A Scalable, Timing-Safe, Network-on-Chip Architecture with an Integrated Clock Distribution Method

Tobias Bjerregaard
Teklatech
Diplomvej, building 377
2800 Lyngby, Denmark
tob@teklatech.com

Mikkel Bystrup Stensgaard
TU of Denmark (DTU), IMM
Richard Petersens Plads
2800 Lyngby, Denmark
mikkel.stensgaard@imm.dtu.dk

Jens Sparsø
TU of Denmark (DTU), IMM
Richard Petersens Plads
2800 Lyngby, Denmark
jsp@imm.dtu.dk

Abstract

Growing system sizes together with increasing performance variability are making globally synchronous operation hard to realize. Mesochronous clocking constitutes a possible solution to the problems faced. The most fundamental of problems faced when communicating between mesochronously clocked regions concerns the possibility of data corruption caused by metastability. This paper presents an integrated communication and mesochronous clocking strategy, which avoids timing related errors while maintaining a globally synchronous system perspective. The architecture is scalable as timing integrity is based purely on local observations. It is demonstrated with a 90 nm CMOS standard cell network-on-chip design which implements completely timing-safe, global communication in a modular system.

1. Introduction

Physical issues of deep submicron technologies as well as design complexity issues push for a modularized design approach. There is a general consensus that the challenges of developing future billion transistor system-on-chip (SoC) designs can be addressed by plugging together individually verified blocks (IP cores), using shared, segmented chip-area interconnection networks [5][4][10][8], so called Networks-on-Chips (NoCs). NoCs facilitate a truly modular and scalable design approach for SoC.

Meanwhile, due to high clock speeds and increasing performance variability of new technologies, strict global synchrony is becoming prohibitively difficult to implement in large chips [1]. Increasingly complex clock distribution techniques used to minimize clock skew, e.g. involving distributed active skew control [22][11], are taking an increasing portion of the total power consumption. Ultimately, failure to live up to the challenges of implementing a clocking signal properly, may render an entire chip dysfunctional, e.g. due to hold time violations.

In this paper we describe an integrated communication and clock distribution scheme which enables modular and scalable SoC designs, while maintaining a globally synchronous system perspective. The method handles timing issues by distributing the clock along the branches of a NoC with a tree topology and ensuring data timing integrity along the branches of the tree. Since timing integrity is secured locally, the architecture is scalable. The NoC features a novel micro-level flow control scheme with integrated fine-grained clock gating. The architecture expresses graceful performance degradation, meaning that its timing can be made robust under any amount of performance variability, by lowering the clock frequency. In Section 2 we provide some background on prior solutions to securing global timing in modularized SoCs. In Section 3 we describe the basics of the presented NoC architecture. In Section 4 we explain the timing of the architecture and Section 5 describes the flow control scheme. In Section 6 we demonstrate the architecture with a 90 nm standard cell design. Section 7 proposes future work, and Section 8 provides a conclusion.

2. Background

Despite the problems of distributing a global clocking signal, many published NoCs assume a completely synchronous timing view [9][14][18]. Among drawbacks of globally synchronous operation are a large peak current at the clock edge, leading to ground bounce and voltage drops, which in turn induce jitter in both clock and data. Also, increasing performance variability, due to process variations, make it ever more difficult to match the delay in different branches of a global clock tree. To reduce the delay variations, large power hungry buffers are used. Most importantly, globally synchronous systems are not scalable, as the challenges faced only become worse as the system grows, and as fabrication technologies scale down.

978-3-9810801-2-4/DATE07 © 2007 EDAA

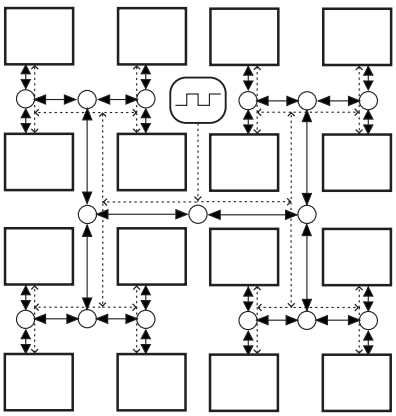

Figure 1. Tree-based IC-NoC architecture.

