BRINGING NOCS TO 65 NM

VERY DEEP SUBMICRON PROCESS TECHNOLOGIES ARE IDEAL APPLICATION FIELDS FOR NOCS, WHICH OFFER A PROMISING SOLUTION TO THE SCALABILITY PROBLEM. THIS ARTICLE SHEDS LIGHT ON THE BENEFITS AND CHALLENGES OF NOC-BASED INTERCONNECT DESIGN IN NANOMETER CMOS. THE AUTHORS PRESENT EXPERIMENTAL RESULTS FROM FULLY WORKING 65-NM NOC DESIGNS AND A DETAILED SCALABILITY ANALYSIS.

Antonio Pullini
Politecnico di Torino

Federico Angiolini
Università di Bologna

Srinivasan Murali
École Polytechnique Fédérale de Lausanne

David Atienza
Universidad Complutense de Madrid

Giovanni De Micheli
École Polytechnique Fédérale de Lausanne

Luca Benini
Università di Bologna

Architectural and physical scalability concerns make the interconnect subsystem one of the most important areas of multiprocessor system-on-chip (MPSoC) design. Architectural concerns arise from the performance pressure associated with several system cores that simultaneously demand communication resources, requiring adequate bandwidth and latency for each core. Physical scalability concerns arise from the intrinsic problems of designing wires that must span the entire chip—problems such as propagation delay, noise, and crosstalk. Traditional shared-bus interconnects have been the main interconnect option for a long time; thus, they have reached a relative maturity, supported by design automation tools. Although designing them is easy, they do not scale. Hence, improvements both in protocol and topology have been introduced. Protocol improvements include outstanding transactions with out-of-order delivery, and topology improvements include bridges and crossbars. Nevertheless, scalability is still suboptimal. Protocol improvements hit a bandwidth limit imposed by the available physical resources. Bridges require multiple buses or spaghetti-like design, and crossbars require large area overheads in routing structures.

The network on chip (NoC) is a promising solution to the scalability problem.[1,2] NoCs build upon improvements in bus architecture—for example, in terms of topology design. By bringing packet-based communication paradigms to the on-chip domain, NoCs address many issues of interconnect fabric design better than buses do.[3] For example, NoC designers can control wire lengths by matching network topology with physical constraints, and they can boost bandwidth simply by increasing the number of links and switches. Furthermore, compared with irregular, bridge-based assemblies of processing element clusters, NoCs also help tackle design complexity issues.[4,5]

Although these key advantages of NoCs have been widely discussed, the practical implementation of NoCs in very deep submicron technology (90 nm and below) is still an open challenge. For example, it is not yet clear whether NoC designers should steer toward very spread-around topologies such as meshes (large numbers of low-radix switches) or more concentrated topologies such as stars (small numbers of high-radix switches). Full crossbars represent the most extreme form of the latter design choice. Although the answer depends largely on the design, two physical-level enablers and constraints pull designers in both directions: 1) The effective routability of large crossbars determines the feasibility of high-radix switches. 2) Although global wires are intrinsically segmented, allowing much better control of crosstalk, load delay, and propagation delay than exists in shared

0272-1732/07/$20.00 © 2007 IEEE Published by the IEEE Computer Society.

buses, the maximum interswitch link length still plays a key role in topology design.

Other questions without a clear answer are the following:

- How do we assess the degrees of freedom available at the technology library level?
- What is the impact of leakage in a design's total power consumption?
- How do we choose the right type of back-end tooling?
- What floorplanning constraints does very deep submicron technology impose?
- What are the clock tree implications of laying a high-performance design across an entire chip?

This article presents key challenges and discoveries for next-generation technologies. (The "Related work" sidebar summarizes other NoC research.) To answer the questions just listed, we present several studies of real NoC designs.

Wire design in 65-nm technologies

Wires are a very important element in sub-100-nm technologies. Our experience with a 65-nm design flow has shown that wires are critical in both NoC modules and intermodule links.

Wire load models and placement-aware logic synthesis

The traditional flow of standard cell design features logic synthesis and placement as two clearly separate stages. Although our in-house experience has shown that this flow achieves reasonable results for 130-nm NoC designs,[3] we found it substantially inadequate at the 65-nm node. The origin of the problem is the same concept that enables splitting the two steps—namely, wire load models. Wire load models are precharacterized equations, supplied by technology libraries, that attempt to predict a gate's capacitive load on the basis of its fan-out and the overall design area. These predictions work very well as long as wire loads don't become too large. Otherwise, wire load models become a very inaccurate representation of the physical reality, which is different net by net. In our 65-nm tests, we experienced unacceptable performance degradation caused by under- or overestimation of wire loads. After the logic synthesis step, even when synthesizing single NoC modules (that is, even without considering long links), tools expect some target frequency to be reachable. However, after the placement phase, the results become up to 30 percent worse and become even slower after routing.

To address this issue, we leverage placement-aware logic synthesis tools, such as Synopsys Physical Compiler. In this type of flow, after a very quick initial logic synthesis based on wire load models, the tool attempts a coarse placement of the current netlist, estimates the resulting wire loads, and keeps optimizing the netlist while keeping this design-specific insight into account. Therefore, the final resulting netlist already considers placement-related effects; after we feed this netlist to the actual placement tool, performance results don't incur major penalties.

