PVS-NoC: Partial Virtual Channel Sharing NoC Architecture

Khalid Latif[1,3], Amir-Mohammad Rahmani[1,3], Liang Guang[1], Tiberiu Seceleanu[2], Hannu Tenhunen[1,3]
[1]Department of Information Technology, University of Turku, Finland
[2]ABB Corporate Research, Vasteras, Sweden [3]Turku Centre for Computer Science (TUCS), Finland
{khalid.latif, amir.rahmani, liang.guang, hannu.tenhunen}@utu.fi, tiberiu.seceleanu@se.abb.com

***Abstract*—A novel architecture aiming for ideal performance and overhead tradeoff, PVS-NoC (Partial VC Sharing NoC), is presented. Virtual channel (VC) is an efficient technique to improve network performance, while suffering from large silicon and power overhead. We propose sharing the VC buffers among dual inputs, which provides the performance advantage as conventional VC-based router with minimized overhead. We reason theoretically and demonstrate quantitatively the benefits of proposed architecture by comparing to state-of-the-art NoC routers, with various traffic patterns. Extensive experiments with synthetic and real benchmarks show significant area and power saving with similar performance compared to latest VC based NoC architectures.**

***Index Terms*—Virtual Channel; Networks-on-Chip (NoC); Resource utilization;**

I. Introduction

Ever-increasing requirements on electronic systems are one of the key factors for evolution of the integrated circuit technology. Multiprocessing is the solution to meet the requirements of upcoming applications. Multiprocessing over heterogeneous functional units require efficient on-chip communication [13]. Network-on-Chip (NoC) is a general purpose on-chip communication concept that offers high throughput, which is the basic requirement to deal with complexity of modern systems. All links in NoC can be simultaneously used for data transmission, which provides a high level of parallelism and makes it attractive to replace the typical communication architectures like shared buses or point-to-point dedicated wires. Apart from throughput, NoC platform is scalable and has the potential to keep up with the pace of technology advances [10]. All these enhancements come at the expense of area and power. In the Raw Architecture Workstation (RAW) multiprocessor system with 16 tiles of processing elements, interconnection network consumes 36% of the total chip power while each router dissipates 40% of individual tile power [18].

A typical NoC system consists of processing elements(PEs), network interfaces (NIs), routers and channels. The router further contains switch and buffers. Buffers consume the largest fraction of dynamic and leakage power of the NoC node (router + link) [4][3]. Storing a packet in buffer consumes far more power as compared to its transmission [20]. Thus, increasing the utilization of buffers and reduction in number and size of buffers with efficient autonomic control enhances the system performance and reduces the area and power consumption.

Wormhole flow control have been proposed to reduce the buffer requirements and enhance the system throughput. But on other hand, one packet may occupy several intermediate switches at the same time. In typical NoC architectures, when a packet occupies a buffer for a channel, the physical channel cannot be used by other channels, even when the original message is blocked [13]. This introduces the problem of deadlock and livelock in wormhole scheme. Virtual Channels (*VCs*) are used to avoid deadlock and livelock. Fig.1 shows a typical VC router architecture [14]. VC flow control exploits an array of buffers at each input port. By allocating different packets to each of these buffers, flits from multiple packets may be sent in an interleaved manner over a single physical channel. This improves the throughput and reduce the average packet latency by allowing blocked packets to be bypassed. By inserting the VC buffers, we increase the physical channel utilization but utilization of inserted VC buffers is not considered.

Now, the issue that needs attention is the utilization of both, the physical as well as the VC utilization. A well designed network exploits available resources to improve performance [2]. ViChaR architecture [16] points out the VC buffer utilization, discussed later in section II. Even for ViChaR architecture, it can be observed that if there is no communication on some physical channel at some time instant and at the same time, neighboring channel is overloaded, free buffers of one physical channel cannot contribute for congestion control by sharing the load of neighboring channel. Adaptive routing technique provides a solution to these issues but introduces some other problems like packet reordering.

Buffer utilization can be enhanced by sharing them with the other ports, while not used by the current port. Fully shared buffer architectures can deliver a high throughput at the expense of area and power consumption with maximum degree of buffer utilization. On other hand, typical NoC architectures without any sharing of buffers require less power to operate but throughput is also affected negatively. The buffer utilization is not good as well. If a buffer is available to receive the data but there is no data on input port, the buffer cannot be used by neighboring overloaded ports. Thus each architecture has its own pros and cons and an ideal tradeoff between performance, power and area is needed.

The available autonomic NoC architectures are fault tolerant

1066-6192/11 $26.00 © 2011 IEEE
DOI 10.1109/PDP.2011.87

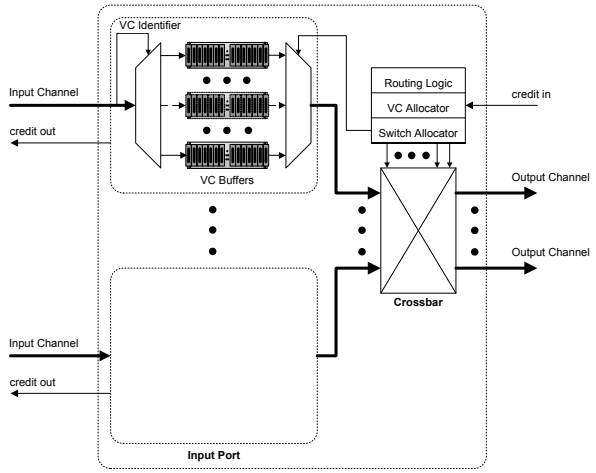


Fig. 1. Typical Virtual Channel Router Architecture [14].

