*IRWIN AND JOAN JACOBS*
*CENTER FOR COMMUNICATION AND INFORMATION TECHNOLOGIES*

# BENoC: A bus-Enhanced Network on-Chip

**Isask'har Walter, Israel Cidon and Avinoam Kolodny**

**CCIT Report #677**
**December 2007**


*DEPARTMENT OF ELECTRICAL ENGINEERING*
*TECHNION - ISRAEL INSTITUTE OF TECHNOLOGY, HAIFA 32000, ISRAEL*

CCIT Report #677 December 2007

# BENoC: A Bus-Enhanced Network on-Chip

Isask'har Walter[1], Israel Cidon[2], Avinoam Kolodny[2]
*Electrical Engineering Department, Technion – Israel Institute of Technology, Haifa 32000, Israel*
*[1]zigi@tx.technion.ac.il , [2]{cidon, kolodny}@ee.technion.ac.il*

## Abstract

**Recent research has shown that Network on-chip (NoC) is superior to a bus in terms of power and area for given traffic throughput requirements. Consequently, NoC is expected to be the main interconnect infrastructure in future System on Chip (SoC) and chip multi-processor (CMP). Unlike off-chip networks, VLSI modules are only a few millimeters apart, hence the cost of off-network communication among the network end-points and routers is quite low. Such off-network communication can circumvent weaknesses of the NoC, such as latency of critical signals, complexity and cost of broadcast operations, and operations requiring global knowledge or central control.**

**In this paper we explore the benefits of adding a low latency, customized shared bus as an integral part of the NoC architecture. While the bus is inferior to NoC in terms of data throughput, it possesses two main advantages: First, the bus is inherently capable to broadcast information. Second, the bus has lower and more predictable propagation latency. Therefore, the bus is superior to a multi-hop network for certain transactions such as broadcast of queries, fast delivery of control signals, quick exchange of small data items, network configuration and power management. Moreover, custom properties can be tailored to this particular bus in order to facilitate these specialized tasks. As a result, the Bus-enhanced NoC (BENoC) is overall more cost-effective than a traditional "busless" NoC.**

**We describe several applications of bus-enhanced networks, such as cache lines lookup and coherency in CMP and efficient management of SoC resources. We present an analytical comparison of the power saving in BENoC versus a network providing similar services. Finally, simulation is used to evaluate the performance of BENoC in a chip multiprocessor system which employs a distributed cache with dynamic non-uniform cache access (DNUCA).**

## Categories and Subject Descriptors

System-Level Design and Co-Design: Network-on-Chip (NoC)

## General Terms

Performance, Design

## Keywords

Network on-Chip, resource management, SoC, NoC support for CMP/MPSoC

## 1. Introduction

There is a large body of work advocating the use of spatial reused networks as the main on-chip interconnection infrastructure (e.g., [1]-[4]). Network architecture has been shown to be more cost effective than a system bus in terms of area, power and performance [5]. In addition, networks generally have good scalability properties, while shared busses cannot withstand the increasing bandwidth and performance requirements already seen in contemporary systems. Consequently, current state-of-the-art VLSI research often presents NoC as the practical choice for future systems. However, conventional interconnect architectures which solely rely on a network have several drawbacks when advanced services are required. In particular, the distributed nature of a network is an obstacle when global knowledge or operation is beneficial. For example, broadcast (sending information to all modules on the chip) is an inherent operation in busses and has no extra cost. However, in a typical NoC a broadcast capability either involves additional hardware mechanisms or a massive duplication of unicast messages. Broadcast is particularly expensive in NoCs that employ wormhole switching [6], as classic wormhole does not support broadcast due to the complexity of the backpressure mechanism and the requirement for small buffers. Similarly, multicast is considerably easier to implement in busses than in typical networks. Finally, multi-hop networks impose an inherent packet propagation latency for the communication between modules. This complicates the design of critical signals between remote modules. Bus properties are also valuable when global knowledge and control are useful. As current NoC implementations are strictly distributed (heavily borrowing concepts from traditional large scale networks), the system behavior and performance is often dictated by multiple local decisions. For example, arbitration for scarce resources is typically

conducted using only local knowledge [7]. Since a bus is inherently shared by many agents allowing them to simultaneously monitor its activity, it can be used to maintain a global view of the system status and as mean for achieving a global decision. Unlike off-chip architectures, modules within a chip are placed in close proximity to each other. This enables off-network communication solutions that are not feasible in large scale networks, e.g., NoC reconfiguration, network search and arbitration for a hot-module [7]. These specialized operations can be performed over an off-network shared bus, at a low latency and low dissipation of power.

