A Scalable Communication-Centric SoC Interconnect Architecture

Cristian Grecu, Partha Pratim Pande, André Ivanov, Res Saleh

SoC Research Lab
Department of Electrical and Computer Engineering
University of British Columbia
2356 Main Mall Vancouver, BC, V6T 1Z4 Canada
Email: {parthap, grecuc, ivanov, res}@ece.ubc.ca

Abstract
System on Chip (SoC) design in the forthcoming billion-transistor era will involve the integration of numerous heterogeneous semiconductor intellectual property (IP) blocks. Some of the main problems in the ultra deep submicron technologies arise from non-scalable global wire delays, failure to achieve global synchronization and difficulties associated with non-scalable bus-based functional interconnect. These problems can be dealt with by using a structured interconnect template to design future SoCs. Recently, we introduced the butterfly fat-tree as an overall interconnect architecture, where IPs reside at the leaves of the tree and switches at its vertices. Here, we analyze this architecture with a particular focus on achieving overall timing closure. The only global wires in this routing architecture are the inter-switch wires and the delays in these global wires can be predicted at the initial stages of design cycle. Our analysis shows that the inter-switch wire delay in the networked SoC can be always designed to fit within one clock cycle, regardless of the system size. We contrast the analysis for our network with that of a bus-based architecture. For the latter, we illustrate how the interconnect delay and system size are interrelated, thereby limiting the number of IP blocks that can be connected by a bus.

Keywords: System on chip, timing closure, switch-based interconnect, butterfly fat-tree, interconnect architecture, global wire delay.

1. Introduction and motivation
According to ITRS 2002 [1], the realization of complex Systems on a Chip (SoCs) consisting of billions of transistors fabricated in technologies characterized by 65 nm feature size and less will soon be feasible. Such SoCs imply the seamless integration of numerous IPs performing different functions and operating at different clock frequencies. The integration of several heterogeneous components into a single system gives rise to new challenges. It is critical that infrastructure IP (IP) be developed for systematic integration of heterogeneous functional IP blocks to enable widespread use of the SoC design methodology.

One of the major problems associated with future SoC designs arises from non-scalable global wire delays. Global wires carry signals across a chip, but these wires typically do not scale in length with technology scaling [2]. Though gate delays scale down with technology, global wire delays typically increase and are controlled by inserting repeaters. Even after repeater insertion the delay may exceed the limit of one clock cycle [3]. In ultra deep submicron processes, 80% or more of the delay of critical paths will be due to interconnect [4] [5]. Another important problem associated with global wires is that such wires are typically implemented in top-level metal layers, with the routing performed automatically in later stages of the design cycle. Such wires end up having parasitic capacitance and resistance that are difficult to predict a priori.

In the forthcoming technologies, global synchronization of all IPs will lose its significance, due to the impossibility of sending signals from one end of the chip to another within a single clock cycle. In fact, many large designs today use FIFOs to synchronously propagate data over large distances to overcome this problem. This solution is ad-hoc in nature. Instead of aiming for global control, one attractive option is to allow self-synchronous IPs to exchange data with one another through a communication-centric architecture [1].

According to recent publications [20] emergence of SoC platforms consisting of large, heterogeneous sets of embedded processors is imminent. A key element of such multiprocessor SoC (MP-SoC) platforms [20] is the interconnect topology. The most frequently used on-chip interconnect architecture is the shared medium arbitrated bus, where all communicating devices share the same transmission medium. The operating frequency of the shared bus depends on the propagation delay in the interconnection wires. This propagation delay, in turn, depends on the number of IP cores connected to the wires. Each core attached to the bus adds a parasitic capacitance, and therefore, the electrical performance will degrade with system growth. For SoCs consisting of tens or hundreds of IP blocks, such bus-based interconnect architectures lead to propagation delays that exceed acceptable limits as all attached devices share the same wires.

To overcome the above-mentioned problems, we recently proposed [16] the use of a network-centric approach to integrate IPs in complex SoCs. We use a butterfly fat tree architecture where the IP blocks are at the leaves of the tree and the switches at its vertices. In our model, a group of IPs is connected to a neighboring switch. Global signals, spanning a significant portion of a die in more traditional bus-based architectures, now only have to span the distance separating switches. In this way the long global wires are divided into segments with propagation time comparable with the clock cycle. Consequently signals propagate along these segments in a pipelined fashion hiding the latency of the global wires. These switches (FPs) are key components of this infrastructure enabling seamless integration of the functional IP blocks. In this scenario, global wires only consist of top level interconnects between switches. The specifics of such interconnect
can be known at early stages of the design process, enabling a better prediction of the electrical parameters of the interconnect, and overall system performance.