and can deliver the desired performance without big impact in case of certain manufacturing faults in architecture by using the intelligent routing algorithms like [21], [8], [6] or [12]. The key problem with these fault tolerant techniques and routing algorithms is that many resources become useless for the system with the fault on one resource. Due to the link failure in typical VC based NoC architectures, the VC buffers connected to it become useless for the system. To reduce the effect of fault on the system performance, such unused resources should be utilized by the system. This paper provides a novel architecture with autonomic sharing of buffers between two input ports with enhanced utilization of resources, and reduced crossbar size and power consumption. The remainder of this paper is organized as follows. Section II presents an overview of the state of the art in NoCs especially using the VCs. Section III presents the link load analysis for NoC based system. Section IV explains our proposed architecture and how its architectural comparison with modern NoC architectures. Finally experimental results and conclusions are drawn.

II. Related Work

Several approaches have been proposed to enhance the buffer utilization in NoC based system. This comes as a performance improvement to utilize the unused buffers. This section reviews several relevant propositions in this theme.

Lan et al. [11] addresses the buffer utilization by making the channels bidirectional and shows significant improvement in system performance. But in this case, each channel controller has two additional tasks: dynamically configuring the channel direction and to allocate the channel to one of the routers, sharing the channel. Also, there is a 40% area overhead over the typical NoC router architecture due to double crossbar design and control logic. The extra tasks for channel controller and increased crossbar size contribute to the power consumption as well.

Nicopoulos et al.[16] presents a Dynamic *VC Regulator* (ViChaR) for NoC routers. The authors address the buffer utilization by using the unified buffered structure (UBS) instead of individual and statically partitioned FIFO buffers. It provides each individual router port with a variable number of VCs according to the traffic load. The architecture provides around 25% improvement in system performance at a small cost of power consumption. The architecture enhances the buffer utilization under heavy traffic load at the port. If there is no load at the port, the buffer resources cannot be utilized by neighboring overloaded ports.

Soteriou et al. [17] introduces a distributed shared buffer (DSB) NoC router architecture. The proposed architecture shows a significant improvement in throughput at the expense of area and power consumption due to extra crossbar and complex arbitration scheme.

Coenen et al. [5] presents an algorithm to optimize size of decoupling buffers in network interfaces. The buffer size is proportional to the maximum difference between the number of words produced and the number of words consumed at any point in time. This approach shows significant reduction in power consumption and silicon area. The buffer utilization in idle time with optimal buffer size can contribute to reduce the overall system power consumption without affect the system performance. If some buffer is idle at some time instant, it can share the load of neighboring input channel and thus increase the utilization of existing resources with a small control logic.

Neishabouri et al. [15] propose the router architecture with *Reliability Aware VC* (RAVC) architecture by [15] allocates more memory to the busy channels and less to the idle channels. This dynamic allocation of storage shows 7.1% and 3.1% latency decrease under uniform and transpose traffic patterns respectively at the expense of complex memory control logic. Though this solution is latency efficient but not area and power efficient, which was not discussed by the authors.

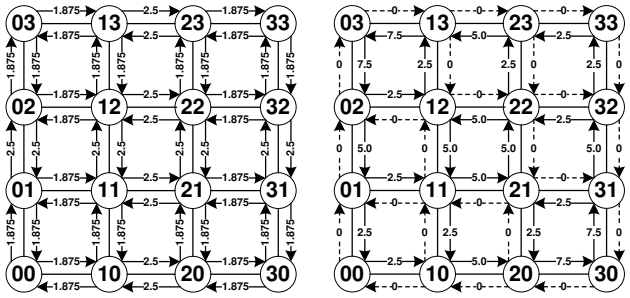
The main motivation of this work is to propose a NoC architecture with considerations of all the issues discussed above. A NoC architecture with an ideal tradeoff between the performance and the mentioned guides to propose the partial sharing of VC buffers.

III. Resource Utilization Analysis

To enhance the utilization of interconnection resources, it is necessary to make the analysis of each network resource individually. Traffic pattern is unpredictable, when multiple applications running on one MPSoC simultaneously. Routing algorithm controls the utilization of communication channels. The system utilizes the router resources according to the load on incoming channels. Thus, the link load analysis can provide the base to formulate the solution for utilization of communication resources.

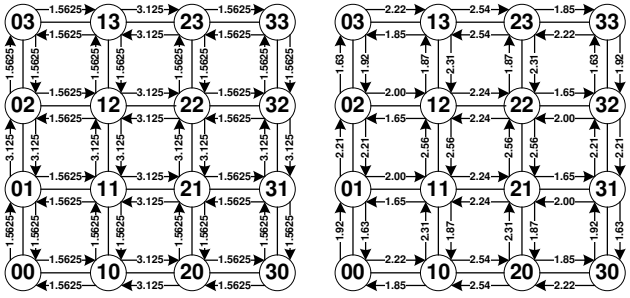
A. Link Load Analysis

In synthetic traffic analysis, the average load for each link is calculated with variety of routing algorithms.As a test case, uniform, transpose, bit complement and NED traffic patterns were analyzed with XY routing. In the uniform traffic pattern, a node sends a packet to any other node with an equal



(a) Uniform traffic load.