Consequently, we propose a new architecture called BENoC (Bus-Enhanced Network on-Chip), composed of both a high performance distributed network, and a complementary low latency, low bandwidth specialized bus (Figure 1). The bus, which is optimized for system-wide distribution of signals and can be centrally arbitrated, is used for low-latency communication and large scale distribution of meta-data in a simple and efficient manner, while the network is used for high-throughput point-to-point communication between modules. As a result, the proposed combination is superior to a conventional NoC. We present several scenarios in which the bus-network hybrid is more cost effective than a pure network implementation. Moreover, we also demonstrate that the custom bus can be equipped with additional simple mechanisms that further facilitate common distributed tasks.

The observation that busses are superior to networks for providing low-latency for low bandwidth signals has already inspired several proposals of bus-NoC hybrids. Typically, such hybrid solutions employ clusters of modules where each cluster shares a local bus. While intra-cluster communication uses the local bus, inter-cluster traffic uses the network [8]. So unlike BENoC, in previous proposals, busses are used as a local NoC alternatives (but only within a cluster), and support the same semantics while not offering additional functionality. In [9], the authors suggest a bus-NoC hybrid for a uniprocessor system. There, the low-latency nature of the bus is used to accelerate the access to an array of distributed cache banks. By replacing groups of adjacent links and routers with fast bus segments, the hop count is reduced and the system performance is improved. In contrast with these approaches, BENoC does not employ the bus as an additional hierarchy layer in the interconnect fabric or an extension of the network but rather as a synergetic component operating in parallel with the network at each network endpoint, improving traditional functionality and offering new services.

The rest of this paper is organized as follows: in Section 2, we discuss the usage of the bus-enhanced NoC architecture. In Section 3 we analyze the energy consumption of the BENoC architecture while Section 4 evaluates the proposed architecture using simulations. Finally, Section 5 concludes the paper.

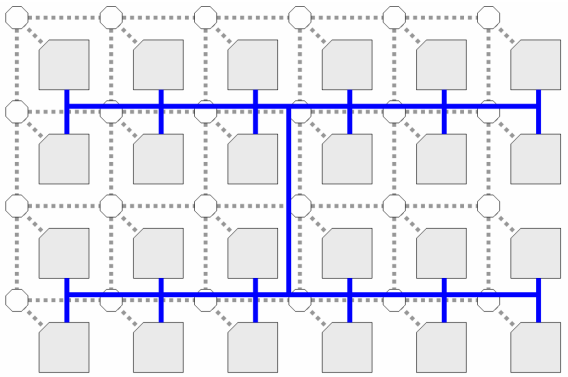

**Figure 1: An example of a BENoC architecture**
In this example, a single segment bus spans through all system modules while the data network is organized in a mesh topology. A dotted line marks network links; a solid one represents the bus.

## 2. BENoC Built-in Services

BENoC is composed of two tightly related parts: a packet switched network (e.g., AEthereal [1], QNoC [2], XPipes [3], Hermes [4] that takes care of point-to-point massive data transfers, and a custom bus that concurrently functions as a low latency broadcast/multicast capable media. The bus is used for NoC subsystem control, propagation of critical signals and special custom services. In this section we describe some of the benefits of using BENoC.