In this paper we show how the switch-to-switch wire length in our proposed model varies with technology nodes and how the propagation delay between the switches can be predicted. We contrast these delay characteristics with those associated with bus-based SoCs. The remainder of this paper is organized as follows. In Sec. 2, an overview of the related work is given. Sec. 3 details the proposed interconnect architecture. Sec. 4 deals with future SoC architecture trends. Secs. 5 and 6 address the wire delay in switch-based and bus-based systems, respectively. Sec. 7 concludes by highlighting the salient features of our work and pointing to future directions.

2. Related work

Current SoC designs predominantly use shared-medium bus-based functional interconnects to integrate IP blocks. There are mainly three types of commercially used SoC interconnect specifications, ARM AMBA [6] bus, Wishbone [7] and IBM CoreConnect [8]. All of them suffer the drawback of non-scalability.

A few on-chip micro network proposals for SoC integration can be found in the literature. Sonic’s Silicon Backplane [9] 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) [10]. Agents communicate with each other using TDMA (Time Division-Multiple Access) bus access schemes. These agents effectively decouple the IP cores from the communication network. The basic interconnect architecture is still bus-based and will hence suffer from performance degradation trends common for buses.

MIPS Technologies has introduced an on-chip switch integrating IP blocks in a SoC [11]. The switch called SoC-it is intended to provide a high-performance link between a MIPS processor and multiple third party IP cores. It is a central switch connecting different peripherals, but only in a point-to-point mode.

Kumar [12] and Dally [13] 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. Each switch is thereby connected to four neighboring switches and one IP block. In this case, the number of switches is equal to the number of IPs.

Saastamoinen [14] describes the design of a reusable switch to be used in future SoCs. The interconnect architecture is however not specifically discussed.

Guerrier and Greiner [15] proposed the use of a tree based interconnect (SPIN) and addressed system level design issues. In [16], [17], [18], an interconnect architecture for a networked SoC, as well as the associated design of required switches and addressing mechanisms, are described.

All of the above mentioned works proposed some kind of interconnect architecture to solve the global wire delay problem, however none of them specifically deals with this. Addressing the wire delay problem in a communication-centric SoC is precisely the focus of this paper.

3. Interconnect architecture

We proposed a novel interconnect template to integrate numerous IPs of an SoC following a butterfly fat-tree architecture as shown in Fig. 1. In our network, the IPs are placed at the leaves and switches placed at the vertices. Fig. 2 illustrates the physical placement of a butterfly fat-tree with 64 IPs, where the darker blocks denote the switches (I^2Ps), and white squares represent the functional IP blocks. A pair of coordinates labels each node, (j, p), where j denotes a node’s level and p denotes its position within that level. In general, at the lowest level, there are N IPs with addresses ranging from 0 to (N-1).

![Fig. 1: Butterfly fat-tree graph with N = 64 IP blocks.](image1)

![Fig. 2: Floorplan of a 64-IP butterfly fat tree network.](image2)
At each subsequently higher level of the tree the number of required switches reduces by a factor of 2. In this way the total number of switches, S, is calculated as:

$$S = \frac{N}{4} + \frac{1}{2} \frac{N}{4} + \frac{1}{4} \frac{N}{4} + \cdots \left(\frac{1}{2}\right)^{\text{levels}} \frac{N}{4} = \left(1 - \left(\frac{1}{2}\right)^{\text{levels}}\right)\frac{N}{4}$$

(3.1)

which illustrates that S tends to N/2 as N grows arbitrarily large. In the case of 64 IPs the number of switches is 28 as shown in Fig. 1.

In regards to the interconnect wire delay, this switch-based architecture offers the advantage that wires between IP blocks and between switches are logically structured and therefore constrained such that their lengths can be largely predictable and consistent across the entire network.

4. SoC microarchitecture trends and assumptions

In a conventional digital ASIC design flow, several iterations of logic synthesis and physical design are 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 constitute a significant portion of the critical path delay.

