Virtual Channels in Networks on Chip:
Implementation and Evaluation on Hermes NoC

Aline Mello
alinev@inf.pucrs.br

Leonel Tedesco
ltedesco@inf.pucrs.br

Ney Calazans
calazans@inf.pucrs.br

Fernando Moraes
moraes@inf.pucrs.br

Pontifícia Universidade Católica do Rio Grande do Sul (FACIN-PUCRS)
Av. Ipiranga, 6681 - Prédio 30 / Bloco C - 90619-900 - Porto Alegre – RS – BRASIL

ABSTRACT

Networks on chip (NoCs) draw on concepts inherited from distributed systems and computer networks subject areas to interconnect IP cores in a structured and scalable way. Congestion in NoCs reduces the overall system performance. This effect is particularly strong in networks where a single buffer is associated with each input channel, which simplifies router design, but prevents packets from sharing a physical channel at any given instant of time. The goal of this work is to describe the implementation of a mechanism to reduce performance penalization due to packet concurrence for network resources in NoCs. One way to reduce congestion is to multiplex a physical channel using virtual channels (VCs). VCs reduce latency and increase network throughput. The insertion of VCs also enables to implement policies for allocating the physical channel bandwidth, which enables to support quality of service (QoS) in applications. This paper has two main contributions. The first is the detailed implementation of a NoC router with a parameterizable number of VCs. The second is the evaluation of latency and throughput in reasonably sized instances of the Hermes NoC (8x8 mesh), with and without VCs. Additionally, the paper compares the features of the proposed router with others employing VCs. Results show that NoCs with VCs accept higher injections rates w.r.t. NoCs without VCs, with a small standard deviation in the latency values, guaranteeing precise packet latency estimation.

Categories and Subject Descriptors

B.4.3 [**Input/Output and Data Communications**]: Interconnections (Subsystems) – asynchronous/synchronous operation, fiber optics, interfaces, parallel I/O, physical structures (e.g., backplanes, cables, chip carriers), topology (e.g., bus, point-to-point).

General Terms

Design, Experimentation, Measurement, Performance.

Keywords

Network-on-chip, virtual channel, performance.

1. INTRODUCTION

Increasing transistor density, higher operating frequencies, short time-to-market and reduced product life cycle characterize today's

semiconductor industry scenery [1]. Under these conditions, designers are developing ICs integrating complex heterogeneous functional elements into a single chip, known as a System on a Chip (SoC). Intellectual property cores, interconnection architectures and interfaces to peripheral devices [2] compose a SoC.

Usually, the interconnection architecture employs dedicated wires or shared busses. Dedicated wires are effective for systems with a small number of cores, but the number of wires around the core increases as the system complexity grows. Therefore, dedicated wires have poor reusability and flexibility. A shared bus is more scalable and reusable when compared to dedicated wires. However, busses allow only one communication transaction at a time, all cores share the same communication bandwidth in the system and scalability is limited to a few dozens of cores [3]. Using separate busses interconnected by bridges or hierarchical bus architectures may reduce some of these constraints, since different busses may account for different bandwidth needs, protocols and increase communication parallelism. Nonetheless, scalability remains a problem for hierarchical bus architectures.

In this context, applying concepts from computer and telecom networks to embedded systems, a new interconnection structure, named Network on Chip (NoC) is emerging [3][4][5]. NoCs [6] can replace busses due to the following features: (i) energy efficiency and reliability [1]; (ii) scalability of bandwidth when compared to traditional bus architectures; (iii) reusability; (iv) distributed routing decisions [4][5].

The throughput of interconnection networks is limited to a fraction (typically 20% to 50%) of the network capacity due to coupled allocation of resources [7]. Two main resources compose interconnection networks: buffers and channels. Typically, a single buffer is associated with each channel. However, a channel can be multiplexed in n virtual channels (VC). VCs provide multiple buffers for each channel, increasing the resources allocation for each packet. The insertion of VCs also enables the use of special policies to allocate the physical channel bandwidth, allowing support to quality of service (QoS).

