Virtual Channels vs. Multiple Physical Networks: a Comparative Analysis

Young Jin Yoon, Nicola Concer, Michele Petracca, Luca Carloni
Department of Computer Science, Columbia University
{youngjin, concer, petracca, luca}@cs.columbia.edu

ABSTRACT

Packet-switched networks-on-chip (NoC) have been proposed as an efficient communication infrastructure for multi-core architectures. Adding virtual channels to a NoC helps to avoid deadlock and optimize the bandwidth of the physical channels in exchange for a more complex design of the routers. Another, possibly alternative, approach is to build multiple parallel physical networks (multi-planes) with smaller channels and simpler router organizations. We present a comparative analysis of these two approaches based on analytical models and on a comprehensive set of experimental results including both synthesized hardware implementations and system-level simulations.

Categories and Subject Descriptors

C.1.2 [**Multiple Data Stream Architectures (Multiprocessors)**]: Interconnection architectures

General Terms

Design, Performance

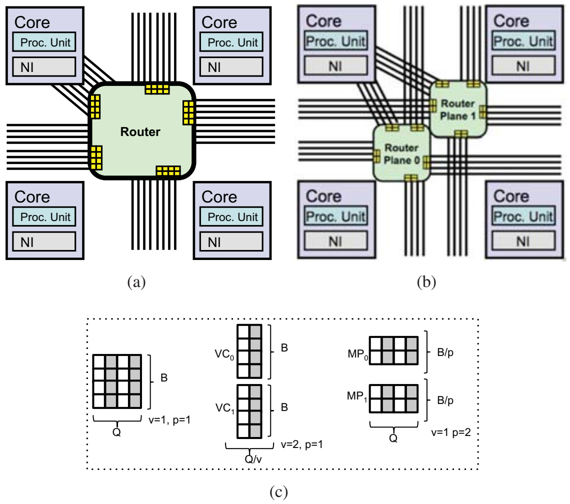
Keywords

Channel Slicing, Virtual Channel, Network-on-Chip.

1. INTRODUCTION

Packet-switched networks-on-chip (NoC) have been proposed as an alternative solution to standard bus-based interconnects to address the global communication demands of future chip-multi-processors (CMP) and system-on-chip (SoC). While these communication demands continue to grow as more cores are integrated on a chip, the on-chip power-dissipation budget is expected to remain very limited due to packaging constraints. Hence, the challenge is not only to design NoCs that can deliver high-bandwidth at low latency for inter-core communication, but also to make sure that this is done in a very power-efficient way [1].

Virtual channels (VC) have been proposed as a buffer-management flow control that extends *worm-hole* flow control (WH) by associating more than one logical channel to each physical I/O port of the router [2]. This is obtained through the partitioning of the storage resources to enable selectively buffering of the incoming worms so that they do not interfere during the forwarding process. VC can be used to avoid routing deadlock, improve performance under congestion [3], and separate different classes of traffic to avoid protocol deadlock [4]. Both WH and VC flow controls are appealing for NoC design because they require less buffering space in the routers. Indeed, in comparison with macro-level networks, NoCs must be designed while considering that, in a chip, buffers are generally more expensive resources than wires in terms of area and power. VC flow control aims at improving performance by investing in more flit buffering space to better exploit the available channel bandwidth. On the other hand, the use of *multiple physical* networks on the same chip has been proposed to improve performance and keep traffic classes separate. For instance, in the RAW processor four separate and independent NoCs are used: two NoCs are statically routed and two are dynamically routed [5].



(a)

(b)

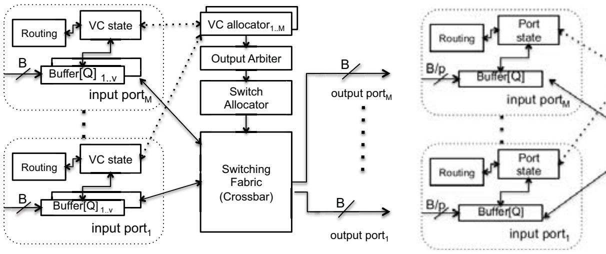
(c)

Figure 1: VC NoC (a) vs. MP NoC (b). Relative storage allocation for reference, VC, and MP NoC (c).

