SAME 99

Session : System-On-Chip

# A Scalable Architecture for System-On-Chip Interconnections

**Pierre Guerrier** **Alain Greiner**

**Université Pierre et Marie Curie**
Laboratoire d'Informatique de Paris VI
Architecture des Systèmes Intégrés et Microélectronique

{Pierre.Guerrier, Alain.Greiner}@lip6.fr

4 place Jussieu, F-75252 PARIS CEDEX 05, FRANCE

## Abstract :

*This paper reviews the emerging problems in the design of bus-based systems-on-a-chip, to show that they will severely limit their performances in the near future. In this context, we discuss the potential benefits ,for all aspects of deep submicron physical design, of turning to a different system interconnect scheme based on switch fabrics. Eventually we evaluate the impact of this interconnect solution on the behavioral design level, and conclude on the need to maintain backward compatibility at this level, with bus-oriented IP portfolios.*

## 1. Introduction

The architectural interconnection of a large number of hardware IP blocks is one of the concerns in System-Level Design. The availability of a standard interconnect solution is a prerequisite to the globalization of IP-based design. Such initiatives as the *VSI* Alliance [1] address the specification of interfaces for IP interconnection, but leave the implementation of the communication medium still open. This need is currently addressed in the industry by on-chip busses such as the OMI standard *PI-Bus* [2].

However, busses do not scale easily to sizes like 30 masters and throughputs in the 10-100 Gbit/s, and a novel approach to on-chip communication is needed. In this paper, we present our first investigations on a *Scalable, Programmable, Integrated Network* (SPIN) that could address this problem.

The paper is organized as follows. Section 2 will explain why deep submicron manufacturing and the Moore's Law hinder bus-based designs. Section 3 will give an overview of the SPIN architecture, and show how it helps tackling the integration issues. Section 4 analyzes the differences between bus and network at the behavioral level, and their impact on the system complexity.

## 2. Limitations of On-Chip Busses

### 2.1 Electrical Limitations

By definition, a bus-based interconnect uses long metal wires interconnecting a large number of sub-unit. Every sub-unit is susceptible of driving the data wires. Those wires have a very high capacitance because of their length, which adds to a large number of gate capacitances, growing with the complexity of the SoC (number of sub-units). Because they are not unidirectional, bus wires are not easy to buffer or pipeline. Therefore it is not possible to achieve high clock frequency on a bus beyond a certain number of interconnected sub-units, and bus wires may not benefit from process integration which increases interconnect delays [3].

The first solution to break this "size wall" and create high complexity SoC, is to cluster sub-units in separate busses interconnected by bridges, or in a hierarchy of busses. This is already used in many industrial designs. The clustering is case-specific, requiring non-recurrent engineering. And it is very cumbersome to cross more than one bridge for any transaction, as this would lock a large number of shared resources (all busses and bridges traversed) which are critical bottlenecks.

The IBM CoreConnect architecture [4] works around the capacitance limitation by implementing the bus as a *hub* instead of multipoint wires: the bus driver's signals are selected by a cascade of multiplexers and fed into a tree of buffers to reach all targets. There are no tri-state gates in this scheme. Higher frequencies can be achieved (200 Mhz) at the expense of silicon area. However other problems still limit the number of sub-units attached and the global system performance, as we see next.

### 2.2 Bandwidth Limitations

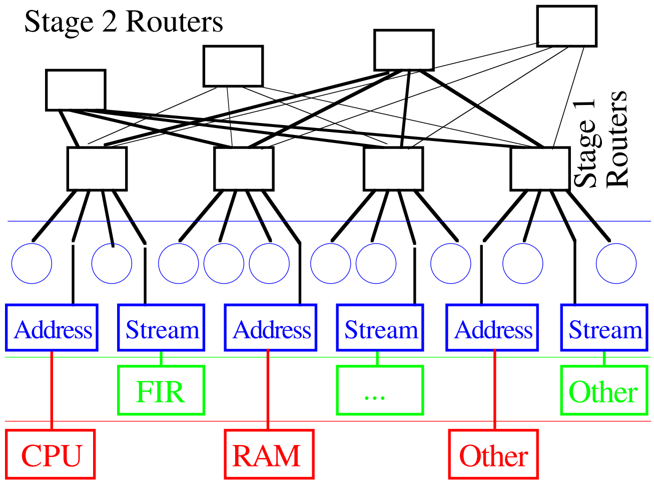
The *total* bandwidth available in a bus interconnect is equal to $w.f$ where $w$ is the width of the data path in bits and $f$ is its clock frequency (whether the bus is implemented as a hub or as multipoint wires). This trivial equation shows that present busses may deliver at most about 10 Gbit/s. This bandwidth is *shared* between all the attached sub-units. There are commonly more than a dozen, which means that they cannot operate concurrently unless their individual data throughputs are constrained below 1 Gbit/s. This figure is optimistic given that bus arbitration is not perfect and wastes a portion of the available clock cycles, especially for larger systems. Unfortunately higher data rates will be required by future SoC, e.g. for Gigabit Ethernet, IEEE 1394, HDTV and other multimedia features.

