Structured Interconnect Architecture: A Solution for the Non-Scalability of Bus-Based SoCs

Cristian Grecu, Partha Pratim Pande, André Ivanov, Res Saleh
Department of Electrical and Computer Engineering
SoC Research Lab, University of British Columbia, 2356 Main Mall, Vancouver V6T1Z4, Canada
{grecuc, parthap, ivanov, res}@ece.ubc.ca

ABSTRACT

Multi-Processor (MP-SoC) platforms are emerging as the latest trend in SoC design. Monolithic bus-based interconnect architectures will not be able to support the clock cycle requirements of these high performance SoCs. Systems having multiple smaller buses, integrated through repeaters or bridges, are possible alternatives. But these kinds of solutions are ad-hoc in nature. By adopting a more structured network-based design paradigm, specific clock cycle requirements can easily be met. The precise focus of this paper is to show how the butterfly fat tree (BFT) can meet this objective when used as the overall MP-SoC interconnect architecture, thereby offering an attractive alternative for SoC interconnect that does not suffer from the non-scalability aspect of the buses in regards to the clock cycle.

Categories and Subject Descriptors
B7.1 [**Integrated Circuits**]: Types and Design Styles – VLSI.

General Terms
Design, Performance
Keywords: MP-SoC, BFT, Pipelining, Bus, Scalability.

1. INTRODUCTION AND MOTIVATION

According to recent publications [1], the emergence of SoC platforms consisting of large, heterogeneous sets of embedded processors is imminent. A key element of such multiprocessor SoC (MP-SoC) platforms [1] is the *interconnect topology*. To date, the most frequently used on-chip interconnect architecture is the shared medium arbitrated bus.

The advantages of shared-bus architecture are simple topology, low area cost, and extensibility. 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 exceed the specified clock cycle. One solution for such cases is to split the bus into multiple segments and introduce a hierarchical architecture. We consider that multiple forms of buses connected through a hierarchical architecture can be viewed as ultimately converging to a structured form of network. A common problem shared by all interconnect topologies implemented in ultra deep submicron (UDSM) technology is the communication latency that arises from non-scalable global wire delays.

Even with repeater insertion, the intra-chip propagation delay can exceed the limit of one clock cycle [2] [3].

Based on the premise that structured solutions can be more easily implemented and automated, our approach to the problem aims at developing highly structured solutions that, by their nature, can contribute significantly to easing the design of large multi-core SoCs. Here, we essentially propose that the on-chip interconnect topology 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 among one other through the help of intelligent switches. As such, the switches can be considered as infrastructure IPs (I^2Ps) [4] providing a robust data transfer medium for the functional IP modules.

As an interconnect topology, we use a butterfly fat tree (BFT) architecture. In this architecture, the functional IP blocks reside at the leaves of the tree and the infrastructure IP blocks, i.e., switches, reside at its vertices. A set of functional IPs is connected to a neighboring switch. The specific aim of this paper is to compare the performance of the BFT with that of the bus-based architectures. We show how the clock cycle supported in a traditional bus-based system is limited by system size, whereas in a BFT-based architecture it is independent of system size.

2. RELATED WORK

A few on-chip micro network proposals for SoC integration can be found in the literature. Sonic's Silicon Backplane [5] is one example. This is a bus-based architecture in which the IP blocks are connected to the bus through specialized interfaces called *agents*. Each core communicates with an agent using the Open Core Protocol (OCP) [6]. MIPS Technologies has introduced an on-chip switch integrating IP blocks in a SoC [7]. Kumar [8] and Dally [9] have proposed mesh-based interconnect architectures. These architectures consist of an *m x n* mesh of switches interconnecting computational resources (IPs) placed along with the switches. In this case, the number of switches is equal to the number of IPs. Saastamoinen [10] describes the design of a reusable switch to be used in future SoCs. The interconnect architecture is however not specifically discussed. Guerrier and Greiner [11] proposed the use of a tree based interconnect (SPIN) and addressed system level design issues. In [12], [13] an interconnect architecture for a networked SoC, as well as the associated design of required switches and addressing mechanisms, are described. None of the above works quantitatively discusses the effect of the system size on the achievable clock cycle in a multi-core SoC.

3. PLATFORM-BASED MP-SoC

In a conventional digital ASIC design flow, several iterations of logic synthesis and physical design are typically required before convergence to design specifications is achieved. During synthesis, the capacitances of the global wires are generally unknown, and

wire-load models are typically used as estimators. The accuracy of such estimations is generally acceptable for short wires, but increasingly unacceptable as the wire delays reach levels where they correspond to a significant portion of the critical path delay.