The goal of this work is to describe the implementation and evaluation of VCs in NoCs. Implementation of routers with VCs is complex due to the degree of freedom to choose schemes for buffering, internal interconnections, arbitration and routing. Silicon area constrains the complexity of these schemes, once NoC primary function is communication, and not processing. The evaluation of latency and throughput of NoCs with VCs enables designers to parameterize the network according to application requirements.

This paper is organized as follows. Section 2 presents an overview of the state of the art in NoCs using VCs. Section 3 details the main contribution of this work, the implementation of VC in the Hermes NoC. Section 4 evaluates latency and throughput in NoCs

with and without VCs. Section 5 presents conclusions and directions for future work.

2. RELATED WORKS

Several approaches have been proposed to organize the design, implementation and maintenance of reconfigurable systems. This Section reviews several relevant propositions in this theme.

This Section reviews the state of the art in NoCs using VCs and summarizes the analysis in Table 1, where each row corresponds to a NoC or NoC router.

The second column of Table 1 presents the network topology. Most networks use 2D mesh, 2D torus or folded 2D torus. The reason for this choice derives from the facilitated implementation, simplicity of the routing algorithm and network scalability. Torus topology is an option to reduce the network diameter [8], then reducing latency. The folded 2D torus [9] is an option to reduce the increased cost in wiring when compared to a standard 2D torus.

The second parameter is the communication channel width. The network proposed by Dally [9] proposes wide communication channels (~300 bits). This network targets future SoCs. The remaining networks have flit sizes between 8 and 64 bits, a data width similar to current processor architectures.

The next parameter in Table 1 is the routing strategy. All networks employ wormhole routing. The Æthereal NoC [10] combines the wormhole routing (used by best-effort traffic) with circuit switching (used by guaranteed throughput traffic). Kavaldjiev [11], Dally and Æthereal use source routing, simplifying router architecture, once it need not take decisions dynamically.

The fifth column of Table 1 presents the router buffering strategy. Kavaldjiev and QNoC [12] routers adopt input queuing, which reduces the area overhead, but are susceptible to the head-of-line (HOL) blocking problem. Marescaux [8], Xpipes [13] and Æthereal routers use output queuing, with an increased buffer area overhead but no HOL blocking problem. Dally and MediaWorm [14] routers employ an intermediate solution, input and output queuing, combining the advantages of both, input and output queuing strategies [15].

The sixth column details the crossbar structure used by routers. The full crossbar has the number of input and output ports equals to the total number of VCs (m) multiplied by the number of router ports (n). The multiplexed crossbar has the number of input/output ports equals to n. A multiplexed crossbar contains a multiplexer at each crossbar input port and a demultiplexer at each crossbar output port. Full crossbar favors performance, but increases the implementation cost. Multiplexed crossbar has smaller area but requires arbiters in input and output ports. According to [14], multiplexed crossbars are applicable for high number of VCs.

The seventh column of Table 1 presents the flow control used in each one of the networks. QNoC and Dally networks adopt credit based flow control. In credit based flow control, routers keep counters for the available space in buffers. When receiving a flit, the counter is decremented, and when transmitting a flit the counter is incremented. If the counter reaches zero, the buffer is full and the router accepts no further flits until buffer space becomes available again. The router transmitting flits (upstream router) receives the credit information of the target node (downstream router) to send data, if credit is available [16]. Marescaux and Xpipes networks adopt the handshake flow control. In the handshake flow control, the upstream router sends flits whenever they become available. If the downstream router has available buffer space, it accepts the flit and sends an acknowledgment (ack) to the upstream router. Otherwise, the downstream router drops the flit and sends a negative acknowledgement (nack). According to Dally [16], the Ack/Nack flow control is less efficient than the credit based flow control, because the flits remain stored in the buffers for an additional time, waiting for an acknowledgment.

The last three columns of Table 1 present VCs features: (i) the number of VCs; (ii) VCs selection strategy (TDM, priority or virtual clock); (ii) support to QoS.

Kavaldjiev and Marescaux routers employ time division multiplexing (TDM). Kavaldjiev router has four time-multiplexed VCs. Each VC, if it has data to transmit, receives at least ¼ of the physical channel bandwidth. This router provides QoS by sending a packet with infinite size, allocating a given VC to a dedicated connection, as in the circuit switching approach. The disadvantage of circuit switching is the reservation of a physical path, even if there is no data to transmit, reducing the overall network performance. Marescaux router has two time-multiplexed VCs.