In addition, other wiring- and placement-related problems can occur within soft macros due to congestion. In our test designs, placement tools perform poorly when they must place modules within fences that are either too narrow or too wide. Although the former case is clearly understandable, we attribute the unexpected latter effect to the placement tool heuristics, which are probably performing worse when the solution space becomes very large. Thus, we must solve the problem by properly tuning the spacing among the soft macro fences. Consequently, accurate area models of the NoC modules are required to avoid time-consuming manual interventions in the synthesis process.

Wire routability

All the problems (such as crosstalk) that apply to global wires also apply, to a lesser extent, to local wires. This means that local wires are also increasingly critical. As a result, in wire-intensive components such as NoC switches, which are essentially crossbars, it is difficult to simultaneously achieve signal integrity, timing closure, and routability (laying the wires according to

Related work

Several researchers have established the need for NoC-based designs.[1-4] Recent research has focused on efficient synthesis methods for NoC-based interconnects and comparisons with bus-based SoCs.[5-7] Several researchers have addressed the design of application-specific NoC architectures for known communication patterns.[8-12] Others have developed tool chains for designing application-specific NoCs.[4,13]

We build on previous work[5,14,15] to address the problem of NoC topology design, accounting for technology and back-end tooling effects at the 65-nm node. Compared to the previous work, here we add several more studies and experiments on issues such as routability, leakage, and clock trees. A very interesting study investigates the impact of technology scaling on the energy efficiency of standard topologies such as meshes, tori, and their variants.[16] Our current work differs from this research in two ways: First, we consider the design of platform-specific NoC topologies and architectures. Second, we use a complete design flow integrated with standard industrial tool chains to perform accurate physical implementations of the NoCs.

References

1. L. Benini and G. De Micheli, ''Networks on Chip: A New SoC Paradigm,'' *Computer*, vol. 35, no. 1, Jan. 2002, pp. 70-78.
2. W.J. Dally and B. Towles, ''Route Packets, Not Wires: On-Chip Interconnection Networks,'' *Proc. 38th Conf. Design Automation* (DAC 01), ACM Press, 2001, pp. 684-689.
3. P. Guerrier and A. Greiner, ''A Generic Architecture for On-Chip Packet-Switched Interconnections,'' *Proc. Design Automation and Test in Europe Conf.* (DATE 00), IEEE Press, 2000, pp. 250-256.
4. K. Goossens, J. Dielissen, and A. Radulescu, ''Aethereal Network on Chip: Concepts, Architectures, and Implementations,'' *IEEE Design & Test*, vol. 22, no. 5, Sept.-Oct. 2005, pp. 414-421.
5. F. Angiolini et al., ''Contrasting a NoC and a Traditional Interconnect Fabric with Layout Awareness,'' *Proc. Design, Automation and Test in Europe Conf.* (DATE 06), IEEE Press, 2006, pp. 124-129.
6. J. Hu and R. Marculescu, ''Exploiting the Routing Flexibility for Energy/Performance Aware Mapping of Regular NoC Architectures,'' *Proc. Design, Automation and Test in Europe Conf.* (DATE 03), IEEE Press, 2003, pp. 10688-10693.
7. S. Murali, L. Benini, and G.D. Micheli, ''Mapping and Physical Planning of Networks-on-Chip Architectures with Quality-of-Service Guarantees,'' *Proc. Asia and South Pacific Design Automation Conf.* (ASP-DAC 05), IEEE Press, 2005, vol. 1, pp. 27-32.
8. A. Pinto, L.P. Carloni, and A.L. Sangiovanni-Vincentelli, ''Efficient Synthesis of Networks on Chip,'' *Proc. 21st Int'l Conf. Computer Design* (ICCD 03), IEEE CS Press, 2003, pp. 146-150.
9. W.H. Ho and T.M. Pinkston, ''A Methodology for Designing Efficient On-Chip Interconnects on Well-Behaved Communication Patterns,'' *Proc. 9th Int'l Symp. High-Performance Computer Architecture* (HPCA 03). IEEE CS Press, 2003, pp. 377-388.
10. A. Hansson, K. Goossens, and A. Radulescu, ''A Unified Approach to Constrained Mapping and Routing on Network-on-Chip Architectures,'' *Proc. 3rd IEEE/ACM/IFIP Int'l Conf. Hardware/Software Codesign and System Synthesis* (CODES+ISSS 05), IEEE CS Press, 2005, pp. 75-80.
11. K. Srinivasan, K.S. Chatha, and G. Konjevod, ''An Automated Technique for Topology and Route Generation of Application Specific On-Chip Interconnection Networks,'' *Proc. IEEE/ACM Int'l Conf. Computer-Aided Design* (ICCAD 05). IEEE CS Press, 2005, pp. 231-237.
12. T. Ahonen et al., ''Topology Optimization for Application-Specific Networks-on-Chip,'' *Proc. Int'l Workshop System-Level Interconnect Prediction* (SLIP 04). ACM Press, 2004, pp. 53-60.
13. S. Kolson et al., ''A Network on Chip Architecture and Design Methodology,'' *Proc. IEEE Computer Soc. Ann. Symp. VLSI* (ISVLSI 02). IEEE Press, 2002, pp. 105-112.
14. S. Murali et al., ''Designing Application-Specific Networks on Chips with Floorplan Information,'' *Proc. IEEE/ACM Int'l Conf. Computer-Aided Design* (ICCAD 06), IEEE Press, 2006, pp. 355-362.
15. A. Pullini et al., ''NoC Design and Implementation in 65 nm Technology,'' *Proc. 1st Int'l Symp. Networks-on-Chip* (NOCS 07), IEEE Press, 2007, pp. 273-282.
16. W. Hang-Sheng, P. Li-Shiuan, and S. Malik, ''A Technology-Aware and Energy-Oriented Topology Exploration for On-Chip Networks,'' *Proc. Design, Automation and Test in Europe Conf.* (DATE 05), IEEE Press, 2005, pp. 1238-1243.

