On-chip networks: A scalable, communication-centric embedded system design paradigm

Jörg Henkel†, Wayne Wolf‡ and Srimat Chakradhar†
†NEC Laboratories America, Princeton, NJ
‡Princeton University, Princeton, NJ

Abstract

As chip complexity grows, design productivity boost is expected from reuse of large parts and blocks of previous designs with the design effort largely invested into the new parts. More and more processor cores and large, reusable components are being integrated on a single silicon die but reuse of the communication infrastructure has been difficult. Buses and point to point connections, that have been the main means to connect components on a chip today, will not result in a scalable platform architecture for the billion transistor chip era. Buses can cost efficiently connect a few tens of components. Point to point connections between communication partners is practical for even fewer components. As more and more components are integrated on a single silicon die, performance bottlenecks of long, global wires preclude reuse of buses. Therefore, scalable on-chip communication infrastructure is playing an increasingly dominant role in system-on-chip designs. With the super-abundance of cheap, function-specific IP cores, design effort will focus on the weakest link: efficient on-chip communication.

Future on-chip communication infrastructure will overcome the limits of bus-based systems by providing higher bandwidth, higher flexibility and by solving the clock skew problem on large chips. It may, however, present new problems: higher power consumption of the communication infrastructure and harder-to-predict performance patterns. Solutions to these problems may result in a complete overhaul of SOC design methodologies into a communication-centric design style. The envisioning of upcoming problems and possible benefits has led to intensified research in the field of what is called NoCs: Networks on Chips. The term NoCs is used in a broad meaning, encompassing the hardware communication infrastructure, the middleware and operating system communication services, and a design methodology and tools to map applications onto a network on chip. This paper discusses trends in system-on-chip designs, critiques problems and opportunities of the NoC paradigm, summarizes research activities, and outlines several directions for future research.

1 Current Trends in SoC Complexity

By the end of this decade, silicon technology will allow chip complexities of up to 1 billion transistors on a single piece of silicon (see [5]). The embedded systems market will especially profit from this development since we can virtually build a whole system (processing elements, memories, communication infrastructure, analog I/O etc) on one chip (SoC). Though a large part of these transistors may be used in embedded memory, a significant share will be used for increasing the number of on-chip processing units in order to increase system performance. Assuming the complexities of today's state-of-the-art embedded processors (e.g. [2]), several hundreds or even thousands of embedded processors can fit on a single chip.

Today's complex SoCs comprise of hardly more than 10-15 processors on a single chip (example: [3]). Unfortunately, we

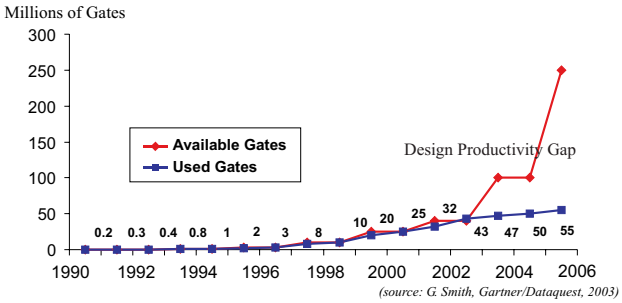

Figure 1: Chip complexity and design complexity crisis [1]

are unable to exploit the maximum possible amount of transistors (silicon-technology-wise) per chip. This fact is even more obvious when the number of transistors per SOC *without* the embedded memory (i.e. in the absence of easy to design regular structures) is excluded. Clearly, real-world SOCs' complexities currently (2003) lag behind the capabilities of current silicon technologies even though there is certainly a demand for higher complexities from an application demand point of view.

Fig. 1[1] gives a possible answer: it shows the predicted productivity gap. It is measured as the number of available gates per chip for a given silicon technology for SOCs on the one side (red graph) and the number of actually used gates per chip for a given silicon technology on the other side (blue graph). The gap is predicted assuming that *no* ESL (Electronic System Level Design) methodologies are deployed for designing future complex SOCs. In other words: the Design gap might be avoided if more ESL methodologies would be deployed in all areas of system level design like specification/modeling, synthesis etc.

However, with current activities in ESL methodologies, we think this gap will actually be closed and thus will indeed allow the system designer to integrate all 1 billion transistors on a single chip (see also [4]) as silicon technology provides these potential complexity by the end of the decade.

Application areas for 1-billion-transistor SoCs are manifold ranging from security systems (e.g. video surveillance), control systems (e.g. automotive control), individual health systems (e.g. hearing aids) to main stream consumer products in such areas as personal communication (e.g. cell phones), personal computing (e.g. PDA), entertainment (e.g. MP3 players), video/photo (e.g. digital still/video cameras) and many more. It can be observed that many of these devices feature already today a high complexity with a rising tendency as new device generation feature a larger functionality and hence need a more complex SoCs for their implementation.