(b) Transpose traffic load.



(c) Bit complement traffic load.

(d) NED traffic load

Fig. 2. Traffic load analysis for XY-routing.

probability while in transpose traffic pattern, each node (*i*,*j*) communicates only with node (*j*,*i*). For bit complement traffic load, each node (*i*,*j*) communicates only with node (*M*-*i*,*N*-*j*), if mesh size is MxN. The NED is a synthetic traffic model based on Negative Exponential Distribution where the likelihood that a node sends a packet to another node exponentially decreases with the hop distance between the two cores. This synthetic traffic profile accurately captures key statistical behavior of realistic traces of communication among the nodes [1].

Fig.2 shows the percentage of the load for each link on the network. Now consider the node '12' in fig.2(b). The input ports to receive data from nodes '11' and '13' are not used at all during the whole simulation independent of the total simulation time. But the input ports from left and right receive the traffic load. The traffic load from node '22' is double than the load from node '02'. In case of typical NoC architectures, the link between nodes '12' and '22' is overloaded all the time but cannot utilize the available resources of other ports. Similar behavior can be observed for odd-even routing.

For any router, two ports are overloaded as compared to other two for all the traffic patterns. In most cases, the identical traffic load value is for another input port as well. Thus, there are only two load values and each value is for two ports of same router. Another interesting observation is that the traffic load values are different for opposite directions. For load balancing and resource utilization, the resources can be shared among a pair of input ports. The pair of ports is selected in such a way that one port has higher load value and other one has a lower load value. The resources can be

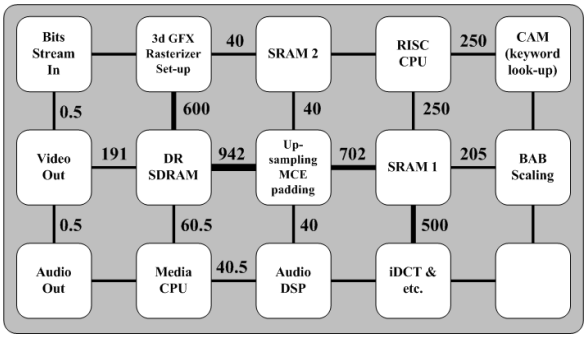


Fig. 3. MPEG4 application [7].

shared among all the input ports but it increases the input cross bar size, which increases the power consumption, area and average packet latency due to more complex controller. Another important and interesting issue is that sharing the resources among two ports with balanced loads can deliver the degree of resource utilization and the throughput closer to the the *Fully VC shared NoC* (FVS-NoC) architecture with significant reduction in power consumption.

B. Real Benchmark Analysis

The MPEG4 application presented by [7] was selected for resource utilization analysis. The application with bandwidth requirements is shown in fig.3. Now consider the link loads for DR-SDRAM node. If right and down side ports share the router resources, while left and up side ports also share the resources. The heavy load value 942 MB/sec from right port can utilize the resources from down port, which contains a smaller load value of 60.5 MB/sec. Thus by sharing communication resources among two ports can balance the input load on all ports with significantly increasing the crossbar size. The average load on input ports will be similar to the full sharing of resources by using this approach.

IV. The Proposed Router Architecture: PVS-NoC

To enhance the performance of typical VC architecture, new VC buffers should be inserted because a congested port cannot utilize the resources of neighboring free port. Thus, the resource utilization is the issue for VC architecture. Network resources can be fully utilized by sharing the resources among all the input ports. This increases the control logic complexity, crossbar size and power consumption. Thus the tradeoff between resource utilization and power consumption is needed. Instead of sharing the VC buffers among all the input ports, if the buffers are shared among two ports, the buffer utilization will be increased and approach the utilization closer the fully shared architecture as discussed in section III without big power consumption and silicon area overhead. Number of buffers can be reduced because of enhance utilization. This reduces the power consumption significantly because the router buffers consume the major fraction of power and area for NoC interconnection as discussed in section I. Thus Partial VC Sharing NoC (PVS-NoC) architecture has been

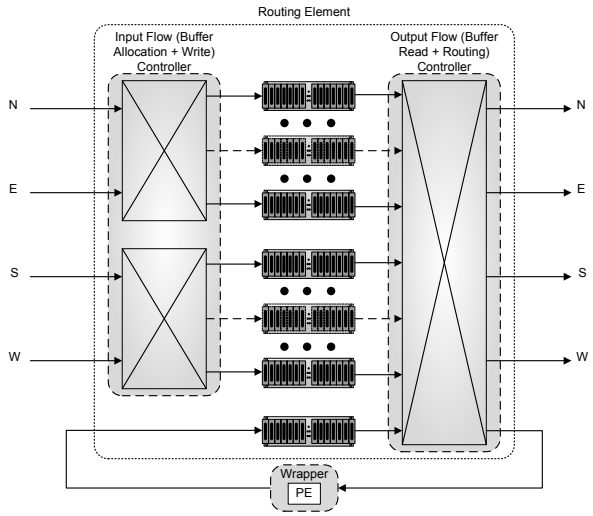


Fig. 4. System Level specification of Router Architecture.