For IP blocks constituted of 50 - 100K gates, such interconnect delay estimation related problems can be reasonably well tackled by existing CAD tools. Moreover, various publications show that global wires in blocks of 50 – 100K gates tend to scale with technology [14]. Therefore, problems in ultra deep submicron processes arising from non-scalable global wire delay and poor back annotation mechanisms can be assumed to be readily surmountable when these are limited to such blocks. There is plenty of evidence in support of IP blocks amounting to such sizes [15].

The trend for future SoC integration will be based on a design paradigm where an array of IP blocks (embedded processors) consisting of around 100K gates will be integrated according to a specific interconnect template [1]. Conventional single bus-based systems will not be able to support these large number of IP blocks. Smaller buses hierarchically interconnected through bridges offer possible solutions.

Here, instead of building a huge multi-processor SoC around an ad-hoc interconnection of multiple buses, we propose the adoption of an efficient parallel computing architecture tailored for the SoC domain. The BFT has found extensive applications in parallel machines. Among other attractive features, they are generally hardware efficient [16]. In view of our objectives, we consider the BFT as an interconnect template for a MP-SoC.

4. NON-SCALABILITY IN A BUS-BASED SoC

Here, we develop a model to analyze how the system size affects the achievable clock cycle in a bus-based SoC. In a bus-based SoC, multiple IP blocks share the same transmission media. This negatively impacts propagation delay, and, ultimately, the achievable clock cycle. This thus limits the number of IP blocks that can be connected to the bus, and thereby the system scalability. Each attached IP block will capacitively load the bus wires. For ease of analysis (but without loss of generality), we assume this extra capacitance to be evenly distributed along the wire and model it as a parasitic capacitance.

As many existing on-chip buses are multiplexer - based they are basically unidirectional and can therefore easily be buffered.

Attaching IP blocks to a bus adds an equivalent parasitic capacitance of C_P per unit length of wire. As a result, the driving capability of the bus will be negatively affected, and buffer insertion is required to accommodate multiple IPs while satisfying a propagation delay within one clock cycle. If a bus wire is divided into N segments, then each wire segment will have a capacitance of $(C_w + C_P)$ per unit length and the delay in the buffered bus wire will be given by equation (4.1), where t_{inv} is the delay of an inverter sized for equal rise and fall propagation delays, M is the size of the inverters, C_G is the gate capacitance of the minimum size inverter, R_{eqn} is the resistance of n-type diffusion region in $\Omega/\square$, R_w and C_w are the resistance and capacitance per unit length of the wire, respectively, and L_{bus} is the bus length.

$$D_{buffered} = N_{bus}t_{inv} + \left(C_G R_w M + \frac{(C_w + C_P)R_{eqn}}{M} \right) L_{bus} + 0.4R_w(C_w + C_P)\frac{L_{bus}^2}{N_{bus}^2} \quad (4.1)$$

Consequently the achievable clock cycle will be given as $1/D_{buffered}$. As C_p increases beyond a certain value the clock cycle exceeds the limit of 15FO4 delay units [17]. The value of C_p can be considered

as a metric for the scalability of a bus-based system as it relates to how many IP blocks can be appended to a bus before the delay exceeds one clock cycle. One way to deal with this scalability problem is to split the bus into smaller segments. A bus of shorter length can support more parasitic capacitance, C_p, arising from appended IP blocks. Figure 1 shows the variation of C_p with bus length.
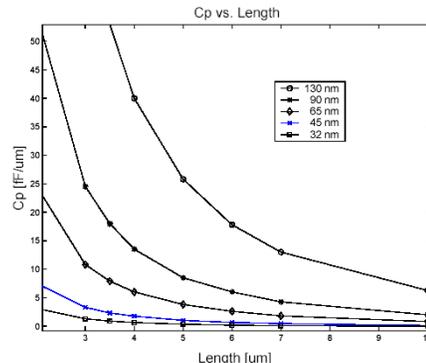

Figure 1: Variation of C_p with bus wire length.

From Figure 1 it is evident that more parasitic capacitance C_p can be supported by reducing the length of the bus, on the other hand, for any bus length, C_p remains constrained by the 15F04 limit.

