Enabling On-Chip Diversity Through Architectural Communication Design[†]

Tudor Dumitras, Sam Kerner, Radu Marculescu

Electrical and Computer Engineering Department
Carnegie Mellon University
Pittsburgh, PA 15213-3890, USA

{tdumitra,skerner,radum}@ece.cmu.edu

Abstract - In this paper, we explore a new concept, called *on-chip diversity*, and introduce a design methodology for such emerging systems. Simply speaking, on-chip diversity means mixing different architectures and/or technologies in a multiple voltage/frequency island setup in order to achieve the highest levels of performance, fault-tolerance and the needed flexibility in SoC design. As the main contribution, we present the challenges in implementing an efficient communication architecture for on-chip diversity and outline a unified framework which addresses some of these issues. We then provide comparative experimental results and make a qualitative analysis of different architectural choices in the design of the on-chip communication. Having the acoustic beamforming as driver application, we show that an efficient communication infrastructure can be constructed by carefully analyzing the characteristics of the application and the required levels of power, performance and fault-tolerance.

I. Introduction and Objectives

Moore's law, the observation that the density of the integrated circuits roughly doubles every two years, has held for the past three decades. Presently, the advances in wiring and manufacturing technology, as well as the device scaling below the 100 nm threshold, seem to allow Moore's law to continue only for a few more years. Indeed, shrinking transistor dimensions, smaller interconnect features and higher operating frequencies lead to a higher sensitivity of deep submicron (DSM) circuits to neutron and alpha radiation, significantly higher soft error rates, and an increasing number of timing violations [5]. It has become clear that, in order reduce the cost of design and verification, the 100% correctness requirement for VLSI circuits has to be relaxed [1]. This means that, in the future, circuits will have to be designed with some degree of architectural and system-level fault-tolerance embedded in their structure [4][12].

Furthermore, in the near future, power dissipation, increasing complexity, and parameter variations will prohibit designers from taking advantage of the full performance and integration capacity that the emerging technologies have to offer. For example, CMOS technologies offer high computational power and the advantage of well-established design methodologies. On the other hand, nanoelectronic solutions promise unprecedented levels of device density, as well as low-power and high frequencies, while MEMS add very precise sensing and actuation capabilities. It would be then desirable to have a design methodology able to combine all these technologies in order to fulfil the increasing functionality, cost and complexity demands.

CAD tools and methodologies have been developed for all these areas of design, however a system-level *integrating approach* is yet to be defined. In order to take full advantage of the potential of new technologies and design styles, a design framework that enables *hybrid systems* (that is, systems that blend synchronous and asynchronous domains, mixed-clock and mixed-signal circuits, silicon and non-silicon technologies, etc.) needs to be developed. We epitomize such systems under the name of *on-chip diversity* (see Figure 1) and believe this represents the key to obtaining inexpensive and highly scalable SoCs, capable of meeting the functionality and robustness requirements. In fact, the generic system in Figure 1 depicts a new *design platform* for integrating multiple clock domains, CMOS technologies, nanoelectronics and MEMS-based devices in a unitary system that combines the best properties of all these structures. Combining heterogeneous architectures and technologies in a multiple voltage/frequency island environment allows circuits to achieve the highest levels of performance. It also introduces a new dimension of flexibility in SoC design: for instance, the system in Figure 1 can sense, actuate and process information in a highly integrated manner. By allowing such design to be partially based on existing CAD tools and design practices, the transition to these novel structures would certainly be smoother than a sudden paradigm shift to a completely new technology, not necessarily silicon based.

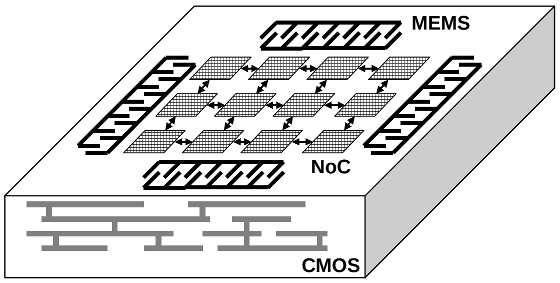

Figure 1. An emerging design platform for on-chip diversity

Characterizing the design space available for on-chip diversity is important for identifying the tradeoffs that can be made for the design of efficient systems. Within the class of solutions that seem feasible in the current industrial context, we identify two categories:

- The combination of different *architectural styles*, which means partitioning the chip into several islands with separate voltages/frequencies [6][10], with the purpose of optimizing a specific parameter, such as energy consumption;
- The combination of different *technologies*, which means assembling together, for instance, CMOS, nanoelectronics and/or MEMS devices in order to add new features to the design.