design rules). As tools automatically try to make wires as straight and short as possible to improve timing, and insert spacing among them to avoid crosstalk, design rule check (DRC) violations can occur, including overlapping or shorted wires. Back-end tools automatically try to remove DRC violations—for example, by means of search-and-repair iterations. The design is virtually split into subblocks, and the tools begin trying to resolve routing violations one block at a time. If many violations occur, it is unlikely that all will be automatically fixed, so designers must resort to alternate methods, including the following:

- *Manual intervention on the layout*. Of course, this is extremely time-consuming and is normally undertaken only when violations are very few.

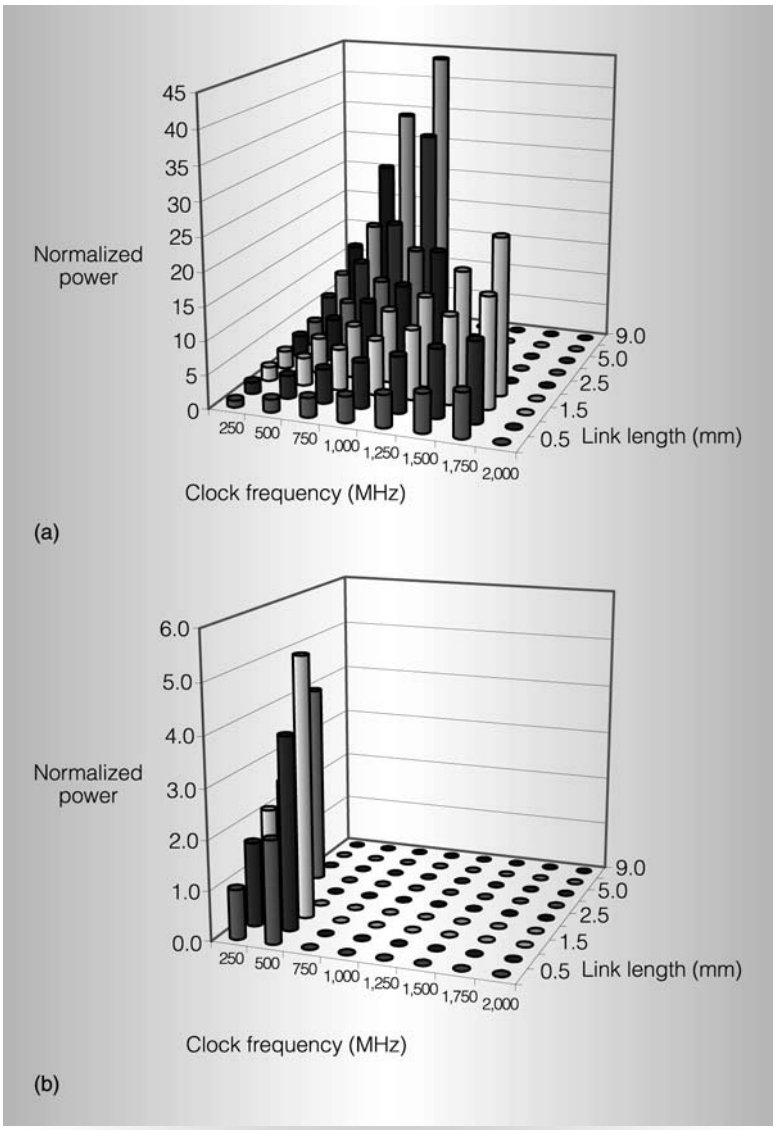

Figure 1. Power consumption of 38-bit links of varying lengths at different operating frequencies: performance/power-oriented 65-nm, low-threshold-voltage (LV_T) library (a); very low power 65-nm, high-threshold-voltage (HV_T) library (b). Values are normalized to shortest link at slowest frequency. Missing columns represent infeasible length-frequency combinations.