This trend will significantly change the way SoCs are designed, both from a design methodology point of view (not covered in this article) as well as from an architectural point

Proceedings of the 17th International Conference on VLSI Design (VLSID'04)
1063-9667/04 $ 20.00 © 2004 IEEE

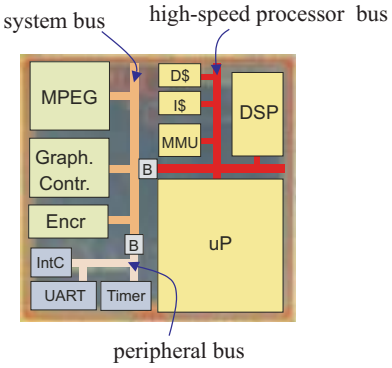

Figure 2: An example for a typical bus-based SoC

of view: hundreds or even thousands of processors will have to be integrated on a single chip and will require a sophisticated communication infrastructure. Bus-based communication infrastructures, even those utilizing hierarchies of buses e.g. high-speed processor bus, system bus and low-speed peripheral bus separated by bridges (similar to the hierarchy of buses offered by ARM[10]); (see also Fig. 2) will not be sufficient for several reasons:

a) a (single) bus does not provide concurrent transactions: depending on the arbitration algorithm, access to the bus is granted to that medium/IP/device that has the highest priority. All other requests occurring during other current bus transactions have to be postponed to a later point in time. Thus, the bus is blocking other transactions that could potentially executed in parallel.

b) large bus lengths are prohibitive in future designs since the combination of (geometrically) large future SoCs and high clock frequencies (up to 10GHz by the end of the decade[5]) would lead to non-manageable clock skews on a bus-based system.

The scalability and success of the Internet has inspired researchers to borrow the ideas of switch-based (routers) networks and packet-based communication for on-chip communication. Most of the terminology used for on-chip packet switched communication is adapted from the computer networking area. Packet-switched communication, besides providing scalability, offers the possibility of standardization and reuse of communication architecture. These features are crucial to chip designers to lower design effort, and meet time to market constraints for new products.

Networks-on-Chip, NoCs, is emerging as a new design paradigm to overcome the limitations of today's bus-based communication infrastructure. An example of a NoC as shown in Fig.3 may work as follows: packets are sent via links from an origin IP to the input channel of a router (see also Fig.5) and switched by, for example, a cross bar switch within the router from where it leaves the router via an output channel and link to the next switch. Or, the packet might be routed to the local IP attached to that switch. A NoC features the following, most prominent, characteristics:

a) it transmits packets instead of words. Dedicated address lines like in bus systems are not necessary since the destination address of a packet is part of the packet.

b) transactions can be conducted in parallel if the network provides more than one transmission channel between a sender and a receiver.

c) routers in the network provide for decoupling such that clock skew issues in large SoCs are not a concern. Other effects like cross talk are of smaller concern, too.

d) the wiring of the on-chip network is structured[1] as shown in Fig. 3: IP cores (like processors, memories) are tiled on the chip in a regular manner connected via a structured on-chip network. In such a layout, routing of wires is not an issue any more.

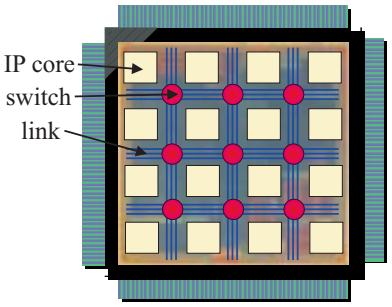

Figure 3: Network on Chip shown with IP cores, links and router in a mesh layout

In this article we will restrict our discussions to on-chip networks for embedded systems. This overview paper is structured as follows: Section 2 characterizes on-chip networks with particular emphasis on comparisons to large scale networks. Section 3 describes opportunities to customize on-chip networks. In Section 4, we summarize some representative research work with focus on architecture, power issues and simulation of on-chip networks. Finally, Section 5 gives hints to future research needs.

2 Characteristics of on-chip networks

In order shed some light in what an on-chip-network is, we shall point out the most prominent differences by comparing them to large scale and/or board-level networks. Proposals for on-chip networks share a common set of characteristics: they all support packet-based communication, are scalable and are capable of providing transport functionality for any heterogenous assembly of IP components.

2.1 Wiring Resources

As pointed out in [6] on-chip-networks have abundant resources for wiring. Dally et al. [6] report that their design can feature up to 6,000 wires on each metal layer and they argue that in total it would be possible that 24,000 pins[2] could cross a network tile of their design. This is plausible since chip area can be dedicated to wires instead of logic at the designers demand without any other penalty (except the chip area spent). This is in sharp contradiction to board-level communication infrastructures where pin count of chips is a determining cost factor of the chip since a pin has very high fabrication costs. Wiring is also very costly in large scale networks since its cost is determined by infrastructure efforts for fiber lines or wireless networks (base stations)

In non-network-based systems, wires may cross the whole chip in a non-regular manner causing hard-to-predict and hard-to-avoid cross-coupling effects. Also, long wires on SoCs need repeaters that are to be inserted by routing tools. This may not only lead to non-routable wires but also to a conservative and hence slow design. On-chip networks with a regular layout on the other hand (see Fig.3) prescribe clearly

[1] Though this is not mandatory, it has many advantages as we show later on

[2] Note, that 'pin' has a different meaning here as it simply denotes the connections of an IP (e.g. a processor) to its environment but not a physical pin.

where to insert repeaters. In addition, all links are of the same length. Delays on wires are thus very predictable as well as cross talk issues are. Moreover, due to the predictable cross-talk pattern a non-conservative line-width sizing and layout can be used and will thus lead to higher clock rates the wires can be operated at. For the same reason, the specific power consumption of a wire will decrease, too.