In the first category we can include the Globally-Asynchronous, Locally-Synchronous (GALS) architectures [6], which try to avoid driving a unique clock signal to the entire chip by partitioning the chip into multiple clock domains, each having a local clock. Extending the traditional GALS concept, some regions of the circuit can run at different voltages and even at different frequencies without a significant performance

[†]This research was supported in part by NSF under grant CCR-00-93104 and MARCO/DARPA Gigascale Silicon Research Center (GSRC).

penalty [10]. Certain parts of the chip, like memories or control logic, do not require as high a voltage as processor cores and therefore, by placing them on different voltage islands, the total power consumed by the design can be significantly reduced.

The idea of having multiple voltage/frequency islands can be pushed even further, leading to the second category of systems. Because of the increased overall capacity and the hard to predict side effects that characterize DSM circuits, the full potential of the current technologies cannot be obtained out of CMOS alone. It has been recently proposed to assemble together CMOS and nano-technologies in order to combine the computational power of silicon with the very high device densities enabled by nano-technologies [9]. MEMS-based devices are currently used in the IC industry in order to give sensing and actuation capabilities to the silicon. While the cost of IC design and manufacturing is becoming a severe constraint even for ASICs, the additional costs, risks and complexities associated with integrating heterogeneous technologies make such integration affordable only for the highest volume products. However, since such technologies are the "egg-shell"[1] through which digital information processing technology interfaces the physical world (interface between humans and smart ambients, for instance), the need for their integration on the same chip becomes apparent.

The dream of having true on-chip diversity depends essentially on solving the problem of *communication* between heterogeneous components. Indeed, the system-level integration of the various technologies can only be achieved if an efficient communication infrastructure can be designed and implemented. This, however, is not an easy task, because the communication architecture must provide the appropriate interfaces between the heterogeneous structures specific to on-chip diversity.

Furthermore, such an "inter-technological" communication is error-prone because of the different, and sometimes conflicting, properties of the multiple technologies. It has been emphasized that even when CMOS technology is used alone the on-chip failures are becoming extremely hard to avoid in the DSM domain [5]; when CMOS is complemented with the other elements of the on-chip diversity, designers will no longer be able to insure the dependability of SoCs unless a systematic approach is used for a reliable architectural communication synthesis.

The design of efficient communication architectures is also important because communication is becoming the most important source of on-chip power consumption. Fortunately, on-chip diversity can also bring important energy savings. Indeed, GALS architectures eliminate the power consumed by driving the clock tree to all the regions of the chip, while voltage island-based designs reduce the overall active power by running at a high voltage and a high frequency only the performance-critical regions of the chip.

A Contributions of this paper

As we can see, the *communication problem* is at the very heart of all the open issues in on-chip diversity-based design. Being able to design effective and yet simple communication infrastructures is the key enabling factor for on-chip diversity. Consequently, a unified framework for the overall design of communication architectures targeting such hybrid structures needs to be defined. With this objective in mind, the contributions of this paper are twofold:

[1] This will allow MEMS to be independently designed and digitally interfaced to the on-chip communication architecture.

- To propose a novel framework for architectural communication design which is able to address some of the issues specific to on-chip diversity. This enables designing hybrid systems, where the best properties of the various structures can be combined in order to obtain the best of each structure.
- To provide qualitative experimental data and do a comparative analysis of the performance of traditional and newly emerging communication architectures.

We emphasize the fact that, in many cases, a combination of these design styles may be desirable, in order to achieve the best performance and to make the transition to the novel communication schemes smoother than a sudden paradigm shift to a completely new design methodology.

B Related work

On-chip diversity has a major impact on the design of interconnect fabrics, as the heterogeneous regions must be able to communicate efficiently. A suitable platform for this type of communication is the Network-on-Chip (NoC) approach, which has been recently proposed [3]. In this approach, the chip is partitioned into several regions called tiles (see Figure 1), which can accommodate multiple IP cores that communicate using an appropriate networking protocol.

The tiles of a NoC can be designed according to the GALS paradigm, where a single chip has to accommodate multiple clock domains. This technique can lead to important power savings, because the clock tree is responsible for an important part of the power consumed in a traditional design but, at the same time, it introduces the problem of synchronization between different clock regions. Moreover, the voltage island-based design [10] proposes to run different regions of the chip at different supply voltages in order to reduce the active power consumption even further, without affecting the overall performance.