Table 1 – State of the art in networks/routers that use VCs.

network/ router	Topology	Communication Channel	Routing Strategy	Buffering	Crossbar	Flow Control	VC	VC Selection	QoS Support
Kavaldjiev	2D Mesh	NA	Wormhole Source	Input queue	Full	NA	4	TDM	Yes
QNoC	2D Mesh regular or irregular	16 data bits (parameterizable) + 10 control bits	Wormhole XY	Input queue	Full	Credit based	4	Priority and availability in buffer	Yes
Dally	2D Folded Torus	256 data bits + 38 control bits	Wormhole XY Source	Input queue + 1 output position	Multiplexer	Credit based	8	NA	Yes
Marescaux	2D Torus	16 data bits + 3 control bits	Wormhole XY	2 output positions	Multiplexer	Handshake	2	TDM	Yes
Xpipes	Arbitrary (design time)	32, 64 or 128 bits	Wormhole Street sign	Output queue	Multiplexer	Handshake	Parameterizable	Priority	No
Æthereal	2D Mesh	32 bits	Circuit Switching (GT) e Wormhole Source (BE)	Output queue	NA	NA	3	Priority	Yes
MediaWorm	not applicable	NA	Wormhole	Input and output queue	Multiplexer	NA	2	Virtual Clock	Yes
Hermes NoC	2D Mesh	16 data bits + (parameterizable) 6 control bits	Wormhole XY / partially adaptive	Input queue	Full	Credit based	2 - 4	TDM adaptive	No

NA – Data not available

QNoC, Xpipes and Æthereal networks use a priority mechanism to define which VC has permission to access the physical channel. QNoC has four VCs, each dedicated to a specific service class. Each service class has a priority defined by its communication requirements. QNoC sends a flit when there is available buffer space and no packet with higher priority is waiting. Once a higher priority packet arrives, the router preempts the current packet transmission. Transmission of preempted packets resume after serving all higher priority packets. Æthereal has three VCs primarily used for GT traffic. Between GT traffics, a long-lived scheduling is used. The BE traffic employs bandwidth not reserved to GT traffic or idle. The use of dedicated network resources for each traffic class simplifies implementation, but increases the overall cost.

Virtual Clock [14] is the third method used to select VCs. Virtual Clock regulates the bandwidth usage of each message. It assumes the existence of a global time reference and two values associated to each message, AuxVC and Vtick. AuxVC is initialized to the global time value at the time the message enters the network and is incremented by the value of Vtick each time a message packet is transmitted. Vtick is computed as the inverse of the negotiated throughput of the message. Small Vtick means that packets must be sent as soon as possible, while higher values mean that packets may wait some period before being transmitted. BE traffic assigns an infinite Vtick to messages.

The last row of the Table presents the features of the proposed architecture. The designer can parameterize the flit size, buffer depth, network size and number of VCs. The Hermes router has n time-multiplexed VCs. Each VC, if it has flits to transmit and credit for transmission, uses at least $1/n$ of the physical channel bandwidth. The same packet can completely occupy the bandwidth, if the remaining VCs do not have data to transmit. This adaptive TDM (Available Bit Rate strategy) optimizes channel usage. Currently, Hermes NoC does not have support to QoS.

The main motivation of this work is to perform a detailed analysis of the VC impact on performance under different traffic patterns and loads. This analysis will guide future work on QoS implementation in the Hermes NoC.

3. HERMES NOC ARCHITECTURE

The Hermes NoC [17] is an infrastructure used to implement low area overhead packet switching NoCs, using mesh topology. This infrastructure was extended to implement virtual channels.

The NoC transmits packets divided into flits. The flit size may be parameterized, and the maximum number of payload flits in a given packet is 2(flit size, in bits). The first and the second flit of a packet are header information, being respectively the address of the target router, named *header flit*, and the number of flits in the packet payload.