### 2.1 BENoC for short latency signaling

In typical NoC-based systems, packets that traverse a path of multiple hops suffer from high latency, due the routing delay accumulated along its way. This latency is often unacceptable for short but urgent signaling messages required for the timely operation of the system, and is considered as one of the NoC's main obstacles that discourage architects from an early adoption a NoC-based architecture. BENoC's custom bus, which is designed for low bandwidth and optimized for short latency, offers a valuable

alternative: such urgent messages may be sent over the bus, traversing only a single arbitration stage. This enables quick delivery of time critical signals (e.g., interrupts, semaphore lock operations) between modules.

## 2.2 BENoC multicast services

BENoC also enables efficient implementation of communication services common in large distributed SoCs. For example, a high performance ASIC or FPGA may include multiple resources with the same functionality distributed in different locations across the chip (e.g., DSP processors, ALUs, multipliers, memory banks, etc.). Instead of performing a complex computation locally, a processor may complete its task in a more efficient manner by sending the data to be processed to one (or more) of these specialized resources. Note that in such cases, the processor does not know which of these resources are idle, so in a basic NoC-system it can only probe them using the network. For such cases, BENoC can easily provide an *anycast* service: In such an operation, the origin processor targets any module that owns a certain type of resource and fulfills certain conditions. For instance, in the above scenario, the processor may initiate a bus transaction destined at "any idle multiplier". In response, idling multipliers may arbitrate for the bus in order to send back their ID or use the network to do so. Note that the bus is only used for control messages and metadata, while the data itself is delivered point-to-point over network. More sophisticated buses may include a custom *convergecast* mechanism that facilitates the efficient collection of acknowledgements or negative result back to the initiator. Such a mechanism may use a daisy-chain circuit along the bus route to feedback back to the origin the identity of the available resource or the complete lack of such a free resource.

As mentioned above, the bus implements certain communication services in a better cost effective manner. The most basic service is a broadcast operation: In order to deliver a message from one source to multiple destinations in a basic NoC, the sender has to generate multiple unicast messages [10]. In addition to an increased latency, this process is energy consuming, as the same information is repeatedly transmitted over the same lines. While the NoC routers may include a built-in broadcast mechanism (e.g., [11]), this extra hardware cannot match the simplicity, low-cost and short latency of the proposed bus.

## 2.3 BENoC for CMP Cache

A broadcast operation is extremely valuable in shared memory CMP systems. Typically, each of these processors is equipped with a local (L1) cache and they all share a distributed L2 cache (Figure 2). In order to facilitate cache coherency, the system should provide a mechanism that prevents applications from reading stale data. More specifically, when a processor issues a read exclusive (i.e., read for ownership) command to one of the L2 caches, all other processors holding a copy of that cache line should invalidate their local copy, as it no longer reflects the most updated data. Such invalidation signal is best propagated using a broadcast/multicast service.

As wire latency becomes a dominant factor, the L1 miss penalty is heavily affected by the distance between the processor and the L2 cache bank holding the fetched line. This observation gave rise to the DNUCA (Dynamic Non-Uniform Cache Architecture) approach: instead of having a few statically allocated possible L2 locations, cache lines are moved towards processors that access them [12][13]. Ideally, all cache lines that are accessed by a certain processor reside in nearby L2 cache banks.

There are several issues to resolve in order to make DNUCA a practical cache management scheme. Examples are finding an efficient line migration policy, handling lines that are accessed by multiple, distant processors and cache line migration schemes. Another major difficulty in implementing DNUCA is the need to lookup cache lines: whenever a processor needs to conduct a line fill transaction (fetch a line into its L1 cache), it needs to determine its location, i.e., the identity of the L2 cache bank/processor storing its updated copy.

As described above, in a network-based interconnect, the line can be looked for using multiple unicast messages. BENoC offers a much more efficient alternative: the low latency bus can be used to broadcast the query to all cache banks. The particular cache storing the line can acknowledge receiving the request on the auxiliary bus and simultaneously send the line's content over the NoC. As queries are composed of small meta-data (the initiating processor's ID and the line's address), they do not create substantial load on the auxiliary bus.

The proposed scheme has two main advantages: First, it reduces the power consumption of the system interconnect as the single bus transaction performs the broadcast operation, instead of multiple messages in the NoC. Second, as the time-critical line search is performed over a dedicated single-hop medium instead of competing for shared network resources, the system performance is improved. In Section 4 we evaluate BENoC for a DNUCA system.

