Performance Evaluation and Design Trade-Offs for Network-on-Chip Interconnect Architectures

Partha Pratim Pande, *Student Member*, *IEEE*, Cristian Grecu, Michael Jones, André Ivanov, *Senior Member*, *IEEE*, and Resve Saleh, *Senior Member*, *IEEE*

Abstract—Multiprocessor system-on-chip (MP-SoC) platforms are emerging as an important trend for SoC design. Power and wire design constraints are forcing the adoption of new design methodologies for system-on-chip (SoC), namely, those that incorporate modularity and explicit parallelism. To enable these MP-SoC platforms, researchers have recently pursued scaleable communication-centric interconnect fabrics, such as networks-on-chip (NoC), which possess many features that are particularly attractive for these. These communication-centric interconnect fabrics are characterized by different trade-offs with regard to latency, throughput, energy dissipation, and silicon area requirements. In this paper, we develop a consistent and meaningful evaluation methodology to compare the performance and characteristics of a variety of NoC architectures. We also explore design trade-offs that characterize the NoC approach and obtain comparative results for a number of common NoC topologies. To the best of our knowledge, this is the first effort in characterizing different NoC architectures with respect to their performance and design trade-offs. To further illustrate our evaluation methodology, we map a typical multiprocessing platform to different NoC interconnect architectures and show how the system performance is affected by these design trade-offs.

Index Terms—Network-on-chip, MP-SoC, infrastructure IP, interconnect architecture, system-on-chip.

1 Introduction and Motivation

SOC design methodologies will undergo revolutionary changes in the years to come. According to recent publications [1], [2], [3], the emergence of SoC platforms consisting of a large set of embedded processors is imminent. A key component of these multiprocessor SoC (MP-SoC) platforms [2] is the interconnect topology. Such SoCs imply the seamless integration of numerous IPs performing different functions and operating at different clock frequencies. The integration of several components into a single system gives rise to new challenges. It is critical that infrastructure IP (I^2P) [4] be developed for a systematic integration of numerous functional IP blocks to enable the widespread use of the SoC design methodology.

One of the major problems associated with future SOC designs arises from nonscalable global wire delays. Global wires carry signals across a chip, but these wires typically do not scale in length with technology scaling [5]. Though gate delays scale down with technology, global wire delays typically increase exponentially or, at best, linearly by inserting repeaters. Even after repeater insertion [5], the delay may exceed the limit of one clock cycle (often, multiple clock cycles). In ultra-deep submicron processes, 80 percent or more of the delay of critical paths will be due to interconnects [6], [7]. In fact, many large designs today use FIFO (first-in, first-out) buffers to synchronously propagate data over large distances to overcome this problem. This solution is ad hoc in nature. According to ITRS (2003 update) [8], "*Global synchronization becomes prohibitively costly due to process variability and power dissipation, and cross-chip signaling can no longer be achieved in a single clock cycle.*" Thus, system design must incorporate networking and distributed computation paradigms with communication structures designed first and then functional blocks integrated into the communication backbone.

The most frequently used on-chip interconnect architecture is the shared medium arbitrated bus, where all communication devices share the same transmission medium. The advantages of the shared-bus architecture are simple topology, low area cost, and extensibility. However, for a relatively long bus line, the intrinsic parasitic resistance and capacitance can be quite high. Moreover, every additional IP block connected to the bus adds to this parasitic capacitance, in turn causing increased propagation delay. As the bus length increases and/or the number of IP blocks increases, the associated delay in bit transfer over the bus may grow to become arbitrarily large and will eventually exceed the targeted clock period. This thus limits, in practice, the number of IP blocks that can be connected to a bus and thereby limits the system scalability [9]. One solution for such cases is to split the bus into multiple segments and introduce a hierarchical architecture [10], however, this is ad hoc in nature and has the inherent limitations of the bus-based systems. For SoCs consisting of tens or hundreds of IP blocks, bus-based interconnect architectures will lead to serious bottleneck problems as all attached devices must share the bandwidth of the bus [9].

To overcome the above-mentioned problems, several research groups, including our group, have advocated the use of a communication-centric approach to integrate IPs in complex SoCs. This new model allows the decoupling of the processing elements (i.e., the IPs) from the communication fabric (i.e., the network). The need for global synchronization

• *The authors are with the SOC Research Lab, Department of Electrical and Computer Engineering, University of British Columbia, 2356 Main Mall, Vancouver, BC, V6T 1Z4 Canada.*
E-mail: {parthap, grecuc, michaelj, ivanov, res}@ece.ubc.ca.

Manuscript received 28 May 2004; revised 21 Nov. 2004; accepted 8 Mar. 2004; published online 15 June 2005.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-0183-0504.

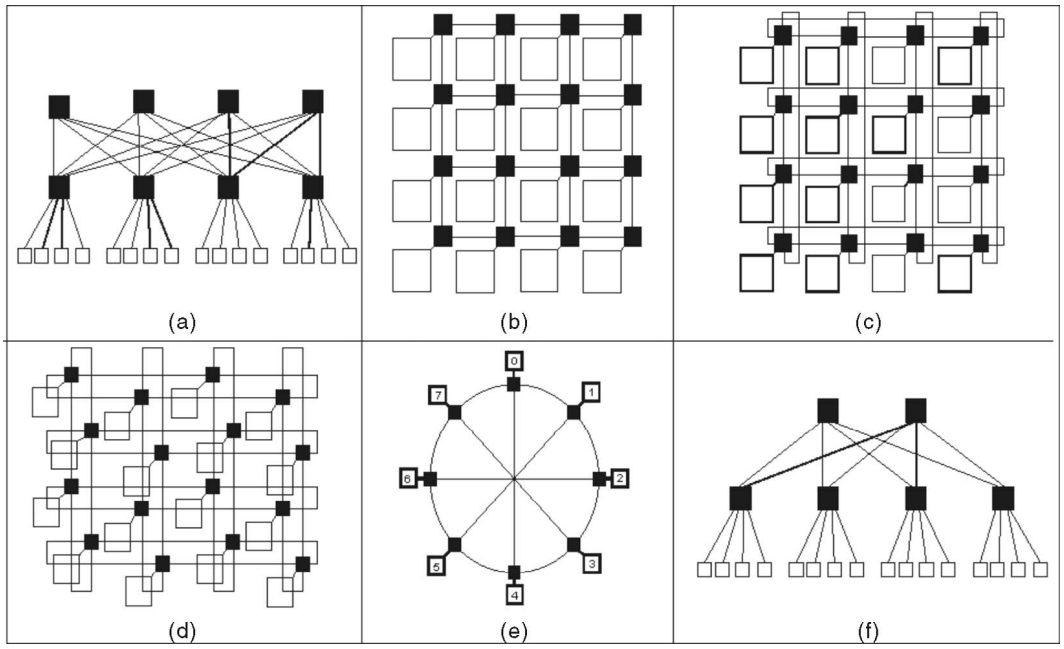

Fig. 1. NoC architectures. (a) SPIN, (b) CLICHÉ, (c) Torus, (d) Folded torus, (e) Octagon, (f) BFT.

can thereby disappear. This new approach employs explicit parallelism, exhibits modularity to minimize the use of global wires, and utilizes locality for power minimization [3].

In a network-centric approach, the communication between IPs can take place in the form of packets. We suggest that a network-on-chip (NoC) resemble the interconnect architecture of high-performance parallel computing systems. The common characteristic of these kinds of architectures is that the functional IP blocks communicate with each other with the help of intelligent switches. As such, the switches can be considered as infrastructure IPs (I^2Ps) [3] providing a robust data transfer medium for the functional IP modules.

A number of different interconnect architectures for MP-SoC platforms have been proposed. Their origins can be traced back to the field of parallel computing. However, a different set of constraints exists when adapting these architectures to the SoC design paradigm. High throughput and low latency are the desirable characteristics of a multiprocessing system. Instead of aiming strictly for speed, designers increasingly need to consider energy consumption constraints [3], especially in the SoC domain. None of the existing works on NoCs has compared the proposed interconnect architectures relative to throughput, latency, and energy. The main focus of this paper is the detailed comparative evaluation of a set of recently proposed NoC architectures with realistic traffic models. Our work furthers the body of knowledge associated with the design and analysis of such complex architectures and our analysis allows us to identify useful design trade-offs that are critical for the optimal development of integrated network-based designs.

2 Related Work

Current SoC designs predominantly use shared-medium bus-based functional interconnects to integrate IP blocks. There are mainly three types of commercial bus-based SoC interconnect specifications: ARM AMBA [11] bus, Wishbone [12], and IBM CoreConnect [13]. In [10], bus splitting has been proposed as an efficient solution for energy savings. A few on-chip micronetwork proposals for SoC integration can be found in the literature. Sonic's Silicon Backplane [14] is one example. In this architecture, IP blocks are connected to the communication fabric through specialized interfaces called agents. Each core communicates with an agent using the Open Core Protocol (OCP) [15]. Agents communicate with each other using time division-multiple access (TDMA) bus access schemes. These agents effectively decouple the IP cores from the communication network. MIPS Technologies has introduced an on-chip switch integrating IP blocks in an SoC [16]. The switch, called SoC-it, is intended to provide a high-performance link between a MIPS processor and multiple third-party IP cores. It is a central switch connecting different peripherals, but only in a point-to-point mode. None of these involves any specific interconnect architecture. Hence, we omit treating this approach any further in the remainder of this paper.

In the following, we briefly describe the different NoC architectures proposed recently. For the purpose of illustration, the functional IP blocks are denoted by white squares, while the infrastructure IPs (switches) are denoted by dark squares.

Guerrier and Greiner [17] have proposed a generic interconnect template called **SPIN** (Scalable, Programmable, Integrated Network) for on-chip packet switched interconnections, where a fat-tree architecture is used to interconnect IP blocks. In this fat tree, every node has four children and the parent is replicated four times at any level of the tree. Fig. 1a shows the basic SPIN architecture with $N = 16$ nodes, representing the number of functional IP blocks in the system. The size of the network grows as $(NlogN)/8$. The functional IP blocks reside at the leaves and the switches reside at the vertices. In this architecture, the number of switches converges to $S = \frac{3N}{4}$, where N is the system size in terms of number of functional IPs.

Kumar et al. [18] have proposed a mesh-based interconnect architecture called **CLICHÉ** (Chip-Level Integration of Communicating Heterogeneous Elements). This architecture consists of an $m \times n$ mesh of switches interconnecting computational resources (IPs) placed along with the switches, as shown in Fig. 1b in the particular case of 16 functional IP blocks. Every switch, except those at the edges, is connected to four neighboring switches and one IP block. In this case, the number of switches is equal to the number of IPs. The IPs and the switches are connected through communication channels. A channel consists of two unidirectional links between two switches or between a switch and a resource.