2.2 Traffic Patterns

The traffic patterns in large scale networks are stochastic and follow in general the Poisson distribution[12]. On-chip networks in embedded systems do not necessarily follow the Poisson distribution since the traffic pattern may be related to a chain of *predictable* events. An example: lets consider a video surveillance system with a camera that takes 50 shots a second. Each frame is stored in a local buffer. Digital signal processing may include sharpening, rendering, motion detection etc. Each of these steps can be processed in a pipelined fashion by a set of application specific DSPs. Data needs to be passed via a high bandwidth interconnect. Assuming that the frame size ($W \times H$) is constant, the traffic pattern (amount of data to be transferred as well as source of all transmissions) can be well predicted within certain bounds.

On-chip networks can exploit these characteristics through *pre-scheduling*: a packet that is pre-scheduled may move from one link to another without arbitration. This makes transmissions very fast and actually increases actual band width.

Fig.4 shows a traffic distribution that we measured on a specialized video surveillance system. As can be seen the traffic pattern does not follow a Poisson distribution (implications will be discussed later).

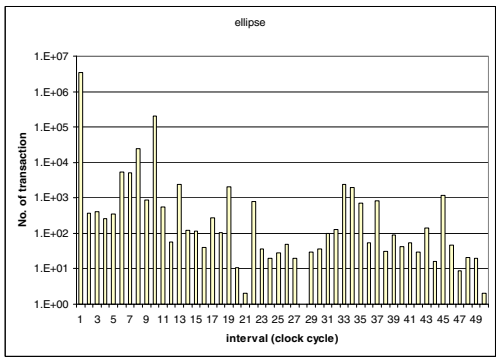

Figure 4: Traffic distribution as a function of intervals in an embedded video system

2.3 Heterogeneity of NoCs

Systems-on-chips tend to run heterogeneous applications on heterogeneous architectures. Many SoC applications perform several different types of algorithms; the processing elements are chosen to match the computational load of each process to the processing element on which it runs. It is reasonable to expect that network-on-chip architectures will also be heterogeneous. Communication loads will vary considerably between pairs of points in a typical SoC architecture. Most SoCs are not large enough to aggregate truly large loads that satisfy the Law of Large Numbers. Specializing the network architecture to better match the traffic helps to reduce the network size and power consumption.

2.4 Power Consumption

Power consumption is virtually a non-issue in large scale networks. However, it will be one of the most important issues in on-chip networks. Already in today's SoCs with hierarchies of buses as the common communication infrastructure, power consumption is a major problem since the per mm^2 power consumption is rising to levels that are increasingly unmanageable due to increased clock frequencies in conjunction with ever decreasing feature sizes i.e. high integration ratios. With the introduction of on-chip networks the scenario might worsen: though the specific power consumption of a wire in an on-chip network might decrease (as discussed earlier), the sheer amount of added wires (as part of the links between routers/switches) will increase sharply due to the increasing number of IP cores residing on future SoCs. In addition the communication will likely increase, too. As a result, the power consumption of the interconnect is no more negligible and might, in fact, significantly contribute to the power budget of future SoCs. On the other side, compared to a bus-based system, a network-based system is still more power efficient (relatively) since unlike to a bus, the information (word in case of a bus; packet in case of a network) does not have to be broadcast to any possible recipient as pointed out in [9]. As a summary, when designing and customizing NoCs, power needs to be treated as a major design constraint.

2.5 Testability

The reliability of electronic systems is no longer limited to critical applications in military, aerospace or nuclear industries, where failures may have catastrophic consequences. Electronic systems are ubiquitous and testing of electronic products to ensure that the design implemented in silicon is free of manufacturing defects has become mandatory. Testing of embedded cores requires three components: (1) a source (on or off-chip) for generating test stimuli, and observing test responses, (2) a test access mechanism to move test data to and from the source to an IP component, and (3)a test scheduling strategy that allows concurrent testing of multiple cores. The three components largely determine the testing cost for the SoC. Hence, design-for-test strategies that minimize test volume, test hardware overhead, test application time while increasing the defect coverage are necessary.