Multiple local busses don't solve the problem at its roots as the bus bridges will remain shared resources limiting the total system bandwidth. Busses must increase $w.f$ to keep up. Since $f$ is almost fixed for the reasons we explained above, designers have started increasing $w$, to values like 1024 bits for the AMBA specification [5]. However, it is our belief that such busses will be very hard to use efficiently: any data transfer that is smaller than a CPU cache line or an embedded DRAM row, would waste most of the available bus width.

## 3. On-Chip Networking for Deep Submicron

### 3.1 Architecture Overview

Our project of *Scalable, Programmable, Integrated Network* (SPIN), consists in a toolbox of macro-cells and methods for fast design of high-performance, versatile interconnections for systems-on-chip. The core technology inside SPIN, is a packet-switched multi-stage network [6]. The main building block of this network is an 8x8 wormhole router, which naturally lends itself to firm-IP-based design. Duplicated instances of the router can form a scalable network using a fat-tree topology (on top of the figure), where the number of routers grows at most like ($n$.log $n$)/8 for an $n$-port network. The network ports are to be wrapped with special interfaces described in the next section (middle layer of the figure). Eventually, different kinds of application IPs are plugged into the interfaces (bottom layers).




**Synopsis of Network Features**

The SPIN router is in essence similar to existing discrete component routers, notably [7] which was designed at our laboratory and is now a commercial product. But the micro-architecture has been specifically re-designed for an integrated environment. In the case of SPIN, the switched communication channels are parallel datapaths instead of serial links, because wiring volume is almost unlimited on-chip. The routing tables are replaced with tiny logic functions, as the topology of the network is fixed and regular. The buffering can be limited to only a few kilobits, because the packets to be routed are the size of a bus burst (much smaller than typical network frames). The core switching functionality remains implemented by a crossbar, but it is not full as some connections do not make sense within the SPIN topology.

### 3.2 Physical Implementation Strengths

The router macro-cell is expected to occupy only approximately 1 $mm^2$ in a .25 µm, 6 metal layer process, by taking advantage of massive over-cell routing of the crossbar wires. The clocking target is a modest 100 Mhz, due to possible crosstalk coupling in the crossbar wires. This means that a 32-port network (16-routers) is already affordable with such process, and larger systems will be feasible with future integration.

The data-paths between routers are full duplex 32-bit wide point-to-point links. All multipoint wires are eliminated. The flow control mechanism on these links is credit-based and tolerates arbitrary latency to relax global timing constraints for deeply submicronic ICs. It is possible to buffer and/or pipeline the global wires without impacting the functionality of the network. This way, the routers can be spread at any distance from each other on the chip, and the target frequency can always be achieved.

The wiring of the links and the router buffers can be efficiently tested altogether by a specific BIST. Testing is based on a daisy-chaining of all edges in the fat-tree graph. The chain can then be fed with a pattern generator. Eventually a loop-back BIST is used for the crossbars. Such pseudo-functional tests eliminate scanpaths, which have too much area overhead for router buffers, and are too slow for networks with potentially tens of thousands of global wires.

### 3.3 Scalable Bandwidth with Low Latency

The fat-tree is a non blocking network topology: it is always possible to carry on any set of simultaneous communications through the network, as long as they are non-conflicting (i.e. the unit link throughput is not exceeded on any network output). The total bandwidth is equal to $w.f.n$ where $n$ is the number of network ports. This bandwidth grows with the system, and every sub-unit is always offered the same $w.f$ throughput whatever the size of the system into which it is plugged. For $w = 32$ bits and $f = 100$ Mhz, every sub-unit has a peak bandwidth of 3 + 3 Gbit/s (with simultaneous input + output). This is a decisive advantage for high-bandwidth applications such as the presentation stages of HDTV.

Furthermore, packets are delivered after a handful of routing hops, because the diameter of the network is logarithmic. Each hop has only 4-cycle latency, thanks to the wormhole design of the router. In this design, the header of a packet is forwarded ASAP, without waiting for the trailer. A wormhole network can then deliver data with a typical latency of some 100 ns under light load condition. This compares favorably with the delay of a smart bus arbiter with a large number of masters.

## 4. Network Interface Behavior

### 4.1 Aspects of Network Behavior

The network packets natively provide a lossless datagram service. The fat-tree network featuring adaptivity, these datagrams may be delivered out-of-order. This means that two packets $P_1$ and $P_2$ sent in this order by A to B, may be received by B in the order $P_2$, $P_1$. This is of course completely different from a bus, where the order of transactions is strictly identical from the point of view of any sub-unit connected to the bus.