- *Decreasing the row utilization*. This means spreading the module out into a larger area. Ideally this leaves more space for wire routing, but because it can also affect placement output (possibly causing placement to diverge from timing closure), this alternative must be experimentally verified. In any case, this approach implies at least an area cost.
- *Decreasing the target frequency*. Wires are allowed to take less straight paths to their destinations without violating timing constraints, making crosstalk less of an issue, and allowing tighter wire packing. This strategy is effective at removing DRC violations, but its obvious cost is lower performance.
- *Hierarchical floorplanning*. This approach tries to better direct the routing tool algorithms by allowing preoptimizations and splitting the problem complexity. Its effectiveness depends on the specific module at hand and must be weighed against the extra design effort at the scripting level. Furthermore, hierarchical floorplanning prevents several optimizations that tools can perform on flattened designs. In the case of NoC switches, this strategy seems to be of limited use; if the designer must manually position even subblocks of switches, simply deploying additional smaller switches would require far less effort.

Link delay and link power

To assess the impact of global wires, we studied 65-nm NoC links in isolation from the NoC modules. Figure 1 shows an overview of the results. Several factors must be considered in link design, including length and desired clock frequency. Short or slow-clocked links don't pose problems. However, as either length or target frequency increases, an undesired power consumption increase occurs. The reason is that when high-performance links are required, back-end tools automatically insert large numbers of buffering gates, dramatically increasing the links' energy cost. In our validation experiments, the feasibility threshold of high-frequency or very long links was often set by the inability to further decrease delay. In some cases, however, the limit was posed by crosstalk concerns—in other words, the added buffers would sometimes be too large to be safely deployed. The trade-off between link power and maximum supported link length is crucial to the designer, especially accounting for the degree of freedom provided by the alternate solution of link repeaters.

Another extremely important design dependency we discovered was the specific technology library used. A single ''technology library'' no longer exists for standard cell design, especially at the 65-nm node. In fact, manufacturing technologies are spreading across a variety of libraries optimized for specific uses, such as low power or high performance, with several intermediate levels featuring, for example, different threshold voltage values. If very low-power libraries are selected by the designer, the size and speed of the buffers interleaved along wires become dramatically inferior, resulting in much tighter constraints on operation frequency or length. Figure 1a shows power consumption for a 65-nm low-power library tuned for a low threshold voltage (LV_T) and therefore enabling power-performance trade-offs. Figure 1b is based on a 65-nm low-power library tuned for a high threshold voltage (HV_T) and therefore providing minimum power consumption. As these figures indicate, the HV_T library is substantially more power-effective than the LV_T library but puts much tighter constraints on link feasibility. Hence, it becomes crucial to have floorplan-aware high-level design automation tools to reduce the NoC-based design space for 65-nm and lower technologies and to select the right libraries for each design according to its particular constraints.

Another way to tackle timing violations on long links is by pipelining the links. Pipeline stages are clocked registers interleaved along the links. By providing one or more extra clock periods to traverse long distances, they solve the link infeasibility problem at a much lower cost than that of deploying whole NoC switches in the middle of the links. In some cases, pipelining can even produce more power-effective solutions than regular wire buffering along particularly critical links, but at a performance cost (one extra latency cycle). The major drawback is that we must extend flow control to account for the fact that feedback signals now return after multiple clock cycles instead of in the same clock period. There are two ways to handle this: either deploying deeper buffers at the link endpoints and using plain registers as pipeline elements, or pipelining the link with flow-control-aware elements, without touching the buffers and logic at the endpoints. The latter approach has proved better in our experience.[6]

Tool flow

We present complete details of our flow for designing NoCs for MPSoCs in other works.[7,8] The flow comprises several seamlessly integrated tools, which span core interconnection to physical layout. SunFloor automates the front-end NoC design process, synthesizing and configuring the best topology for the application. Next, the ×pipesCompiler automates the architecture generation phase, leveraging the ×pipes library of NoC components and generating the RTL code for the desired topology. Finally, several industrial tools handle the back-end processes, logic synthesis, and physical design. To ensure that the designed topology will satisfy timing constraints after placement and routing, we precharacterize the network components' timing models and use the information in the topology synthesis phase. We obtain the timing, area, and power models of the different network components from layouts with back-annotated resistance, capacitance information, and the components' switching activity.

As mentioned, we had to extend our design flow to include support for pipelined links at all levels, from high-level tools to CAD layout software. Therefore, SunFloor automatically pipelines long links in the design, according to the targeted operating frequency. When a link is pipelined and its latency increases, SunFloor considers this information to determine the NoC's average latency, which it takes into account in its cost metrics.

A key issue to consider in the process is floorplanning. The criticality of links makes simply connecting cores located at opposite corners of the floorplan infeasible. Instead of designing a logical topology and then mapping it to the chip floorplan later, we must perform both steps together to physically connect logically connected components without effort. Several problems can arise. For example, NoC links can be easily pipelined, but NoC interfaces with