Dally and Towles [19] have proposed a 2D torus as an NoC architecture, shown in Fig. 1c. The Torus architecture is basically the same as a regular mesh [22]; the only difference is that the switches at the edges are connected to the switches at the opposite edge through wrap-around channels. Every switch has five ports, one connected to the local resource and the others connected to the closest neighboring switches. Again, the number of switches is $S = N$. The long end-around connections can yield excessive delays. However, this can be avoided by folding the torus, as shown in Fig. 1d [28]. This renders to a more suitable VLSI implementation and, consequently, in our further comparative analysis, we consider the Folded Torus of Fig. 1d.

Karim et al. [20] have proposed the **OCTAGON** MP-SoC architecture. Fig. 1e shows a basic octagon unit consisting of eight nodes and 12 bidirectional links. Each node is associated with a processing element and a switch. Communication between any pair of nodes takes at most two hops within the basic octagonal unit. For a system consisting of more than eight nodes, the octagon is extended to multidimensional space. The scaling strategy is as follows: Each octagon node is indexed by the 2-tuple (i, j), $i, j \in [0, 7]$. For each $i = I$, $I \in [0, 7]$, an octagon is constructed using nodes $\{(I, j), j \in [0, 7]\}$, which results in eight individual octagon structures. These octagons are then connected by linking the corresponding i nodes according to the octagon configuration. Each node (I, J) belongs to two octagons: one consisting of nodes $\{(I, j)j \in [0, 7]\}$ and the other consisting of nodes $\{(i, J)i \in [0, 7]\}$. Of course, this type of interconnection mechanism may significantly increase the wiring complexity.

We proposed an interconnect template following a **Butterfly Fat-Tree** (**BFT**) [21] architecture, as shown in Fig. 1f. In our network, the IPs are placed at the leaves and switches placed at the vertices. A pair of coordinates is used to label each node, (l, p), where l denotes a node's level and p denotes its position within that level. In general, at the lowest level, there are N functional IPs with addresses ranging from 0 to $(N - 1)$. The pair $(0, N)$ denotes the locations of IPs at that lowest level. Each switch, denoted by $S(l, p)$, has four child ports and two parent ports. The IPs are connected to $N/4$ switches at the first level. In the jth level of the tree, there are $N/2^{j+1}$ switches. The number of switches in the butterfly fat tree architecture converges to a constant independent of the number of levels. If we consider a 4-ary tree, as shown in Fig. 1f, with four down links corresponding to child ports and two up links corresponding to parent ports, then the total number of switches in level $j = 1$ is $N/4$. At each subsequent level, the number of required switches reduces by a factor of 2. In this way, the total number of switches approaches $S = \frac{N}{2}$, as N grows arbitrarily large [21].

3 Switching Methodologies

Switching techniques determine when and how internal switches connect their inputs to outputs and the time at which message components may be transferred along these paths. For uniformity, we apply the same approach for all NoC architectures. There are different types of switching techniques, namely, *Circuit Switching*, *Packet Switching*, and *Wormhole Switching* [22].

In *circuit switching*, a physical path from source to destination is reserved prior to the transmission of the data. The path is held until all the data has been transmitted. The advantage of this approach is that the network bandwidth is reserved for the entire duration of the data. However, valuable resources are also tied up for the duration of the transmitted data and the set up of an end-to-end path causes unnecessary delays.

In *packet switching*, data is divided into fixed-length blocks called packets and, instead of establishing a path before sending any data, whenever the source has a packet to be sent, it transmits the data. The need for storing entire packets in a switch in case of conventional packet switching makes the buffer requirement high in these cases. In an SoC environment, the requirement is that switches should not consume a large fraction of silicon area compared to the IP blocks.

In *wormhole switching*, the packets are divided into fixed length flow control units (*flits*) and the input and output buffers are expected to store only a few flits. As a result, the buffer space requirement in the switches can be small compared to that generally required for packet switching. Thus, using a *wormhole switching* technique, the switches will be small and compact. The first flit, i.e., *header flit*, of a packet contains routing information. Header flit decoding enables the switches to establish the path and subsequent flits simply follow this path in a pipelined fashion. As a result, each incoming data flit of a message packet is simply forwarded along the same output channel as the preceding data flit and no packet reordering is required at destinations. If a certain flit faces a busy channel, subsequent flits also have to wait at their current locations.

One drawback of this simple wormhole switching method is that the transmission of distinct messages cannot be interleaved or multiplexed over a physical channel. Messages must cross the channel in their entirety before the channel can be used by another message. This will decrease channel utilization if a flit from a given packet is blocked in a buffer. By introducing virtual channels [22] in the input and output ports, we can increase channel utility considerably. If a flit belonging to a particular packet is blocked in one of the virtual channels, then flits of alternate packets can use the other virtual channel buffers and, ultimately, the physical channel. The canonical architecture of a switch having virtual channels is shown in Fig. 2.

4 Performance Metrics

To compare and contrast different NoC architectures, a standard set of performance metrics can be used [22], [27]. For example, it is desirable that an MP-SoC interconnect

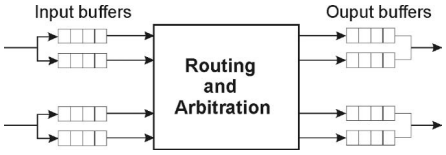

Fig. 2. Virtual-channel switch.

architecture exhibits high throughput, low latency, energy efficiency, and low area overhead. In today's power constrained environments, it is increasingly critical to be able to identify the most energy efficient architectures and to be able to quantify the energy-performance trade-offs [3]. Generally, the additional area overhead due to the infrastructure IPs should be reasonably small. We now describe these metrics in more detail.

4.1 Message Throughput

Typically, the performance of a digital communication network is characterized by its bandwidth in bits/sec. However, we are more concerned here with the rate that message traffic can be sent across the network and, so, *throughput* is a more appropriate metric. Throughput can be defined in a variety of different ways depending on the specifics of the implementation. For message passing systems, we can define message throughput, TP, as follows:

$$TP = \frac{(Total\ messages\ completed) \times (Message\ length)}{(Number\ of\ IP\ blocks) \times (Total\ time)}, \tag{1}$$

where *Total messages completed* refers to the number of whole messages that successfully arrive at their destination IPs, *Message length* is measured in flits, *Number of IP blocks* is the number of functional IP blocks involved in the communication, and *Total time* is the time (in clock cycles) that elapses between the occurrence of the first message generation and the last message reception. Thus, message throughput is measured as the fraction of the maximum load that the network is capable of physically handling. An overall throughput of $TP = 1$ corresponds to all end nodes receiving one flit every cycle. Accordingly, throughput is measured in flits/cycle/IP. Throughput signifies the maximum value of the accepted traffic and it is related to the peak data rate sustainable by the system.

4.2 Transport Latency

Transport latency is defined as the time (in clock cycles) that elapses from between the occurrence of a message header injection into the network at the source node and the occurrence of a tail flit reception at the destination node [21]. We refer to this simply as *latency* in the remainder of this paper. In order to reach the destination node from some starting source node, flits must travel through a path consisting of a set of switches and interconnect, called *stages*. Depending on the source/destination pair and the routing algorithm, each message may have a different latency. There is also some overhead in the source and destination that also contributes to the overall latency. Therefore, for a given message i, the latency L_i is:

$$L_i = sender\ overhead + transport\ latency + receiver\ overhead.$$

We use the average latency as a performance metric in our evaluation methodology. Let P be the total number of messages reaching their destination IPs and let L_i be the latency of each message i, where i ranges from 1 to P. The average latency, L_{avg}, is then calculated according to the following:

$$L_{avg} = \frac{\sum_l^P L_i}{P}. \tag{2}$$

4.3 Energy

When flits travel on the interconnection network, both the interswitch wires and the logic gates in the switches toggle and this will result in energy dissipation. Here, we are concerned with the dynamic energy dissipation caused by the communication process in the network. The flits from the source nodes need to traverse multiple hops consisting of switches and wires to reach destinations. Consequently, we determine the energy dissipated by the flits in each interconnect and switch hop. The energy per flit per hop is given by

$$E_{hop} = E_{switch} + E_{interconnect}, \tag{3}$$

where E_{switch} and $E_{interconnect}$ depend on the total capacitances and signal activity of the switch and each section of interconnect wire, respectively. They are determined as follows:

$$E_{switch} = \alpha_{switch} C_{switch} V^2, \tag{4}$$

$$E_{interconnect} = \alpha_{interconnect} C_{interconnect} V^2. \tag{5}$$

$\alpha_{switch}, \alpha_{interconnect}$ and $C_{switch}, C_{interconnect}$ are the signal activities and the total capacitances of the switches and wire segments, respectively. The energy dissipated in transporting a packet consisting of n flits over h hops can be calculated as

$$E_{packet} = n \sum_{j=1}^{h} E_{hop,j}. \tag{6}$$

Let P be the total number of packets transported, and let E_{packet} be the energy dissipated by the ith packet, where i ranges from 1 to P. The average energy per packet, $\overline{E}_{packet}$, is then calculated according to the following equation:

$$\overline{E}_{packet} = \frac{\sum_{i=1}^{P} E_{packet_i}}{P} = \frac{\sum_{i=1}^{P} \left(n_i \sum_{j=1}^{h_i} E_{hop,j} \right)}{P}. \tag{7}$$

The parameters α_{switch} and $\alpha_{interconnect}$ are those that capture the fact that the signal activities in the switches and the interconnect segments will be data-dependent, e.g., there may be long sequences of 1s or 0s that will not cause any transitions. Any of the different low-power coding techniques [29] aimed at minimizing the number of transitions can be applied to any of the topologies described here. For the sake of simplicity and without loss of generality, we do not consider any specialized coding techniques in our analysis.

4.4 Area Requirements

To evaluate the feasibility of these interconnect schemes, we consider their respective silicon area requirements. As the switches form an integral part of the active components, the

infrastructure, it is important to determine the amount of relative silicon area they consume. The switches have two main components: the storage buffer and logic to implement routing and flow control. The storage buffers are the FIFOs at the inputs and outputs of the switch. Another source of silicon area overhead arises from the interswitch wires, which, depending on their lengths, may have to be buffered through repeater insertion to keep the interswitch delay within one clock cycle [9]. Consequently, this additional buffer area should also be taken into account. Another important factor that needs to be considered when analyzing the area overhead is the wiring layout. One of the main advantages of the NoC design methodology is the division of long global wires into smaller segments, characterized by propagation times that are compatible with the clock cycle budget [30]. All the NoC architectures considered here achieve this as a result of their inherent interconnect structure. But, the segmented wire lengths will vary from one topology to another. Consequently, for each architecture, the layout of interswitch wire segments presents different degrees of complexity. Architectures that possess longer interswitch wires will generally create more routing challenges, compared to those possessing only shorter wire segments. Long wires can block wiring channels, forcing the use of additional metal layers and causing other wires to become longer. The determination of the distribution of interswitch wire lengths can give a first-order indication of the overall wiring complexity.