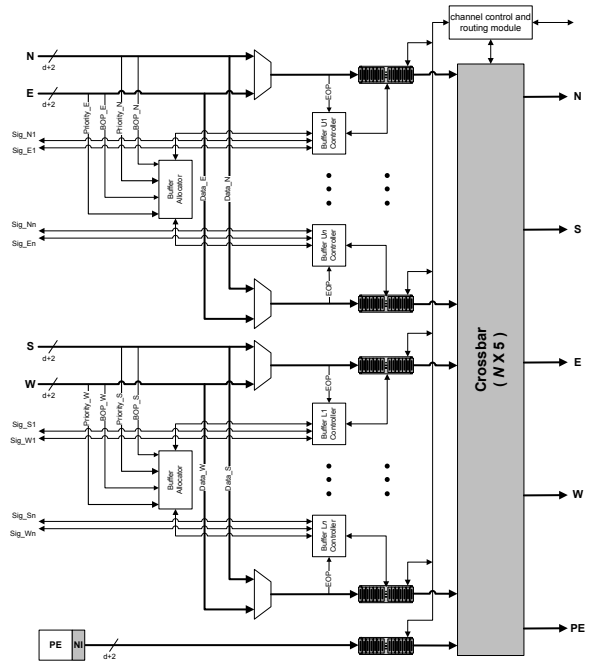


Fig. 5. System Level specification of Router Architecture.

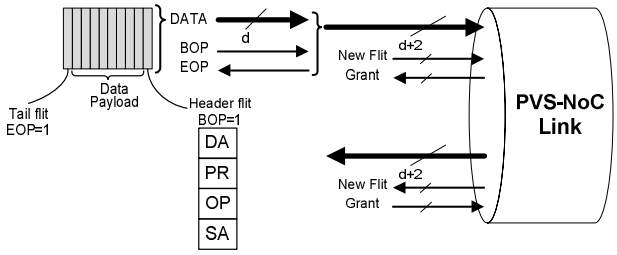


Fig. 6. Data transmission format.

proposed as tradeoff between resource utilization and power consumption.

PVS-NoC architecture uses the hybrid control logic. Input uses the hierarchy of arbiters while central control logic is used for output crossbar. The system level PVS-NoC architecture is shown in Fig.4. The input crossbars and control logic are responsible for buffer allocation and receiving the data packets as explained in section IV-A. The output crossbar and control logic use the typical NoC out architecture and its tasks are route computation and packet transmission as discussed in section IV-B. Both control logics operate according to the VC buffer status and does not communicate with each other. The detailed PVS-NoC architecture is shown in fig.5. The signalling details for each component are explained later in this section.

Data is injected to the network in the form of packets produced by network interface (NI). The format is shown in Fig. 6. While receiving the packet, NI also depacketize it and deliver the payload to the PE. Header flit carries the source address (SA), the operational code (OP), the priority level (PR) and the destination address (DA). BOP and EOP are the indicators of header and tail flits respectively.

A. The Input Controller and Buffer Allocation

The contribution of PVS-NoC architecture stands on the input side because the issue is to enhance the buffer utilization by allocating the free buffers to overloaded ports, which is the task of the input control logic as discussed at beginning of this section. The VC buffers are shared between neighboring input ports as shown in fig.5. Input architecture of two ports is replicated for other pairs of input ports and can be extended to any number of input ports according to the topology requirements. The PE uses the dedicated buffer for packet injection to the network. The detailed architecture of input side for pair of neighboring ports is shown in Fig.7.

The buffer dedicated to the local PE is not shared with any other port. In this section, two input ports are used to explain the complete operation. The control of two ports which share the VC buffers is completely independent from the other input control logic as shown in fig.5. The control logic for buffer allocator is completely independent from the other buffer allocators. The task of buffer allocator is to keep track of free buffers and allocate them to the incoming traffic. After allocation, Buffer write control directly communicates withe the neighboring nodes. This distributed control reduces the communication overhead and power consumption. The input control system operates in following phases:

***Buffer Allocation*:** The *Buffer allocator* receives the requests from neighboring routers for new buffer allocation in the form of *BOP* signal. With the status of all the buffers, the allocator allocates the free buffers to the requesting router. If only one buffer is available and both routers are requesting for the buffer, the router with higher priority value (*PR*) in header flit is allowed to use the buffer for packet transmission.

***Local Signalling*:** After the allocation, the corresponding *Buffer write* controller is informed by raising the *Allocated*

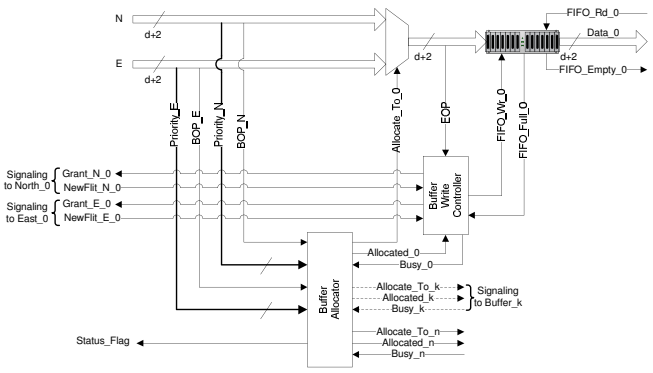


Fig. 7. Architecture of Input part of Router.

signal from buffer allocator to the buffer write controller. In response, the Buffer write controller raises the *Busy* signal. At the same time, the buffer allocator signals the corresponding multiplexer by *Allocated_To* signal, to which input, corresponding buffer has been alloted.