Contributions. We present a comparative analysis in terms of network performance, area occupation and power consumption of using virtual-channels versus multiple physical planes. For a fair comparison, we explore the design space as we vary the flit width B, the queue depth Q, the number of planes p and the number of virtual channels v while keeping the aggregate input-port storage and channel bandwidth constant. Fig. 1 compares the basic architectural organization of a *multi-plane* (MP) NoC with $p = 2$ with an equivalent VC NoC with $v = 2$. Note that the MP NoC is obtained by *partitioning* equally the same number of wires of the VC NoC across the two *planes*. The count of the total number of FFs in the routers' input queues is the same for both architectures but in the VC router the storage is spread into different virtual queues while in the MP case the storage is partitioned among smaller, simpler, and independent WH-routers (Fig. 1(c)). In the sequel we study the cases of MP NoCs with $p = 2$ and $p = 4$ and compare them with a single-plane NoC without virtual channels as well as single-plane NoCs with $v = 2$ and $v = 4$. Our analysis includes: (a) the RTL design, logic synthesis and technology mapping of various routers across three technology process generations (b) an extensive set of system-level simulations with synthetic traffic patterns, and (c) ISA simulations of a 16-core CMP running PARSEC benchmarks.

Related Work. Balfour and Dally present a comprehensive comparative analysis of NoC topologies and architectures in [6], where they also discuss the idea of duplicating certain NoC topologies, such as Mesh and *CMesh*, to improve the system performance. Carara *et al.* also propose to replicate the physical networks taking advantage of the abundance of wires between routers and compared this solution to the VCs approach [7]. Our work differs from these analyses because instead of duplicating the NoC we actually *partition* (or slice [2]) it in a number of sub-networks while keeping the overall amount of wire and buffering resources constant. Noh *et al.* propose a multi-plane-based design for a VC-enabled router [8]: the internal crossbar switch is replaced with a number of parallel crossbars (planes) that increase the flit-transfer rate between input and output queues. This results in a router with a simpler design which performs better than a single-plane router with a larger number of VCs. Differently from our study, Noh *et al.* maintain the flit-width constant as they scale the number of additional lanes.

DAC'10, June 13-18, 2010, Anaheim, California, USA
Copyright 2010 ACM 978-1-4503-0002-5 /10/06...$10.00



Terms	Definitions
B	flit width of single-plane NoC
Q	size of router input buffer
S	input storage per port (=B × Q)
p	number of physical channels
v	number of virtual channels

(a) (b) (c)

Figure 2: Block diagrams of VC (left) and MP routers (center), NoC parameters used in our comparative study (right).

Q	2			4			8			16			32		
Tech $[nm]$	90	65	45	90	65	45	90	65	45	90	65	45	90	65	45
$p = 1$		1			1			1			1			1	
$p = 2$	1.39	1.05	1.07	1.36	1.16	1.06	1.21	1.10	1.04	1.00	1.00	1.05	1.00	1.04	1.03
$p = 4$	1.40	1.13	1.21	1.32	1.23	1.16	1.56	1.21	1.22	1.12	1.18	1.14	1.16	1.15	1.15
$p = 1, v = 2$		N/A		2.03	2.42	1.92	1.68	1.32	1.36	1.01	0.87	0.88	0.79	0.69	0.67
$p = 1, v = 4$		N/A			N/A		2.51	2.33	2.37	1.53	1.55	1.53	1.06	1.02	1.01

Table 1: Power dissipation ratio (w.r.t. 1-plane WH reference NoC) for different technologies and values of p, v with $B = 128$.

2. ANALYTICAL MODEL

Fig. 2(a) shows the block diagram of a classic 5-port VC router for a 2-D Mesh network. Each I/O port is connected to a physical channel that has a data parallelism of B bits, which matches the flit size. In a VC-router with v virtual channels each input port is equipped with: (1) a routing logic block; (2) a set of v queue buffers; and (3) a VC control block that holds the state needed to coordinate the handling of the flits of the various packets. Each queue buffer can store up to Q flits. In a VC-router, a set of *VC allocator*s arbitrates the matching between input and output VCs. A *switching fabric* that forwards flits from the I/O ports is configured dynamically based on input routing and output arbitration.