4.5 Evaluation Methodology

In order to carry out a consistent comparison, we developed a simulator employing flit-level event-driven wormhole routing to study the characteristics of the communication-centric parameters of the interconnect infrastructures. In our experiments, the traffic injected by the functional IP blocks followed Poisson [31] and self-similar distributions [31]. In the past, a Poisson distributed injection rate was frequently used when characterizing performance of multiprocessor platforms [32]. However, the self-similar distribution was found to be a better match to real-world SoC scenarios [33]. Each simulation was initially run for 1,000 cycles to allow transient effects to stabilize and, subsequently, it was executed for 20,000 cycles. Using a flit counter at the destinations, we obtain the throughput as the number of flits reaching each destination per unit time. To calculate average latency and energy, we associate an ordered pair, (L_{switch}, E_{switch}), with each switch and an ordered pair, $(L_{interconnect}, E_{interconnect})$, with each interconnect segment, where $L_{switch}, L_{interconnect}$ and $E_{switch}, E_{interconnect}$ denote the delays and energy dissipated in the switch and interconnect, respectively. The average latency and energy dissipation are calculated according to (2) and (7).

To estimate the silicon area consumed by the switches, we developed their VHDL models and synthesized them using a fully static, standard cell-based approach for a 0.13 μm CMOS technology library. Starting from this initial estimation, by using an ITRS (**International Technology Roadmap for Semiconductors**) suggested scaling factor of 0.7, we can project the area overhead in future technology nodes.

5 Infrastructure IP Design Considerations

One common characteristic of the communication-centric architectures described in this paper is that the functional IP blocks communicate with each other with the help of intelligent switches. The switches provide a robust data transport medium for the functional IP modules. To ensure the consistency of the comparisons we later make in this paper, we assume that similar types of switching and routing circuits are used in all cases. These designs are now described in more detail.

5.1 Switch Architecture

The different components of the switch port are shown in Fig. 3. It mainly consists of input/output FIFO buffers, input/output arbiters, one-of-four MUX and DEMUX units, and a routing block. In order to have a considerably high throughput, we use a virtual channel switch, where each port of the switch has multiple parallel buffers [22].

Each physical input port has more than one virtual channel, uniquely identified by its virtual channel identifier (VCID). Flits may simultaneously arrive at more than one virtual channel. As a result, an arbitration mechanism is necessary to allow only one virtual channel to access a single physical port. Let there be m virtual channels corresponding to each input port; we need an $m:1$ arbiter at the input. Similarly, flits from more than one input port may simultaneously try to access a particular output port. If k is the number of ports in a switch, then we need a $(k-1):1$ arbiter at each output port. The routing logic block determines the output port to be taken by an incoming flit.

The operation of the switch consists of one or more processes, depending on the nature of the flit. In the case of a header flit, the processing sequence is: 1) *input arbitration*, 2) *routing*, and 3) *output arbitration*. In the case of *body* flits, switch traversal replaces the routing process since the routing decision based on the header information is maintained for the subsequent body flits. The basic functionality of the input/output arbitration blocks does not vary from one architecture to another. The design of the routing hardware depends on the specific topology and routing algorithm adopted. In order to make the routing logic simple, fast, and compact, we follow different forms of deterministic routing [22]. In our routing schemes, we use distributed source routing, i.e., the source node determines only its neighboring nodes that are involved in message delivery. For the tree-based architectures (SPIN and BFT), the routing algorithm applied is the least common ancestor (LCA) and, for CLICHÉ and Folded Torus, we apply the *e*-Cube (dimensional) routing [21]. In the case of Octagon, we adopt the hierarchical address-based routing as proposed in [19]. The corresponding routing blocks have been implemented for all the above-mentioned cases.

The arbiter circuit essentially consists of a priority matrix, which stores the priorities [23] of the requesters, and grant generation circuits used to grant resources to requesters. The matrix arbiter stores priorities between n requesters in a binary *n-by-n* matrix. Each matrix element $[i, j]$ records the binary priority between each pair of inputs. For example, suppose requester i has a higher priority than requester j, then the matrix element $[i, j]$ will be set to 1, while the corresponding matrix element $[j, i]$ will be 0. A requester will be granted the resource if no other higher priority requester is bidding for the same resource. Once a

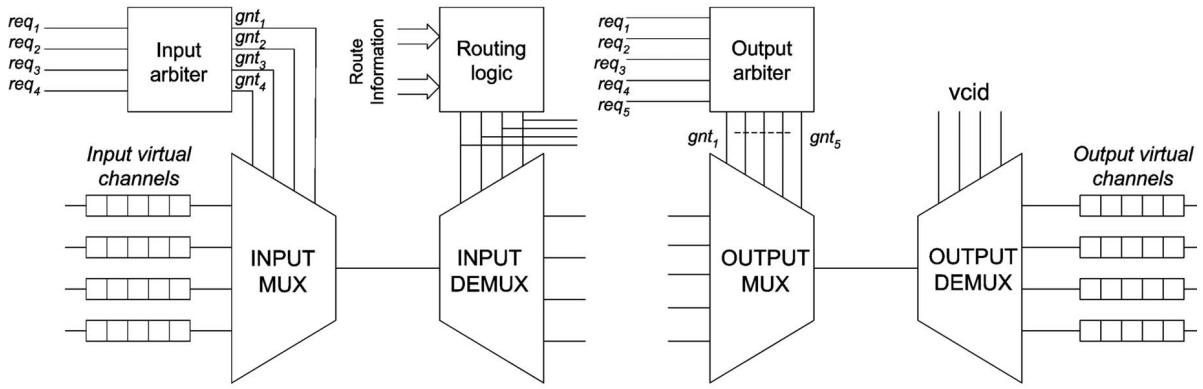

Fig. 3. Block diagram of a switch port.

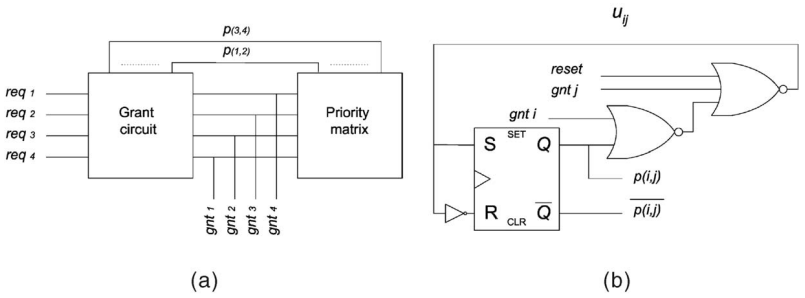

Fig. 4. (a) Block diagram of an arbiter; (b) one element of the priority matrix.

requester succeeds in being granted a resource, its priority is updated and set to be the lowest among all requesters. A block diagram of the arbiter and one element of the priority matrix circuit is shown in Fig. 4.

The FIFO buffers are also critical components of the switch. Their operating speed should be high enough not to become a bottleneck in a high-speed network. More specifically, the switches at level one need to be interfaced with the SoC's constituent IP blocks. Hence, the switches should be able to receive and transmit data at the rated speed of the corresponding IPs. Furthermore, the FIFOs should be able to operate with different read and write clocks as the SoC's constituents IPs are expected to generally operate at different frequencies. Instead of using separate counters to implement read and write pointers, two tokens are circulated among the FIFO cells to implement read and write operations [24]. A FIFO cell can be read from or written into only if it holds the corresponding token. After a token is used in a given cell, it is subsequently passed on to the adjacent cell.

5.2 Virtual Channel Allocation

The virtual channel allocation determines which output virtual channel is taken by a message at each of the intermediate switch nodes. Each switch input port has a separate queue buffer corresponding to the virtual channels. When a flit first arrives at an input port, its *type* is decoded. If it is a header flit, then, according to its VCID field, it is stored in the corresponding virtual channel buffer. The routing logic determines the output port to be taken by this flit and assigns the incoming flit to an available output virtual channel. The VCID of the flit is modified accordingly. When the subsequent body flits arrive, they are queued into the buffer of the input virtual channel and subsequently inherit the particular output virtual channel reserved by the header. Instead of reserving output ports for the entire duration of a packet, the switch allocates output ports on a flit-by-flit basis.

5.3 Network Interfacing

The success of the NoC design paradigm relies greatly on the standardization of the interfaces between IP cores and the interconnection fabric. The Open Core Protocol (OCP) [15] is an interface standard receiving wide industrial and academic acceptance. Using a standard interface should not impact the methodologies for IP core development. In fact, IP cores wrapped with a standard interface like the OCP interface will exhibit a higher reusability and greatly simplify the task of system integration. The network interface will have two functions:

1. injecting/absorbing the flits leaving/arriving at the functional IP blocks;
2. packetizing/depacketizing the signals coming from/reaching to OCP compatible cores in form of messages/flits.

As shown in Fig. 5, for a core having both master and slave interfaces, the OCP compliant signals coming out of the

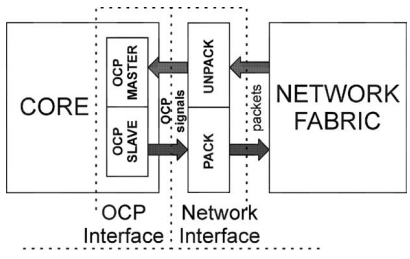

Fig. 5. Interfacing of IP cores with the network fabric.

TABLE 1
Simulation Parameters

Topology	Message Length (Flits)	Buffer Depth (Flits)[1]	Port Count
SPIN	64	2	8
CLICHÉ	64	2	5
FOLDED TORUS	64	2	5
OCTAGON	64	2	3/6[2]
BFT	64	2	6

[1] *The average message latency decreases when buffer size increases [22]. According to [22], the effect of buffer size on performance is small. Consequently, to avoid excessive silicon area consumption in our switch design, here we considered the buffer depths to be equal to two flits.*
[2] *The bridge nodes connecting two adjacent OCTAGONS have six ports.*

functional IP blocks are packetized by a second interface, which sits between the OCP instances and the communication fabric.