***Packet Receive*:** After local control signalling, the Buffer write controller signals the *Grant* to the corresponding neighbor router. The latter uses the *Grant* signal as VC identifier for requested packet and always raises the equivalent *NewFlit* signal, while transmitting flits for that packet.

***Buffer De-Allocation*:** The *Busy* signal from the Buffer write controller goes down upon arrival of tail flit marked by *EOP* signal. After receiving *Not Busy* signal from the Buffer write controller, the Buffer allocator marks the corresponding buffer as free. If not already raised, the *Status_Flag* is raised.

***Status Flag*:** The *Status_Flag* signal is the logical *AND* operation of all the *Busy* signals from buffer write controllers. When all buffers have been allocated, the *Status_Flag* is raised to inform the neighboring routers that no buffer is available for transmission of new packets. If at least one buffer is available for allocation, the *Status_Flag* is in down state.

The *Buffer allocation* unit works only when the *BOP* signal is received. Once the buffer has been tied to the requesting router, the Buffer allocator goes into the sleep mode to save power. It does not consume power until the next BOP signal is received. Similarly, the Buffer write controller works only after receiving the *Allocated* signal from buffer allocator and goes in sleep mode at the *EOP* signal.

B. The Output Controller and Routing Algorithm

The *Output* part is modeled by a typical N×5 crossbar, where N is the total number of buffers in the router including the buffer dedicated to local PE. The crossbar size can be customized according to the topology requirements. The mesh topology was considered to decide the number of I/O ports of the router. Distributed control logic was used here as well. There is one central controller which is the key part of the router. The central output controller decides the routing policy and computes the route for the packet. The port controller controls the packet transmission and the communicates with the destination router, without involving the central controller. It also decides the flow control as well. A worm-hole flow control was used, which makes efficient use of buffer space as small number of flit buffers per VC are required [19]. The Output controller operation can be divided into several steps like route computation, data transmission and de-allocation of buffers.

C. Comparison with Existing Architectures

To make the comparison with existing architectures, 10-port router with 2-ports per direction was selected with 10 internal buffers. Table. I. shows a comparison of PVS-NoC architecture with other NoC architectures.

The typical VC NoC represents a conventional VC NoC architecture with 2 VCs per port and use unidirectional channels to communicate with neighboring routers. Thus two channels are required between two neighboring routers for two way communication. In case of heavy traffic load on a certain port, typical VC NoC architecture can provide only 2 VCs to receive the packets on that port. The PVS-NoC can provide 4 VCs to the same port under heavy traffic load on same port. Thus, the VC utilization has been doubled with slight overhead of crossbar size.

BiNoC has two bidirectional channels, which can be used according to the requirements to communicate in any direction [11]. As compared to the BiNoC architecture with 10-in-out channels, the PVS-NoC approach provides 5-input and 5-output channels. PVS-NoC can provide 4-input and 4-output VCs for bidirectional communication between two nodes. On other hand, BiNoC provides only two physical channels for packet transfer between two nodes with in any direction. The option of direction selection is provided at the cost of big crossbar size. Another issue to be addressed here is scalability. The number of VC buffers can be selected according to the application and topology requirements for the proposed architecture. To insert a new VC, the buffer and a controller are needed without any modification in existing logic and slight increase in crossbar resources. To insert a new buffer in BiNoC architecture, a separate buffer allocator is required and cross bar size will increase exponentially.

The DSB-175 and DSB-300 router architectures have already been explained by [17]. To make the exact comparison, we define DSB-160 architecture in accordance with [17]. The DSB-160 is the router with 160-flits of aggregate buffering. The buffers are divided between 5 middle memory banks with 16-flit buffers per bank and aggregate 80-flit input buffers comprising two 8-flits buffers (VCs) at each input port. Five memory banks are not considered in comparison table I because only one flit can be written into and read from a middle memory in DSB architecture, which reduce the utilization of memory banks. Thus the static power consumption without increasing the system performance is the major overhead of DSB architecture as compared to the PVS-NoC.

For comparison purpose, FVS-NoC was used. In FVS-NoC architecture, any of 10 VC buffers can be allocated to any input port to receive the traffic. All the channels are unidirectional for FVS-NoC. The architecture provides the

Architecture⇒ Resource⇓	Typical VC NoC	BiNoC	Distribute Shared Buffer NoC (DSB-160)	FVS-NoC	PVS-NoC
Number of Buffers	10	10	10	10	10
Channels/Direction	1-in 1-out	2-inout	1-in 1-out*	1-in 1-out	1-in 1-out
Maximum VCs/Physical Channel	2	1	2	10	4
Buffer Size	16 flits	16 flits	8 flits	16 flits	16 flits
Total Buffer Size	160 flits	160 flits	160 flits	160 flits	160 flits
Crossbar	10X5	10X10	2(5X5)	5X10 and 10X5	2(2X4) and 10X5

TABLE I
COMPARISON WITH EXISTING NOC ROUTER ARCHITECTURES

maximum utilization of VC buffers at the cost of extra big crossbar, which makes the solution area and power expensive as compared to the proposed architecture.