Partitioning chip functionality into IP cores allows a timing-wise partitioning as well. By the concept of *globally asynchronous locally synchronous* (GALS) systems, a SoC is implemented as synchronous islands which communicate asynchronously [16]. A subclass of NoCs are implemented entirely using asynchronous (clockless) circuit techniques [2][6][3][19], synchronizing with individual IP core clocks at the network boundary. A drawback of this approach is limitations in the availability of design tools for asynchronous circuits. Also, the lack of global clock-level synchronization complicates the implementation of guaranteed communication services [7].

Alternatively, mesochronously clocked systems employ a single clock across the entire system, but at arbitrary phases. Like GALS, mesochronous systems leverage existing synchronous design tools and know-how, while avoiding drawbacks of strict global synchrony. Power dissipation in the clock distribution network is significantly reduced, since power hungry clock buffers, used to reduce global clock skew, are avoided. Also, mesochronous systems are scalable: since the phase difference between clock-phase regions is arbitrary, the systems can have any size.

In the most general form of mesochronous clocking, nothing can be said concerning the phase alignment between clocks in different parts of the system. Thus metastability may occur when passing data between clock phase domains. In [15] metastability detectors are implemented, and a controlled delay is introduced to the data path until no transmission errors are detected. In [20], instead the delay of the clock signal is adjusted, to the same effect. Similarly [13] presents a scheme for detecting whether the sampling clock edge falls within the switching zone of the incoming data signal. If it does, the data is sampled on the negative clock edge instead. Common for these schemes however is that complex phase detection is needed, making the circuit overhead non-negligible. Another drawback is the need for an initialization phase.

In [17] the aim is to *contain* the clock skew in a mesochronous system. A directed clock distribution method is presented, in which the clock is distributed along the edges of a mesh-type NoC. It is shown how data may ride on the clock wave up and down the edges along which the clock is distributed. However the implications of transmitting data on edges orthogonally to the clock distribution is not fully covered. It appears that in order to allow such communication, delay balancing along the clock edges, similar to that needed when implementing a global clock tree, is needed. Also, as the delay must be balanced in terms of fractions of the clock period, the communication is functional in a very slim clock frequency range.

3. The IC-NoC Architecture

The IC-NoC architecture (Integrated Clocking Network-on-Chip) presented herein integrates global communication and clock distribution, implementing scalable, timing-safe, synchronous, on-chip communication. Figure 1 shows a basic, homogeneous instance of the architecture. By distributing the clock along the branches of the tree, the clock skew between the nodes of the network is predictable and furthermore correlated with the delay of the data. Due to the tree topology required by the clock distribution, no converging paths are allowed in the network.

Only few NoCs employ tree-based architectures, designers arguing that routing in a tree is less flexible than in a mesh. Rightfully so, situations may occur in which data needs to be routed to the very root of the tree, in order to get to a destination quite close geographically to the source. With proper application mapping however, cores which communicate a lot will be clustered and locality can be exploited to a much larger degree than in a mesh. For example, communication between two neighboring cores in a binary tree only has to pass a single 3x3 routers, hence decreasing both latency and power compared to a mesh. Also, though routing data in a tree may require transgressing a longer physical wire length, the worst-case number of hops is smaller than in a mesh ($2logN-1$ vs. $2\sqrt{N}$). In [12] it was shown that even with no link power reduction methods (e.g. low swing or encoding to reduce power dissipation in the wires), a tree is a power-wise better choice than a mesh for a 0.18 μm CMOS technology. This holds for several generations of future technologies. Finally, in a tree there are fewer routers than in a mesh, thus the area and the leakage current of the NoC is minimized.

A major advantage of the IC-NoC architecture is the fact that the system is completely scalable, as timing integrity is ensured locally, on a node-to-node basis. Additionally, once the node-to-node timing is shown to hold, the system can be conceived as globally synchronous, from a system-level perspective. Hence, a system designer does not need to take into account its mesochronous nature.

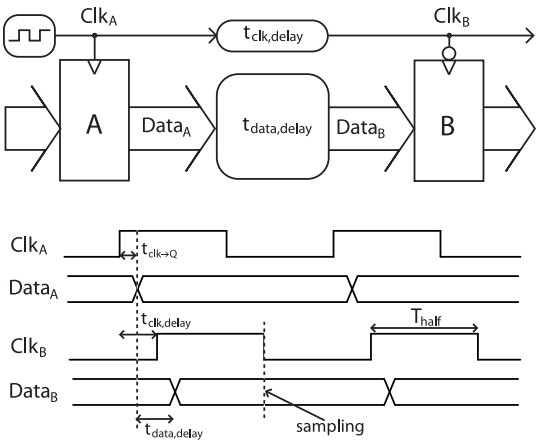

Figure 2. Downstream communication.