Fig. 2(b) shows the block diagram of a MP router that can be used on each plane of a multi-plane 2-D Mesh NoC (*MP-router*). Its structure is simpler than the VC-router because it implements the basic WH flow control with a single queue per each input port and hence does not need VC allocators.

The table in Fig. 2(c) reports the parameters of our model. We compare the two router architectures across different NoCs by keeping the amount of storage installed on the interconnect constant. We first define a *reference NoC architecture* based on WH flow-control routers with flit-width B and input-queue depth Q. The input storage at each port is $S = B \times Q$ bits. This can be seen either a VC-router with one virtual channel ($v = 1$) or a MP-router for a single-plane NoC ($p = 1$). Then, we vary the number v of virtual channels and number p of planes by partitioning the available storage S according the following rules: (1) if $v > 1$, the queue length of a virtual channel is $Q_{VC} = Q/v$ and $B_{VC} = B$; (2) if $p > 1$, the flit width B_i of each plane is constrained by $\sum_{i=1}^{p} B_i = B$ (in case of uniform partitioning $B_i = B/p, \forall i$) and $Q_{MP} = Q$. These rules enforce $S_{MP} = S_{VC}$ which in the following is the configuration that we consider unless differently stated (*competitive sizing*).

The most common flow-control on router-to-router links in a NoC is *credit-based* flow control, which uses credits to allow the upstream router to keep track of the storage availability in the input queue of the downstream router. In order to guarantee minimal zero-load latency, this flow control imposes a constraint on the minimum size of the router input queue, which should be at least equal to the round-trip-time (RTT) of one credit on the link [9]. In the best case, the RTT is equal to 2 clock cycles, thus $Q_{min} = 2$ (*minimum sizing* constraint) [2]. This translates in an aggregate amount of storage across the p routers of a node in the MP NoC that is equal to $S_{min_{MP}} = 2 \times \sum_{i=1}^{p} B_i = 2 \times B$. Instead, the minimal storage for a VC-router is $S_{min_{VC}} = 2 \times v \times B$ because every virtual channel requires a minimum-size queue.

In summary minimum-sizing allows reducing the number of bits in the router input queues to the minimum value for a minimal zero-load latency. Since the traffic load on a NoC is often limited, longer queues do not bring relevant benefits in terms of offered throughput while having larger area and power overheads. Indeed, for those applications where the NoC is not required to achieve high throughput but only to provide connectivity among the on-chip cores, minimum-sizing can bring interesting advantages in terms of power reduction.

3. SYNTHESIS-BASED COMPARISON

We used logic synthesis to implement the two NoC architectures introduced in the previous section. For the RTL designs we took advantage of the NoC Emulator (NoCem) [10]. We performed the synthesis using *Synopsys Design Compiler* with industrial standard-cell libraries for three different technology processes ($90nm$, $65nm$, and $45nm$), and with target clock frequencies set to $500Mhz$, $1Ghz$, and $2Ghz$, respectively. We analyzed the power dissipation of the synthesized designs with *Synopsys Primetime PX*, assuming at each input port a uniformly distribution of the data bits and a traffic load of 0.4 flits/clock-cycle (roughly the saturation throughput of a 2D-Mesh). We synthesized various configurations of 2×2 NoCs[1] obtained by varying the values of the parameters of the table in Fig. 2(c). Specifically, we considered $v \in \{1, 2, 4\}$ virtual channels versus $p \in \{1, 2, 4\}$ planes with queue lengths $Q \in \{2, 4, 8, 16, 32\}$. The WH router for the reference single-plane architecture has flit-width B and queue length Q. Each plane in a multi-plane NoC has an MP-router with flit width $B_{MP} = B/p$ and input-queue length $Q_{MP} = Q$. In a VC-router, instead, the flit width is $B_{VC} = B$, while a portion of the input queue of $Q_{VC} = Q/v$ is reserved to each virtual channel. We report results for $B = 128$ bits. The trends are similar for $B = 64$ and $B = 256$.