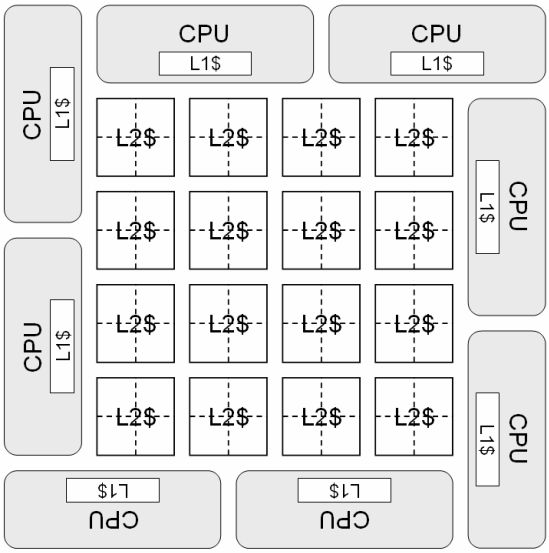


**Figure 2: An example of a CMP System**
The system is composed of eight processor cores and 16 L2 cache banks. Each L2 bank is divided into 4 sub-banks.

## 2.4 BENoC for system management

The custom bus can also facilitate the configuration and management of the NoC itself. For example, when changing the system's operation mode ("usecases" in [14]), the network resources may need to be re-configured. Such configuration may include updating routing tables, adjusting link speeds or turning some of them completely off and remapping the system modules address space. Interestingly, although these operations are not performed during the normal run-time of the system, they should be handled with care: Since the configuration of different network resources is performed independently, they may interfere with each other. For example, if a configuration packet turns off a certain link (or a router), other configuration messages may not be able to reach their destination due to "broken paths". Similarly, trying to update routing table while the network is being used to deliver other configuration messages is problematic. Alternatively, configuration can be done using the custom bus. As a result, the configuration process becomes simpler to design and implement. In fact, special side-band signals are often implemented in bus-based interconnect to ease bootstrap configuration of the system (e.g., PCI bus, Power PC's DCR bus).

It may also be desirable to completely shut off parts of the NoC when they are not expected to be used for a long time in order to save power. However, a major complication in the implementation of such a mechanism is caused by the inability to switch on inactive units fast enough when they are needed, as the "wakeup" packets cannot traverse though sleeping links or routers. Using the bus, units that were switched off can be awakened in a simple, fast and direct manner. Moreover, the bus can be used to handle the communication between the modules during the NoC initialization and power-up time.

## 3. Analysis

In this section we approximate the energy required for a broadcast operation in a regular NoC and in BENoC. For simplicity, we assume the network has a regular mesh topology. We use the following notation:

n = The number of modules in the system

$\Delta V$ = Voltage swing [V]

$C_0$ = Global wire capacitance per unit of length [F/mm]

P = Tile size [mm]

$C_{ld}$ = NoC link driver input capacitance [F]

$C_{bd}$ = Bus driver input capacitance [F]

$C_{min}$ = Minimal inverter input capacitance [F]

We model the time needed for a driver to charge a capacitor using the following equation [15]:

$$T = \frac{\tau}{C_{in}} C_{Load} + \tau \qquad (1)$$

where $C_{in}$ is the driving buffer's input capacitance and $C_{load}$ is the load's capacitance. The constant $\tau$ is determined by the technology [16]:

$$\tau \simeq R_{\min} C_{min} \qquad (2)$$

where $R_{min}$ and $C_{min}$ are the effective resistance and the input capacitance of a minimal inverter. The energy required to charge $C_{load}$ is

$$E = \Delta V^2 \cdot C_{load} . \quad (3)$$

First, we approximate the latency and energy of a broadcast transaction in a NoC-based system which relies on multiple unicast messages. Assuming each NoC link is approximately P millimeters long, its capacitance is

$$C_{link} = P \cdot C_0 . \quad (4)$$

Using (1), the time required for a link driver to transmit a single bit is

$$T_{link} = \frac{\tau}{C_{ld}}(C_{link} + C_{in}) + \tau \ , \quad (5)$$

where $C_{in}$ is the input capacitance in the input port to which the link is connected.