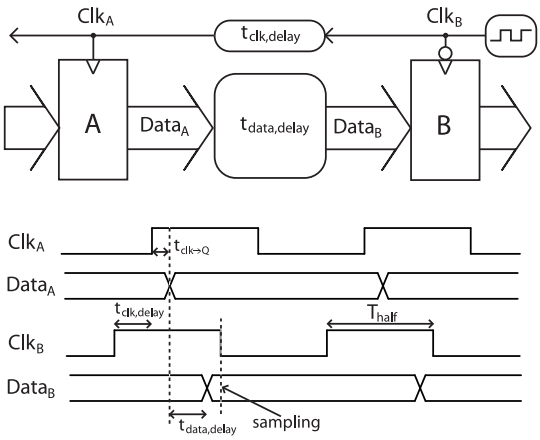

Figure 3. Upstream communication.

4. Link Timing

Data is communicated either in the same or opposite direction of the clock signal. We denote this downstream and upstream communication, respectively. Network nodes are clocked at alternating clock edges. Setup and hold time requirements are calculated in the following, and it is shown how these are dependent on the clock period. By slowing down the clock, both setup and hold times can be controlled. Hence performance is gracefully declining, as timing is guaranteed to hold at *some* clock frequency, no matter what the process variation is. The system is, so to say, correct by construction. As shown in Section 5 the back pressure flow control scheme of the IC-NoC benefits additionally from utilizing both phases of the clock.

In the following t_{setup}, t_{hold}, and $t_{clk\rightarrow Q}$ denote the setup time, hold time and clock-to-data output propagation delay for the registers. For simplicity, the contamination delay is disregarded. Typical values for a 90 nm standard cell flip flop are $t_{setup} = 60\ ps$, $t_{hold} = 20\ ps$, and $t_{clk\rightarrow Q} = 60\ ps$. T_{half} is half the clock period. We assume a 50% duty cycle.

For downstream transmissions the data experiences positive clock skew. Figure 2 illustrates this scenario. For the setup time of register B not to be violated, the following inequality must hold:

$$\begin{aligned} t_{clk\rightarrow Q} + t_{data,delay} + t_{setup} &< T_{half} + t_{clk,delay} \\ t_{data,delay} - t_{clk,delay} &< T_{half} - t_{clk\rightarrow Q} - t_{setup} \\ \Delta_{diff} &< T_{half} - t_{clk\rightarrow Q} - t_{setup} \end{aligned} \quad (1)$$

where Δ_{diff} is the delay difference between data and clock. Additionally, for the hold time to be respected:

$$\begin{aligned} t_{clk\rightarrow Q} + t_{data,delay} + T_{half} &> t_{hold} + t_{clk,delay} \\ t_{data,delay} - t_{clk,delay} &> t_{hold} - T_{half} - t_{clk\rightarrow Q} \\ \Delta_{diff} &> t_{hold} - T_{half} - t_{clk\rightarrow Q} \end{aligned} \quad (2)$$

The two equations (1) and (2) bound the tolerable difference in delay between data and clock:

$$t_{hold} - T_{half} - t_{clk\rightarrow Q} < \Delta_{diff} < T_{half} - t_{clk\rightarrow Q} - t_{setup} \quad (3)$$

At a clock frequency of 1 GHz, using the typical values for a 90 nm standard cell flip flop, (3) evaluates to

$$-540\ ps < \Delta_{diff} < 380\ ps \quad (4)$$

For upstream transmissions the data experiences negative clock skew. Figure 3 illustrates the clock thus being distributed in the opposite direction of the data. For the setup time of register B not to be violated, the following inequality must hold:

$$\begin{aligned} t_{clk\rightarrow Q} + t_{data,delay} + t_{setup} &< T_{half} - t_{clk,delay} \\ t_{data,delay} + t_{clk,delay} &< T_{half} - t_{clk\rightarrow Q} - t_{setup} \\ \Delta_{sum} &< T_{half} - t_{clk\rightarrow Q} - t_{setup} \end{aligned} \quad (5)$$

where Δ_{sum} is the sum of the data and clock delays. Additionally, for the hold time to be respected:

$$\begin{aligned} t_{clk\rightarrow Q} + t_{data,delay} + T_{half} &> t_{hold} - t_{clk,delay} \\ t_{data,delay} + t_{clk,delay} &> t_{hold} - T_{half} - t_{clk\rightarrow Q} \\ \Delta_{sum} &> t_{hold} - T_{half} - t_{clk\rightarrow Q} \end{aligned} \quad (6)$$