6 Experimental Results and Analysis

We applied our evaluation methodology to all the proposed NoC architectures described earlier in this paper. The wormhole routing simulator was used to compare and contrast the NoC topologies in terms of throughput and latency. In this simulator, the user may choose between uniform and localized traffic patterns for the packets. There are options of using both Poisson and self-similar message injection distributions. Self-similar traffic has been observed in the bursty traffic between on-chip modules in typical MPEG-2 video applications [33] and networking applications [32]. It has been shown that modeling of self-similar traffic can be obtained by aggregating a large number of ON-OFF message sources [32]. The length of time each message spends in either the ON or the OFF state should be selected according to a distribution which exhibits long-range dependence. The Pareto distribution ($(\mathrm{F(x)} = 1 - \mathrm{x}^{-\alpha}$, with $1 < \alpha < 2$) has been found to fit well to this kind of traffic. A packet train remains in the ON state for $t_{ON} = (1-r)^{\frac{-1}{\alpha_{ON}}}$ and in the OFF state for $t_{OFF} = (1-r)^{\frac{-1}{\alpha_{OFF}}}$, where r is a random number uniformly distributed between 0 and 1, $\alpha_{ON} = 1.9$, and $\alpha_{OFF} = 1.25$ [31]. The destination IP selection depends on the traffic pattern adopted. The simulator is capable of handling variable message length. Message lengths may vary depending on the application. On the other hand, message length and buffer depth are strongly correlated. In an SoC environment, buffer depth is of extreme importance as it adds to the silicon area overhead due to the switches. In addition, switch parameters can also be specified. These include input/output port buffer depths (in flits), number of ports, and the number of virtual channels per switch port. Messages arriving at destinations are immediately consumed at the rate of one flit per time step, i.e., no blocking is encountered at the destinations. All resource contention is handled without bias in the sense that granting of resources to packets is done on a first come, first-serve basis.

The energy dissipation of NoC fabrics arise from two different sources: 1) the switch blocks, which include the buffers, and 2) interswitch wire segments. To study the energy efficiency of the interconnect architectures, we determine the energy dissipated in each switch, E_{switch}, by running Synopsys Prime Power on the gate-level netlist of the switch blocks, including the FIFO buffers. Our energy estimation methodology involved feeding a large set of data patterns to the switch blocks. Through functional simulation using Synopsys Prime Power, the average values for the activity factors were determined. The experimental data set included long sequences of 1s and 0s to account for the possible cases where low transition activity data were to be transported. To determine interconnect energy, $E_{interconnect}$,

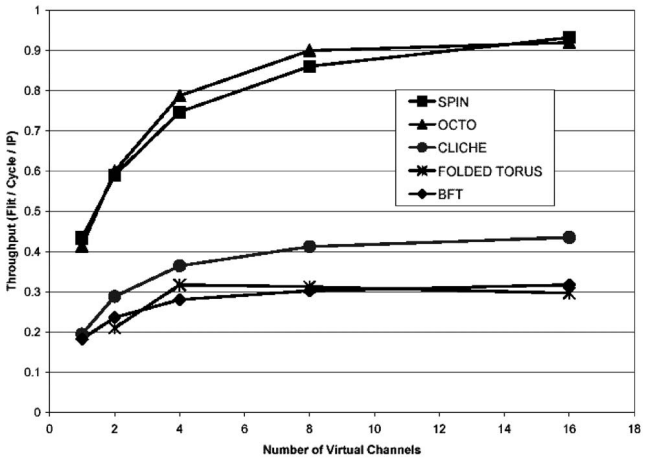

Fig. 6. Variation of throughput under spatially uniform traffic distribution.

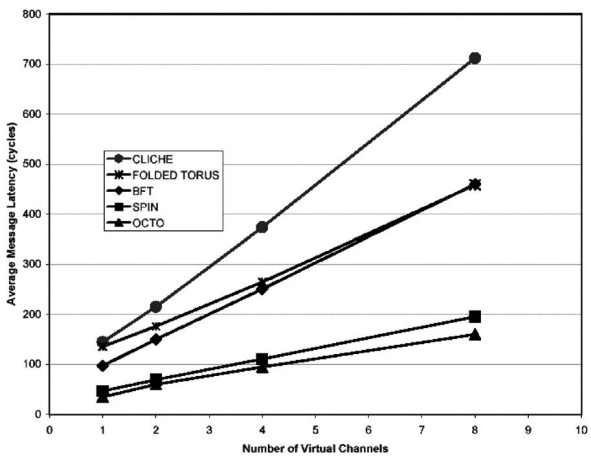

Fig. 7. Variation of latency with virtual channels.

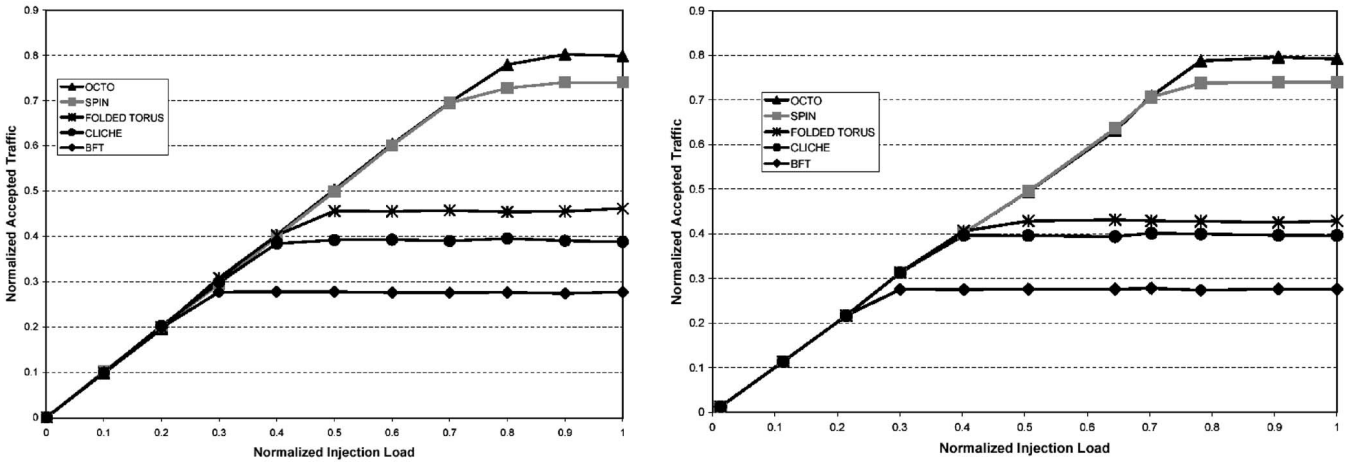

Fig. 8. Variation of accepted traffic with injection load. (a) Poisson. (b) Self-similar.

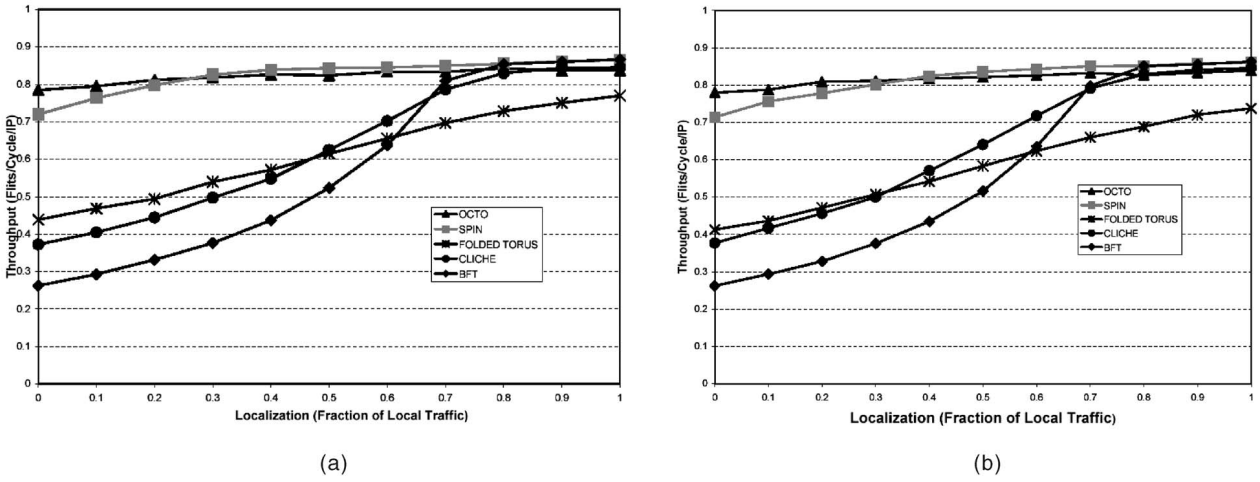

Fig. 9. Variation of throughput under localized traffic (number of vc = 4). (a) Poisson. (b) Self-similar.

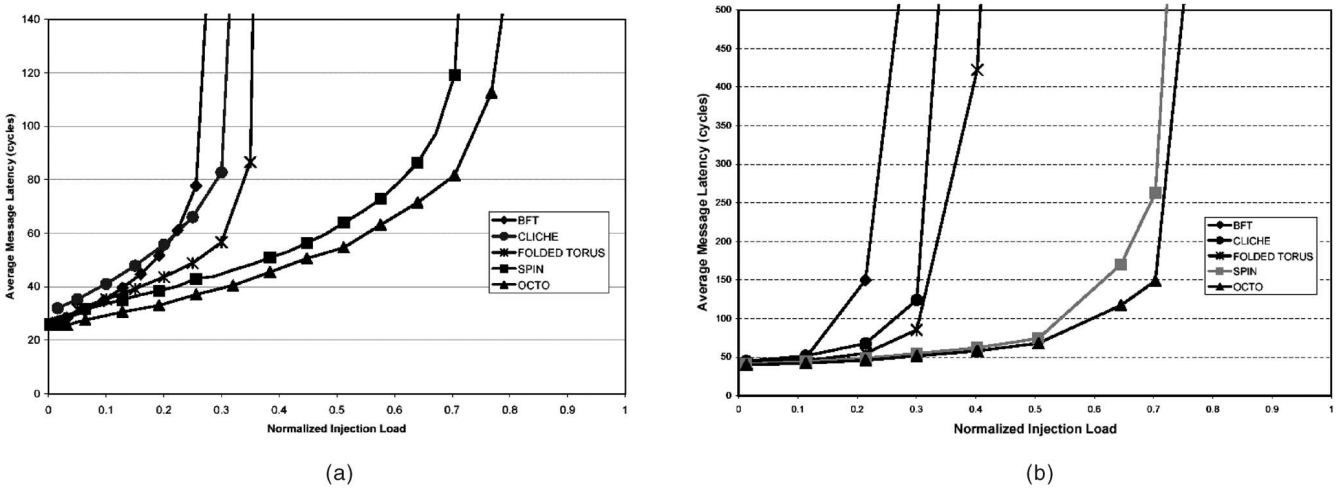

Fig. 10. Latency variation with injection load for spatially uniform traffic distribution. (a) Poisson. (b) Self-similar.

the capacitance of each interconnect stage, $C_{interconnect}$, is calculated taking into account the specific layout of each topology. $C_{interconnect}$ can be estimated according to the following expression:

$$C_{interconnect} = C_{wire} \cdot w_{a+1,a} + n \cdot m \cdot (C_G + C_J), \qquad (8)$$

where C_{wire} is the wire capacitance per unit length and $w_{a+1,a}$ is the wire length between two consecutive switches; C_G and C_J are the gate and junction capacitance of a minimum size