For IP blocks consisting 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 [2] [4]. 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. For example, a 32-bit DSP core can have around 145K gates [19]; a MPEG2 decoder can consist of approximately 60K gates [19], and a general purpose 32-bit RISC microprocessor can amount to around 50K gates [19].

We are already at a point where many new designs coming out from industry consist of 100 embedded processors [20]. By extension to the above, we conjecture that the trend for future SoC integration will be based on a hierarchical design paradigm where an increasing number of IP blocks consisting of 100K gates will be integrated according to a specific interconnect template. One possible interconnect template is the butterfly fat-tree as shown in Figs. 1 and 2.

Assuming IP blocks consisting of 100K gates, Table 1 shows the maximum number of IP blocks that can be integrated in a single SoC in different ITRS technology nodes [5]. In the foregoing, we assume that such blocks would be integrated according to a butterfly fat-tree microarchitecture template. 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. In the next section, we show that the increased number of levels does not negatively impact the achievable clock cycle rates for the SoC.

### 5. Inter-switch wire delay

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

$$W_{a+1,a} = \sqrt{\frac{\text{Area}}{2^{\text{levels}-a}}}$$

(5.1)

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). Table 2 shows the inter-switch wire length in mm for different technology nodes. X denotes that the particular inter-switch wire is not present in the concerned technology node. The die size is assumed to remain unchanged at 20 mm.

### Table 1: Maximum number of IP blocks (100K gates/IP block) [5] and corresponding fat-tree levels as technology scales

<table>
<thead>
<tr>
<th>Technology Node</th>
<th>Max. No. of IPs</th>
<th>No. of Fat-Tree levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>130 nm</td>
<td>500</td>
<td>6</td>
</tr>
<tr>
<td>90 nm</td>
<td>1000</td>
<td>7</td>
</tr>
<tr>
<td>65 nm</td>
<td>2500</td>
<td>9</td>
</tr>
<tr>
<td>45 nm</td>
<td>7500</td>
<td>10</td>
</tr>
<tr>
<td>32 nm</td>
<td>10000</td>
<td>11</td>
</tr>
</tbody>
</table>

### Table 2: Inter-Switch wire lengths in mm

<table>
<thead>
<tr>
<th>Technology Node</th>
<th>No. of levels</th>
<th>$W_{11,10}$</th>
<th>$W_{10,9}$</th>
<th>$W_{9,8}$</th>
<th>$W_{8,7}$</th>
<th>$W_{7,6}$</th>
<th>$W_{6,5}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>130 nm</td>
<td>6</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>10.000</td>
</tr>
<tr>
<td>65 nm</td>
<td>9</td>
<td>X</td>
<td>X</td>
<td>10.000</td>
<td>5.000</td>
<td>2.500</td>
<td>1.250</td>
</tr>
<tr>
<td>45 nm</td>
<td>10</td>
<td>X</td>
<td>10.000</td>
<td>5.000</td>
<td>2.500</td>
<td>1.250</td>
<td>0.625</td>
</tr>
<tr>
<td>32 nm</td>
<td>11</td>
<td>10.000</td>
<td>5.000</td>
<td>2.500</td>
<td>1.250</td>
<td>0.625</td>
<td>0.312</td>
</tr>
</tbody>
</table>

We can compute the intrinsic RC delay [21] of a wire according to the equation below:

$$D_{\text{unbuffered}} = 0.4R_nC_uL^2$$

(5.2)

where $R_n$ and $C_u$ are the resistance and capacitance per unit length of the wire, respectively, and $L$ is the wire length. The minimum conceivable clock cycle time considering a highly pipelined design style can be assumed to equal the value of 15FO4, with FO4 defined as the delay of an inverter driving four identical ones [2]. In different technology nodes, FO4 can be estimated as $423\mu s_{\text{min}}$ [ps] where $s_{\text{min}}$ is the minimum feature size in each technology node [1]. 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 a total of \( N \) inverters inserted, then the total delay of the buffered wire will be according to the following expression [21]:

\[
D_{\text{buffered}} = N t_{\text{inv}} + \left( C_g R_w M + \frac{C_r R_{\text{equiv}}}{M} \right) L + 0.4R_n C_w L^2 \quad \frac{L}{N}
\]