5. A TREE-BASED STRUCTURED INTERCONNECT ARCHITECTURE

We proposed a novel interconnect template to integrate numerous IPs of an SoC following a butterfly fat-tree architecture. Figure 2 shows an SoC with **64** IPs integrated according to BFT, where the darker blocks denote the switches (I²Ps), and white squares represent the functional IP blocks. Each switch has four child ports and two parent ports. The total number of switches converges to N/2 [12] when the number of functional IP blocks in the system, N, is arbitrarily large.
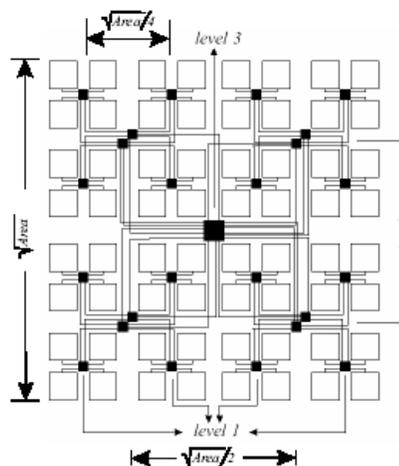

Figure 2: Floorplan of a 64-IP butterfly fat tree network.

In the case of 64 IPs the number of switches is 28 as shown in Figure 2. Data is assumed to be transferred among the IP blocks in terms of packets. Wormhole routing technique, where packets are divided into fixed length flow control units (flits) is adopted.

The switches (I^{2}Ps) consist of multiple stages. These stages together with the structured wires between the I^{2}Ps provide a highly pipelined communication medium. We can differentiate between two types of delays, namely intra- and inter-switch. Through detailed circuit level design and analysis, we can constrain the delay of each pipelined stage to be within the ITRS suggested limit of 15 FO4 delay units.

5.1 Switch-to-Switch Wire Delay

The inter-switch wire length depends on the number of levels in the BFT, which in turn depends on the system size. Assuming IP blocks consisting of 100K gates, Table 1 shows the maximum number of 100K gates IP blocks that can be integrated in a single SoC in different ITRS technology nodes [17]. As reported in Table 1, as the number of IP blocks increases, the number of required levels in the butterfly fat-tree also increases. Table 1 reports the number of required levels for each technology node.

Table 1: **Maximum number of IP blocks (100K gates/IP block) and corresponding fat-tree levels as technology scales**

Technology Node	Max. No. of IPs	No. of Fat-Tree levels
130 nm	500	6
90 nm	1000	7
65 nm	2500	9
45 nm	7500	10
32 nm	10000	11

The wire length between switches in the butterfly fat-tree architecture depends on the levels of the switches. From Figure 2, this inter-switch wire length is given by the following expression:

$$w_{a+1,a} = \frac{\sqrt{Area}}{2^{levels-a}} \qquad (5.2)$$

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). The die size is assumed to remain unchanged at 20 mm [17].

We can compute the intrinsic *RC* delay of a wire according to the equation below [18]:

$$D_{unbuffered} = 0.4R_wC_wL^2 \qquad (5.3)$$

where L is the wire length. In different technology nodes, *FO4* can be estimated as $425*L_{min}$ [ps] where L_{min} is the minimum gate length in each technology node [17]. For long wires, the intrinsic delay will easily exceed this *15FO4* limit. In those cases, the delay can, at best, be made to increase linearly with wire length by inserting buffers. If the wire is divided into n segments and n inverters inserted, then the total delay of the buffered wire will be according to the following expression [18], where m is the size of the inverters:

$$D_{buffered} = nt_{inv} + \left(C_GR_wm + \frac{C_wR_{eqn}}{m}\right)L + 0.4R_wC_w\frac{L^2}{n} \qquad (5.4)$$

From equation (5.3), the length of global wires (inter-switch connections), which require buffering, can be determined. It is noticed that most of the inter-switch wires need not be buffered [19]. Consequently, the inter-switch propagation delay always remains within one clock cycle.

The preceding inter-switch wire length and delay analysis and corresponding results do not strongly depend on the IP block size assumption. If the number of gates in the IP blocks were to largely

exceed 100K gates, or were much smaller than 50K, then the total number of IP blocks in the SoC would vary accordingly. Consequently, only the number of levels in our template would change. The inter-switch wire length and delay would remain largely unaffected.