Using typical values for a 90 nm standard cell flip flop, the right hand side of (6) is always negative, and the upstream timing requirement limits to the setup time requirement (5). At 1 GHz, this evaluates to

$$\Delta_{sum} < 380\ ps \quad (7)$$

For a given 90 nm standard cell technology, a wire has a capacitance of $0.2\ pF/mm$ and a resistance of $0.4\ K\Omega/mm$. Dividing Δ_{sum} equally between clock and data delay such

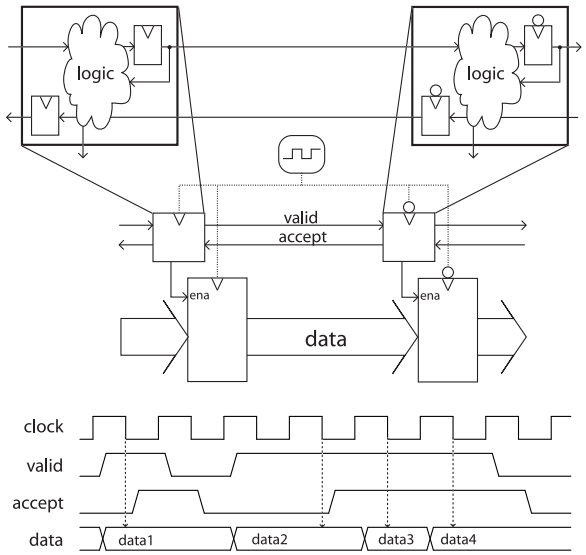

Figure 4. Handshaked pipelining using the two phases of the clock. Arrows show when data is captured by the consumer.

that each must maximally be 190 ps, this corresponds approximately to a 1.5-2 mm wire. This is further detailed in Section 6.

The flow control scheme presented in Section 5 requires transmissions both up- and downstream (handshaking signals), irrespective of the direction of the data flow. The downstream timing requirements are quite easy to satisfy, as these are dependent on the difference in clock and data delay, delays which can be matched. It is seen that the upstream timing represents the performance limiting factor. It is important to note that the skew tolerance can be made arbitrarily large by lowering the clock frequency.

5. Flow Control

In order to illustrate the generalized system timing concepts explained in Section 4, we have implemented an on-chip packet-routing network, the IC-NoC. The back pressure flow control scheme employed in the IC-NoC leverages important challenges of state-of-the-art SoC design, by facilitating fine-grained clock gating, micro-level flow control and skew tolerant links. This is required in order to enable a scalable architecture. Traditionally, in order to realize back pressure flow control, extra stall buffers are needed to absorb incoming data, when the forward path is congested. Alternatively, the pipeline should be clocked at double the speed of the data – at double clock frequency or using dual-edge triggered registers – reserving one cycle for data transfer, one for congestion control.

The protocol presented herein enables flow control, without the need for stall buffers, yet still clocking each pipeline stage only at the speed of the data. This is accomplished as shown in Figure 4, by clocking pipeline stages at alternating clock edges. Hence both edges of the clock are utilized, without needing dual-edge triggered registers.

We adapt concepts from asynchronous circuit design. Local back pressure flow control is inherent in asynchronous circuits, as the movement of data is governed by local handshaking mechanisms [21]. In order to realize such handshaking between a data producer and consumer, a minimum of two handshake phases are required; a request phase during which data is forwarded from the producer, and an acknowledge phase during which the consumer subsequently accepts the data. This is known in its general form as 2-phase handshaking. Adapting this for the synchronous domain, we utilize the two phases of the clock.

