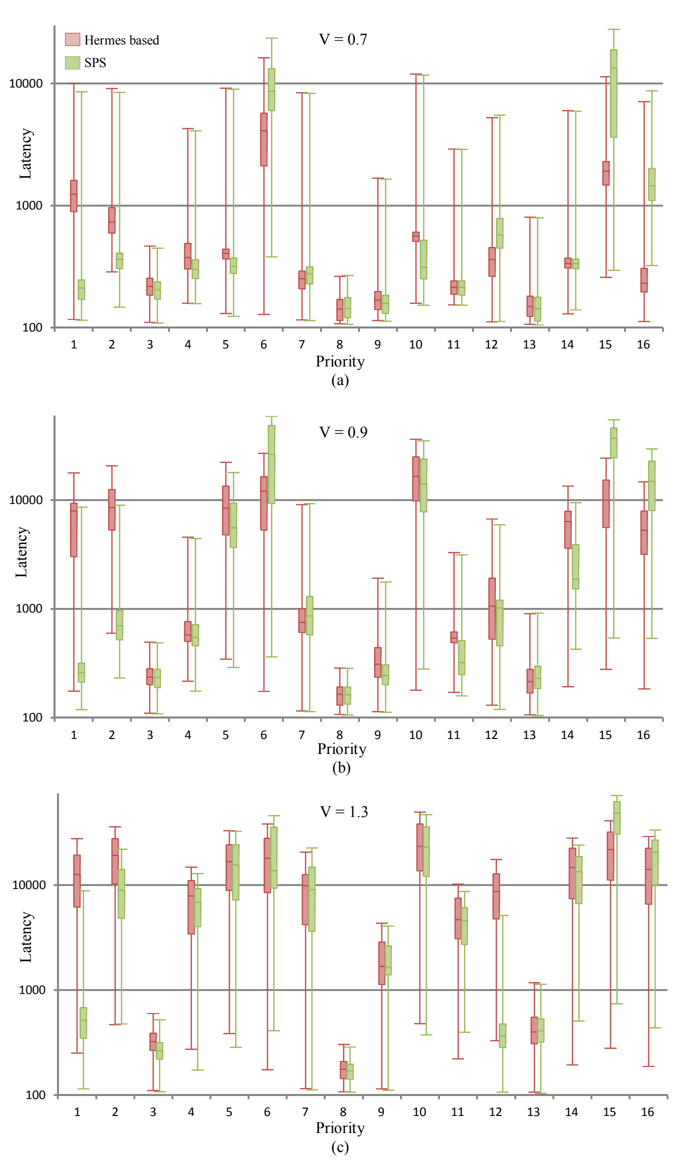


Figure 5: Performance with increased header flit count

(a) V=0.7 (b) V=0.9 (c) V=1.3

As observed with the tests in previous sub section, the use of SPS was found to be effective with the increase in the load as the high priority packets were observed to be causing lower variability and magnitude in latency than the basic non-preemptive NoC.

d) Effect of RF and PD variation

To verify the effect of RF variation on performance, we tested a NoC with RF condition set to 1, ¾ and ½ of the packet size. With those tests the SPS logic was set to trigger only if the flits left to send owing to the lower priority was equal to or greater than 1, ¾ the packet size and ½ of the packet size respectively and the result is interpreted as Figure 6.

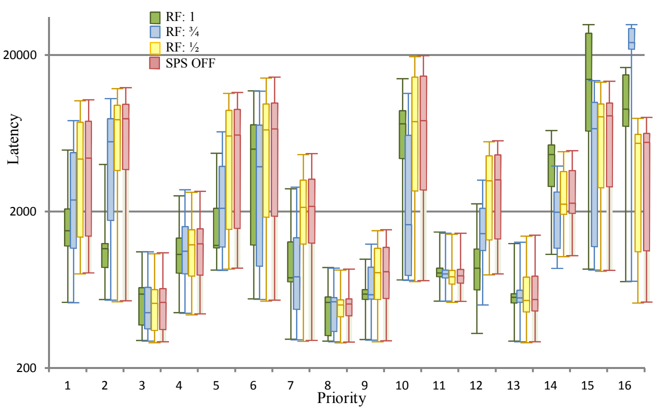


Figure 6: Effect of RF variation

In Figure 6, it can be observed that the effect of SPS can be scaled by varying the RF condition. With RF set to 1, the high priority packets were given the maximum preference and hence high priority packets are seen to be with the lowest variability and magnitude of latency. This comes at the cost of the low priority packet performance as those suffer higher variability and magnitude in latency. With the subsequent tests with RF set to ¾ and ½, the effect get moderated towards the low priority packet performance ultimately getting closer to the performance of the basic non-premptive NoC.