Since a broadcast message has to travel at least $\sqrt{n}$ modules away from the source, the minimal time to complete the broadcast (neglecting delay within the router) is

$$\begin{aligned} T_{net} &= \sqrt{n} \cdot T_{link} = \sqrt{n}\left(\frac{\tau}{C_{ld}}(C_{link} + C_{in}) + \tau\right) \\ &= \sqrt{n}\left(\frac{\tau(P \cdot C_0 + C_{in})}{C_{ld}} + \tau\right) \end{aligned} . \quad (6)$$

Note that (6) underestimates the broadcast latency, as messages are withheld at least one clock cycle in each router along their path. In addition, if no priority is given to such packets, they might also be delayed due to network congestion.

In order to calculate the total energy needed for NoC broadcast, we should first determine the number of times a packet is transmitted. Note that in a regular mesh, a source node may have at most 8 modules at a distance of one, 16 modules two hops away, 24 modules three hops away and so on. In the energy-wise best case, the broadcasting module is located exactly in the middle of the mesh. The broadcasting module therefore has to send 8 messages that would each travel a single link each, 16 messages that travel two links, and in general, 8j messages to a distance of j hops, until transmitting a total of n-1 messages. It can be easily shown that if $\sqrt{n}$ is an integral, odd number, then the Manhattan distance between the module in the middle of the mesh and the ones in its perimeter is exactly

$$D_{\max} = \frac{\sqrt{n} - 1}{2} . \quad (7)$$

Since a message transmitted to a destination j hops away has to traverse j links, the minimal number of transmissions required to complete the broadcast is

$$\begin{aligned} K &= 8 \cdot 1 + 16 \cdot 2 + 24 \cdot 3 + ... + 8D_{\max} \cdot D_{\max} \\ &= 8\sum_{j=0}^{D_{\max}} j^2 = 8\frac{D_{\max}(D_{\max} + 1)(2D_{\max} + 1)}{6} \\ &= \frac{16D_{\max}^3 + 24D_{\max}^2 + 8D_{\max}}{6} \end{aligned} \quad (8)$$

Consequently, the lower bound of the total energy consumed by a single broadcast operation according to (3) is

$$E_{net} = \Delta V^2 \cdot K(C_{ld} + C_{link} + C_{in}) . \quad (9)$$

Similarly, we now evaluate the latency and energy that characterize a broadcast on a bus. We assume that the bus is composed of $\sqrt{n}$ horizontal sections (of length $\sqrt{n} \cdot P$ each), connected together using a vertical segment of the same length. As the total bus length is approximately $\left(\sqrt{n} + n\right)P$ long, and assuming that it is connected to n loads of $C_{in}$ each, its total capacity is approximately

$$C_{bus} \simeq \left(\sqrt{n} + n\right)PC_0 + nC_{in} . \quad (10)$$

The resulting broadcast transmission delay according to (1) is:

$$\begin{aligned} T_{bus} &= \frac{\tau}{C_{bd}}C_{bus} + \tau \\ &= \frac{\tau}{C_{bd}}\left(\left(\sqrt{n} + n\right)PC_0 + nC_{in}\right) + \tau \end{aligned} \quad (11)$$

Using (3), the total energy required to drive the bus:

$$\begin{aligned} E_{bus} &= \Delta V^2(C_{bus} + C_{bd}) \\ &= \Delta V^2\left(\left(\sqrt{n} + n\right)PC_0 + nC_{in} + C_{bd}\right) \end{aligned} \quad (12)$$