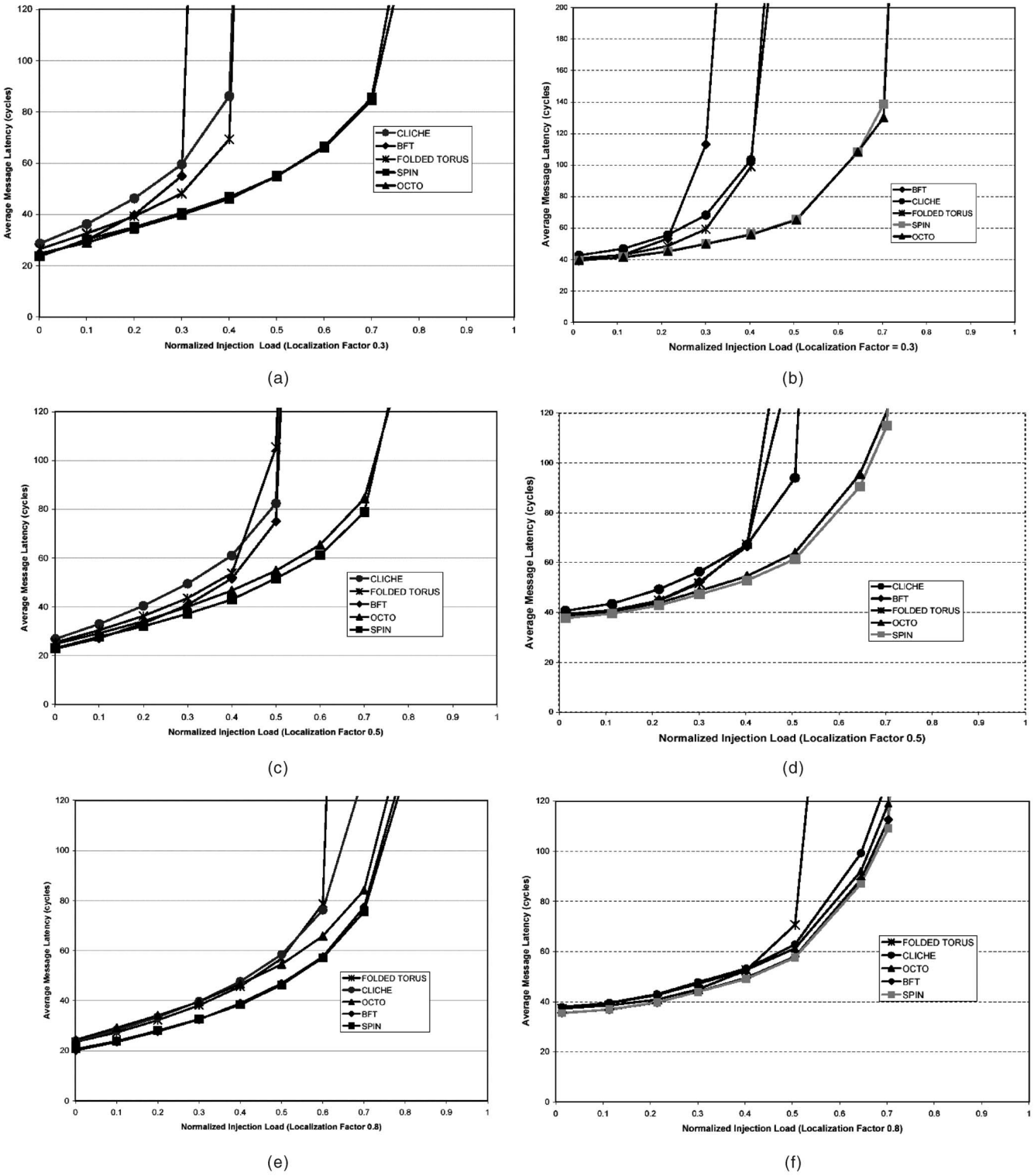

Fig. 11. Top: Latency variation with injection load (localization factor = 0.3). (a) Poisson. (b) Self-similar. Middle: Latency variation with injection load (localization factor = 0.5). (c) Poisson. (d) Self-similar. Bottom: Latency variation with injection load (localization factor = 0.8). (e) Poisson. (f) Self-similar.

inverter, respectively, n denotes the number of inverters (when buffer insertion is needed) in a particular interswitch wire segment, and m is their corresponding size with respect to a minimum size inverter. While calculating C_{wire}, we have considered the worst-case switching scenario, where the two adjacent wires switch in the opposite direction of the signal line simultaneously [6].

In all the subsequent experiments, we consider each system to be consisting of 256 functional IP blocks, i.e., $N = 256$. Table 1 summarizes the simulation parameters.

6.1 Throughput and Latency

We now compare the throughput and latency characteristics of the various NoC architectures. The throughput of the communication infrastructure generally depends on the traffic pattern. Fig. 6 shows the variation of throughput with the number of virtual channels for all the topologies, determined through simulation using (1). Measuring throughput under uniform spatial distribution assumptions is an accepted metric [22] for evaluating parallel systems. Throughput is the maximum traffic accepted by the network and it relates to the peak data rate sustainable by the system. The accepted traffic depends on the rate at which the functional IP blocks are injecting data into the network. Ideally, accepted traffic should increase linearly with this injection load. However, due to the limitation of routing resources (switches and interconnect wires), accepted traffic will saturate at a certain value of the injection load. Similarly to the throughput, the unit of measure for injection load is also flits/cycle/IP. For both Poisson and self-similar injection rates, the variation of throughput with virtual channels has similar characteristics. From Fig. 6, when the number of virtual channels is increased beyond four, there is a trend toward throughput saturation. However, each additional virtual channel implies an increased silicon area.

Fig. 7 shows the variation of latency with the number of virtual channels. The average message latency depends on the number of virtual channels and injection load. In this case, the average latency generally increases with the number of virtual channels. To keep the latency low while simultaneously maintaining a considerable throughput, the number of virtual channels is constrained to four in the design of the switches. Consequently, a system with four virtual channels strikes an appropriate balance between high throughput, low latency, and conservation of silicon area. This result is consistent with previous research on the optimal number of virtual channels [36] and, in part, validates the modeling and simulation approach used to generate the results in this paper.

The plots in Fig. 6 also indicate that, under the uniform traffic assumption, BFT, CLICHÉ, and Folded Torus provide a lower throughput than do SPIN and Octagon. This happens due to the fact that SPIN and Octagon have more links between a source and a destination pair than do the others.

The role of injection load on the accepted traffic was also studied and shown in Fig. 8. We observe that the accepted traffic increases linearly with the injection load up to the throughput saturation point. Fig. 8b shows that self-similar traffic saturates the networks at slightly lower average data rates.

While these results are as one would expect, the assumption of spatial uniformity of traffic is not very realistic in an SoC environment since different functions will be mapped to different parts of the SoC and they will exhibit highly localized patterns. Hence, we studied the effect of traffic localization on throughput for both types of injection processes and considered the illustrative case of spatial localization where local messages travel from a source to the set of the nearest destinations. In the case of BFT and SPIN, localized traffic is constrained within a cluster consisting of a single subtree, while, in the case of CLICHÉ and Folded Torus, it is constrained within the four destinations placed at the shortest Manhattan distance [14].

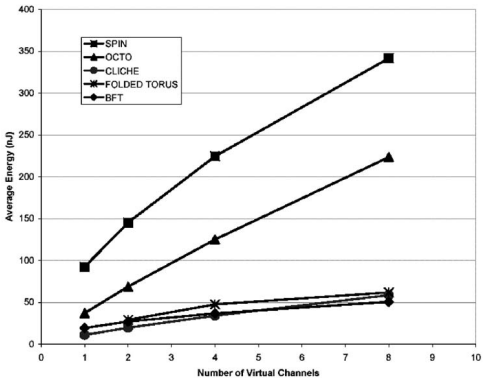

Fig. 12. Average energy dissipation per packet.

In the case of Octagon, the local traffic is the traffic constrained within the basic group of eight IP blocks. We define the *localization factor* as the ratio of local traffic to total traffic. For example, if the localization factor is 0.3, then 30 percent of the traffic generated by an IP occurs within its cluster, while the rest of the traffic is randomly distributed in the remainder of the entire SoC.

Fig. 9 shows the effect of traffic localization on throughput for all the topologies. We assumed that the number of virtual channels (vc) is four, based on the previously described experiments. Localization of traffic does not have much impact on SPIN and Octagon, but it enhances the throughput of BFT, CLICHÉ, and Folded Torus considerably. Though SPIN and Octagon have very high throughput for the uniformly distributed traffic, they lack the ability to exploit the traffic localization inherent in SoC architectures.

In Fig. 10, we show the variation of latency with injection load in the case of both Poisson and self-similar distributions for uniform traffic. The injection load directly affects the average message latency. As the injection load approaches the accepted traffic (throughput) limit, there will be more message contention and latency will increase. At the limit, latency increases exponentially when the injection load reaches the saturation point. Consequently, the desirable point of operation for the system should be well below network saturation. The self-similar traffic distribution yields higher average message latency, principally due to its bursty nature. We considered the effect of traffic localization on the latency. Variation of latency with localization factors of 0.3, 0.5, and 0.8 is shown in Fig. 11a, Fig. 11b, Fig. 11c, Fig. 11d, Fig. 11e and Fig. 11f. One important effect of localization on the latency characteristic is that it allows a higher injection load. Consequently, more traffic can be processed without the network being saturated. Eventually, this will enhance the overall datarate of the system.

It is seen from Fig. 11 that, similar to the case of throughput characteristics, traffic localization does not have significant impact on the latency variations for SPIN and Octagon.

6.2 Energy Dissipation

While evaluating the feasibility of an interconnect infrastructure, its energy dissipation profile must be considered as it can be a significant portion of the overall SoC energy budget. The metric for comparing the NoC architecture

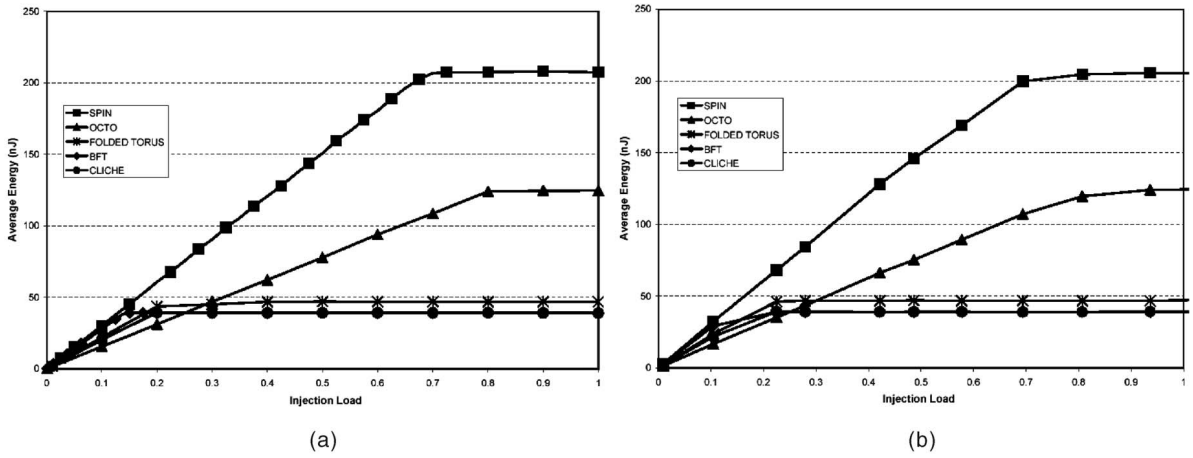

Fig. 13. Energy dissipation profile for uniform traffic. (a) Poisson. (b) Self-similar.

with respect to the energy dissipation is the average dynamic energy dissipated when a packet moves between a pair of source and destination IP blocks. This average energy dissipation, in turn, depends on the number of virtual channels and injection load. Fig. 12 shows the variation of average energy dissipation per packet as a function of the number of virtual channels (the same for both Poisson and self-similar traffic) assuming the networks to be operated at the peak sustainable data rate. We can observe that the energy dissipation increases linearly with the number of virtual channels for all the architectures. Consequently, a system with four virtual channels per physical link will give reasonably low energy dissipation without compromising throughput.