D. Virtual Channel Sharing under Faults

The additional feature of PVS-NoC is to retain the system performance till certain level after the occurrence of fault. In NoC based interconnection platform, the fault can occur in three types of components: physical link, buffer and the controller. Consider that the fault has been occurred on a network inter-router link. In typical architectures, the input/output buffers connected to the link cannot be utilized in future. Suppose that there are 'X' faulty links in NoC based system and each input port contains 'V' VCs with buffer depth 'd' and each flits size or buffer width is 'f' bits. If there is a small VC controller for each VC, the resources useless after the fault on physical channel can be estimated by Eq. 1.

$$Resource\ Overhead = X.V.d.f_{(Memory\ Cells)} + X.V.f_{(wires\ in\ Output\ Crossbar)} + X_{(VC\ Control\ Logic)} \quad (1)$$

Though the power gating techniques can be applied to save the static power consumption for resources mentioned in Eq.1. But if these buffers are used by the NoC system, the fault will not impact the system performance significantly and routing intelligence can help to retain the performance. Consider the NoC platform shown in fig.8. For node '11' as source and node '13' as destination, the route of packet will be just to go up vertically in normal situation. If the communication link between nodes '11' and '12' is broken as shown in fig.8, the packet will be rerouted as shown. Similar is the situation, when node '10' is the source and node '30' is the destination. Now, the resources on the new routes and especially the router at node '21' will be overloaded. The input ports from left and down side will be overloaded due to the faulty links, while other inputs will operate with normal loads in typical architectures. Now, to balance the load on input buffers and provide a relief to the loaded ports, if the right and down input ports share the VC buffers and left and upper input ports as well, the extra load due to faulty links will be shared by all the ports. PVS-NoC architecture uses this sharing approach to retain the performance in case of fault on any communication resource.

To retain the performance after the fault occurrence, the problem needs to be approached from a different way. Fault on physical link also makes the buffer and controller useless for the system. In case of PVS-NoC architecture, all the VC buffers and controllers can be used by the neighboring port. This reduces the impact of fault on system performance. In case of fault on buffer, the load on corresponding port can be shared by neighboring port and impact of faulty buffer on port load becomes exactly half for the PVS-NoC architecture and congestion can be avoided. In case of fault in control logic, none of available architectures can utilize the the rest of resources related to the fault link.

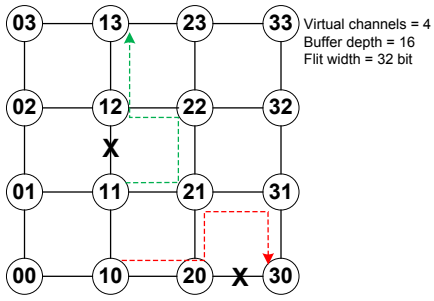


Fig. 8. Different routes between two source-destination pairs in presence of faulty links.

V. EXPERIMENTAL RESULTS

To demonstrate the better performance characteristic of the proposed architecture (PVS-NoC), a cycle-accurate NoC simulation environment was implemented in HDL. The packets had a fix length of seven flits, the buffer size was eight flits and the data width was set to 32 bits. The 5x5 2D mesh topology was used for interconnection. With same parameters, typical VC NoC and FVS-NoC architectures were analyzed. Static XY routing was used. The proposed architecture was analyzed for synthetic and real application traffic patterns.

A. Synthetic traffic

In synthetic traffic analysis, the performance of the network was evaluated using latency curves as a function of the packet injection rate. The packet latency was defined as the time duration from when the first flit is created at the source node to when the last flit is delivered to the destination node. For each simulation, the packet latencies were averaged over 50,000 packets. Latencies were not collected for the first 5,000 cycles to allow the network to stabilize. To perform the simulations, XY wormhole routing algorithm [9] under uniform, transpose and Negative Exponential Distribution (NED) [10] traffic patterns was used.

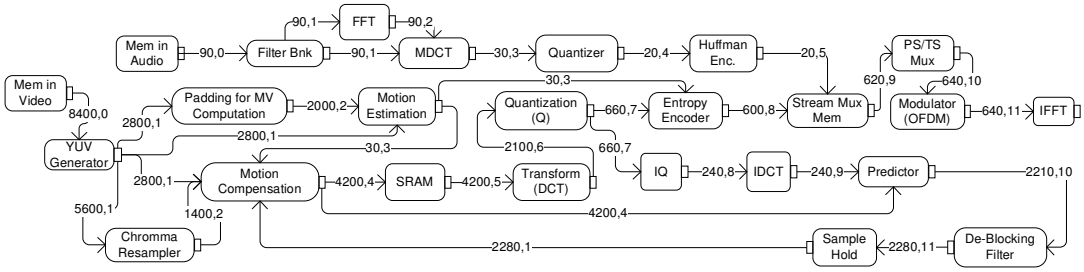
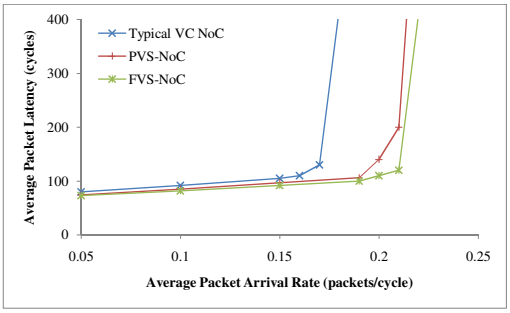
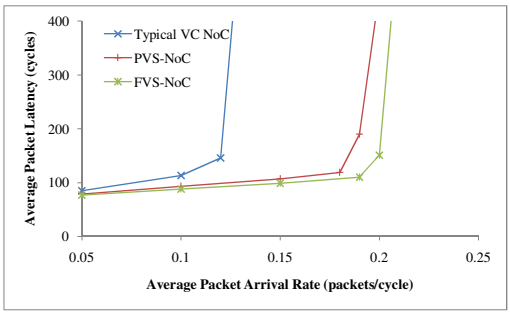