Clearly, the bus driver should be much more powerful (and energy consuming) than a link driver. In order to choose an appropriate sizing for the bus driver, we require:

$$\frac{T_{net}}{T_{bus}} = \beta \tag{13}$$

where β is a parameter reflecting the network-to- bus broadcast speed ratio.
Using equations (6), (11) and (13), we get:

$$\begin{aligned} &\frac{\tau}{C_{bd}}\left(\left(\sqrt{n}+n\right)PC_0 + nC_{in}\right)+\tau \\ &= \frac{\sqrt{n}}{\beta}\left(\frac{\tau(P\cdot C_0 + C_{in})}{C_{ld}}+\tau\right) \end{aligned} \tag{14}$$

and therefore, for achieving a desired speed ratio β the bus driver should have an input capacitance of

$$C_{bd} = \frac{\tau\left(\sqrt{n}PC_0 + nPC_0 + nC_{in}\right)}{\frac{\sqrt{n}}{\beta}\left(\frac{\tau(P\cdot C_0 + C_{in})}{C_{ld}}+\tau\right)-\tau}. \tag{15}$$

Using (12), we get to the total energy consumption required for a bus broadcast:

$$\begin{aligned} E_{bus} &= \Delta V^2\left(\left(\sqrt{n}+n\right)PC_0 + nC_{in}\right) \\ &+\Delta V^2\left(\frac{\tau\left(\sqrt{n}PC_0 + nPC_0 + nC_{in}\right)}{\frac{\sqrt{n}}{\beta}\left(\frac{\tau(P\cdot C_0 + C_{in})}{C_{ld}}+\tau\right)-\tau}\right) \end{aligned} \tag{16}$$

In order to complete the analysis, we use typical values for the various electrical parameters for 0.65um technology [16]. The tile size (P) is assumed to be 1mm, and $C_{ld}$ is selected so that the resulting single-wire link bandwidth is 20Mb/sec.
Figure 3 shows the energy required for unicast and broadcast transmissions in a NoC. It also shows the energy required for a transmission in BENoC for two bus speeds (values of β). As expected, the bus is no match for the NoC when a message should be delivered to a single destination. The energy required for the delivery of a unicast message traveling an average distance in a mesh NoC is proportional to $\sqrt{n}$ while in the bus the energy is approximately linear with respect to the number of modules using reasonable values of the speed ratio beta. Obviously, trying to provide the total network throughput capacity on the bus would be extremely wasteful in terms of power. However, when broadcast operations are compared, the bus is considerably more energy efficient than the network, as shown by the "network broadcast" curve compared with the "bus transaction" curves, for system size n of ~25 or more.

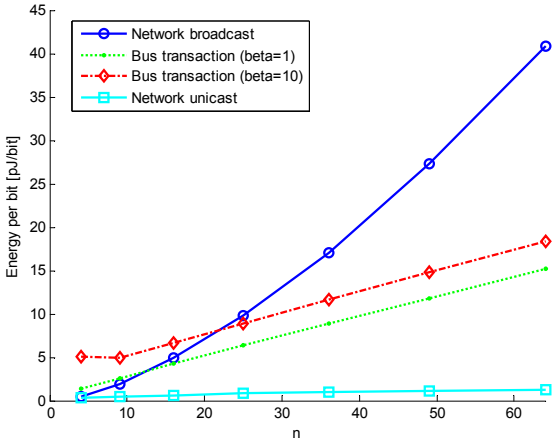


**Figure 3: Energy consumption in NoC and BENoC**
The energy consumed by a network unicast and broadcast, and by a bus transmission.

## 4. Experimental Results

In this section, we evaluate the BENoC and a regular NoC interconnected for a classical CMP system depicted in Figure 2, supporting dynamic non-uniform cache access architecture which consists of 8 processors and 64 distributed cache banks. In order to demonstrate the core properties of BENoC, we assume that the bus uses centralized arbitration.
We focus on two time-critical operations in a DNUCA system. The first one is the basic line-fill ("read") transaction, which is performed by a processor that tries to read a line into its L1 cache. If an L2 cache has a valid copy of the line, it must provide its content to the reading processor. If the most updated copy resides in a L1 cache of another processor, it is asked to "writeback" the line. Else, the line is fetched from a lower memory hierarchy level (L3 cache/memory). When the operation is completed, the processor becomes a "sharer" of the line.