The effect of injection load on the energy dissipation for a uniform traffic distribution for both Poisson and self-similar injection process is shown in Fig. 13. Similar to the nature of accepted traffic variations, the energy dissipation profiles show a saturating characteristic occurring when the injection load reaches the throughput limit. Beyond saturation, no additional messages can be injected successfully into the system and, consequently, no additional energy is dissipated.

We also consider the effect of traffic localization on the energy dissipation profile for all the NoC architectures. Fig. 14a, Fig. 14b, Fig. 14c, Fig. 14d, Fig. 14e, and Fig. 14f show the energy dissipation profile for localization factors of 0.3, 0.5, and 0.8, respectively. The benefits of traffic localization are evident from these figures: Increasing the amount of traffic localization causes more messages to be injected without increasing the average energy dissipation. This happens due to the fact that, on the average, messages will traverse fewer stages in the case of a greater amount of localization. Consequently, the functional mapping should be performed so as to exploit the advantages of spatial locality, i.e., the blocks that communicate more frequently should be placed close to each other. This will reduce the use of long global paths and the energy dissipation.

From Figs. 13 and 14, we can infer that the architectures with a higher degree of connectivity like SPIN and Octagon have greater average energy dissipation at saturation than the others, though they provide higher throughput and lower latency on the average.

6.3 Area Overhead

In the NoC design paradigm, the silicon area overhead arises due to the presence of the switches, the interswitch repeaters, and the interfaces between the functional IP blocks and the network.

Regardless of the specific topology used by the interconnect infrastructure, each functional IP blocks needs to have the OCP-IP interface. Consequently, in our analysis, we consider the whole interfacing circuitry a part of the functional IP blocks.

From our detailed circuit level design and synthesis, we deduce that, within a switch, the buffer area significantly dominates the logic [25]. The buffer area, in turn, largely depends on the number of virtual channels and the flit length. In a networked SoC, IPs can be divided into two groups, *functional* and *infrastructure* IPs (switches). Hence, the distribution of functional IPs and I^2Ps depends on their respective sizes and interconnect topology. Letting A_{FIP} denote the area of the functional IP blocks and A_{I^2P} denote the area of the switches, then

$$Area_{chip} = N_1 \cdot A_{FIP} + N_2 \cdot A_{I^2P}, \tag{9}$$

where N_1 and N_2 are the number of functional and infrastructure IPs, respectively, and $Area_{chip}$ is the total area of the SoC under consideration. For a BFT, $N_1 = 2N_2$, for SPIN architecture, $N_1 = \frac{4}{3}N_2$, and, for all the others, $N_1 = N_2$. These numbers help to determine the distribution of functional and infrastructure IP blocks in an SoC. Through RTL level design and synthesis, we found that the switches consist of approximately 30K gates,[3] while the OCP-IP interface accounted for around 1,400 gates. Using (9) and the constraints on N_1 and N_2, we can determine the distribution of functional and infrastructure IP blocks in all the topologies. We consider the case where functional IP blocks are constrained to be of the order of 100K gates, as suggested in [6]. Table 2 shows the maximum number of 100K blocks that can be integrated within an SoC in different technology nodes [9].

The distribution of functional and infrastructure IP blocks is indicated in Table 3.

Under these assumptions, we determined the silicon area consumed by the switch blocks for all the architectures.

3. Here, we consider a 2-input NAND structure as a reference gate.

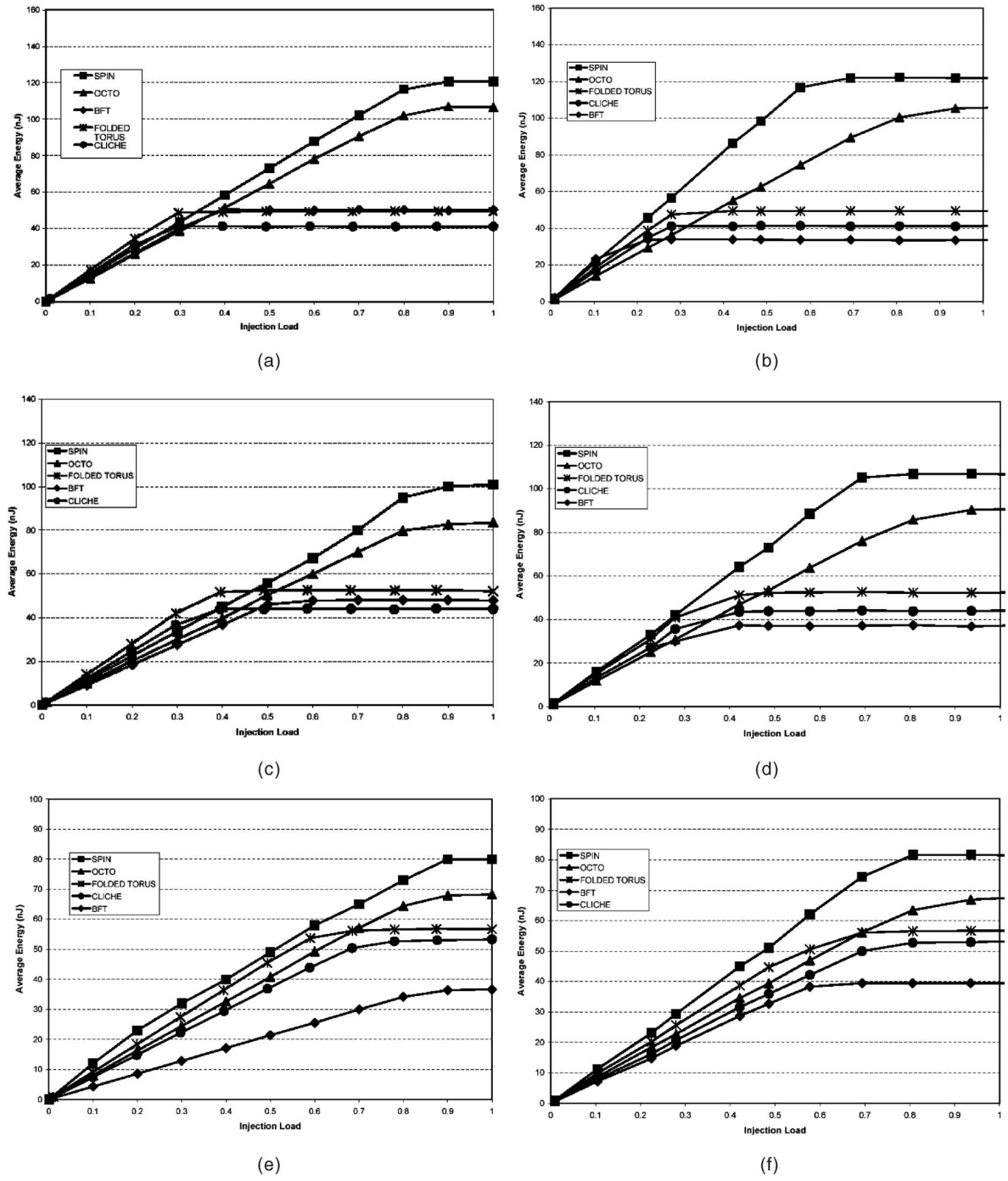

Fig. 14. Top: Energy dissipation profile for localized traffic (localization factor = 0.3). (a) Poisson. (b) Self-similar. Middle: Energy dissipation profile for localized traffic (localization factor =0 .5). (c) Poisson. (d) Self-similar. Bottom: Energy dissipation profile for localized traffic (localization factor = 0.8). (a) Poisson. (b) Self-similar.

TABLE 2
Maximum Number of 100K IP Blocks in Different Technology Nodes

Technology Node	130 nm	90nm	65nm	45nm	32nm
Max. no. of 100 k gates IP Blocks	500	1000	2500	7500	10000

The other factors contributing to the area overhead are the interswitch repeaters. The wire length between switches in the BFT and SPIN architectures depends on the levels of the switches. Consequently, to keep the interswitch wire delay within one clock cycle, some of them need to be buffered [8]. In CLICHÉ and Folded Torus, all the interswitch wires are of equal length and their delay is always within one clock cycle [8]. Therefore, no repeater insertion is required. In Octagon, the interswitch wires connecting functional IP blocks in disjoint octagon units [19] need to be buffered.

TABLE 3
Distribution of Functional and Infrastructure IP Blocks

Technology node	No. of Functional IPs			No. of I²Ps		
	BFT	SPIN	CLICHÉ / FOLDED TORUS/ OCTAGON	BFT	SPIN	CLICHÉ / FOLDED TORUS/ OCTAGON
130 nm	428	400	375	214	300	375
90 nm	856	800	750	428	600	750
65 nm	2142	2000	1875	1071	1500	1875
45 nm	6426	6000	5625	3213	4500	5625
32 nm	8568	8000	7500	4284	6000	7500

The silicon area overhead in different technology nodes can be estimated for all the interconnect architectures as the sum of the area due to the switches ($Area_{I^2P}$) and repeaters ($Area_{repeaters}$):

$$Area_{overhead} = Area_{I^2P} + Area_{repeaters}. \quad (10)$$

Fig. 15 reports the silicon area overhead for all the platforms under consideration across different technology nodes, assuming a die size of $20mm \times 20mm$ and that each functional IP block consists of 100K equivalent 2-input NAND gates.

From Fig. 15, we see that both SPIN and Octagon have a considerably higher silicon area overhead. This happens due to the fact that both these architectures provide a higher degree of connectivity. The percentage of silicon area overhead for different platforms increases slightly with technology scaling. However, the relative area overhead between any two platforms remains almost unchanged.