5.2 Switch Design and Delay Analysis

According to Figure 2 the switch has six ports. In order to have a considerably high throughput, we use a virtual channel switch, where each port of the switch has four parallel buffers [13]. The different components of the switch are shown in Figure 3.
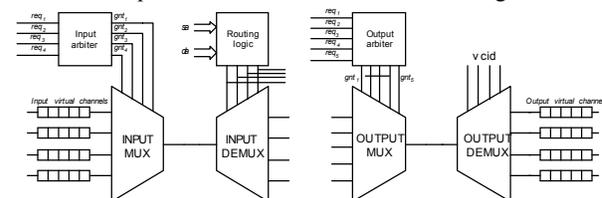

Figure 3: Block diagram of a switch port.

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 therefore necessary to allow only one virtual channel to access a single physical port. As there are four virtual channels corresponding to each input port, we need a 4:1 arbiter at the input. Similarly, flits from more than one input port may simultaneously try to access a particular output port. Consequently, on the output side, we need a 5:1 arbiter since among the six ports of the switch; any five may try to access a particular output port [13]. 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 sequence of the processes is: (1) **Input Arbitration;** (2) **Routing;** and (3) **Output Arbitration**. In the case of **body** flits, **Switch Traversal** replaces the routing process as the routing decision based on the header information is maintained for the subsequent body flits as indicated in Figure 4.

The blocks involved in the input arbitration process are the 4:1 arbiter and the input mux; similarly, the blocks in the output arbitration process are the 5:1 arbiter and the output mux. The routing process is performed by the combination of the routing block and the input demux.
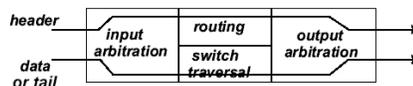

Figure 4: Intra-switch pipeline stages.

The switch traversal mainly involves the chain of four muxes and demuxes. Each of these processes takes place in different clock cycles. Consequently, we are interested in calculating the delays incurred by these processes. The arbiter circuit mainly consists of a priority matrix, which stores the priorities of the requesters and grant generation circuits, granting resources to requesters. The matrix arbiter stores priorities between n requestors in a binary *n-by-n* matrix.

As shown in Figure 3, for the input side, there are four virtual channels competing for the resources and the grant circuit generates four grant signals denoted by gnt_1 to gnt_4.

We use the method of *logical effort* [20] to determine the delays in all the processes involved in the switch operation. We report the

delay relative to the *FO4* unit of delay. The critical path of the arbiter circuit is shown in Figure 5.
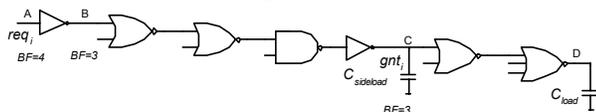

Figure 5: Critical path of the arbiter.

The first step in the implementation of the routing logic involves the comparison (XOR) of the source and destination addresses taking the most significant ($M = (log_2N - 2^l))$ bits, where N is the number of functional IP blocks in the system and l denotes the level number of the switch. Subsequently, the result of the comparison is checked, i.e., whether a "1" results from the XOR operation. As a result of these two logical operations the critical path of the routing block is as shown in Figure 6.
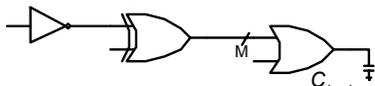

Figure 6: Critical path of the routing block.

The final M-input OR gate of Figure 6 is modeled as a tree of 2-input NOR gates [20].

The critical path involved in switch traversal consists of the combination of four mux/demux according to Figure 3.

We developed a VHDL model for the switch and synthesized it using Synopsys' synthesis tool in CMOS 0.18μ technology. We compared results obtained from the logical effort analysis with those given by Synopsys' tool. These comparisons are reported in Table 2.

Table 2: Comparison of Delays: Calculated vs. Synopsys' tool-generated

Process	Delay (LE analysis)	Delay (Synopsys' Tool)
Input Arbitration (t_{ia})	8.9 *FO4*	9.1*FO4*
Routing (t_r)	5*FO4*	6.2*FO4*
Output Arbitration (t_{oa})	10.3*FO4*	9.6*FO4*
Switch Traversal (t_{st})	3.5*F04*	4.1*FO4*

Table 2 shows that the delays estimated by our logical effort analysis closely match those obtained from Synopsys' tools. This indicates that the delay associated with each of the processes involved with the operation of the switch is well below the ITRS-suggested limit of 15*FO4* and can therefore be driven by a clock with a period of *15FO4*.