Table 1 reports the *power dissipation* of each NoC router architecture normalized to the reference architecture ($p = 1$). For $Q \leq 8$ the management of VCs costs $32 - 142\%$ in terms of power, while the overhead for having MPs is only $4 - 56\%$, and sometimes negligible when having just $p = 2$. For $Q \geq 16$ the efficiency in area translates in a power efficiency, which is actually amplified as power savings are obtained also in a $Q = 16, v = 2$ configuration. The trends for area occupation are similar to those observed for the power analysis. Note that the results on both area occupation and power dissipation are independent from the technology pro-

[1] Notice that since the results presented in this section are given in terms of relative numbers they are independent from the network size. Hence, the analysis of a 2×2 NoC is sufficient to expose the main trends in the comparison of the alternative NoC configurations, which for instance are valid also for larger $n \times n$ 2D-Mesh NoCs.

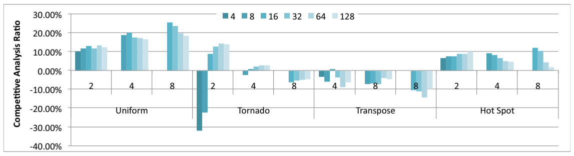


Figure 3: Throughput improvement ratio (TIR) for 2D Mesh.

cess. For short queues, $Q \leq 8$, the management of virtual channels causes substantial area overhead $(59-136\%)$, while the area overhead when implementing multiple separate planes is significantly smaller $(8-36\%)$. For long queues, $Q \geq 16$, the partitioning of an input queue into shorter queues to support multiple virtual channels delivers efficiency in terms of occupied area. E.g. for $Q = 32$, splitting the storage with $v = 2$ virtual channels delivers almost a 20% reduction in area.

In summary, when the total amount of storage in terms of sequential elements (flip-flops) that can be assigned to each router is small, it is convenient to build a MP NoC instead of an equivalent VC NoC while the opposite is true if there is room for more storage.

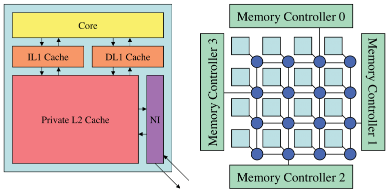
We also collected data on the critical path of the router for each configuration. This depends inversely on B and v. In a MP NoC to reduce B by a factor of $p \in \{2,4\}$ leads to a delay improvement of 1-10%. Further, a MP NoC with $p = 2$ planes can run at a clock frequency 15-25% higher than an NoC with $v = 2$ VCs, while a MP NoC with $p = 4$ planes can run at a clock frequency that is up to 25-35% faster than an NoC with $v = 4$ VCs. [2]

4. SYSTEM-LEVEL ANALYSIS

We developed an event-driven simulator with detailed models of routers and NIs and high-level models for the cores that generate and consume the network traffic. Each NI is connected to one or more routers depending on the number of NoC planes. We used a 4×4 Mesh and four synthetic traffic patterns: Uniform Random Traffic (URT), Tornado, Transpose, and 4-HotSpot [2, 9]. We set the flit width of the single-plane reference NoC to $B = 256$ while partitioning the MP NoCs with uniform width (e.g. $B_i = B/p$). We run the simulation with the offered load close to saturation.

Both VCs and MPs improve the system throughput[3]. The improvement depends on the application and the amount of buffering installed on the router's input queues. With a MP-NoC different packets assigned to different planes can be processed in parallel during a single clock cycle. Moreover MPs can be used to improve the performance also when the available storage is reduced to the minimum (i.e. $S = 2B$), whereas VCs can not be implemented because the virtual queues would become shorter than the minimum sizing constraint imposed by the flow control.