6.4 Wiring Complexity

The complexity of the wire area estimation problem involves determination of the longest wire segments that may arise in each architecture and their distribution. The long wire segments block wiring channels and force other wires to become longer. From our experience, there will be additional area consumed by the wires than what is predicted by the first order analysis. Assuming this kind of overhead, our aim is to estimate the distribution of wire lengths in all the interconnect architectures under consideration. In an NoC environment, the interswitch wire segments are the longest on-chip wires except for clock, power, and ground wires [34]. Due to the structured nature of NoC-based interconnects, the interswitch wire lengths can be determined a priori. The wire lengths between switches in the BFT and the SPIN architectures depend on the levels of the switches. On the other hand, the number of switch levels can be expressed as a function of system size (N) as $levels = \log_4 N$ for both, where N is the number of functional IP blocks in an SoC. The interswitch wire length is given by the following expression [21]:

$$w_{a+1,a} = \frac{\sqrt{Area}}{2^{levels-a}}, \quad (11)$$

where $w_{a+1,a}$ is the length of the wire spanning the distance between level a and level $a+1$ switches, where a can take integer values between 0 and $(levels-1)$. For CLICHÉ and Folded Torus, all wire segments are of the same length, respectively; the interswitch wire lengths of the CLICHÉ architecture can be determined from the following expression:

$$w = \frac{\sqrt{Area}}{\sqrt{N}-1}, \quad (12)$$

while, for the Folded Torus, all the interswitch wire lengths are double those for CLICHÉ [28].

Considering a die size of $20mm \times 20mm$ and a system size of 256 IP blocks, we determined the number of interswitch links and their lengths for all the NoC architectures under consideration. We calculated the interswitch wire lengths in the cases of CLICHÉ, Folded Torus, BFT, and SPIN using (11) and (12) and Figs. 1 and 16.

For Octagon, we determined the interswitch wire lengths following the low wiring complexity scaling strategy shown in [20]. As shown in Fig. 16, in each basic octagon unit, we can differentiate three types of links: long links (connecting blocks 0-4 and 7-3), medium length links (connecting blocks 0-7, 3-4, 1-5, and 2-6), and short links (connecting blocks 0-1, 1-2, 2-3, 4-5, 5-6, and 6-7). These link lengths can be determined by assuming the 256 IP blocks divided into 32 basic octagons, with four placed horizontally and eight placed vertically.

Fig. 17 shows the distribution of interswitch wire lengths for all of the NoC architectures. From this figure, we can qualitatively infer that SPIN will have the highest degree of wiring complexity, while CLICHÉ and Folded Torus will have the lowest complexity. Consequently, for the SPIN topology, the layout of interswitch wire segments will have the greatest impact on area overhead. CLICHÉ and Folded Torus are the simplest ones from a layout perspective, while BFT and Octagon stand between these two groups.

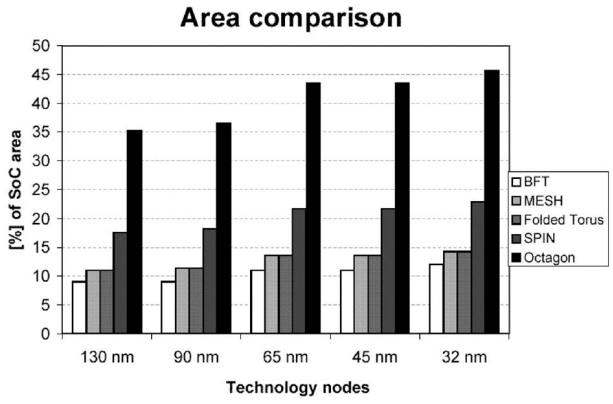

Fig. 15. Area overhead.

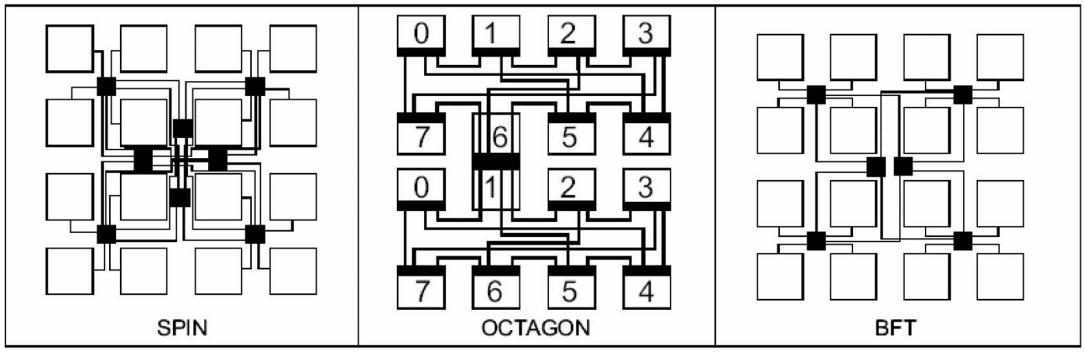

Fig. 16. Simplified layout examples of SPIN, OCTAGON, and BFT.

7 Case Study

To illustrate how system designers can use the analytical and experimental procedures outlined in this paper to estimate the performance of an SoC application, we simulated a multiprocessor SoC, a network processing platform, mapped to different NoC communication fabrics we described in earlier sections. Among all the architectures under consideration, Octagon and SPIN have the higher throughput, but their energy dissipation is much greater than that of the others. In addition, the silicon area overhead due to the infrastructure IP blocks is also higher. Taking these facts into account, we considered the architectures with a lower energy dissipation profile, i.e., BFT, CLICHÉ, and Folded Torus, for further evaluation. For illustrative purposes, we mapped the network processing platform onto these three interconnect architectures. The functional block diagram of the network processor is shown in Fig. 18, based on a commercial design [26]. All the functional blocks are divided into five clusters. Initially, we assumed the traffic to be uniformly distributed among these five clusters. The micro-engines (MEs) in clusters 2 and 3 are the programmable engines specialized for network processing. MEs do the main data plane [26] processing for each packet and communicate in a pipelined fashion within each ME cluster. Consequently, the traffic will be highly localized within these two clusters (clusters 2 and 3).

As discussed earlier, we assumed localization factors of 0.3, 0.5, and 0.8 for the traffic within these two clusters, while the rest of the traffic is assumed to be uniformly random. We also assumed a self-similar injection process.

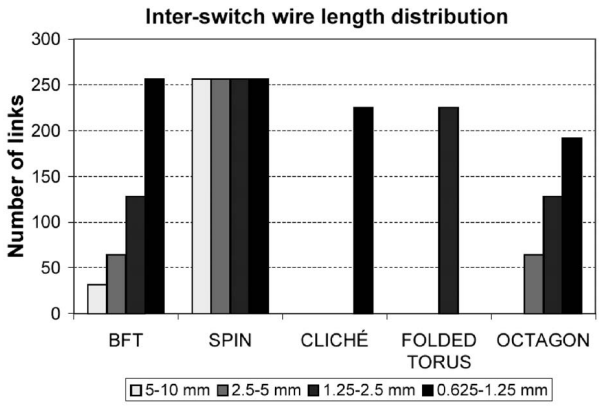

Fig. 17. Interswitch wire length distribution.

Under the stated traffic distributions, we simulated the performance of the network processor SoC shown in Fig. 18. From the throughput characteristics, we can project the aggregate bandwidth [35] sustainable by the SoC platform by using the following expression:

$$Aggregate\ Bandwidth = (Number\ of\ IP\ blocks) \times (Flit\ length) \times (Accepted\ traffic) \times (clock\ rate).$$

Table 4 shows the projected bandwidth, assuming a clock rate of 500 MHz (typical for an SoC implemented in a 130 nm process) and 16-bit flit length, average message latency, and average energy dissipation.

As expected and discussed in Section 6.1, throughput increases significantly with traffic localization, which in turn gives rise to higher aggregate bandwidth. The value of average latency is measured at an injection load below saturation. The effect of traffic localization on average latency is that it allows a higher injection load without saturating the network. The message latency, at a lower injection load (below saturation), remains largely unaffected by traffic localization. While measuring the average energy dissipation, to have a consistent comparison, we kept the system throughput at the same level for all the architectures, while varying the amount of localized traffic. When the factor of localization is varied from 0.3 to 0.8, the bit energy savings relative to the uniformly distributed traffic scenario vary from 20 percent to 50 percent.

As shown in this case study, it is possible to project the achievable performance of a typical multicore SoC implemented using the NoC design paradigm.

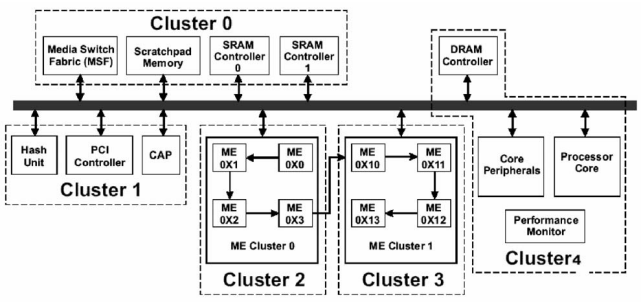

Fig. 18. Functional block diagram of a typical network processor.

TABLE 4
Projected Performance of a Network Processor SoC Platform in NoC Design Paradigm

	Uniform			0.3 Localization			0.5 Localization			0.8 Localization		
	CLICHÉ	FOLDED TORUS	BFT	CLICHÉ	FOLDED TORUS	BFT	CLICHÉ	FOLDED TORUS	BFT	CLICHÉ	FOLDED TORUS	BFT
Throughput	0.58	0.60	0.40	0.63	0.65	0.57	0.75	0.70	0.63	0.80	0.78	0.82
Aggregate Bandwidth (Gbps)	92.8	96	64	100.8	104	91.2	120	112	100.8	128	124.8	131.2
Average Latency (Cycles)	33	30	28	33	30	28	33	30	28	33	30	28
Energy per packet (nJ)	2.1	2.2	2.9	1.70	1.80	2.32	1.50	1.62	1.74	1.25	1.30	1.5

8 Conclusions

Networks-on-chip (NoC) are emerging as a viable interconnect architecture for multiprocessor SoC platforms. In this new paradigm, infrastructure IPs are used to establish the on-chip communication medium. NoC-based architectures are characterized by various trade-offs with regard to functional, structural, and performance specifications. Here, we carried out detailed comparisons and contrasted different NoC architectures in terms of throughput, latency, energy dissipation, and silicon area overhead. We illustrated that some architectures can sustain very high data rates at the expense of high-energy dissipation and considerable silicon area overhead, while others can provide a lower data rate and lower energy dissipation levels. Our principal contribution lies in the establishment and illustration of a consistent comparison and evaluation methodology based on a set of readily quantifiable parameters for NoCs. Our methodology sets an important basis for the optimal evaluation and selection of interconnect infrastructures for large and complex SoCs. Though the parameters considered in our benchmarking are considered by experts in the field to be among the most critical, they do not constitute a unique set nor are they exhaustive. Different applications or circumstances may require this set to be altered or augmented, e.g., by including parameters such as testability, dependability, and reliability. However, they are an important set to characterize the emerging NoC architectures.