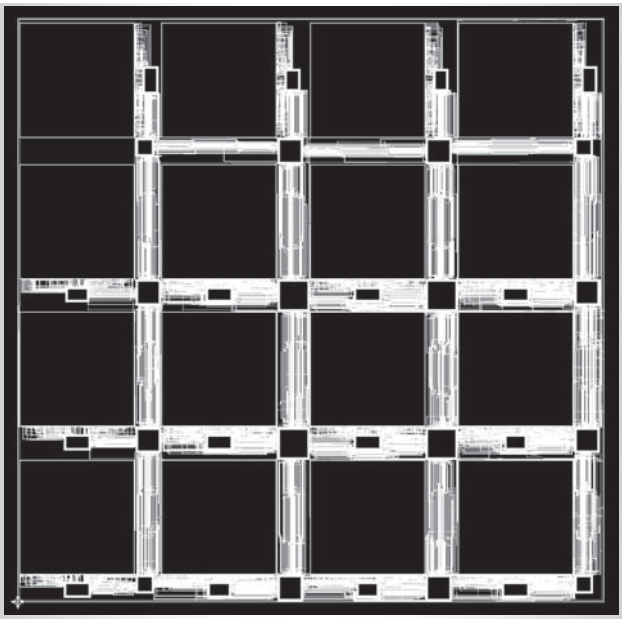

Figure 2. A 65-nm, 4 × 4 mesh built with ×pipes library components.

attached cores generally cannot. This means that we need mapping algorithms that value the distance between IP cores and NoC interfaces more than the distance between, for example, two switches. Furthermore, to allow for link routing and possibly repeater deployment, we must leave gaps between some of the IP cores. Our tool flow captures these requirements.

Experimental results

We performed a thorough study of the trends imposed by deep-submicron manufacturing processes in NoC designs. We used 65-nm NoC designs, including features such as pipelined links. As a new contribution that builds on physical-level awareness, we investigated the viability of high-radix switches. We also studied how leakage and clock trees behave in 65 nm, in increasingly complex blocks and at the complete-platform level.

Design space in 65-nm technology

As we have mentioned, at the 65-nm node, multiple technology libraries optimized for performance or power, and featuring various supply and threshold voltage levels, are available. To investigate the use of these libraries, we implemented the 4 × 4 mesh design shown in Figure 2 with different library choices: LV_T (fast), HV_T (low-power), and MV_T (multiple threshold voltages). The third option consists of choosing gates from multiple libraries at different threshold voltages for an ideal mix; that is, gates in the critical path came from the fastest library, and the other gates came from the library optimized for power. For the MV_T case, we studied two configurations: In one of them, we aimed for a high frequency to show the advantages over the LV_T library; in the other, we studied a power-performance trade-off. To make the experiment more accurate, we usually enabled clock gating. Since clock gating implies a slight performance penalty, we made an exception for the LV_T scenario, in which performance is of paramount importance. We chose nominal operating conditions for all instances. Table 1 summarizes our findings.

As Table 1 shows, there is almost an order of magnitude difference in the power/performance ratios achievable from the LV_T or HV_T libraries. System architects should take this into account when choosing the ideal NoC configuration.

The MV_T scenario proved to be a particularly attractive option, with performance approaching the LV_T library (the difference shown in the table is due to the addition of clock gating) at a lower power consumption. HV_T proved most effective in terms of power per available bandwidth. The MV_T design at 300 MHz achieved almost the same power per available bandwidth metric, because it was far from the maximum frequency point and therefore contained a large majority of HV_T gates.

Trade-offs in large switch design

Figure 3 shows how area, frequency, and power scale when we implement ×pipes switches of increasing cardinality in a 65-nm MV_T 1.2-V technology, with clock gating. The area metric is the size of the whole box enclosing the switch logic; the cell area itself is smaller. We characterized power with two simulations on the post-routing netlist annotated with parasitics. The first simulation was in complete

Table 1. Postrouting performance-power comparison of meshes in variants of a 65-nm technology library.

	Library variant			
Property	**HV_T**	**MV_T (first)**	**MV_T (second)**	**LV_T**
Frequency goal	Max.	300 MHz	Max.	Max.
Clock gating	Enabled	Enabled	Enabled	Disabled
Frequency (MHz)	142	300	714	952
Bandwidth (Gbytes/s)	27	57	137	183
Power (mW)	11	25	88	145
Power/bandwidth (mJ/Gbyte)	0.41	0.44	0.64	0.79

idleness; the second featured worst-case traffic, with all input ports injecting flits and the maximum number of bits switching (each flit payload flipping all the bits of the previous one). We then simply averaged the two resulting power figures. Although we believe this is a pessimistic metric, it is far more accurate than tool-generated power estimates because we inject real functional traffic, whereas tools only assume a certain switching-activity value. We typically observed a mismatch of about a factor of two between our results and the tools' automatically generated outputs, the latter being overly pessimistic.

As we expected, area and power increased with the switch radix, while frequency decreased dramatically. The first conclusion we can draw is that placement-aware synthesis worked as expected; there were no significant gaps between the timing predictions of Physical Compiler and the timing actually reached by the Synopsys Astro placement and routing tool (Figure 3a).