The NoC can be considered as just another core in the SoC (like other SoCs, NoC-based SoC has to be tested for manufacturing defects) but the NoC offers new opportunities for testing: NoC is often composed of many identical components (routers, network interfaces, etc.), and it interconnects every component in the SoC. These characteristics of an NoC give rise to several unique test challenges. First, the regular and hierarchical structure of the NoC allows test re-use. All identical blocks in the NoC can reuse the same test data [26]. This test data set can be broadcast and applied to all identical elements at the same time. Responses can be compared against each other and mismatches can be sent off-chip for analysis. Second, timing tests are important because clock boundaries between cores are now inside the network interface, and the NoC, spread over the entire SoC, may have long wires. Interconnection elements are vulnerable to timing and crosstalk errors. Delay test patterns with a high fault coverage are necessary to increase the overall defect coverage and reduce the defect levels. Third, on-chip communication links are relatively short compared to those in computer networks, and communication wires are relatively abundant in NoCs. High-speed interconnect testing will dominate the test development, test volume and test application costs. Fourth, presence of communication protocol stacks complicates testing. Embedded software executing on communication-oriented processor cores assumes that the underlying processor platform is free from functional as well as timing defects. Automatic, at-speed processor self-test methodologies [16] will play an important role in the overall NoC test strategy. Finally, NoC QoS characteristics will determine whether or not test stimuli can be delivered to the IP components to enable at-speed test [24]. Bandwidth resource management schemes will determine the volume and speed of test stimulus delivery to an IP component. These schemes will have a first order effect on SoC test

scheduling, and achievable at-speed delay defect coverage.

2.6 Limitations

Scalability and reusability are critical for any on-chip communication platform. Packet switched network proposals are promising but several limitations must be solved before these proposals will find their way into commercial products.

First, memory access between IP cores and memory cores will be a performance bottleneck if a packet switched fabric has to be traversed on every memory access. The problem can be ameliorated by adding local memory resources. However, many data intensive real time applications, especially in media processing, require memory buffers to be shared among many computation cores. Therefore, it may be necessary to evolve a hybrid infrastructure that can provide both circuit-switched access to memory cores and allow packet switched data transfer for general data communication.

Second, there are no standards for on-chip networks. On the one hand, absence of standards offers immense opportunities to customize the communication fabric. However, without some standardization, it will be difficult to promote large scale reusability of the communication backbone and motivate IP vendors to create design independent communication IP cores.

Third, adoption of networks on chips will be slow. Today, a large majority of designers are not experts in network concepts. Furthermore, unlike IP components for computation, scalable communication IP cores that can be reused across designs are far and few. Therefore, design methodologies and tools that can encapsulate and automate many of the design aspects are necessary to reign in the communication design costs. Models of computation that capture the communication requirements as well as abstract the capabilities of a communication architecture can enable efficient matching of application requirements and architectural capabilities. Technologies and tools to perform the match are necessary to reduce the design effort.

3 Customization Potential for On-chip Networks

Large scale networks underlay strong restrictions to follow standardizations in order to enable network clients of virtual all performance, size and cnad cost levels to properly talk to each other. Another reason is downward compatibility i.e. to enable older standards to use newest network infrastructure. Standardization of on-chip networks on the other side is a rather minor issue since the IP core deployed can be freely chosen/designed to fit to any NoC. It is desirable, however, that conventions are defined that provide standards for interface definition in order to employ IP cores of different vendors in one SoC with an on-chip network.

In the following, we discuss several ways to customize on-chip networks in order to find the best compromise between the main design constraints performance, power and area for a specific NoC design. The customization levels can be categorized in architectural, protocol, QoS and software issues.

3.1 Architectural Customization

Bus-based systems are not scalable i.e. the more processing nodes are added to the bus the more the bus becomes the bottleneck for the performance of the system. Not so networks. In fact, adding additional routers/switches and links to a network, effectively increases the network's bandwidth. This is one of the most prominent reasons for the NoC paradigm. Widely used are *direct networks* (also referred to as point-to-point networks) where nodes (a router/switch and local IP) are connected to a few (dependent on topology) other nodes via bi-directional links (also called channels). An example of a 2D mesh network is shown in Fig. 3. Network topologies can be characterized by a) the number of links connecting a node to other nodes, b) the maximum distance between of any two

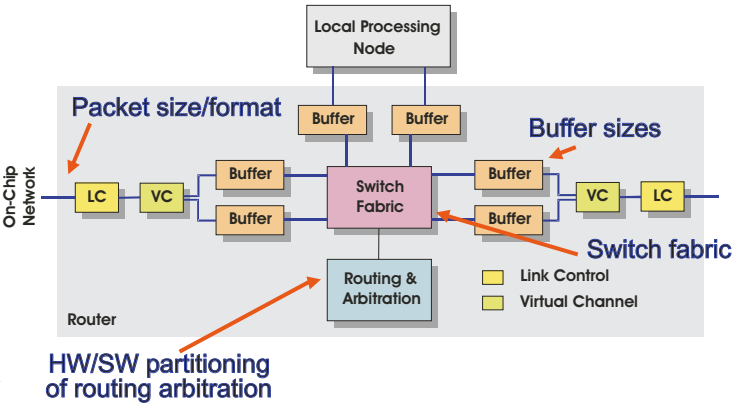

Figure 5: A generic router with some parameters listed

nodes, c) the *regularity* i.e. whether or not all nodes have the same number of neighbors and d) the *symmetry* i.e. whether or not the networks looks the same from all locations.