A *valid* signal is routed alongside the data, indicating its validity. In the reverse direction, an *accept* signal is used to indicate that the receiving stage has accepted the data, storing it in its register. Note that whereas control signals in asynchronous circuits are edge sensitive, our implementation is level sensitive, normal for synchronous circuits, using the clock edge as trigger event. Since the stages are clocked at alternating clock edges, it is possible to send the data, and receive acknowledgment from the next stage, within the same clock cycle. This allows transmitting of data at full clock speed along the pipeline, stop in an instance if congestion is detected, and resume transmission without delay once the congestion is resolved.

The control signals are used to derive an enable signal for the pipeline registers. If the forward path is congested, the registers will not be enabled. Similarly, if no data valid signal is sensed at the input, the registers will also not be enabled. Thus fine-grained clock gating is an inherent characteristic of the flow control method. This is particularly important in network-on-chip architectures, as traffic is expected to be of a bursty nature. This means that the network will lay idle for long periods, and power consumption during idleness is of a major concern.

As mentioned in Section 4, a handshake requires signaling both up- and downstream. Hence the performance is limited by (5), irrespective of the direction of the data flow.

6. Results

As a demonstration of the IC-NoC architecture we have implemented a 64 port packet-switched NoC for a 10 mm$\times$10 mm chip. The demonstrator system constitutes a homogeneous multiprocessor system as illustrated in Figure 5. It consists of 32 processing tiles, each with a microprocessor and a local memory. The prioritization within the routers is balanced such that a processor always has priority to accessing its local memory. The network has a 32-

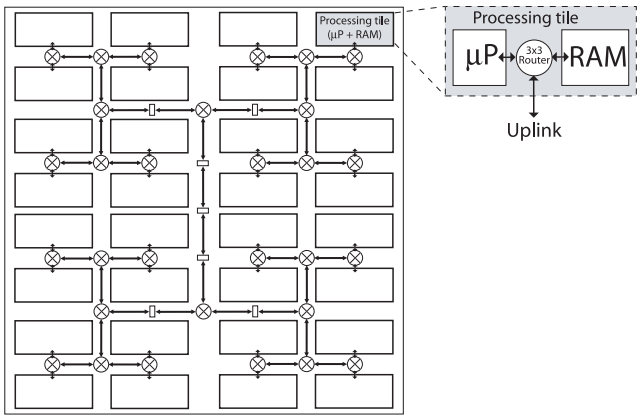

Figure 5. Demonstrator system.

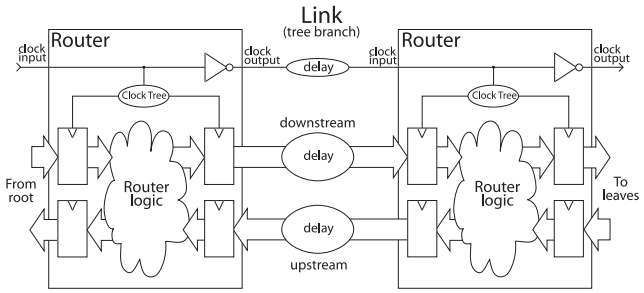

Figure 6. The clock is forwarded along the link between the routers.

bit wide data path. A commercially available 90 nm standard cell technology is used. Results are obtained by back-annotated simulations, using nominal timing parameters at 1 V supply.

Figure 6 shows the basics of two routers and a bidirectional link between them. Links are implemented as two independent, unidirectional handshake channels: upstream and downstream. The figure illustrates how the clock is inverted and forwarded on the link, implementing the 2-phase flow control and timing scheme described in Sections 4 and 5.

The baseline performance of the flow control scheme is measured by simulating a simple pipeline with pipeline stages placed head-to-head (clock and data wire delay both zero). The flow control logic and registers alone take 220 ps. Adjacent pipeline stages are clocked at opposite clock edges, and with control signal buffering the pipeline operates at up to 1.8 Ghz. Figure 7 illustrates the operating speed of a pipeline, with back-annotated timing from layout, as a function of the wire length between the stages (the length of pipeline *segments*). The area of a 32-bit pipeline stage is 0.0015 mm^2.

The routers are pipelined for optimal speed. This determines the forward latency of packets through the network: $2\frac{1}{2}$ cycles per 5×5 router and $1\frac{1}{2}$ cycle per 3×3 router. The 5×5 routers operate at 1.2 GHz, while 3×3 routers operate at 1.4 GHz. Matching the router and pipeline speeds, it is seen that the optimal pipeline segment length is 0.9 mm when using 5×5 routers and 0.6 mm when using 3×3 routers. The area of a 5×5 router is 0.022 mm^2 while the area of a 3×3 router is 0.010 mm^2.