The second operation is the read-for-ownership ("read-exclusive") transaction, which is similar to the basic line-fill operation, but also implies that the reading processor wishes to have the single valid copy of the line as it is about to update its content. In order to complete the transaction, all other L1 copies of the line (held by an owning processor or by sharers) must be invalidated.

A processor performing a read/read exclusive operation does not know the exact state of the requested line. More precisely, the line might be owned by another processor, shared by one or more processors or it may not be present in any of the L2 caches at all. In addition, even if the line is in an L1/L2 cache, the reading processor does not know its location. In a typical DNUCA implementation, the processor has therefore to lookup the line prior to the read/read exclusive operation. In this work, we assume a classic model in which each L2 cache line includes some extra bits to keep track of the current sharers/owner of the line [17].

In order to evaluate the proposed technique, we use two simulators. In order to simulate the BENoC architecture we use Opnet [18]. The model accounts for all network layer components, including wormhole flow control, virtual channels, routing, finite router buffers and link capacities. It addition, it simulates the bus arbitration and propagation latencies. The DNUCA system was modeled using Simics [19] which is a well-known parallel execution simulator. Our benchmarks are composed of SPLASH-2 [20] traces executed on a CMP system. Since we are interested in the parallel sections of the programs, we fast forward through the initial sequential part of each program and measure performance only in the parallel part of the code.

Figure 4 presents the decrease in the line fill transaction time in BENoC relative to the average duration of the same transactions in a standard NoC system, for various network-to-bus speed ratios (i.e., different values of β). As expected, BENoC significantly reduces the average transaction time. This is also true for slow custom busses, which are also very power efficient (Section 3). Note that even when an extremely high latency bus is used, BENoC achieves a significant performance improvement. This results from the fact that in the above analysis we have used a lower bound for network latency. In a real network, broadcast messages of cores are likely to collide, as they repeatedly compete for the network resources. In addition, even when no collisions occur, routers introduce some additional latency.

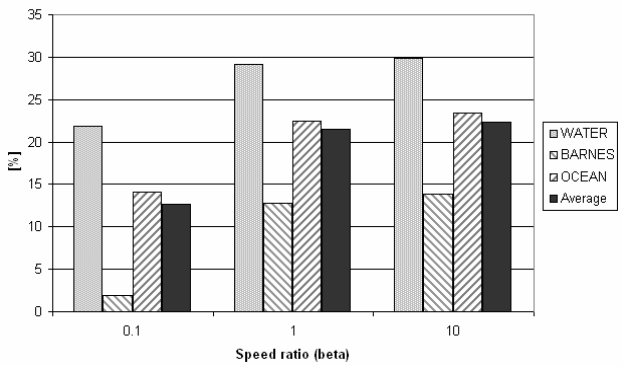


**Figure 4: L2 access time improvement**
The reduction in the line fill transaction time in benchmark programs, for different network-to-bus speed ratios.

## 5. Summary

A salient feature of on-chip systems is the proximity of all components within a distance of several millimeters, which enables low-latency communication among them. This is in contrast with macro networks, where link delays are inherently dominant in the system. Therefore, traditional networks usually cannot benefit from out-of-band communication and they use their standard links for all operations, while NoCs can leverage a side-bus to enhance system functionality.

The bus-enhanced NoC architecture described in this paper suggests to combine a customized bus with a NoC for getting a best of breed communication infrastructure. The customized bus can circumvent some weaknesses of the NoC, such as latency of critical signals, complexity and cost of broadcast operations, and operations requiring global knowledge or central control. It is used to support the network in specialized operations and services, such as broadcast, anycast and convergecast, which are essential for common operations such as cache line search and cache invalidation. The bus can also be used point-to-point to support low-latency critical signals with a small number of bits. BENoC is superior to classical NoC in terms of delay and power. Our approximate analysis shows that BENoC advantage over NoC starts at relatively small system

size around 10-20 modules, and becomes very significant as system size grows.

In conclusion, the scalability requirements of future SoCs can be served by a NoC providing high throughput and parallelism for massive data transfer, enhanced by an integral customized-bus providing low-latency and broadcast capabilities for control operations and specialized services, in a BENoC architecture.