In the super computing domain, we will find symmetric and regular networks according to the nature of its applications. However, we believe that one architectural customization for an embedded system SoC lies in fact that it is neither symmetric nor regular: there will be computational nodes that have a larger need for communication with many other nodes and there will be those nodes that have lesser communication with a small number of nodes. The heterogeneity of computation should be reflected in the NoC architecture in order to prevent a (too expensive) over-design (for a more detailed overview of network topologies see [11]). An example for a non-symmetric, non-regular NoC is given in [13][7].

Fig.5 shows a (generic) node of a router-based network: The link control LC controls the flow of physical channels (links) and refers messages to the channel controller which arranges queuing of messages in the virtual channels (two shown in the example). The core of the router represents the switch and the according routing and arbitration algorithms. The router is connected to the local IP (processing elements via buffered links).

A crucial component for customization is the number of *virtual channels*. If a message occupies the buffer of a physical channel, no other message can enter the channel. Virtual channels try to minimize those situations. The choice of number of virtual channels needs to be carefully adapted to the number of sources and destinations messages are expected to be received from and to be sent to. The number of virtual channels needs also to be adapted to the switching technique.

The **buffer size** is another important parameter to customize a router. If the message/packet size is large, it may occupy more channels and thus degrade performance. On the other side, if the message size is small and wormhole switching is used, the increase in performance through increased buffer size is insignificant.

3.2 Protocol Customization

A set of rules and methods are required for transfer of information from one IP component to another IP component in an SoC. These rules are referred to as *communication protocols*. Hardware handshaking with request and acknowledge signals is a simple example of a protocol for communication between two units connected through direct wires. A number of bus protocols exist which allow reliable communication among connected components. The bus protocols provide rules for usage of communication wires for information exchange and for resolving conflicts amongst components. It also provides upper limits on the transfer rate or bandwidth. Similarly, in NoC, communication protocols will determine how a compo-

nent is connected to the network as well as how the information flows from one IP component to another. However, the protocols used will be much more complex. Communication workload will be partitioned and organized into layers with well-defined functionality and interfaces. The architecture defining the protocol layers is referred to as a *protocol stack*. Application-specific customization of the protocol stack can provide huge power and performance benefits.
In the following we discuss routing and packet size as two important customization characteristics of a network protocol.

a) **Routing**
Routing determines the path a packet takes in the network starting from the origin to its destination. Considerations for routing algorithms are (see also [11]) *connectivity* (can packets routed from any origin to any destination in the network?), *adaptivity* (can alternative paths be used?), *deadlock and livelock freedom* (may packets block the network or travel forever?), *fault tolerance* (can a packet be routed if there is a fault in the network?). As for connectivity, it is quite clear in an embedded system which IP core may be a potential recipient or sender of a package. If an IP is neither one, no connectivity (to the on-chip[3]? network) needs to be provided. Adaptivity adds overhead since the routing tables need to be updated in a regular manner. If the traffic pattern is quite predictable a pre-routing and pre-scheduling might be preferred over a large degree of adaptability.
Furthermore, in order to find the right routing algorithm it has to be determined whether the packet is to be routed to one (*unicast routing*) or to multiple recipients *multicast routing*. In the first case the question arises whether the routing should be centralized or distributed etc.

b) **Packet Size**
Finding the right packet size is crucial to make optimum use of the network resources. The optimum size highly depends on the characteristics of the application. If a message has to be split in too many packets which have to be re-assembled at destination to obtain the initial message, the overhead incurred might be too high. On the other side, if the packet is too large, the packet might block the link for too many cycles and potentially block other traffic with side effects on the performance of the whole system. The packet size is also crucial in conjunction with the buffer size of the router/switch: if, for example, the packet size exceeds the buffer size, certain routing algorithms cannot be applied.
The packet size may also be adapted to the application characteristics: a data dominated application may require a larger buffer size as opposed to control dominated applications. Fig.6 shows experiments conducted in our lab: the per-bit energy consumed for routing and switching through a router architecture similar to Fig.5 dependent upon packet size and buffer size is shown. As can be seen, the energy can vary in a wide range spanning orders of magnitudes. Of course, throughput (not shown here) will also be affected by these parameters. Obviously, an on-chip network bears a large customization potential.

3.3 Customization for QoS

When components are combined, the performance of the combination must be validated.This can be done by analyzing the complete system implementation. However, going forward, this analysis may be too hard. It may become necessary to abstract the behavior of NoC, and the IP cores to underscore their service requirements rather than requiring second-guessing of the inner workings of a component. The services will describe all that is required to know, and component interaction can be validated at the service abstraction level rather than the component implementation that is subject to periodic revisions. Almost any form of quality commitment will require communication bandwidth resource management in the NoC. If a commitment has been given by the NoC to an IP component, then the service is guaranteed, otherwise the service defaults to a best-effort service. For example, consider reliable data transmission over unreliable media (wires in NoC suffer increasingly from interference such as cross talk and voltage drops). To ensure reliable delivery, data can be retransmitted to overcome transmission errors or forward error correction can be employed. Re-transmission takes a variable, possibly unbounded, amount of time whereas error correction can be performed in constant time. Hence, service level time guarantees like minimum throughput through the NoC cannot be delivered by retransmission based error recovery approaches.