Hermes NoC implements either handshake or credit based flow control strategies. The VC implementation employs credit based flow control, due to the advantages over handshake, as mentioned before. Figure 1 illustrates the interface between routers. The following signals compose the output port: (1) *clock_tx*: synchronizes data transmission; (2) *tx*: indicates data availability; (3) *lane_tx*: indicates the VC (lane) transmitting data; (4) *data_out*: data to be sent; (5) *credit_i*: indicates available buffer space, for each lane.
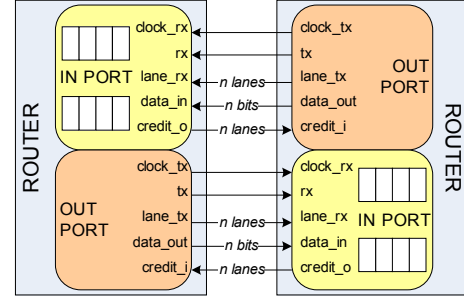

Figure 1 – Hermes NoC physical router interfaces.

The router has one centralized switching control logic and five bi-directional ports: East, West, North, South, and Local. The Local port establishes a communication between the router and its local core. The other ports of the router are connected to neighbor routers. Each port contains two unidirectional physical channels. A physical channel may support multiplexed VCs (lanes). Figure 2 presents a Hermes router with two lanes per physical channel. The local port of the router is not multiplexed because it is assumed that the core connected to this port is not able to receive or send more than one packet simultaneously. Although multiplexing of physical channels may increase the switching performance [6], it is important to keep a compromise between performance, complexity and router area.
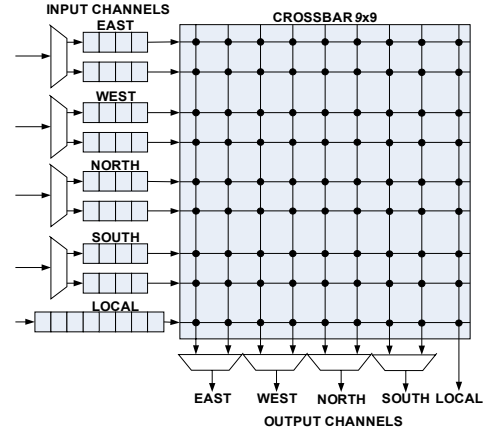

Figure 2 – Hermes router with two VCs. The local port is not multiplexed.

Each input port has a buffer for temporary flit storage, with a depth d. When *n* lanes are used, a buffer with d/n depth is associated to each lane. The input port receives flits, storing them in the buffer indicated by signal *lane_rx* (Figure 1), decrementing the number of lane credits. When an output port transmits a flit, this flit is removed from the buffer and the number of credits is incremented. Credit availability reaches a neighbor router through signal *credit_o* (Figure 1).

Multiple packets may arrive simultaneously in a given router. Centralized round-robin arbitration grants access to incoming packets. The priority of a lane is a function of the last lane having a routing request granted.

If the incoming packet request is granted by the arbiter, a routing algorithm is executed to connect the input port to the correct output port. Four routing algorithms may be used: one deterministic (dimension-ordered or XY) and three partially adaptive (West

First, North Last and Negative First) [17]. When the algorithm returns a busy output port, the header flit as well as all subsequent flits of this packet will be blocked.

When the routing algorithm finds a free output port, the connection between the input lane and the output lane is established and the switching table is updated. Three vectors compose the switching table: in, out and free. The in vector connects an input lane to an output lane. The out vector connects an output lane to an input lane. The free vector is responsible for modifying the output lane state from free (1) to busy (0). The in and out vectors receive an identifier constructed by combining the port number (np) and the lane number (nl), as presented in Equation 1.

$$id = \left(np \times number\ of\ lanes\right) + nl \qquad (1)$$

The East, West, North, South and Local ports are numbered from 0 to 4, respectively. The L1, L2 ... Ln lanes are numbered from 0 to n-1, respectively. Consider a router with two lanes per physical channel. Lane L1 from the North port has the identifier 4 ($(2 \times 2) + 0)$) and lane L2 has the identifier 5 ($(2 \times 2) + 1)$). The switching table presented in Figure 3(b) represents the switching illustrated in Figure 3(a). Consider the North port. The output lane L1 is busy (free=0) and is being driven by the input lane L1 of the West port (out=2). The input lane L1 is driving the output L1 lane of the South port (in=6). The switching table contains redundant information, but this organization is useful to enhance the routing algorithm efficiency.
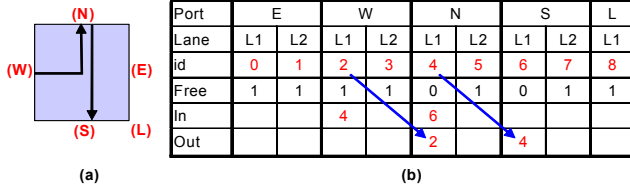

Figure 3 – a) switching; b) switching table.