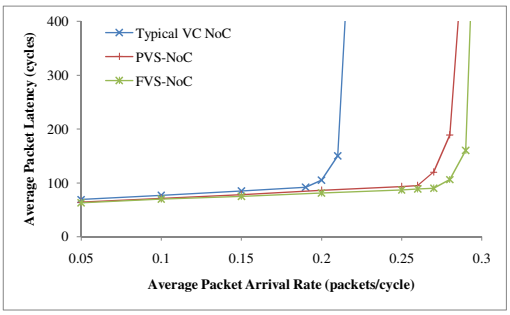
Fig. 10. Video conference encoder application



(a) Uniform traffic load.



(b) Transpose traffic load.



(c) NED traffic load.

Fig. 9. Average packet latency (APL) vs. Packet injection rate for 5×5 Mesh 2D NoC.

The throughput curves for uniform, transpose and NED traffic patterns are shown in fig.9. It can be observed for all the traffic patterns, the PVS-NoC architecture saturates at higher injection rates as compared to the typical VC architecture but slightly less than FVS-NoC architecture. The reason is that partial VC sharing makes the load balanced. In case of proposed architecture, bandwidth limitations are managed by proper resource utilization without increasing the communication resources. The saturation point of PVS-NoC is just before FVS-NoC because FVS-NoC provides more buffer utilization by sharing the VC buffer among all input ports. But FVS-NoC is not a power efficient solution as verified for realistic application traffic pattern.

B. Real Benchmark

For real benchmark analysis, the encoding part of video conference application with sub-applications of H.264 encoder, MP3 encoder and OFDM transmitter was used. The application graph with 25 nodes is shown in fig.10.

The application graph consists of processes and data flows; data is, however, organized in packets. Processes transform input data packets into output ones, whereas packet flows carry data from one process to another. A transaction represents the sending of one data packet by one source process to another, target process, or towards the system output. A packet flow is a tuple of two values (P, T). The first value 'P' represents the number of successive, same size transactions emitted by the same source, towards the same destination. The second value 'T' is a relative ordering number among the (packet) flows in one given system. For simulation purposes, all possible software procedures are already mapped within the hardware devices. The application mapped to 5x5 2D-mesh NoC is shown in fig.11.

To estimate the power consumption, we adapted the high level NoC power simulator presented provided by [9]. The power consumption of the interconnection network (NoC switches, bus arbiters, intermediate buffers, and interconnects) is based on 35nm standard CMOS technology. The simulation results for average packet latency (APL) and power consumption for video conference encoding application are shown in Table.II. The PVS-NoC showed 18% reduction in power consumption but 6% more APL over the FVS-NoC architecture. On other hand, the PVS-NoC showed 22.32% reduction in APL value but 7.9% more power consumption over the typical VC architecture. Thus, the proposed architecture PVS-NoC provides an optimal tradeoff between APL and power consumption.

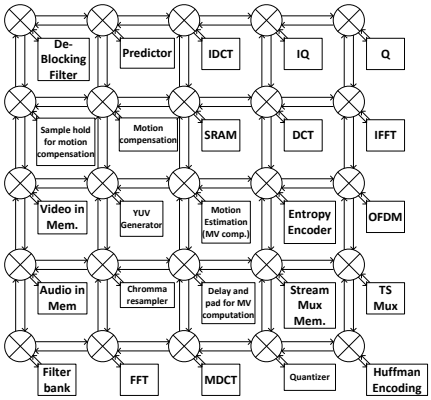


Fig. 11. Video conference encoder application mapped to 5x5 Mesh NoC

Architecture	Power Consumption (mW)	APL (cycles)
Typical VC	66.4	112
PVS-NoC	72.1	87
FVS-NoC	87.9	82

TABLE II
EXPERIMENTAL RESULT FOR POWER CONSUMPTION AND APL OF VIDEO CONFERENCE ENCODER APPLICATION RUNNING AT 50MHZ.

VI. CONCLUSIONS

PVS-NoC architecture with an ideal tradeoff between VC utilization, system performance and power consumption was proposed. The VC buffers are shared between two input ports in PVS-NoC. Apart from system performance, the architecture is also fault tolerant. In case of any link failure, the VC buffers are used by the other physical channel sharing the VC buffers and system performance is not affected severely.