The variation of PD had a similar effect on latency performance and the experimental results are added as Figure 7.

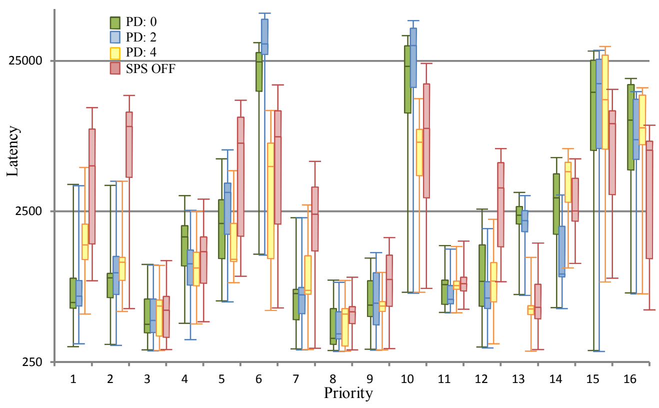
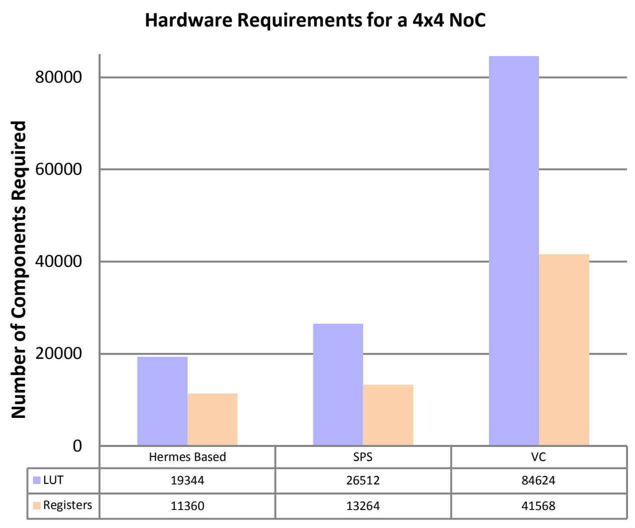


Figure 7: Effect of PD variation

Though PD variation has similar characteristics as RF variation, the later would have a better reliability in performance scaling applications than PD. This is due to the fact that as PD scaling relays on the priority of the competing packets, it can get affected by the specific characteristics related to each of the used task mapping whereas RF is a more generic parameter.

B. Hardware Overhead

The hardware overhead associated with the technique was evaluated using Xilinx Vivado™ Design Suite by synthesising the design for an FPGA. While the baseline router (2-position input buffers) based on Hermes with priority arbitration utilised 1209 Look Up Tables (LUTs) and 710 of Slice Registers of the chosen FPGA, the SPS-enabled NoC (2-position input buffers) utilised 1657 LUTs and 829 Slice registers which is minimalistic. When a comparable Virtual Channel (4 VCs with 2-position buffers) based design implemented upon the same basic NoC architecture was tested, Vivado generated a LUT requirement of 5289 and Slice Register requirement of 2598 thereby demonstrating the overhead associated.



	Hermes Based	SPS	VC
LUT	19344	26512	84624
Registers	11360	13264	41568

Figure 8: Hardware overhead comparison

The hardware overhead comparison for the techniques on a 4x4 NoC architecture is interpreted in Figure 8. It can be observed that SPS has moderate hardware requirements compared to the basic non-premptive NoC while the Virtual Channel based NoC costs more than twice as much.

V. Future Work

Even though the technique addresses the issue of link utilisation of lower priority packets, it doesn't make the NoC immune from Head-of-line (HOL) blocking. HOL blocking is one of the main issues that deteriorate the predictability of non-preemptive NoCs and quoting Huang et al from [15], "due to HOL blocking, the throughput of the links is typically limited to 58%".

As HOL blocking can affect the performance of SPS enabled routers, the technique would be improved by coupling it with Priority Forwarding [16] which would then enable the routers to neutralise HOL blocking scenarios as it occurs. This is achieved by forwarding the priority of the high priority packet encountering HOL blocking to the low priority packet blocked down the line and this can be performed alongside SPS without any side effects.