Fig. 3 shows the throughput improvement ratio TIR as function of the number of given VCs and MPs for competitive sizing: $TIR = 1 - \frac{Th_{MP}}{Th_{VC}}$. A value $TIR > 0$ indicates that VCs perform better than MPs, whereas a value $TIR < 0$ indicates that MPs perform better. The performance of the two NoCs varies notably as function of the application and available storage. In the case of random traffics, such as URT and 4HS where packets incur many channel contentions, VCs outperform MPs by offering a up to 20%. However, when we tested the 4HS traffic in a 8×8 Mesh, therefore reducing the randomness of the traffic we noticed that MPs achieve similar throughput values as VCs. In Tornado and Transpose instead the traffic flows are very regular as each core communicates with only one single other core defined by the traffic pattern itself. Here MPs exploit the reduced contention probability and outperform VCs by up to 30%. As the storage increases though VCs can store multiple flits in their queues reducing the contentions and hence improving their performance as in the case of Tornado.



(a) (b)

Figure 4: Target system: (a) logical design of a node and (b) topology of the 16-CMP with 4 memory controllers.

Processors	16 in-order 1-way SPARC cores
L1 Caches	Split IL1 and DL1 with 16KB per core, 4-way set associative, 64B line size,and 1 cycle access time.
L2 Caches	1 MB per core with 4-way set associative, 64B line size, and 3 cycle access time.
Directory Caches	256kB per memory controller with 4-way set associative, 4 cycle access time.
Memory	4 memory controllers on chip, 275-cycle DRAM access + on-chip delay

Table 2: Basic parameters of the target 16-CMP system.

Initially the improvement of throughput comes at the expense of serialization latency: using p MPs, each NI has a channel width b that is p times narrower than the reference configuration. As a consequence, each packet traveling on a MP network is made of p times more flits than in the single-plane NoC, e.g. a packet of 1 Kbit is composed by 4 flits when $b = 256$ bits or 8 flits when $b = 128$ bits. As the average load increases, however, MP NoCs can better handle the higher traffic volume, thus reducing the overall system latency and raising its maximum throughput. These trends have been observed in all of the traffic patterns that we analyzed.

5. CASE STUDY: SHARED-MEMORY CMP

We completed full-system simulation of a 16-core shared-memory chip-multiprocessor (CMP) similar to the one used by Peh et al. [4]. We used Virtutech Simics [11] with the GEMS toolset [12], augmented with GARNET, a cycle-accurate model for packet-switched NoC that supports pipelined routers with either WH or VC flow controls [13]. We extended GARNET to accommodate the modeling of heterogeneous multi-plane NoCs with different flit sizes per plane and to support on-chip directory caches. We run simulations with eight benchmarks from the PARSEC suite.: with the *simsmall* input dataset [14].

Target System. We assume that the 16-core CMP is designed with a 45*nm* technology and runs at $2Ghz$. Each core is connected to a node of a 2D-mesh NoC through a network interface as illustrated in Fig 4(a). The NoC provides support for communication with the off-chip DRAM memory through four memory controllers as illustrated in Fig 4(b). Cache access latency were characterized using CACTI [15] and cache coherence is based on the MOESI directory protocol [16]. Each memory controller has a 256kB directory cache, where each blocks consists of a 16-bit vector matching the number of private L2-caches in the CMP. The bandwidth of DRAMs, off-chip links, and memory controllers was assumed to be ideal, i.e. high enough to support all outstanding requests. The basic simulation parameters are summarized in Table 2.

Network-on-Chip Configurations. Cache coherence protocols are characterized by a number of functionally-dependent data and control messages. The MOESI cache-coherence implementation in GEMS, has four classes of messages that are exchanged among the private L2-caches and the memory controllers (Table 3): Data Request (REQ), Request Forward (FWD), Data Transfer (DATA), and Write Back (WB). Causality dependencies across messages of different classes may cause *message-dependent,* or *protocol deadlock* [17]. A common way to guarantee the absence of message-dependent deadlock is to introduce an ordering in the use of the

[2] Note that in the following system-level simulation section we conservatively assume the same clock frequency for all NoC configurations.

[3] Measured considering the bundle of MPs on each channel when $p > 1$.