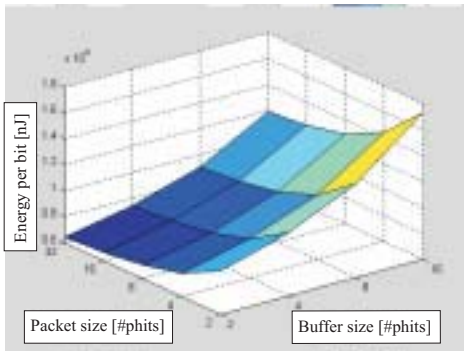

Figure 6: Energy per transmitted bit on NoC as a function of network parameters

In some cases, an IP component connected to the NoC may not always be able to accept incoming data. Since the input buffers to the IP component are finite, several solutions can be envisioned but these solutions have a direct consequence on the quality of service. For example, data that arrives after the buffers are full can be dropped but error recovery approaches are necessary to recover from lost data. Such schemes preclude realization of latency and jitter guarantees necessary for multimedia applications. A clear, service level specification of the NoC services is necessary to ensure that communication patterns with hard, real-time deadlines can be realized by the communication fabric. An essential part of any quality of service is flow-control. Reliable data delivery requires that the the sender does not overrun the receiver by dumping data at a rate faster than what the receiver can cope with. It becomes the responsibility of the NoC to provide the necessary flow-control infrastructure to facilitate communication between arbitrary IP components.

Quality-of-service, an important design goal for networks-on-chips, changes the network design in several ways. Most importantly, NoCs must implement a resource management strategy that allocates bandwidth, enforces the allocation at the processing elements, and notifies the IP components (attached to the NoC) of that allocation. QoS-based design enables IP re-use and the decoupling of applications and architectures. In particular, it ensures that global problems, such as clocking, are solved by combining local, decoupled solutions (eg. GALS) and the solution has a global predictable behavior.

[3] Note, an IP core like a memory may only need *local* connectivity e.g. to an adjacent processor which, itself, is connected to the on-chip network

This is especially important for time-sensitive communication between IP components in an SoC.

3.4 Customization in Hardware/Software Trade-off

In Fig.5 it was implicitly assumed that router/switch on the one side and local IP (a sub-system that can comprise a processor and local memory) are different entities. However, this is not necessarily the case as the local processor may assume parts or all of the duties of the router. For example, routing and arbitration algorithms may be assumed by the local processor and even the virtual channels may be managed by the local processor and buffers may actually use the local memory instead of dedicated hardware. There is a wide variety of constellations possible that will depend on many factors as the expected traffic passing the router, utilization of the local processor etc.

4 State-of-the-Art Research in On-chip networks

Research for NOCs intensified around 2000 when it became clear from the ITRS (International Technology Roadmap for Semiconductors) that new on-chip communication infrastructures are necessary to overcome problems like clock skew on large chips and power consumption combined with high integration density of up to a billion transistors per chip by the end of this decade.

We give a representative (but by far not complete) overview of research activities concerning NoCs with emphasis on architectural issues, power consumption and simulation/modeling. Lahiri/Raghunathan/Dey [15] have conducted early work on bus-based communication architectures with a focus on protocols. Benini/DeMicheli [8] and Jantsch/Hemani [14] have initiated research for NoCs from a high-level design point of view whereas Horowitz et al.[17] and Dally et al.[6] have predicted problem from a physical (wire) point of view.

Let us start with architectural issues. The importance of buffer sizes in switches has already been pointed out before. In [18] the authors present an approach to adapt buffers efficiently to the SoC architecture in order to trade-off performance and cost. Associated with buffer size is the packetization of data. This problem has been studied in [19]. A design methodology for designing a network-centric architecture has been presented in [20]. The authors in [21] focus on designing a router architecture that has been adapted to trade-off cost and efficiency. An all-over SoC architecture for NoCs using a parallel programming model, implemented through multithread processors, interleaved memory modules and a high capacity interconnection network, is proposed in [22]. Further communication architectures have been proposed by [23] and [25]. Commercially available (bus-based) interconnect architectures include CoreConnect from IBM [27], AMBA from ARM [10], MicroNetworks from Sonics [28], Wishbone form Silicore [29], Palmbus from Palm [30] and Altera's Avalon [31].

As for power/energy issues in NoCs, initial work has been conducted in estimating a router's power consumption. In [32], the authors use stochastic traffic models to obtain transition activity and packet arrival and departure events. They then propose a simulation-based framework to estimate energy. In [33], [34] a stochastic model is deployed. Simulation is then used to estimate energy. In [35] the authors propose a bit level energy estimation of network routers. Their model is targeted for NoCs with one router. The authors in [36] extend that to modeling the energy consumption of a bit as it is transmitted from one tile to another in a tile-based network architecture. In [37] the authors propose a model of the interconnect to compute the power at system level.