As shown in Figure 2, the components of the application that do not require a high level of performance can be mapped on an island running at a lower voltage and possible at a lower frequency. In order to accommodate the communication differences between the two islands, the buffers connecting them should support asynchronous or partially synchronous communication, as described later in Section C (see Figure 7).

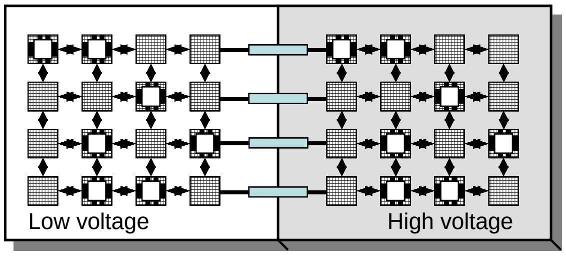

Figure 2. Communicating voltage islands

The problem of designing NoCs for fault-tolerance has been addressed only recently. From a design perspective, in order to deal with node failures, Valtonen et al. [12] proposed an architecture based on autonomous, error-tolerant cells, which can be tested at any time for errors and, if needed, disconnected from the network by the other cells. A more detailed failure model, including data upsets and omission failures, as well as a methodology to deal with these errors, are described in [15]. However, it is unclear what fault-tolerance strategy is best suited for on-chip diversity and what is the best way to implement it. This paper presents a general design methodology that addresses the issues of diversity, performance and fault-tolerance in a unified manner. As this methodology is extremely flexible and significantly simplifies the tasks of the designer, we believe that it can become an important part of the future SoC design portfolio.

C Structure of the paper

The remainder of this paper is organized as follows: in Section II, we identify the major challenges in working with on-chip diversity. In Section III, we present several candidate communication architectures and propose an efficient methodology to combine these architectures, while in Section IV we present our experimental results. We then conclude by summarizing our novel ideas for communication synthesis for SoC design.

II. CHALLENGES OF ON-CHIP DIVERSITY

Building a system based on on-chip diversity raises the challenge of integrating the different technologies and architectures and implementing efficient communication between the heterogeneous islands of the chip. Furthermore, the systems designed this way are subject to various types of failures that have to be tolerated by the communication strategy. On-chip communication architectures must therefore be designed with the purpose of increasing the reliability of chips through system-level and architectural methods. These methods must provide enough flexibility to utilize the full potential offered by integrated technologies, but at the same time they must be part of a unified framework that provides a holistic view of the system and a way to manipulate coherently the design choices offered by the on-chip diversity. As on-chip communication becomes the bottleneck for high-performance circuits, the communication architectures must be designed to increase the performance and minimize the power consumption for the communication in a failure-prone environment.

In order to enable the production of low-cost, high-performance communication architectures, we need to study more closely the failure modes for on-chip diversity. The faults that may appear in such a circuit are either transient of permanent. The transient errors, also known as *data upsets*, are caused by fluxes of neutron and alpha particles, power supply and interconnect noise, electromagnetic interference, or electrostatic discharge and are by far the most common problem in future VLSI circuits [5]. The rate of occurrence of these errors is increasing as technology scales down into the deep submicron domain. Permanent faults reflect irreversible physical changes in the structure of the circuit and they make recovery very hard or even impossible. Fortunately, these errors occur infrequently [5] and do not pose a serious threat to the mass production of VLSI chips. Furthermore, improvements of the semiconductor design and manufacturing techniques have led to a significant decrease of the permanent error rates during the past decade.

In the context of on-chip diversity, however, there are additional, more subtle error modes that can appear. The high coupling capacities of the interconnect and the tighter integration favor the Miller effect, which significantly affects on-chip delays [14]. As a consequence, it becomes more difficult to achieve delay determinism. Furthermore, in GALS circuits, the communication between the different clock domains introduces the problem of *synchronization errors*. This emphasizes the need to develop accurate failure models for on-chip diversity in order to allow engineers to design effective ways to increase the dependability of SoCs.

In what follows, we focus on implementing reliable communication in a failure-prone environment. The modules of a SoC based on on-chip diversity will communicate through a generic and reusable scheme, which has good performance and fault-tolerance characteristics.

III. SEAMLESS COMMUNICATION FOR ON-CHIP DIVERSITY

There are several communication architectures that can be used for SoC applications. We argue next that a combination of such architectures might be the best solution for the on-chip diversity and we present a communication paradigm that enables such hybrid structures.