The most interesting result, however, concerns large switches. The logic synthesis tools were now aware of placement but not yet of routing. Starting from 14 × 14 cardinality, the wire density in the switch crossbars became too high to simultaneously comply with timing objectives, guarantee crosstalk freedom, and resolve DRC violations. Because of the goal priorities we set in our scripts, we achieved the first two objectives, but got an increasing number of DRC violations, ranging from hundreds (for the 14 × 14 switch) to tens of thousands (30 × 30). This number of DRC violations is unacceptable for manual fixing and must be resolved automatically. As discussed earlier, two options for fixing violations are increasing the switch area or decreasing the switch frequency. The former alternative is only partially effective. Typical industrial utilization rates are close to 85 percent. We achieved the 85 percent goal without problems in our tests until we reached 10 × 10 cardinality; at that point, widening the target fences became necessary. For example, we can properly route the 14 × 14 switch only after tweaking its row utilization to about 70 percent. In fact, we need the remaining "unused" space to route resources. However, in the 30 × 30 case, even a final row utilization of 50 percent (that is, leaving half the switch floorplan unfilled) doesn't fix the violations. This area overhead is clearly unacceptable.

The alternate option of trying to fix DRC issues by decreasing the switch frequency returns somewhat better results, making 14 × 14 and 18 × 18 switches routable at a 25 to 30 percent frequency cost, but still fails on larger switches. Similarly, even after more than halving the frequency targets, 30 × 30 switches remain unroutable. Even in cases where we can fix DRC violations by some means, our results suggest that avoiding excessively large switches may be the best option. Moreover, using large centralized blocks can also result in system-level effects that are not immediately apparent from the plots in Figure 3. For example, the many cores connected to such a switch ideally should be physically placed immediately around it, causing obvious congestion in the floorplan. Alternatively, we could spread the cores over the chip, but

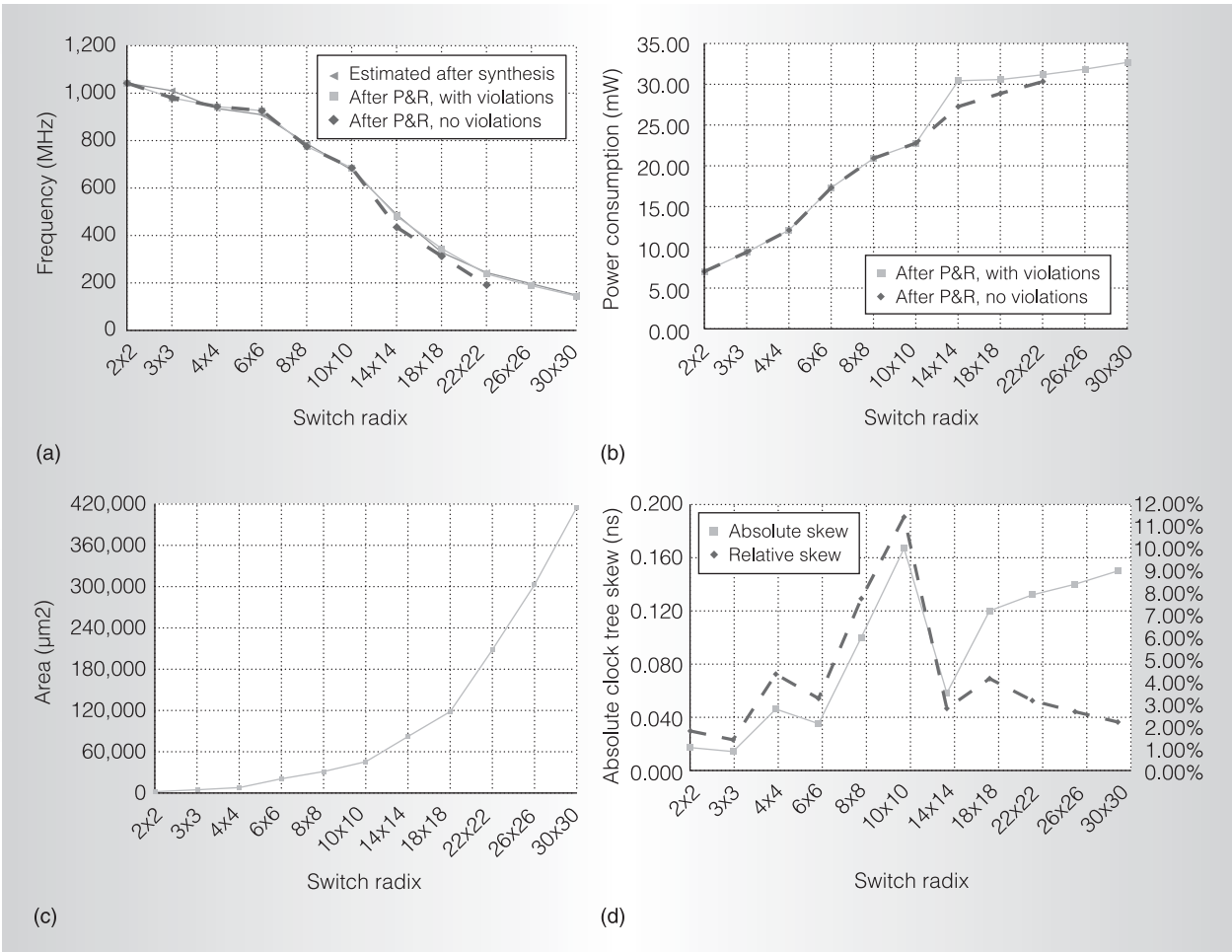

Figure 3. Frequency (a), power (b), macro area (c), and intrablock clock tree skew (d) of switches of increasing cardinality.

then we would need several long links to connect remote cores to the switch. These links would require pipelining, with additional latency, area, and power costs.