After routing, the output port is responsible to allocate the bandwidth between the n lanes. Each lane, if it has flits to transmit and credit for transmission, uses at least $\frac{1}{n}$ of the physical channel bandwidth. If a single lane satisfies this condition, it uses all physical channel bandwidth.

The connection is closed after transmitting all flits composing the packet. The router has one flit counter for each output lane. The counter of a specific lane is initialized when the second flit of a packet arrives, indicating the number of flits of the payload. The counter is decremented for each subsequent flit. When the counter value reaches zero, the connection is closed and the free vector corresponding to the position of the output lane goes to one (free=1).

Equation 2 gives the minimal latency to transfer a packet from source to target, in clock cycles.

$$minimal\ latency = \left(\sum_{i=1}^{n} R_i\right) + P \qquad (2)$$

In this Equation:

- n is the number of routers in the communication path (source and target included);
- R_i is the router latency (arbitration and routing), at least 6 clock cycles in this implementation;
- P is the packet size.

The theoretical peak throughput of each Hermes router is 1Gbits/s, for a router with 5 input ports, 8-bit flits, and 25MHz clock $\left(5\ ports \times 8\ bits \times 25MHz\right)$.

4. EVALUATION

The network behavior is a function of its architecture and the application running on it. For example, some applications mostly consists in long messages (e.g. streaming), while in others short messages dominate (e.g. controllers). According to [19], the influence of traffic in system performance is greater than that of network design parameters. Thus, it is important to have available traffic generation techniques able to reflect real traffic behaviors.

The most important performance metrics of an interconnection network are latency and throughput [19]. Latency is the time elapsed from the moment a packet is created until the moment where it is received at the destination node, including the queuing time at the source node. Throughput is the maximum traffic accepted by the network, where traffic accepted is the amount of information delivered per time unit.

4.1 Experimental Setup

The network design parameters are: 8x8 routers mesh; XY routing; 16-bit flit size; 8-flit buffers. Smaller network sizes do not favor VCs, except when traffic generation creates a situation where VCs are clearly benefit, e.g., several flows using the same channel. The number of VCs varies from 1 (no VC) to 4.

Three parameters define the workload model: spatial distribution of targets, injection rate, and packet length. The adopted spatial distribution is the complement traffic pattern [19], which concentrates the flows in the network bisection. The random inserted traffic load varies between 10% and 100% of the channel capacity. Performance is evaluated for short (100 flits) and long packets (1,000 flits). In both scenarios, each router transmits 10,000 flits.

4.2 Results

This Section analyses the following performance figures: throughput, latency, bandwidth utilization per link (2 channels, one in each direction) and latency variance.

Figure 4(a) shows the behavior of the accepted traffic with 1, 2 and 4 VCs w.r.t. the inserted traffic. For small injection rates, the accepted traffic follows the inserted traffic. Around 20% the accepted traffic becomes constant, representing a throughput saturation point. Independently of how much traffic is offered to the network after the saturation point, it is not possible to obtain a throughput greater than 20% of network capacity (without VCs). The insertion of VCs increases the maximum accepted traffic. For 2 and 4 VCs the maximum accepted traffic 23% and 25%, respectively. When 2 VCs are used, packets can bypass blocked packets, increasing channel utilization and throughput. Adding more VCs, e.g. 4, no significant increase in routing flexibility is observed, and the buffer size per VC is reduced, explaining the small advantage of 4 VCs over 2 VCs.