All transactions being visible from all sub-units, cache consistency can be enforced by means of snooping the bus. This simple method is not suitable for a switched network, where there is no central point for snooping serialized transactions. It is not possible either to have continuous, zero-dead-time dialogues over a switch network because it only permits a split transaction model with alternated requests and replies. In contrast, the master of a bus can instantly feed and fetch data into a target.

### 4.2 Requirements for IP Reuse

IP-based design is presently a major emerging paradigm. Therefore any new communication template has to be in-line with IP-oriented methods, one of the prerequisites being to maintain compatibility with legacy (digital) IPs so that they can be reused directly. Those IPs fall in several categories, including:

- bus-oriented IPs like RAMs, DMA I/Os, DSPs, CPUs and all the associated software IPs;
- dataflow-oriented IPs such as the high-throughput signal processors of [8];

For instance, VSI defines a standard interface for bus-oriented IPs, which is a sensible goal to meet for any interconnect. Dataflow-oriented IPs require less flexibility from the interconnect as their communication patterns can often be modeled by static flow graphs, but they demand much more bandwidth and quality-of-service. [9] illustrates this case and how an interconnect can be designed to best address it.

We set ourselves the goal of providing interoperation with these two kinds of IPs, because of course no IP provider will take on his side the effort of bridging the semantic gap between packet switching and those legacy I/O paradigms.

### 4.3 Hiding the Network behind Wrappers

Front-end interfaces (wrappers) on the network terminals can hide its internal behavior from the connected IP by providing high-level services, such as the VCI interface for bus-IPs or simple FIFOs for stream IPs.

A protocol has to be implemented in hardware by wrappers at both ends of an address-space transaction (or a point-to-point stream connection). We have started investigations on protocols with low-latency, low-bandwidth overhead, and suitable for lightweight hardware implementations. Distributed Memory multicomputers provide a large array of protocols for address-space transactions, which we can leverage to meet our particular needs. We have also found a streaming protocol in the large class of sender-based network protocols.

We have started implementing those protocols in hardware in order to evaluate the silicon overhead for compatibility. We expect that the cost of wrappers will be similar to the cost of the network itself. This seems to make it less attractive, but one must remember that any on-chip bus will also require VCI-wrappers. Dataflow processors will also require expensive address generators to interface to a bus. Therefore we expect that the total cost of a networked solution will be competitive with on-chip busses.

## 5. Conclusion

We have proposed an architecture for the interconnection of heterogeneous silicon IPs in systems-on-a-chip. This architecture uses packet-switching to address the performance and quality-of-service requirements of such applications as HDTV, with an unmatched scalability towards large and complex systems. It should altogether relieve the burdens of chip-level routing and timing.

Eventually, maintaining compatibility with pre-existing IPs designed for bus or data-flow interconnects appears to offer the most challenging problems of on-chip switching networks. We have proposed a possible solution based on wrappers emulating various legacy communication protocols.

## Bibliography


[1] Virtual Socket Interface Alliance, *On-Chip Bus Attributes*, 1999, `www.vsia.com/library/specs/summary.htm` (document access may be limited to the abstract for non-members).

[2] Open Microprocessor Initiative, *PI-Bus Draft Standard Specification*, 1994, `www.omimo.be/ftp/standard/omi324.ps`

[3] K. Keutzer, "*Chip Level Assembly (and not the Integration of Synthesis and Physical) is the Key to DSM Design*", Proceedings of the ACM/IEEE International Workshop on Timing Issues in Specification and Synthesis of Digital Systems (τ'99), Monterey, CA, USA, March 1999.

[4] IBM Company, *CoreConnect Product Brief*, 1998, `www.chips.ibm.com/products/powerpc/cores/crcon_pb.html`

[5] ARM Company, *Advanced Microprocessor Bus Architecture Specification*, 1998, `www.arm.com/Pro+Peripherals/AMBA/`

[6] C. Clos, "*A Study of Non-Blocking Switching Networks*", in Bell Systems Technical Journal, volume 32, March 1953, pp. 406-424.

[7] B. Zerrouk, V. Reibaldi, F. Potter, A. Greiner, A. Derieux, "*RCube : A Gigabit Serial Link Low Latency Adaptive Router*", in Records of the IEEE Hot Interconnects IV$^{th}$ Symposium, Palo Alto, CA, USA, August 1996.

[8] P. Lippens, J. van Meerbergen, A. van der Werf, et al., "*Phideo: A Silicon Compiler for High-Speed Algorithms*", in Proceedings of the EDAC, Amsterdam, The Netherlands, February 1991, pp. 436-441.

[9] J. Leijten, J. van Meerbergen, A. Timmer, J. Jess, "*Stream Communication between Real-Time Tasks in a High-Performance Multiprocessor*", in Proceedings of the DATE'98 Conference, Paris, France, February 1998.