Clock tree insertion

The last plot in Figure 3 depicts the clock tree skew inside each switch under test. Our clock tree insertion policy, for a whole topology, is as follows: First, we hierarchically synthesize and place each subblock. Second, we logically connect all the blocks together in a topology. Third, we insert the clock tree, either in a single step globally, or by deploying one smaller clock tree into each block and subsequently connecting all of these trees. Fourth, we route all remaining nets.

The two clock tree insertion approaches have different trade-offs. The former, which theoretically guarantees the minimum possible skew because the entire design is visible at once, heavily affects runtime. In fact, this strategy is almost unusable for large 65-nm topologies because of the need for minimum skews over long distances; runtimes would be many hours and memory usage would be several gigabytes. The second approach is more efficient. By synthesizing clock trees locally within each subblock, it better controls local skew (Figure 3d). Absolute skew doesn't increase significantly with switch size, and since the clock period is also increasing, the relative clock skew remains constant. When creating a complete topology, we can join the clock trees to a single common root, compensating (if necessary) for each tree's different delay. The routing tools easily minimize the frequency loss caused by a skew of less than 10 percent to a negligible drop.

Test design: A multimedia benchmark

In another experiment, we considered a 30-core multimedia benchmark consisting of 10 ARM7 processors with caches, 10 private memories (a separate memory for each processor), five custom traffic generators, five shared memories, and interprocessor communication devices. We present a more detailed description of the application elsewhere.[3] Using SunFloor, we synthesized the most power-efficient topology for the application, satisfying the application bandwidth and latency requirements. We set the NoC's flit width at 32 bits and the target network operating frequency to 230 MHz.

In this experiment, we used the 65-nm MV_T technology libraries and leveraged area, timing, and power consumption models of the switches, network interfaces, and links. SunFloor automatically generated the floorplan, accounting for the repeaters. We assumed the sizes of the processor and memory cores were 1 mm × 1 mm, and we input these sizes to the tool flow. During floorplanning, SunFloor placed each network interface together with its core in a combined bounding box, with each core communicating with its network interface through point-to-point wires.

SunFloor can obtain the physical length of NoC links only after floorplanning, so it cannot determine the number and location of the pipeline stages needed along the links beforehand and must assess them after floorplanning. Therefore, it first maps pipeline registers in the geometrically ideal position, but if another component is already occupying that area, the tool resolves the overlap by moving the register to the nearest core's boundary. The bounding boxes of the cores are surrounded by thin, 30-μm-wide gaps; the tool uses this space for placing the pipeline flip-flops and routing the wires in the design. Figure 4 shows the resulting floorplan.

Thanks to our accurate precharacterization of the network components and our taking physical design issues into account during topology design, the Astro placement and routing tool achieved the final layout with little manual intervention. From our experiments, we find that building an accurate precharacterization of the components and bridging the gap across the various design phases are critical to achieving a working NoC design in a reasonable time.

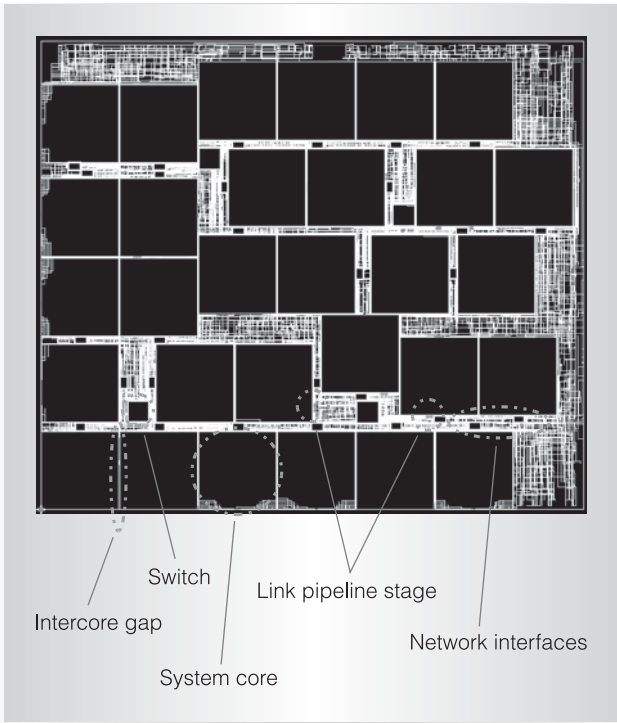

Figure 4. The 30-core multimedia benchmark with link pipeline stages automatically placed by SunFloor.

Leakage power

Leakage power is often mentioned as a major problem in deep-submicron design. Our experiences with a 65-nm NoC switch tend to contradict this assumption, as Table 2 shows for a representative MV_T case at 1.2-V supply voltage. According to our results, leakage represents as little as 0.46 percent of total power consumption when the circuit is active. In idle, absolute leakage remains roughly constant, so that,

Table 2. Dynamic and leakage power in a 22 × 22 switch in active and idle states.