Simulation is another important aspect for on-chip networks as performance is a crucial characteristic that needs to be estimated before the NoC is designed. Simulatative approaches have focused on protocols and topologies. In [39] a statistical approach to a simulator is proposed. The authors in [38] aim at accurate simulation rather than a fast design space exploration. Their simulator provides a user with accurate power models and the capability to model network components in great detail.

5 Future Research Directions

Research on NoCs is still in its infancy. Though basic concepts have been proposed, few concrete implementations of complex NoCs exist to date. And, there is certainly no standardization on NoCs.

It is also questionable whether a **standardization** would help advancement of NoCs because one of the advantages of NoCs as opposed to large scale networks is that they potentially can be customized *without* complying to standards and hence they can be designed in an application specific manner i.e. very efficient for a given set of constraints. On the other side, IP needs to be integrated on the NoC. Thus, standardized interface definition a lá VSIA's (Virtual Socket Interface Alliance) OCB (On-Chip-Bus) might help for NoCs, too.

Simulating NoCs will become quite a challenge since NoCs will comprise many processing units connected through a complex communication infrastructure. Cycle-accurate simulation similar to what has been practiced for low complexity SoCs with one processing units where an ISS (Instruction Set Simulator) was used won't work for several reasons: a) whereas in a single-processing-unit SoC there is one master only, the ISS resembling that processing could be used to simulate the whole system. Not so in a complex NoC where there are many (even heterogeneous) processing units. Here, a communication-centric simulation approach is needed i.e. one that simulates the whole system from the point of the view of the communication infrastructure; b) the sheer complexity might prohibit cycle-accurate simulation. New simulation strategies are needed. Probably *"cycle-approximate"* simulation strategies are a solution. Even though it is not yet defined what exactly it stands for it represents efforts to simulate "sufficiently" accurate with much higher simulation speeds than ordinary cycle-accurate simulators. On the other side *cycle-approximate* certainly does not mean pure statistical simulation. Here it should be noted that communication network designers generally assume that sources on the network are independent and uncorrelated. However, the sources connected to a network-on-chip are likely to be highly correlated. Systems-on-chips often process streaming data from one or a few sources. The various processing elements on the chip pass data related to that stream. A stream's period defines a basic heartbeat for the SoC and NoC. Data derived from that stream may be highly periodic or less so, depending on the processing done on it, but they are all likely to reflect the system heartbeat to some extent. Data presented to an NoC is also less likely to fit the traditional Poisson distribution used in communication networks. So, simulation for NoCs offers plenty of terrain for future research.

We also need to investigate **hierarchical architectures** for NoCs. The very large chips on the horizon will allow us to put 50-100 CPUs on a single chip. The traffic in these systems is unlikely to be uniformly distributed. Huge multiprocessors will probably run clusters of applications that are loosely coupled. Each subapplication will have its own set of CPUs that communicate frequently with each other, while communication between these clusters will require less bandwidth. NoC architectures and protocols can take advantage of the non-uniformity of traffic to reduce power consumption and to improve QoS.

Last but not least there is the **crosstalk problem**: physical

layer components of NoCs will often be delivered as hard IP. Physical connections need to be carefully designed to avoid crosstalk problems; good circuit design can also improve performance and reduce power consumption. However, most of the IP for networks-on-chips will need to be designed to be parameterizable and configurable. As we have seen in this paper, substantial efficiencies can be achieved by customizing the network. In some cases, customization may be required to make the design feasible, given the large differences in traffic characteristics between NoCs and traditional communications applications.

Acknowledgements: we would like to thank A. Ghosh and J. Xu who have provided some of the experiments when they were summer interns at NEC Laboratories America.

References

[1] G. Smith, DAC Panel Presentation, 40th. Design Automation Conference (DAC), Anaheim, June 2003.

[2] Tensilica, http://www.tensilica.com

[3] "NEC Boosts TCP/IP Protocol Processing with 10 CPU Cores on One Chip", NEC Corporation, Press Release, May 16th. 2003,

[4] J. Henkel. "Closing the SoC Design Gap", IEEE Computer Magazine, pp.119-121, September 2003.

[5] ITRS, International Technology Roadmap for Semiconductors, Update 2002.

[6] W. J. Dally and B. Towles, "Route packets not wires: on-chip interconnection networks", *DAC*, 2001.

[7] W. Wolf, B. Ozer, T. Lv, "Architectures for distributed smart cameras", Multimedia and Expo, 2003. IEEE ICME'03, Proc. Vol. 2, pp.5-8, 2003.

[8] L. Benini and G. De Micheli, "Powering networks on chips", IEEE/ACM ISSS Symposium, 2001.

[9] L. Benini and G. De Micheli, "Networks on chip: a new SOC paradigm", *IEEE Computer*, pp.70-78, 2002.

[10] D. Flynn, "AMBA: enabling reusable on-chip designs", IEEE Micro Magazine, Volume: 17 Issue: 4, July-Aug., 1997.

[11] J. Duato, S. Yalamanchili, L. Ni, "Interconnection Networks - An Engineering Approach", Morgan Kaufmann Publishers, 2003.

[12] L. Kleinrock, "Queueing Systems", John Wiley & Sons Inc, 1975.