(5.3)

where \( t_{\text{inv}} \) is the delay of an inverter sized for equal rise and fall propagation delays, and such that \( t_{\text{inv}} = FO4/5 \). \( M \) is the size of the inverter, \( C_g \) is the gate capacitance of the minimum size inverter, \( R_{\text{equiv}} \) is the resistance of n-type diffusion region in \( \Omega \). Differentiating \( D_{\text{buffered}} \) with respect to \( N \) and equating to zero yields the optimum number of segments [21]:

\[
N = \frac{0.4R_n C_w L^2}{t_{\text{inv}}}
\]

(5.4)

\( R_n \) can be calculated according to the following formula:

\[
R_n = \frac{\rho}{T W}
\]

(5.5)

where \( \rho \) is the resistivity of the copper wire (here assumed to be 2.2Ωμm), and \( T \) and \( W \) are the wire thickness and width, respectively.

**Fig. 3:** Cross section of multiple metal layers.

\( C_w \) can be calculated according to the following equation [3]:

\[
C_w = 2 \varepsilon_d \varepsilon_0 \left( \frac{1}{W} + \frac{2}{W} \right)^2 \quad \frac{C_{\text{fringe}}}{(T/W)}
\]

(5.6)

where \( \varepsilon_d \) is the dielectric constant, \( \varepsilon_0 \) is the permittivity of free space. \( C_{\text{fringe}} \) is the fringing capacitance assumed to be constant and equal to 0.04F/μm in all technology nodes [3].

In our calculations of \( R_n \) and \( C_w \), the inter-level dielectric thickness \((H)\), metal thickness \((T)\), intra-level dielectric thickness \((S)\), and wire width \((W)\), are all assumed to be the half pitch [3] for the given technology node, as shown in Fig. 3. Specific values for \( R_n \), \( C_w \) and \( t_{\text{inv}} \) are shown in Table 3 for successive technology nodes.

**Table 3:** Values of \( R_n \), \( C_w \), \( t_{\text{inv}} \) and \( FO4 \) in different technology nodes.

<table>
<thead>
<tr>
<th>Technology node</th>
<th>( R_n ) [Ω/μm]</th>
<th>( C_w ) [F/μm]</th>
<th>( t_{\text{inv}} ) [ps]</th>
<th>( FO4 ) [ps]</th>
</tr>
</thead>
<tbody>
<tr>
<td>130 nm</td>
<td>0.06</td>
<td>0.30</td>
<td>11.05</td>
<td>55.25</td>
</tr>
<tr>
<td>90 nm</td>
<td>0.12</td>
<td>0.22</td>
<td>7.65</td>
<td>38.25</td>
</tr>
<tr>
<td>65 nm</td>
<td>0.20</td>
<td>0.20</td>
<td>5.50</td>
<td>27.5</td>
</tr>
<tr>
<td>45 nm</td>
<td>0.44</td>
<td>0.20</td>
<td>3.82</td>
<td>19.1</td>
</tr>
<tr>
<td>32 nm</td>
<td>0.73</td>
<td>0.20</td>
<td>2.70</td>
<td>13.5</td>
</tr>
</tbody>
</table>

We used the values of \( R_n \), \( C_w \), and \( t_{\text{inv}} \) from Table 3 to calculate unbuffered and buffered global wire delay in different technology nodes.

**Fig. 4:** Buffered global wire delay in different technology nodes.

An important point is that our inter-switch wire length and delay analysis and its 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 an SoC would scale accordingly, i.e., inversely to the size of IP blocks. Consequently, only the number of levels in our template would change. The inter-switch wire length and delay would remain largely unaffected.

### 6. Wire delay in a bus-based SoC

In this section we analyze the effects on delay of connecting IP blocks to a bus. In a bus-based SoC, multiple IP blocks share the same transmission media. As the number of connected IP blocks increases, the capacitance attached to the bus wires increases correspondingly. 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 [6], [7], [8], as shown in Fig. 5, they are basically unidirectional and can therefore easily be buffered.

![Fig. 5: Multiplexer-based bus architecture.](image)

We consider the length of bus wires, \( L_{bus} \), to be equal to the maximum unbuffered wire length at each technology node that can be driven within one clock cycle.