Figure 4(b) shows the average latency behavior w.r.t. the inserted traffic. Up to the saturation point, average latency is constant. The network saturation value increases with the number of VCs, as mentioned before, and the average latency is smaller. The average latency can be 20 times smaller when the number of VCs goes from 1 to 2. Only small advantages are observed for 4 VCs, for the same reasons mentioned before.
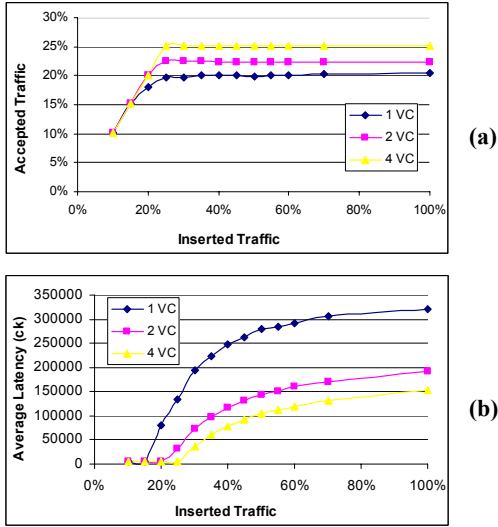
Figure 4 – (a) Accepted traffic and (b) average latency in relation to inserted traffic (100 long packets per router).

Figure 5 groups links in classes based on the amount of bandwidth utilization. It is apparent from the plot how VCs contribute to increase the bandwidth utilization of links. When using VCs, no link presents bandwidth utilization below 40%, while this happens for 20% of the links if VCs are not used.
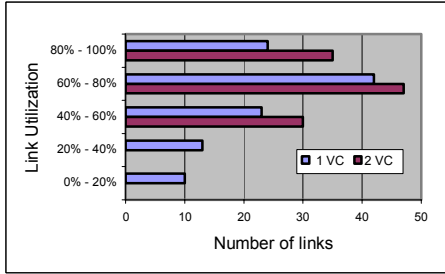

Figure 5 – Cardinality of link classes based on link bandwidth utilization for 1 VC and 2 VC (100 long packets per router).

Figure 6 plots the number of packets with a given latency value, for 1 and 2 VCs, for 25% and 50% inserted traffics. The goal is to show not only the distribution of packet latency values, but also the spreading of these values.

In Figure 6(a), using a 25% inserted traffic, the average latency is

32,567 and 132,534 clock cycles for 1 and 2 VCs, respectively. Reduction of the average latency and smaller spreading still occurs in heavily saturated networks, as depicted in Figure 6(b). The smaller spreading allows estimating the time to transmit packets more precisely, a key factor to implement QoS in NoCs.
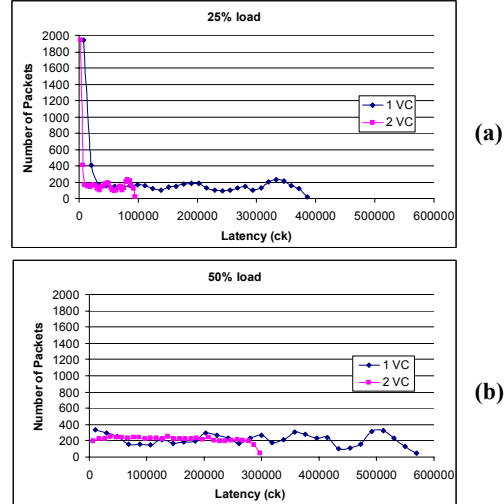
Figure 6 – Number of packets with a given latency for (a) 25% load and (b) 50% load (100 long packets per router).

Table 2 and Table 3 detail the latency and throughput values for long (1,000 flits) and short packets (100 flits), respectively. The last two columns of the "average throughput" row indicate the experimental network saturation point. Without using VCs (1VC) the network saturates with injection rates equal to 20% (long packets) and 17.5% (short packets) of the maximum channel bandwidth. For 2 VCs the network saturates with injection rates equal to 23% (long packets) and 21.5% (short packets).

Beyond the saturation point, the latency values (2nd to 5th rows) grows dramatically. Consider an injection rate of 20% (20% load). In this situation, while the 1 VC implementation is saturated, the 2 VCs is not. Comparing 1 VC to 2 VCs for long packets (Table 2), the average latency is bigger by a factor of 20, the maximum latency by a factor of 40 and the latency standard deviation by a factor of 71. For short packets, the difference is even more remarkable: 220 times in average latency, 209 times in maximum latency and 685 times in standard deviation. The performance of short packets is penalized in the absence of VC, since short packets execute arbitration and routing more frequently,

Table 2 – Simulation results of 100 packets with 1,000 flits each.