Measurement	Active switch	Idle switch
Dynamic power (mW)	52.12	9.46
Leakage power (mW)	0.24	0.23
Relative leakage power (%)	0.46	2.42

even though dynamic power decreases by a factor of five (only the clock tree is switching, and it is gated), relative leakage power remains well below 3 percent.

However, our results could change in different scenarios: non-nominal conditions (high operating temperatures or lower-than-expected transistor threshold voltages), the ability to completely stall the system clock, and the need to use LV_T libraries due to demanding performance requirements. We plan a more thorough assessment of these scenarios in future work.

In summary, our most important conclusions are the following:

- Designers should leverage the degrees of freedom supplied by the large variety of available technology libraries.
- Synchronous design is still feasible at the 65-nm node, even for distributed components such as NoCs, if the clock distribution infrastructure is properly designed.
- High-radix switches are feasible at the 65-nm node until 10 × 10 or 14 × 14, after which their overhead in area and frequency becomes too severe.
- Link pipelining allows maximum flexibility in topology design at a relatively low cost and is becoming a necessity for complying with timing and signal integrity constraints.
- Leakage is not yet critical at the 65-nm node as long as the device is operating in normal conditions.

Many important issues however still need to be tackled. These include a more detailed assessment of leakage power consumption, a thorough analysis of the impact of frequency scaling, and a feasibility study on the usage of multiple voltage islands on the chip. MICRO

Acknowledgments

This work was partially supported by the Swiss National Science Foundation under FNS grant 20021-109450/1, by the Spanish government under research grant TIN 2005-5619, and by STMicroelectronics under a grant to the University of Bologna.

References

1. L. Benini and G. De Micheli, ''Networks on Chip: A New SoC Paradigm,'' *Computer,* vol. 35, no. 1, Jan. 2002, pp. 70-78.
2. W.J. Dally and B. Towles, ''Route Packets, Not Wires: On-Chip Interconnection Networks,'' *Proc. 38th Conf. Design Automation* (DAC 01), ACM Press, 2001, pp. 684-689.
3. F. Angiolini et al., ''Contrasting a NoC and a Traditional Interconnect Fabric with Layout Awareness,'' *Proc. Design, Automation and Test in Europe Conf.* (DATE 06), IEEE Press, 2006, pp. 124-129.
4. A. Jantsch and H. Tenhunen, Eds., *Networks on Chip*, Kluwer Academic, 2003.
5. L. Benini and G.D. Micheli, Eds., *Networks on Chips: Technology and Tools*, Morgan Kaufmann, 2006.
6. A. Pullini et al., ''Fault Tolerance Overhead in Network-on-Chip Flow Control Schemes,'' *Proc. 18th Ann. Symp. Integrated Circuits and System Design* (SBCCI 05), ACM Press, 2005, pp. 224-229.
7. S. Murali et al., ''Designing Application-Specific Networks on Chips with Floorplan Information,'' *Proc. IEEE/ACM Int'l Conf. Computer-Aided Design* (ICCAD 06), IEEE Press, 2006, pp. 355-362.
8. A. Pullini et al., ''NoC Design and Implementation in 65 nm Technology,'' *Proc. 1st Int'l Symp. Networks-on-Chip* (NOCS 07), IEEE Press, 2007, pp. 273-282.

Antonio Pullini is a research assistant at the Politecnico di Torino, Italy. His research interests include low-power digital design and networks on chip. Pullini has an MSc in electrical engineering from the University of Bologna, Italy.

Federico Angiolini is a PhD student in the Department of Electronics and Computer Science of the University of Bologna. His research interests include memory hierarchies, multiprocessor embedded systems, networks on chip, and nanotechnologies. Angiolini has an MSc in electrical engineering from the University of Bologna.

Srinivasan Murali is a post doctoral researcher in the Integrated Systems Laboratory of École Polytechnique Fédérale de Lausanne, Switzerland. His research interests include reliable and efficient design methods for NoCs and SoCs. Murali has an MSc and a PhD in electrical engineering from Stanford University.

David Atienza is an associate professor in the Computer Architecture and Automation Department of the Universidad Complutense de Madrid (UCM) and a postdoctoral researcher in the Integrated Systems Laboratory of École Polytechnique Fédérale de Lausanne. His research interests include emulation techniques for SoCs, NoC design, and dynamic memory management in embedded systems. Atienza has an MSc and a PhD, both in computer science, from UCM.

Giovanni De Micheli is a professor of computer science and electronic engineering and the director of the Integrated Systems Center at École Polytechnique Fédérale de Lausanne. His research interests include design technologies for integrated circuits and systems, with emphasis on synthesis and low-power design. De Micheli has an MSc and a PhD, both in electrical engineering, from the University of California, Berkeley. He is a Fellow of the ACM and the IEEE.

Luca Benini is a professor in the Electronics and Computer Science Department of the University of Bologna. His research interests include all aspects of computer-aided design of digital circuits, with special emphasis on low-power applications, and the design of portable systems. Benini has an MSc and a PhD, both in electrical engineering, from Stanford University. He is an IEEE Fellow.

Direct questions and comments about this article to Antonio Pullini, DAUIN, Politecnico di Torino, corso Duca degli Abruzzi 24, 10129 Torino, Italy; antonio.pullini@polito.it.