Attaching IP blocks to a bus adds an equivalent 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 \( N_{bus} \) 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 can be obtained by modifying equation (5.3). The delay in this case will be as follows:

\[
D_{bus,\text{buffered}} = N_{bus} + \left( C_w R_w M + \frac{(C_w + C_p) R_w}{M} \right) L_{bus} + 0.4R_w (C_w + C_p) \frac{L_{bus}}{N_{bus}}
\]

(6.1)

Similarly to equation (5.4), the optimum number of sections will be given by the following:

\[
N_{bus} = \sqrt{\frac{0.4 R_w (C_w + C_p) L_{bus}^2}{f_{inv}}}
\]

(6.2)

From equation (6.1) one can determine how much parasitic capacitance can be added to a bus wire before \( D_{bus} \) exceeds one clock cycle for a specific wire length of wire. The other hand, if bus wire lengths are increased, then wire capacitance will dominate and result in decreasing the allowable \( C_p \) and hence the number of IPs possibly appended to the bus. In Fig. 6 we illustrate the effect of bus wire length on \( C_p \) for different technology nodes. From the latter, the bus driving capability decreases exponentially with bus wire length.

![Fig. 6: Variation of \( C_p \) with bus wire length.](image)

Fig. 6 illustrates the scalability problem associated with bus-based SoCs. For a fixed bus length, there is an upper limit on the parasitic capacitance (due to IP blocks) that can be accommodated if the bus delay is to be less than one clock cycle. As a result, there is a corresponding upper limit to the number of IPs that can be connected to a bus. Furthermore, in order to meet such delay requirements, the value of allowable parasitic capacitance decreases exponentially with bus length. As a result, the number of IP cores that can be added to the bus decreases.

However, due to heterogeneous nature of constituent IP cores in a SoC (DSPs, MPEG decoders, memories etc.), it is not possible to quantify the number of IPs that can be connected to the bus. By knowing \( C_p \) and the types of IPs that need to be integrated for a particular application, we will be able to determine whether timing closure is achievable when connecting these IPs to a bus.

If the 15\( \times \)FO4 requirement on the clock cycle is relaxed and thereby increased, then the permissible values of \( C_p \) and hence the number of attached IP blocks also increases as shown in Fig. 7. This implies that by stretching the clock cycle, more IP blocks can be added to the bus at the cost of overall system speed degradation.

![Fig. 7: Variation of clock cycle with \( C_p \) in 130nm technology node.](image)

The length of bus wires is difficult to predict in early stages of the design cycle. Hence, typically, system-level timing closure can only be reached post layout and after several iterations.
In contrast, in the case of a networked SoC, the system size does not imply any extra loading on the inter-switch wire. As a result, variations in system size have little effect on the achievable clock cycle. That is, the inter-switch wire delay is largely insensitive to system size and only depends on the levels of the switches in the butterfly fat-tree architecture.

7. Conclusions and future work

This work specifically analyzes global wire delays in a new switch-based interconnect architecture for future generations of SoCs. We assumed a butterfly fat-tree as a system-level interconnect template. As this is a highly structured and regular architecture, the inter-switch wire delay can be estimated accurately, and at initial phases of the design cycle. We constrained this delay to be within one clock cycle, where the latter, is, in turn, dictated by the technology dependent parameter limit governed by \( 15 \text{FO4} \).

The delay in a bus-based SoC depends on the number of connected IP blocks. To further quantify this dependency, we proposed parasitic capacitance, \( C_p \), as a metric, which, in turn, is directly proportional to the number of IPs attached to a bus. We showed upper limits on the value of \( C_p \), and therefore on the number of IPs, for different forthcoming technology nodes, and showed how these limits decrease exponentially against increases in bus wire length.

Looking forward in time, where numerous (hundreds or thousands) IP blocks consisting of 50-100K gates will need to be integrated, single bus-based interconnect templates will face serious limitations. We envisage that multiple forms of buses connected through a hierarchical architecture will ultimately converge to some form of network as the one proposed and analyzed here. As a result, we propose that future design processes start with a network architecture in mind. This will allow for better interconnect delay predictions. The ultimate effect will be shortening of design cycle and less number of iterations.

8. Acknowledgments

The authors wish to thank Micronet, PMC-Sierra, Gennum and NSERC for their financial support and the CMC for providing access to CAD tools.

9. References