A Bus-based communication

Traditionally, the IP cores that form a SoC are connected with an on-chip shared bus or a hierarchy of buses (see Figure 3). Because a bus is a shared communication medium, it requires arbitration in order to ensure the mutual exclusion between the components accessing the channel. Thus, when a component wishes to transfer data over the bus, it needs first to handshake with the arbiter and wait until the access is granted. The arbiter resolves conflicts according to a fixed bus protocol. The information transmitted on the bus is broadcasted to all the connected modules, even to those that are not concerned by the content of the transmission. These bus-centric interconnects are therefore very useful in applications that need to share a lot of data among the modules.

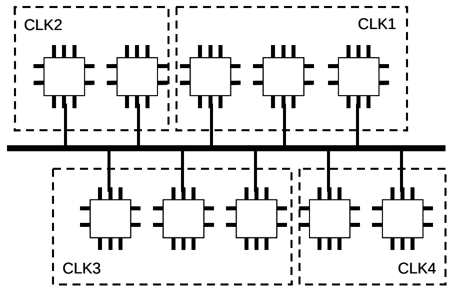

Figure 3. Bus-based communication

One problem arises when a bus needs to connect modules that are situated in different clock domains, as it is the case in GALS architectures (see Figure 3). In this case, communication between the domains has to be done through a special interface, which supports mixed clocks [7]. Because of the special handshake needed before the process of transferring data, the latency in communication may increase and delay determinism is very hard to achieve. A specially designed high level protocol is therefore needed in order to tolerate such synchronization errors.

Bus-based communication architectures are very common today and well established standards and CAD methodologies exist in the industry. However, the bus-based solution is not scalable and it has been recently proposed to switch to network-on-chip communication architectures, which would allow the integration of a large number of IPs [3]. However, in a NoC environment the problem of synchronization errors and data upsets (see Section II) is even more serious and these failures have to be treated at the system level in the design of the communication protocol. In what follows, we'll address these NoC-specific issues.

B Reliable and scalable NoC communication

Perhaps the greatest challenge introduced by the advent of new technologies and on-chip diversity is the necessary shift from design determinism to *design uncertainty* [17][2]. Detailed low-level models for nanoscale technologies are far too inaccurate, while parameter variations introduce too many factors to take into account. Failures that occur in systems based on on-chip diversity can only be characterized by stochastic models, as they are either non-deterministic in nature or too complex to be described by simple models. Therefore, the NoC communication has to be implemented with the awareness of this inherent non-determinism of on-chip diversity.

In order to deal with these realities of the modern design, we have selected for our study a recently proposed probabilistic communication scheme called *stochastic communication* [15]. This methodology implements end-to-end communication between the tiles of a NoC by using a probabilistic broadcast algorithm. At the first round the sender transmits the message to one or several of its nearest neighbors. During the subsequent rounds the sender and the tiles that have previously received the message forward it to a randomly chosen subset of their own neighbors, until the message reaches the destination. For example, in Figure 4 the Producer starts by sending its message to the tiles 2 and 7. At the second round, the three tiles that are aware of the message (i.e. tiles 6, 2 and 7) continue to send the message to randomly chosen neighbors and tiles 1, 3, 8 and 11 receive the data. A message thus propagates from tile to tile until it reaches the destination, i.e. the Consumer in Figure 4. This algorithm runs concurrently on every tile of the network and it is completed when the message has reached its intended destination. The behavior of such a communication scheme is similar with the spreading of an epidemic. The message is spread exponentially fast, and after $O(\log_2 n)$ stages it reaches all the tiles with high probability[1].

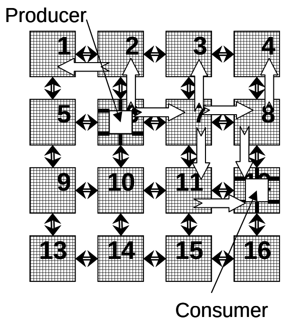

Figure 4. Stochastic Communication: an example

As messages are transmitted multiple times in the network, this redundancy can be exploited to protect the communication against failures. In order to prevent data corruption through transient errors, the packets are protected by a cyclic redundancy code (CRC); if an error is discovered, then the packet will be discarded. Because a packet is retransmitted many times in the network, the receiver does *not* need to ask for retransmission, as it will receive the packet again anyway. CRC encoders and decoders are easy to implement in hardware, as they only require one shift register [11].

A typical tile of such a NoC is shown in Figure 5. The IP core is placed in the center of the tile. On the four edges of the tile there are buffers to hold the messages sent and received by the IP. A CRC decoding circuit checks all the received messages and when an error is discovered, the message is discarded before being fed into the IP. The tile keeps a list of messages that have to be sent in an output buffer. The messages received during the last round and the new messages generated by the IP core are constantly added to the list. However, if a message is already present, a duplicate message *will not* be inserted. So even if the message is received a second time from one of the tiles in the neighborhood, *only one copy* is kept in the send buffer.