## 6. References


[1] K. Goossens, J. Dielissen, and A. Radulescu, "AEthereal Network on Chip: Concepts, Architectures, and Implementations", IEEE Design and Test of Computers, 2005

[2] E. Bolotin, I. Cidon, R. Ginosar and, A. Kolodny, "QNoC: QoS Architecture and Design Process for Network on Chip", Journal of Systems Architecture, Volume 50, February 2004

[3] D. Bertozzi and L. Benini, "Xpipes: A Network-on-Chip Architecture for Gigascale Systems-on-Chip", Circuits and Systems Magazine, IEEE Volume 4, Issue 2, 2004

[4] F. Moraes, N. Calazans, A. Mello, L. Möller, and L. Ost, "Hermes: an Infrastructure for Low Area Overhead Packet-switching Networks on Chip", Integration, the VLSI Journal, Oct. 2004

[5] E. Bolotin, I. Cidon, R. Ginosar and A. Kolodny, "Cost Considerations in Network on Chip", Journal of Systems Architecture, special issue on Network on Chip, Volume 50, February 2004, pp. 105-128

[6] W.J. Dally and C. Seitz, "The Torus Routing Chip", Distributed Computing, vol. 1, no. 3, 1986

[7] I. Walter, I. Cidon, R. Ginosar, and A. Kolodny, "Access Regulation to Hot-Module in Wormhole NoCs", NOCS 2007

[8] Richardson, T.D.; Nicopoulos, C.; Park, D.; Narayanan, V.; Yuan Xie; Das, C.; Degalahal,, V. "A hybrid SoC interconnect with dynamic TDMA-based transaction-less buses and on-chip networks", 19th International Conference on VLSI Design

[9] N. Muralimanohar, R. Balasubramonian, ", Interconnect design considerations for large NUCA caches" International Symposium on Computer Architecture, Proceedings of the 34th annual international symposium on Computer architecture, 2007

[10] E. Bolotin, Z. Guz, I. Cidon, R. Ginosar, A. Kolodny, "The Power of Priority: NoC based Distributed Cache Coherency", NOCS 2007

[11] Y. Jin, E. J. Kim, and K. H. Yum, "A Domain-Specific On-Chip Network Design for Large Scale Cache Systems", in Proceedings of 13th International Symposium on High-Performance Computer Architecture (HPCA-13), Phoenix, 2007

[12] C. Kim, D. Burger, S. W. Keckler, "An Adaptive, Non-Uniform Cache Structure for Wire-Delay Dominated On-Chip Caches", Int. Conf. on Arch. Sup. for Prog. Lang. and Oper. Sys., pp. 211-222, October, 2002

[13] C. Kim, D. Burger, S. W. Keckler, "Nonuniform Cache Architectures for Wire Delay Dominated on-Chip Caches", IEEE Micro, 23:6, pp. 99-107, November/December, 2003

[14] S. Murali, M. Coenen, A. Radulescu, K. Goossens, and G. De Micheli, "A Methodology for Mapping Multiple Use-Cases onto Networks on Chips", Proc. Design, Automation and Test in Europe (DATE) 2006

[15] I. Sutherland, R. F. Sproull, and D. Harris, "Logical Effort: Designing Fast CMOS Circuits", The Morgan Kaufmann Series in Computer Architecture and Design, ISBN: 978-1-55860-557-2

[16] Predictive Technology Model, http://www.eas.asu.edu/~ptm

[17] B. M. Beckmann and D. A. Wood, "Managing wire delay in large chip multiprocessor caches", MICRO 37, pages 319-330, Dec. 2004

[18] OPNET Modeler, www.opnet.com

[19] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, and G. Hallberg, "Simics: A full system simulation platform", IEEE Computer, 35(2):50–58, Feb. 2002

[20] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta., "The SPLASH-2 Programs: Characterization and Methodological Considerations", In Proceedings of the 22nd Annual International Symposium on Computer Architecture, pages 24–37, June 1995