Complement Traffic Pattern	10% Load		15% Load		20% Load		25% Load		30% Load	
	1VC	2VC	1VC	2VC	1VC	2VC	1VC	2VC	1VC	2VC
average latency (ck)	4,094	4,059	4,094	4,059	80,708	3,997	132,534	32,567	194,119	72,901
minimal latency (ck)	1,021	2,009	1,021	2,009	1,018	1,621	1,022	1,128	1,022	1,770
maximum latency (ck)	7,178	6,110	7,591	6,110	346,371	9,096	391,903	96,578	463,498	175,727
latency standard deviation (ck)	1,621	1,429	1,620	1,429	101,441	1,431	127,081	26,922	147,172	46,098
Average throughput	10.09%	10.09%	15.13%	15.13%	18.02%	20.17%	19.75%	22.55%	19.68%	22.44%

Table 3 – Simulation results of 1,000 packets with 100 flits each.

Complement Traffic Pattern	10% Load		15% Load		20% Load		25% Load		30% Load	
	1VC	2VC	1VC	2VC	1VC	2VC	1VC	2VC	1VC	2VC
average latency (ck)	494	459	28237	459	101,375	459	197,936	45,826	246,038	81,789
minimal latency (ck)	121	209	119	209	117	133	118	207	118	210
maximum latency (ck)	878	710	238,841	710	376,099	1,798	501,402	124,839	553,637	189,887
latency standard deviation (ck)	198	157	54,453	157	118,603	173	162,203	36,923	172,198	53,186
Average throughput	10.01%	10.01%	14.62%	15.01%	13.06%	20.02%	17.36%	21.38%	17.45%	22.44%

consuming time. Also, packets may stay blocked for long periods in routers.

4.3 Area Results

The Leonardo synthesis tool generated synthesis data for a smaller, 4x4 NoC. Table 4 presents the area data produced by the synthesis, showing the FPGA resource use. The synthesis of the Hermes NoC with 1 VC takes about 17% of the resources of a XC2V6000 FPGA (i.e. 1,021,800 equivalent gates), the synthesis of the Hermes NoC with 2 VCs takes 32.61% (i.e. 1,956,600 equivalent gates) and the synthesis of the Hermes NoC with 4 VCs takes 75.41% (i.e. 4,524,600 equivalent gates).

It would be expected to have a small difference in area when using virtual channels, since the buffer area is the same. The area increase is due to the Virtex FPGA mapping, which uses 1-bit LUT RAM to implement arrays (i.e. buffers). As the LUT size is equal to 16, buffer capacity from 1 to 16 consumes the same area. Consequently, when 2 VCs are used (two buffers are used per input port), it is expected to double the used area.

Table 4 – 4x4 Hermes NoC area results for 2V6000 FPGA.

Resources	Mapping to Xilinx XC2V6000 FPGA device						
	Used			Available	Used /Available		
	1 VC	2 VCs	4 VCs		1 VC	2 VCs	4 VCs
Slices	5,756	11,018	25,481	33,792	17.03%	32.61%	75.41%
LUTs	11,511	22,036	50,962	67,584	17.03%	32,61%	75.41%
Flip Flops	2,808	5,232	9,504	70,056	4.01%	7.47%	13.57%

Even if the Noc described here is not yet prototyped in hardware, the synthesis data are coherent with extrapolations made from the non-VC, 8-bit flit prototyped Hermes NoC described in [17].

5. CONCLUSIONS AND FUTURE WORK

This work presented the implementation of the virtual channel concept for the Hermes NoC. The impact of VCs on the average latency depends on the network dimension. Small dimension networks with VCs present little latency reduction, while large dimension networks as the 8x8 described here can provide more than 50% reductions in average latency. Also, VCs reduce the spreading of latency values, providing a more predictable underlying structure on top of which to build NoCs that can insure QoS. The clear disadvantage brought about by VCs is the almost linear area increase with the number of VCs per link.

An ongoing work aiming at the prototyping of the VC Hermes NoC has as objective the reduction of area increase. Another work related to area consumption is to investigate the use of heterogeneous networks. In this type of network, there can be routers with or without VCs, with different buffer depths and different flit sizes, for example.