The tradeoff between using 5×5 routers in a quad tree or 3×3 routers in a binary tree concerns the latency, area and throughput of the network. The quad tree has lower latency, as the latency of a 5×5 router is less than the latency of two 3×3 routers. Also it has lower area, as the area of a 5×5 router is less than that of three 3×3 routers. Finally a quad tree has a higher aggregate throughput, as it is possible to route from all inputs to all outputs of a 5×5 router in parallel. This is not possible in a subtree of three 3×3 routers. However, the binary tree has better local performance, as the latency between adjacent leaf nodes is shorter; only $1\frac{1}{2}$ cycles vs. $2\frac{1}{2}$ cycles in a quad tree. Also, the routers are more evenly spread out in a binary tree, so that links near the root are shorter and hence need less pipelining.

The difference however is marginal, and in a relatively small application like the one used herein as a demonstration of the IC-NoC architecture, we use only 3×3 routers in a binary tree topology. We target link segments of 1.25 mm near the root of the tree, and hence get a 1 GHz operating speed. The network was synthesized and wire delays, derived from pipeline layouts, were added according to the link lengths. It was shown to operate to full satisfaction with back-annotated timing. With a tree topology the area scales linearly with the number of network ports: $Area_{total} = (N-1)*Area_{router}+Area_{pipelines}$. The total area of the NoC is 0.73 mm^2, only 0.73% of the chip area.

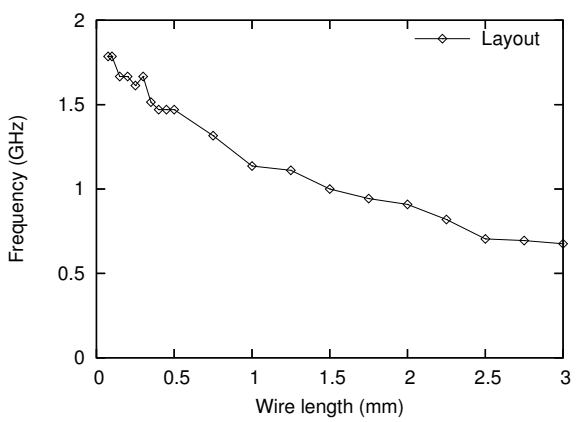

Figure 7. Clocking frequency as a function of the wire length between two pipeline stages.

7 Future Work

The IC-NoC architecture as introduced herein, opens up to a number of potential advantages. We are currently looking at several extensions to the basic architecture. First of all, the 2-phase flow control scheme can be modified to allow the use of latches instead of edge triggered registers. This will reduce the area as well as the power consumption. Secondly, we plan to introduce non-tree topologies by breaking rings using traditional mesochronous communication methods. This allows for much more flexibility while still leveraging the advantages of the presented architecture along the underlying tree. Finally, by the use of weighted skew variation on links, it is possible to distribute power surge temporally, by making sure that the leaves of the tree are not clocked within close temporal proximity.

8. Conclusion

We have presented an integrated clocking and communication architecture, the IC-NoC, which overcomes the challenges of implementing globally synchronous on-chip clocks. The architecture is scalable yet facilitates a globally synchronous system perspective in modular System-on-Chip designs. Scalability is achieved by distributing the clock signal along the branches of a segmented communication infra-structure with a tree topology. Data timing integrity is secured locally on a node-to-node basis, along the branches of the tree. The architecture features a novel flow control scheme which utilizes the two phases of the clock to provide micro-level handshaking. Furthermore fine-grained clock gating and skew tolerance is inherently enabled.

The architecture is demonstrated with a 90 nm IC-NoC design. The demonstrator system, implemented using commercially available standard cells, shows that operation (nominal timing parameters, 1 V supply) is possible at 1 GHz and beyond. This illustrates how the drawbacks of strict global synchrony can be overcome without losing the benefits.

References

[1] International technology roadmap for semiconductors (ITRS) 2003. Technical report, International Technology Roadmap for Semiconductors, 2003.

[2] J. Bainbridge and S. Furber. Chain: A delay-insensitive chip area interconnect. *IEEE Micro*, 22(5):16–23, October 2002.