Message	From → To	Size	MP assignment	
Class		(bits)	Plane ID	(flits)
REQ	Cache →Mem	64	0	8
FWD	Mem → Cache	64	1	8
DATA	Mem → Cache	576	2	18
DATA	Cache → Cache	576	2	18
WB	Cache → Mem	{64,576}	3	{8,18}

Table 3: Plane assignments for $MP_{\mathbf{4}}$ and $MP_{\mathbf{16}}$.

network resources. From a NoC design viewpoint this translates in assigning a separate set of channels and queues to each message type. Therefore, we use VCs or planes to isolate different message classes. The *baseline* virtual-channel NoC ($VC_{\mathbf{4}}^{\mathbf{64}}$) assigns to each message class a distinct virtual channel for a total of $v = 4$ virtual-channels. The flit width, which also corresponds to the channel parallelism, is $B_{VC} = 64$ bits. For each virtual channel the router has an input queue of size $Q_{VC} = 4$ and, therefore, the total buffer storage per input port is 16 flits.

As possible alternative implementations to the baseline VC NoC we consider two MP NoC configurations, called $MP_{\mathbf{4}}$ and $MP_{\mathbf{16}}$, each with $p = 4$. All the NoCs use 5-stage pipelined routers with credit-based flow control. The MP NoC configurations differ for the sizing of the router input queues: $MP_{\mathbf{4}}$ has $Q_{MP_{\mathbf{4}}} = 4$ (minimum sizing) while $MP_{\mathbf{16}}$ has $Q_{MP_{\mathbf{16}}} = 16$ (competitive sizing).

For both multi-plane configurations we partitioned the 64 bits channel parallelism of $VC_{\mathbf{4}}^{\mathbf{64}}$ as follows: $B_0 = B_1 = 8$ bits for Plane 0 and 1, $B_2 = 32$ bits for Plane 2, and $B_3 = 16$ bits for Plane 3. Table 3 reports the plan assignment for each message class together with the message size expressed both in bits and in the number of flits that are necessary when this message is transferred on a MP-NoC. For example, a DATA message, which consists of a cache line of 512 bits and an address of 64 bits, is transmitted as a worm of 18 flits on Plane 2, whose flit width is $B_2 = 32$. Notice that the same message incurs a much smaller serialization latency when transmitted as a sequence of 9 flits on the VC-NoC $VC_{\mathbf{4}}^{\mathbf{64}}$, whose flit width is $B_{VC} = 64$ bits [4]. Similarly, a REQ message, which consists of 64 bits, requires 8 flits to be transmitted on Plane 0 of either $MP_{\mathbf{4}}$ or $MP_{\mathbf{16}}$, but only 1 flit on $VC_{\mathbf{4}}^{\mathbf{64}}$.

Experimental Results. Fig. 5 reports the *average flit latency* measured while running the eight PARSEC workloads on the 16-core CMP for the two MP NoC configurations. The values are normalized with respect to the corresponding values for the VC NoC configuration. The latency is measured from the flit generation to its arrival at the destination and includes the serialization latency (the flits are queued right after identifying the coherent status of L2-cache block). Consequently, it is not surprising that both MP NoCs present a performance loss with respect to the baseline VC NoC. On the other hand, the MP NoCs offer an interesting power/performance trade-off. Under competitive sizing, $MP_{\mathbf{16}}$ offers a 18% average power saving in exchange for a 14% average performance loss across all benchmarks. Under minimum sizing constraints, instead, $MP_{\mathbf{4}}$ reaches an average power saving of about 70% at the cost of a average performance loss of 32%.

6. CONCLUSIONS

We compared virtual channels (VC) with physical multi-plane (MP) networks-on-chip in terms of system-level performance, gate-level area occupation, and power dissipation. First we showed that, independently from the technology process, MP are an efficient solution when the amount of storage resources at the input ports of each router is limited. Instead, VC give interesting advantages when it is possible to equip the router inputs with longer queues that can be efficiently partitioned among the virtual channels. Then, we showed that the choice of implementing VC rather than MP can also be driven by the characteristics of the traffic over the NoC. More irregular traffic patterns take advantage of deeper input queues in the routers, thus working best on a VC NoC. When the traffic is more regular, the collisions are less frequent and, therefore, shorter queues can be deployed in the routers, thereby making MP a good design choice. Finally, we performed a comparative analysis running a suite of real benchmark applications on an instruction-set-architecture for an hypothetical 16-core CMP. Here we showed that as long as the amount of storage resources on the NoC is comparable, the two solutions offer similar power efficiency. However, MP permits to reduce the queue lengths without compromising functionality, which leads to implementations with higher performance-per-watt efficiency when a NoC with long buffering queues would be over-provisioned with respect to the offered load. In summary, to have multiple physical networks appear as an interesting design choice for NoC that need to satisfy low performance requirements with low amount of available resources, but with more stringent needs in terms of power savings. Virtual channels, instead, are a better solution for NoCs in case of high resource availability and performance needs.