The structured inter-switch wires and the processes underlying the switch operations yield four types of pipelined stages. Together, these four types characterize the functionality of the communication fabric. Through careful design and analysis, we have shown how it is possible to constrain the delays of each of these stages such that they are bounded by the clock period limits suggested by ITRS [17] for high performance multi-core SoC platforms.

6. CONCLUSIONS

Multi-core SoC platforms are emerging as the trend for future SoC designs. In this paper, we showed that by adopting BFT as an interconnect architecture; the whole SoC interconnection fabric is divided into multiple pipeline stages. Each of these stages can be driven by a clock signal with ITRS suggested time period of 15FO4 delay units irrespective of the system size. In this way it overcomes the inherent non-scalability associated with traditional bus-based systems. The constraints of the structured architecture that we proposed are well outweighed by the performance benefits and other portability, scalability and design automation benefits.

7. ACKNOWLEDGMENTS

8. REFERENCES

[1] P. Magarshack, P.G. Paulin, "System-on-Chip Beyond the Nanometer Wall", *Proceedings of DAC'03,* June 2-6,2003, Anaheim, USA.

[2] M.A. Horowitz, et al., "The Future of Wires", *Proceedings of the IEEE,* Volume: 89 Issue: 4, April 2001 pp. 490–504.

[3] K. C. Saraswat, et al., "Technology and Reliability Constrained Future Copper Interconnects – Part II: Performance Implications," *IEEE Transactions on Electron Devices*, Vol. 49, No. 4, April 2002 pp. 598-604.

[4] Y. Zorian, "Guest editor's introduction: what is infrastructure IP?"*, IEEE Design & Test of Computers*, Volume: 19 Issue: 3 , May-June 2002 pp. 3 -5.

[5] D. Wingard, " MicroNetwork-Based Integration for SoCs", *Proc. DAC 2001,* pp. 673-677, Las Vegas, Nevada, USA, June 18-22, 2001.

[6] Open Core Protocol, *www.ocpip.org.*

[7] MIPS SoC-it, *www.mips.com.*

[8] S. Kumar, et al, " A Network on Chip Architecture and Design Methodology," *Proceedings of ISVLSI,* pp. 117-124, 2002.

[9] W. J. Dally, and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks", *Proceedings of DAC 2001*, pp. 683-689, Las Vegas, Nevada, USA, June 18-22, 2001.

[10] I. Saastamoinen, et al, "Interconnect IP Node for Future System-on-Chip Designs", *Proceedings of the First IEEE International Workshop on Electronic Design, Test and Applications*, pp. 116 –120, 2002.

[11] P. Guerrier, A. Greiner**,** "A generic architecture for on-chip packet-switched interconnections", *Proceedings of Design, Automation and Test in Europe Conference and Exhibition 2000*, pp. 250 –256.

[12] P. P. Pande, C. Grecu, A. Ivanov, R. Saleh, "Design of a Switch for Network on Chip Applications", Proceedings *of ISCAS*, Bangkok, May 2003.

[13] P. P. Pande, C. Grecu, A. Ivanov, R. Saleh, "High-Throughput Switch-Based Interconnect for Future SoCs", Proceedings *of 3rd IEEE International Workshop on System-on-Chip for Real-Time Applications*, June 30-July 2, 2003, Calgary, Canada.

[14] D. Sylvester, K. Keutzer, "Impact of Small Process Geometries on Microarchitectures in Systems on a Chip", *Proceedings of the IEEE,* Vol. 89, No. 4, April 2001 pp. 467-489.

[15] Design and Reuse website, *http://www.us.design-reuse.com/sip/.*

[16] C. E. Leiserson, "Fat Trees: Universal networks for hardware efficient supercomputing". *IEEE Transactions on Computers,* C-34 (10): pp. 892-901. Oct. 1985.

[17] *ITRS 2002 Documents, http://public.itrs.net/Files/2002Update/Home.pdf.*

[18] D. A. Hodges, H. G. Jackson and R. Saleh, *Analysis and Design of Digital Integrated Circuits*, Third Edition, McGraw-Hill, 2003.

[19] Cristian Grecu, Partha Pratim Pande, Andre Ivanov, Res Saleh, "A Scalable Communication-Centric SoC Interconnect Architecture," IEEE International Symposium on Quality Electronic Design, ISQED 2004 San Jose, California, USA, 22-24 March, 2004.

[20] I. Sutherland, B. Sproull and D. Harris, *Logical Effort: Designing Fast CMOS Circuits*, Morgan Kaufmann, 1999.