[13] J. Xu, W. Wolf, J. Henkel, S. Chakradhar, T. Lv, "A Case Study in Networks-on-Chip Design for Embedded Video", Design, Automation and Test in Europe, March, 2004.

[14] A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. Oberg, M. Millberg, and D. Lindqvist, "Network on chip: An architecture for billion transistor era", In Proceeding of the IEEE NorChip Conference, November 2000.

[15] K. Lahiri; A. Raghunathan; S. Dey, Efficient exploration of the SoC communication architecture design space ICCAD-2000. IEEE/ACM International Conference on CAD, pp.424-430, 2000.

[16] Li Chen, "Software-Based Self-Test and Diagnosis for Processors and System-on-Chips", PhD thesis, University of California, San Diego, June, 2003.

[17] The future of wires Ho, R.; Mai, K.W.; Horowitz, M.A., "The future of wires", Proceedings of the IEEE , Volume: 89 Issue: 4 , April 2001, Page(s): 490 -504, 2001.

[18] S. S. Mukherjee, P. Bannon, S. Lang, A. Spink, and D. Webb, "The Alpha 21364 network architecture", IEEE Micro, 22(1), 2002.

[19] P. Bhojwani and R. Mahapatra, "Interfacing Cores with On-chip Packet-Switched Networks", IEEE Proceedings on VLSI Design, New Delhi, pp. 382-387, Jan 2003.

[20] P. Pande, C. Grecu, A. Ivanov, R. Saleh, "Design of a Switch for Network on Chip Applications," IEEE International Symposium on Circuits and Systems, ISCAS 2003, Vol. V, pp. 217-220, 2003.

[21] E. Rijpkema, K. Goossens, A. Radulescu, J. Dielissen, J. van Meerbergen, P. Wielage, E. Waterlander, "Trade offs in the design of a router with both guaranteed and best-effort services for networks on chip Design", Design Automation and Test in Europe (DATE) Conference, pp.350-355, 2003.

[22] M. Forsell, "A scalable high-performance computing solution for networks on chips", IEEE Micro Magazine, Volume: 22 Issue:5, Page(s):46-55. 2002.

[23] T. Dumitras, R. Marculescu, "On-chip stochastic communication" Design, Automation and Test in Europe Conference, pp.790-795, 2003.

[24] S. Wang, S. T. Chakradhar, K. Balakrishnan, "Reconfigurable Embedded Core Test Protocol" Proceedings of ASP-DAC, January, 2004.

[25] M. Galles, "Spider: a high-speed network interconnect", IEEE Micro, Volume: 17 Issue: 1, Page(s): 34 -39, 1997.

[26] B. Vermeulen, J. Dielissen, K. Goossens, C. Ciordas, "Bringing Communication Networks On Chip: Test and Verification Implications", IEEE Communications Magazine, Volume: 41 Issue: 9, Page(s): 74-81, September 2003.

[27] R. Hofmann, B. Drerup, "Next generation CoreConnect processor local bus architecture", Annual IEEE International ASIC/SOC Conference, pp.221 -225, 2002.

[28] D. Wingard, "MicroNetwork-based integration for SOCs", Design Automation Conference, pp.18-22 June 2001.

[29] Silicore, http://www.silicore.net

[30] B. Cordan, "An efficient bus architecture for system-on-chip design", IEEE Custom Integrated Circuits, pp.623-626, 1999.

[31] http://www.altera.com

[32] Wassal, A.G., Hasan, M.A. "Low-power system-level design of VLSI packet switching fabrics", IEEE Transactions on Computer Aided Design, pp.723-738, June 2001.

[33] C. Patel, S. Chai, S. Yalamanchili, D. Schimmel, "Power constrained design of multiprocessor interconnection networks", Int'l Conference on Computer Design, pp. 408-416, 1997.

[34] Langen, D., Brinkmann, A., Ruckert, U. "High level estimation of the area and power consumption of on-chip interconnects", IEEE Int'l ASIC/SOC Conference, pp. 297-301, 2000.

[35] T. T. Ye; L. Benini; G. De Micheli, "Analysis of Power Consumption on Switch Fabrics in Network Routers", Design Automation Conference, (DAC 2002), Page(s): 524-529, 2002.

[36] J. Hu, R. Marculescu, "Energy-aware mapping for tile-based NoC architectures under performance constraints", Design Automation Conference, IEEE/ACM Proceedings of the ASP-DAC 2003, Asia and South Pacific, pp.233-239, 2003.

[37] Y. Zhang, R. Y. Chen, W. Ye, M.J. Irvin, "System level interconnect power modeling", ASIC Conference 1998. Proceedings. Eleventh Annual IEEE International, pp.289-293, 1998.

[38] H. S. Wang, X. Zhu, L. S. Peh and S. Malik, "Orion: A power-performance simulator for interconnection networks", MICRO 35, pp.294-305, Nov. 2002.

[39] D. Wiklund and D. Liu, "Design of a system-on-chip switched network and its design support", ICCCAS, pp. 1279 -1283 (vol.2), 2002.