The proposed architecture was simulated with synthetic and real application traffic patterns. The performance was compared with typical VC based NoC architecture and FVS-NoC. The PVS-NoC architecture showed significant improvement in system throughput without significant power consumption overhead.

ACKNOWLEDGEMENTS

This work is supported by the Nokia Foundation and DOMES project (123518/2008) funded by the Academy of Finland.

REFERENCES

[1] Massoud Pedram. Amir-Mohammad Rahmani, Ali Afzali-Kusha. Ned: A novel synthetic traffic pattern for power/performance analysis of network-on-chips using negative exponential distribution. In *Journal of Low Power Electronics (American Scientific Publishers)*, volume 5, pages 396–405, 2009.

[2] James Balfour and William J. Dally. Design tradeoffs for tiled cmp on-chip networks. In *Proceedings of the 20th annual international conference on Supercomputing (ICS)*, pages 187–198, 2006.

[3] N. Banerjee, P. Vellanki, and K.S. Chatha. A power and performance model for network-on-chip architectures. In *Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings*, volume 2, pages 1250–1255, 2004.

[4] Xuning Chen and Li-Shiuan Peh. Leakage power modeling and optimization in interconnection networks. In *Low Power Electronics and Design, 2003. ISLPED '03. Proceedings of the 2003 International Symposium on*, pages 90–95, 2003.

[5] M. Coenen, S. Murali, A. Radulescu, K. Goossens, and G. De Micheli. A buffer-sizing algorithm for networks on chip using tdma and credit-based end-to-end flow control. In *Hardware/Software Codesign and System Synthesis, 2006. CODES+ISSS '06. Proceedings of the 4th International Conference*, pages 130 –135, 2006.

[6] T. Dumitras and R. Marculescu. On-chip stochastic communication [soc applications]. In *Design, Automation and Test in Europe Conference and Exhibition, 2003*, pages 790 – 795, 2003.

[7] Donghyun Kim et al. A reconfigurable crossbar switch with adaptive bandwidth control for networks-on-chip. In *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pages 2369 – 2372 Vol. 3, 2005.

[8] D. Fick, A. DeOrio, G. Chen, V. Bertacco, D. Sylvester, and D. Blaauw. A highly resilient routing algorithm for fault-tolerant nocs. In *Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09.*, pages 21 –26, 20-24 2009.

[9] G. Guindani, C. Reinbrecht, T. Raupp, N. Calazans, and F.G. Moraes. Noc power estimation at the rtl abstraction level. In *Symposium on VLSI, 2008. ISVLSI '08. IEEE Computer Society Annual*, pages 475 –478, 2008.

[10] A. Jantsch and H. Tenhunen (Eds.). *Networks on Chip*. Kluwer Academic Publishers, 2003.

[11] Ying-Cherng Lan, Shih-Hsin Lo, Yueh-Chi Lin, Yu-Hen Hu, and Sao-Jie Chen. Binoc: A bidirectional noc architecture with dynamic self-reconfigurable channel. In *Networks-on-Chip, 2009. NoCS 2009. 3rd ACM/IEEE International Symposium on*, pages 266 –275, 2009.

[12] P. Lotfi-Kamran, A. M. Rahmani, M. Daneshtalab, A. Afzali-Kusha, and Z. Navabi. Edxy - a low cost congestion-aware routing algorithm for network-on-chips. *J. Syst. Archit.*, 56(7):256–264, 2010.

[13] Giovanni De Micheli Luca Benini. *Networks On Chips: Technology And Tools*. Morgan Kaufmann Publishers, aug 2006.

[14] R. Mullins, A. West, and S. Moore. Low-latency virtual-channel routers for on-chip networks. In *Computer Architecture, 2004. Proceedings. 31st Annual International Symposium on*, pages 188–197, 2004.

[15] Mohammad Hossein Neishaburi and Zeljko Zilic. Reliability aware noc router architecture using input channel buffer sharing. In *ACM Great Lakes Symposium on VLSI*, pages 511–516, 2009.

[16] C.A. et al. Nicopoulos. Vichar: A dynamic virtual channel regulator for network-on-chip routers. In *Microarchitecture, 2006. MICRO-39. 39th Annual IEEE/ACM International Symposium on*, pages 333 –346, 2006.

[17] V. Soteriou, R.S. Ramanujam, B. Lin, and Li-Shiuan Peh. A high-throughput distributed shared-buffer noc router. *Computer Architecture Letters*, 8(1):21 –24, jan. 2009.

[18] Hangsheng Wang, Li-Shiuan Peh, and S. Malik. Power-driven design of router microarchitectures in on-chip networks. In *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*, pages 105–116, 2003.

[19] Brian Patrick Towles William J Dally. *Principles and Practices of Interconnection Networks*. The Morgan Kaufmann Series in Computer Architecture and Design, 2004.

[20] T.T. Ye, L. Benini, and G. De Micheli. Analysis of power consumption on switch fabrics in network routers. In *Design Automation Conference (DAC), 2002. Proceedings. 39th*, pages 524–529, 2002.

[21] Zhen Zhang, A. Greiner, and S. Taktak. A reconfigurable routing algorithm for a fault-tolerant 2d-mesh network-on-chip. In *Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE*, pages 441 –446, 2008.