VI. Conclusion

For increasing packet predictability in non-premptive NoCs, this paper explored Selective Packet Splitting technique, its performance and overhead. With a prototype, we tested its performance with several random traffic scenarios as well as with varied load intensity on the NoC and the results were analysed. SPS was found to decrease latency variation and latency magnitude of high priority packets at the cost of low priority ones irrespective of the traffic pattern and intensity. SPS also provided additional provision of tuning the performance benefit hence providing a facility to scale the performance depending on the overall criticality of the situation. The paper also evaluated the implementation overhead of the technique and it was found to be nominal (37% Lookup tables and 17% Registers) from the basic non-preemptive NoC.

VII. Acknowledgement

This work is supported by EPSRC project LowPowNoC (EP/J003662/1) and EU FP7 projects T-CREST (288008) and DREAMCLOUD (611411).

VIII. References

[1] R. Dobkin, R. Ginosar, and I. Cidon, ‘QNoC Asynchronous Router with Dynamic Virtual Channel Allocation’, in *First International Symposium on Networks-on-Chip, 2007. NOCS 2007*, 2007, p. 218.

[2] W. J. Dally, ‘Virtual-channel flow control’, in *Proceedings of the 17th annual international symposium on Computer Architecture*, New York, NY, USA, 1990, pp. 60–68.

[3] R. Stefan, A. Molnos, A. Ambrose, and K. Goossens, ‘A TDM NoC supporting QoS, multicast, and fast connection set-up’, in *Design, Automation Test in Europe Conference Exhibition (DATE), 2012*, 2012, pp. 1283–1288.

[4] F. Moraes, N. Calazans, A. Mello, L. Möller, and L. Ost, ‘HERMES: an infrastructure for low area overhead packet-switching networks on chip’, *Integr. VLSI J.*, vol. 38, no. 1, pp. 69–93, Oct. 2004.

[5] Evgeny Bolotin, Israel Cidon, Ran Ginosar and Avinoam Kolodny, *QNoC: QoS architecture and design process for Network on Chip*. .

[6] K. Goossens, J. Dielissen, and A. Radulescu, ‘AEthereal network on chip: concepts, architectures, and implementations’, *Design Test of Computers, IEEE*, vol. 22, no. 5, pp. 414 – 421, Oct. 2005.

[7] T. Bjerregaard and J. Sparso, ‘A router architecture for connection-oriented service guarantees in the MANGO clockless network-on-chip’, in *Design, Automation and Test in Europe, 2005. Proceedings*, 2005, pp. 1226 – 1231 Vol. 2.

[8] S. Park, T. Krishna, C.-H. Chen, B. Daya, A. Chandrakasan, and L.-S. Peh, ‘Approaching the theoretical limits of a mesh NoC with a 16-node chip prototype in 45nm SOI’, in *2012 49th ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2012, pp. 398–405.

[9] ‘NetSpeed ORION: A New Approach to Design On - chip Interconnects’, 26-Aug-2013.

[10] A. Mello, L. Tedesco, N. Calazans, and F. Moraes, ‘Virtual channels in networks on chip: implementation and evaluation on hermes NoC’, in *Proceedings of the 18th annual symposium on Integrated circuits and system design*, New York, NY, USA, 2005, pp. 178–183.

[11] A. Morgenshtein, A. Kolodny, and R. Ginosar, ‘Link Division Multiplexing (LDM) for Network-on-Chip Links’, in *2006 IEEE 24th Convention of Electrical and Electronics Engineers in Israel*, 2006, pp. 245 –249.

[12] P. Kundu, ‘On-Die Interconnects for Next Generation CMPs’. Proc. Workshop On- and Off-Chip Interconnection Networks for Multicore Systems,, Dec-2006.

[13] W. Zhang, L. Hou, J. Wang, S. Geng, and W. Wu, ‘Comparison Research between XY and Odd-Even Routing Algorithm of a 2-Dimension 3X3 Mesh Topology Network-on-Chip’, in *Intelligent Systems, 2009. GCIS ’09. WRI Global Congress on*, 2009, vol. 3, pp. 329 –333.

[14] R. Nikhil, ‘Bluespec System Verilog: efficient, correct RTL from high level specifications’, in *Second ACM and IEEE International Conference on Formal Methods and Models for Co-Design, 2004. MEMOCODE ’04. Proceedings*, 2004, pp. 69–70.

[15] T.-C. Huang, U. Y. Ogras, and R. Marculescu, ‘Virtual Channels Planning for Networks-on-Chip’, in *8th International Symposium on Quality Electronic Design, 2007. ISQED ’07*, 2007, pp. 879–884.

[16] Bharath Sudev, Leandro Soares Indrusiak, ‘PFT- A Low Overhead Predictability Enhancement Technique for Non-Preemptive NoCs’, in *21st IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, Istanbul, Turkey, pp. 342 – 345.