Acknowledgments

The authors thank Micronet, PMC-Sierra, Gennum, and NSERC for their financial support and the CMC for providing access to CAD tools. The authors also thank Pierre Paulin of ST Microelectronics and Allan Nakamoto of PMC Sierra for their feedback. In addition, they wish to thank the expert reviewers for their extremely valuable comments, which helped them enhance the quality of their work.

References

[1] L. Benini and G. DeMicheli, "Networks on Chips: A New SoC Paradigm," *Computer,* vol. 35, no. 1, pp. 70-78, Jan. 2002.

[2] P. Magarshack and P.G. Paulin, "System-on-Chip beyond the Nanometer Wall," *Proc. Design Automation Conf. (DAC),* pp. 419-424, June 2003.

[3] M. Horowitz and B. Dally, "How Scaling Will Change Processor Architecture," *Proc. Int'l Solid State Circuits Conf. (ISSCC),* pp. 132-133, Feb. 2004.

[4] Y. Zorian, "Guest Editor's Introduction: What Is Infrastructure IP?" *IEEE Design and Test of Computers,* vol. 19, no. 3, pp. 3-5, May/June 2002.

[5] M.A. Horowitz et al., "The Future of Wires," *Proc. IEEE,* vol. 89, no. 4, pp. 490-504, Apr. 2001.

[6] K.C. Saraswat et al., "Technology and Reliability Constrained Future Copper Interconnects—Part II: Performance Implications," *IEEE Trans. Electron Devices,* vol. 49, no. 4, pp. 598-604, Apr. 2002.

[7] D. Sylvester and K. Keutzer, "Impact of Small Process Geometries on Microarchitectures in Systems on a Chip," *Proc. IEEE,* vol. 89, no. 4, pp. 467-489, Apr. 2001.

[8] ITRS 2003 Documents, http://public.itrs.net/Files/2003ITRS/Home2003.htm, 2003.

[9] C. Grecu, P.P. Pande, A. Ivanov, and R Saleh, "Structured Interconnect Architecture: A Solution for the Non-Scalability of Bus-Based SoCs," *Proc. Great Lakes Symp. VLSI,* pp. 192-195, Apr. 2004.

[10] C. Hsieh and M. Pedram, "Architectural Energy Optimization by Bus Splitting," *IEEE Trans. Computer-Aided Design,* vol. 21, no. 4, pp. 408-414, Apr. 2002.

[11] AMBA Bus specification, http://www.arm.com, 1999.

[12] Wishbone Service Center, http://www.silicore.net/wishbone.htm, 2004.

[13] CoreConnect Specification, http://www3.ibm.com/chips/products/coreconnect/, 1999.

[14] D. Wingard, "MicroNetwork-Based Integration for SoCs," *Proc. Design Automation Conf. (DAC),* pp. 673-677, June 2001.

[15] Open Core Protocol, www.ocpip.org, 2003.

[16] MIPS SoC-it, www.mips.com, 2002.

[17] P. Guerrier and A. Greiner, "A Generic Architecture for On-Chip Packet-Switched Interconnections," *Proc. Design and Test in Europe (DATE),* pp. 250-256, Mar. 2000.

[18] S. Kumar et al., "A Network on Chip Architecture and Design Methodology," *Proc. Int'l Symp. VLSI (ISVLSI),* pp. 117-124, 2002.

[19] W.J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," *Proc. Design Automation Conf. (DAC),* pp. 683-689, 2001.

[20] F. Karim et al., "An Interconnect Architecture for Networking Systems on Chips," *IEEE Micro,* vol. 22, no. 5, pp. 36-45, Sept./Oct. 2002.

[21] P.P. Pande, C. Grecu, A. Ivanov, and R. Saleh, "Design of a Switch for Network on Chip Applications," *Proc. Int'l Symp. Circuits and Systems (ISCAS),* vol. 5, pp. 217-220, May 2003.

[22] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection Networks—An Engineering Approach.* Morgan Kaufmann, 2002.

[23] H.-S. Wang, L.S Peh, and S. Malik, "A Power Model for Routers: Modeling Alpha 21364 and Infiniband Routers," *Proc. 10th Symp. High Performance Interconnects,* pp. 21-27, 2002.
[24] T. Chelcea and S.M. Nowick, "A Low-Latency FIFO for Mixed Clock Systems," *Proc. IEEE CS Workshop VLSI,* pp. 119-126, Apr. 2000.
[25] P.P. Pande, C. Grecu, A. Ivanov, and R. Saleh, "High-Throughput Switch-Based Interconnect for Future SoCs," *Proc. Third IEEE Int'l Workshop System-on-Chip for Real-Time Applications* pp. 304-310, 2003.
[26] Intel IXP2400 datasheet, http://www.intel.com/design/network/products/npfamily/ixp2400.htm, 2004.
[27] J. Hennessey and D. Patterson, *Computer Architecture: A Quantitative Approach.* Morgan Kaufmann, 2003.
[28] W.J. Dally and C.L. Seitz, "The Torus Routing Chip," Technical Report 5208:TR: 86, Computer Science Dept., California Inst. of Technology, pp. 1-19, 1986.
[29] V. Raghunathan, M.B. Srivastava, and R.K. Gupta, "A Survey of Techniques for Energy Efficient On-Chip Communications," *Proc. Design and Test in Europe (DATE),* pp. 900-905, June 2003.
[30] L. Benini and D. Bertozzi, "Xpipes: A Network-on-Chip Architecture for Gigascale Systems-on-Chip," *IEEE Circuits and Systems Magazine,* vol. 4, no. 2, pp. 18-31, 2004.
[31] K. Park and W. Willinger, *Self-Similar Network Traffic and Performance Evaluation.* John Wiley & Sons, 2000.
[32] D.R. Avresky, V. Shubranov, R. Horst, and P. Mehra, "Performance Evaluation of the ServerNetR SAN under Self-Similar Traffic," *Proc. 13th Int'l and 10th Symp. Parallel and Distributed Processing,* pp. 143-147, Apr. 1999.
[33] G. Varatkar and R. Marculescu, "Traffic Analysis for On-Chip Networks Design of Multimedia Applications," *Proc. Design Automation Conf. (DAC),* pp. 510-517, June 2002.
[34] *Networks on Chip,* A. Jantsch and H. Tenhunen, eds. Kluwer Academic, 2003.
[35] B. Vermeulen et al., "Bringing Communication Networks on a Chip: Test and Verification Implications," *IEEE Comm. Magazine,* pp. 74-81, Sept. 2003.
[36] W.J. Dally, "Virtual-Channel Flow Control," *IEEE Trans. Parallel and Distributed Systems,* vol. 3, no. 2, pp. 194-205, Mar. 1992.

Partha Pratim Pande is completing his PhD degree studies in VLSI design at the Department of Electrical and Computer Engineering, University of British Columbia, Canada. His PhD thesis evolved around the topic "Network on Chip." Before this, he received the MSc degree in computer science from the National University of Singapore in 2001 and the bachelor's degree in electronics and communication engineering from Calcutta University, India, in 1997. After receiving the bachelor's degree, he worked in industry for a couple of years as a digital system design engineer. At the end of 1999, he returned to academia to pursue higher studies. He has received several national scholarships from the government of India for academic excellence. In addition to this, he received the International Graduate student scholarship from the National University of Singapore. He is a student member of the IEEE.

Cristian Grecu received the BS and MEng degrees in electrical engineering from the Technical University of Iasi, Romania, and the MASc degree from the University of British Columbia. He is a doctoral student in the Department of Electrical and Computer Engineering, University of British Columbia, Canada. His research interests focus on design and test of large SoCs, with particular emphasis on their data communication infrastructures.

Michael Jones received the BSc degree in computer engineering from Queen's University, Kingston, Ontario, Canada, in 2002 and the MASc degree from the University of British Columbia, Vancouver, Canada, in 2005. Since 2005, he has moved on to work as a software engineer in the high tech industry.

André Ivanov received the BEng (Hon.), MEng, and PhD degrees in electrical engineering from McGill University. He is a professor in the Department of Electrical and Computer Engineering at the University of British Columbia (UBC). He spent a sabbatical leave at PMC-Sierra, Vancouver, British Columbia. He has held invited professor positions at the University of Montpellier II, the University of Bordeaux I, and Edith Cowan University in Perth, Australia. His primary research interests lie in the area of integrated circuit testing, design for testability, and built-in self-test for digital, analog, and mixed-signal circuits and systems-on-a-chip (SoCs). He has published widely in these areas and holds several patents in IC design and test. Besides testing, he has interests in the design and design methodologies of large and complex integrated circuits and SoCs. He has served and continues to serve on numerous national and international steering, program, and/or organization committees in various capacities. Recently, he was the program chair of the 2002 VLSI Test Symposium (VTS '02) and the general chair for VTS '03 and VTS '04. In 2001, he cofounded Vector 12, a semiconductor IP company. He has published more than 100 papers in conference and journals and holds four US patents. He serves on the editorial board of *IEEE Design and Test* and Kluwer's *Journal of Electronic Testing: Theory and Applications*. He is currently the chair of the IEEE Computer Society's Test Technology Technical Council (TTTC). He is a Golden Core Member of the IEEE Computer Society, a senior member of the IEEE, a fellow of the British Columbia Advanced Systems Institute, and a Professional Engineer of British Columbia.

Resve Saleh received the PhD and MS degrees in electrical engineering from the University of California, Berkeley, and the BS degree in electrical engineering from Carleton University, Ottawa, Canada. He is currently a professor and the NSERC/PMC-Sierra Chairholder in the Department of Electrical and Computer Engineering at the University of British Columbia, Vancouver, Canada, working in the field of system-on-chip design, verification, and test. He received the Presidential Young Investigator Award in 1990 from the US National Science Foundation. He has published more than 50 journal articles and conference papers. He is a senior member of the IEEE and served as general chair (1995), conference chair (1994), and technical program chair (1993) for the Custom Integrated Circuits Conference. He recently held the positions of technical program chair, conference chair, and vice-general chair of the International Symposium on Quality in Electronic Design (2001) and has served as an associate editor of the *IEEE Transactions on Computer-Aided Design*. He recently coauthored a book entitled *Design and Analsys of Ditigal Integrated Circuit Design: In Deep Submicron Technology*. He was a founder and chairman of Simplex Solutions, which developed CAD software for deep submicron digital design verification. Prior to starting Simplex, he spent nine years as a professor in the Department of Electrical and Computer Engineering at the University of Illinois, Urbana, and one year teaching at Stanford University. Before embarking on his academic career, he worked for Mitel Corporation in Ottawa, Canada, Toshiba Corporation in Japan, Tektronix in Beaverton, Oregon, and Nortel in Ottawa, Canada.