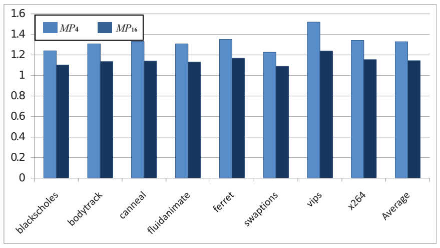


Figure 5: Average flit latency.

Acknowledgments. This work was partially supported by the Gigascale Systems Research Center, one of six research centers funded under the Focus Center Research Program (FCRP), a Semiconductor Research Corporation entity.

7. REFERENCES

[1] J. Owens *et al.*, “Research challenges for on-chip interconnection networks,” *IEEE Micro*, vol. 27, no. 5, pp. 96–108, Sept.-Oct. 2007.

[2] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004.

[3] L.-S. Peh and W. J. Dally, “A delay model for router microarchitectures,” *IEEE Micro*, vol. 21, pp. 26–34, Jan. 2001.

[4] L.-S. Peh *et al.*, “In-network snoop ordering (INSO): Snoopy coherence on unordered interconnects,” in *Int. Sym. on High Perf. Computer Architecture (HPCA)*, Feb. 2009, pp. 67–78.

[5] M. B. Taylor *et al.*, “The Raw microprocessor: A computational fabric for software circuits and general purpose programs,” *IEEE Micro*, vol. 22, no. 2, Mar-Apr 2002.

[6] J. Balfour and W. J. Dally, “Design tradeoffs for tiled CMP on-chip networks,” in *Conf. on Supercomputing*, Nov. 2006, pp. 187–198.

[7] E. Carara *et al.*, “Router architecture for high-performance NoCs,” in *Proc. of the Conf. on Integrated circuits and systems design*, Jan. 2007, pp. 111–116.

[8] S. Noh *et al.*, “Multiplane virtual channel router for network-on-chip design,” in *Int. Conf. on Communications and Electronics*, Oct. 2006, pp. 348–351.

[9] N. Concer *et al.*, “Distributed flit-buffer flow control for networks-on-chip,” in *Proc. of the Int. Conf. on HW/SW Codesign & System Synthesis*, Sep. 2008, pp. 215–220.

[10] Website, www.opencores.org/.

[11] P. S. Magnusson *et al.*, “Simics: A full system simulation platform,” *IEEE Computer*, vol. 35, no. 2, pp. 50–58, Feb. 2002.

[12] M. M. K. Martin *et al.*, “Multifacet’s general execution-driven multiprocessor simulator (GEMS) toolset,” *SIGARCH Comput. Architecture News*, vol. 33, no. 4, pp. 92–99, Nov. 2005.

[13] L.-S. Peh *et al.*, “GARNET: A detailed on-chip network model inside a full-system simulator,” in *Int. Symp. on Perf. Analysis of Systems and Software*, Apr. 2009, pp. 33–42.

[14] C. Bienia *et al.*, “The PARSEC benchmark suite: characterization and architectural implications,” in *Conf. on Parallel arch. and compilation techniques*, Oct. 2008, pp. 72–81.

[15] S. Thoziyoor *et al.*, “CACTI 5.1,” HP, Tech. Rep., 2008.

[16] B. Jackob, S. W. Ng, and D. Wang, *Memory Systems: Cache, DRAM, Disk*. Morgan Kaufmann, 2007.

[17] Y. H. Song and T. M. Pinkston, “A progressive approach to handling message-dependent deadlock in parallel computer systems,” *IEEE Trans. on Par. and Dist. Systems*, vol. 14, no. 3, pp. 259–275, Mar. 2003.

[4] Notice that GARNET does not model the head/tail flits of a worm.