Currently, the Hermes NoC supports only best effort service (BE) [6]. This service offers no QoS guarantee to applications, all applications are treated equally and packets can experiment arbitrarily long delays. For applications with performance requirements, it is necessary to provide guaranteed throughput service (GT) [16]. GT services are essential for applications that require controlled bandwidth, as in the case of video transmission. The insertion of VCs in the Hermes NoC enables future implementation of mechanisms to support Quality of Service (QoS).

6. ACKNOWLEDGMENTS

7. REFERENCES

[1] International Sematech. "International Technology Roadmap for Semiconductors – 2002" Update, 2002. Available at http://public.itrs.net.

[2] Martin, G.; Chang, H. "System on Chip Design". In: 9th International Symposium on Integrated Circuits, Devices & Systems, Tutorial 2, 2001.

[3] Kumar, S.; Jantsch, A.; Soininen, J. P.; Fonsell, M. "A Network on Chip Architecture and Design Methodology". In: Computer Society Annual Symposium on VLSI, 2002.

[4] Benini, L.; De Micheli, G. "Powering networks on chips: energy-efficient and reliable interconnect design for SoCs". In: 14th International Symposium on Systems Synthesis, 2001, pp. 33-38.

[5] Guerrier. P.; Greiner. A. "A generic architecture for on-chip packet-switched interconnections". In: Design Automation and Test in Europe (DATE'00), 2000, pp. 250-256.

[6] Rijpkema, E.; et al. "A Router Architecture for Networks on Silicon". In: PROGRESS'2001.

[7] Dally, W. J. "Virtual-Channel Flow Control". In: 17th International Symposium on Computer Architecture, 1990, pp. 60-68.

[8] Marescaux, T.; et al. "Interconnection Networks Enable Fine-Grain Dynamic Multi-tasking on FPGAs". In: 12th Conference on Field-Programmable Logic and Applications, 2002, pp. 795-805.

[9] Dally, W. J.; Towles, B. "Route Packets, Not Wires: On-chip Interconnection Networks". In: Design Automation Conference, 2001, pp. 684-689.

[10] Goossens, K.; et al. "Guaranteeing the Quality of Service in Networks on Chip". Nurmi, J.; Tenhunen, H.; Isoaho, J.; Jantsch, A., editors, Networks on Chip, Kluwer 2003, pp. 61-82.

[11] Kavaldjiev, N.; Smit, G.; Jansen, P. "Two Architectures for On-chip Virtual Channel Router". PROGRESS, 2004, pp. 96-102.

[12] Bolotin E. et al. "QNoC: QoS architecture and design process for network on chip". Journal of Systems Architecture, 50(2-3), Feb. 2004, pp. 105-128.

[13] Bertozzi, D.; Benini, L. "Xpipes: A Network-on-chip Architecture for Gigascale Systems-on-Chip". IEEE Circuits and Systems Magazine, 4(2), 2004, pp. 18-31.

[14] Yum, K. H.; Kim, E. J.; Das, C.R.; Yousif, M.; Duato, J. "Integrated Admission and Congestion Control for QoS Support in Clusters". In: IEEE International Conference on Cluster Computing, 2002, pp. 325-352.

[15] Chuang, S.-T.; Goel, A.; Mckeown, N.; Prabhakar, B. "Matching output queuing with a combined input output queued switch". IEEE Journal on Selected Areas in Communications, 17(6), 1999, pp.1030-1039.

[16] Dally, W. J.; Towles, B. "Principles and Practices of Interconnection Networks". San Francisco: Morgan Kaufmann, 2004, 550 p.

[17] Moraes, F.; et al. "Hermes: an Infrastructure for Low Area Overhead Packet-switching Networks on Chip". Integration the VLSI Journal, 38(1), Oct. 2004, pp. 69-93.

[18] Glass, C.; Ni, L. "The Turn Model for Adaptive Routing". Journal of the Association for Computing Machinery, 41(5), Sep. 1994, pp. 874-902.

[19] Duato, J.; Yalamanchili, S.; Ni, L. "Interconnection Networks". Elsevier Science, 2002, 600 p.