[3] E. Beigne, F. Clermidy, P. Vivet, A. Clouard, and M. Renaudin. An asynchronous NOC architecture providing low latency service and its multi-level design framework. In *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC 2005)*, pages 54–63. IEEE, 2005.

[4] L. Benini and G. D. Micheli. Networks on chips: A new SoC paradigm. *IEEE Computer*, 35(1):70 – 78, January 2002.

[5] T. Bjerregaard and S. Mahadevan. A survey of research and practices of network-on-chip. *ACM Computing Surveys*, 38, March 2006.

[6] T. Bjerregaard and J. Sparsø. Implementation of guaranteed service in the MANGO clockless network-on-chip. *IEE Proceedings of Computers and Digital Techniques*, 153(4):217–229, July 2006.

[7] T. Bjerregaard and J. Sparsø. A scheduling discipline for latency and bandwidth guarantees in asynchronous network-on-chip. In *Proceedings of the 11th IEEE International Symposium on Advanced Research in Asynchronous Circuits and Systems*. IEEE, 2005.

[8] W. J. Dally and B. Towles. Route packets, not wires: On-chip interconnection networks. In *Proceedings of the 38th Design Automation Conference*, pages 684–689, June 2001.

[9] K. Goossens, J. Dielissen, and A. Radulescu. Æthereal network on chip: Concepts, architectures and implementations. *IEEE Design & Test of Computers*, 2005.

[10] A. Jantsch and H. Tenhunen. *Networks on Chip*. Kluwer Academic Publishers, 2003.

[11] N. Kurd, J. Barkatullah, R. Dizon, T. Fletcher, and P. Madland. Multi-GHz clocking scheme for intel pentium 4 microprocessor. In *Digest of Technical Papers. ISSCC. 2001 IEEE International Solid-State Circuits Conference, 2001.*, pages 404–405. IEEE, 2001.

[12] S.-J. Lee. Realization of an on-chip network for practical application to system-on-chip - a chip designer's view. In *Keynote at the IEEE International Symposium on System-on-Chip 2005*. IEEE, 2005.

[13] B. Mesgarzadeh, C. Svensson, and A. Alvandpour. A new mesochronous clocking scheme for synchronization in SoC. In *Proceedings of the 2004 International Symposium on Circuits and Systems (ISCAS '04)*, pages 605–608. IEEE, 2004.

[14] M. MillBerg, E. Nilsson, R. Thid, and A. Jantsch. Guaranteed bandwidth using looped containers in temporally disjoint networks within the Nostrum network on chip. In *Proceedings of the Design, Automation and Testing in Europe Conference (DATE 2004)*. IEEE, 2004.

[15] F. Mu and C. Svensson. Self-tested self-synchronization circuit for mesochronous clocking. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 48:129–140, 2001.

[16] J. Muttersbach, T. Villiger, K. Kaeslin, N. Felber, and W. Fichtner. Globally-Asynchronous Locally-Synchronous Architectures to Simplify the Design of On-Chip Systems. In *Proc. 12th International ASIC/SOC Conference*, pages 317–321, Sept. 1999.

[17] E. Nilsson and J. Öberg. Reducing power and latency in 2-d mesh NoCs using globally pseudochronous locally synchronous clocking. In *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis 2004*, pages 176–181. ACM, 2004.

[18] M. D. Osso, G. Biccari, L. Giovannini, D. Bertozzi, and L. Benini. Xpipes: A latency insensitive parameterized network-on-chip architecture for multi-processor SoCs. In *Proceedings of the 21st International Conference on Computer Design (ICCD03)*. IEEE, 2003.

[19] D. Rostislav, V. Vishnyakov, E. Friedman, and R. Ginosaur. An asynchronous router for multiple service levels networks on chip. In *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC 2005)*, pages 44–53. IEEE, 2005.

[20] I. Söderquist. Globally updated mesochronous design style. *IEEE Journal of Solid-State Circuits*, 38:1242–1249, 2003.

[21] J. Sparsø and S. Furber. *Principles of Asynchronous Circuit Design - a Systems Perspective*. Kluwer Academic Publishers, Boston, 2001.

[22] S. Tam, S. Rusu, U. N. Desai, R. Kim, J. Zhang, and I. Young. Clock generation and distribution for the first IA-64 microprocessor. *IEEE Journal of Solid-State Circuits*, 35:1545–1552, 2000.