[1] The term "with high probability" indicates the convergence to 1 as the retransmission probability is increased.

The contents of this buffer will be sent to the neighbors during the next round.

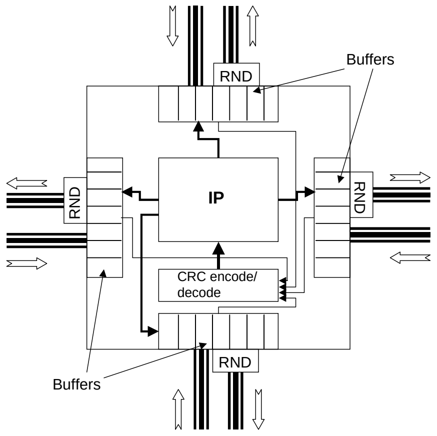

Figure 5. A tile of a NoC

We note that, since a message might reach its destination before the broadcast is completed, the spreading could be terminated even earlier in order to reduce the number of messages transmitted in the network. This is important because this number is directly connected to the bandwidth used and the energy dissipated. To do this, a *time to live* (TTL) is assigned to every message upon creation and it will be decremented at every hop until it reaches 0; then the message will be garbage-collected. Another important parameter that controls the behavior of the algorithm is *p*, the probability that a message is transmitted over a link. This parameter influences both the number of messages transmitted in the network and the number of rounds that a message requires to reach its destination. This is a powerful way to tune the trade-off between performance and energy consumption.

As the experimental results indicate, this communication scheme has an excellent tolerance to DSM failures. Furthermore, because the number of message transmissions is limited by the random choice, it scales very well with the size of the network, while maintaining a low latency. Perhaps one of the most attractive features, however, is the efficiency in adjusting the desired performance and energy dissipation, which gives a lot of flexibility to the SoC designer.

C Choosing the right architecture

As we have seen above, the bus-based interconnects are very efficient when a only few communicating IPs are connected or when the application requires a significant number of message broadcasts. On the other hand, stochastically communicating NoCs are very scalable, can include a large number of IPs and their performance does not degrade significantly under the influence of on-chip failures. However, as most SoC applications do not have a uniform structure and in order to take advantage of the properties of on-chip diversity, a combination of these structures would be more appropriate (see Figure 6). Furthermore, because bus-based solutions are widely used today, these hybrid structures would smoothen the transition to the novel communication architectures, instead of imposing a sudden paradigm shift toward communication entirely based on NoCs.

Whether the on-chip communication architecture is bus-based or has a NoC style infrastructure, if the SoC spans multiple clock domains, the data transfers between the domains have to be routed through a specialized interface (see Figure 7) that allows an asynchronous or a partially synchronous communication. The hybrid structures from Figure 6 would make extensive use of such a component, as the heterogeneity of the interconnection schemes makes the synchronization hard to achieve.

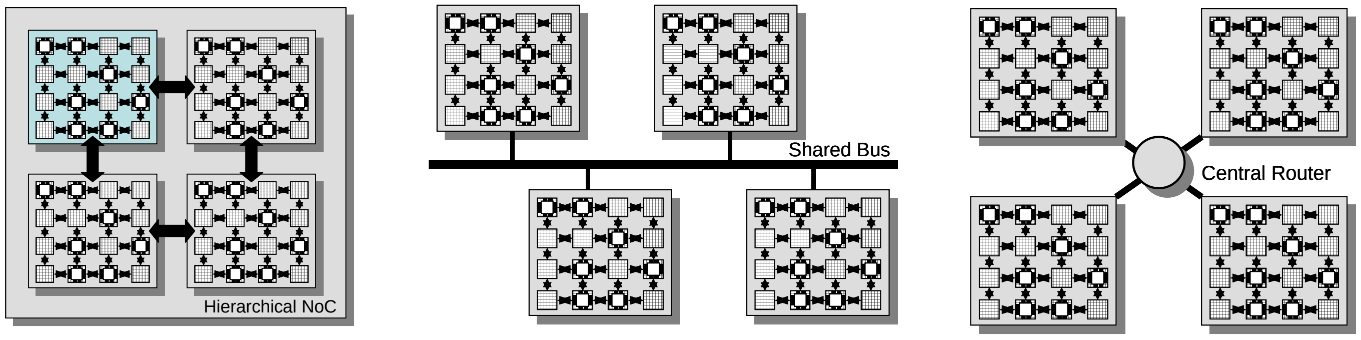

Figure 6. On-chip communication: three possible hybrid structures

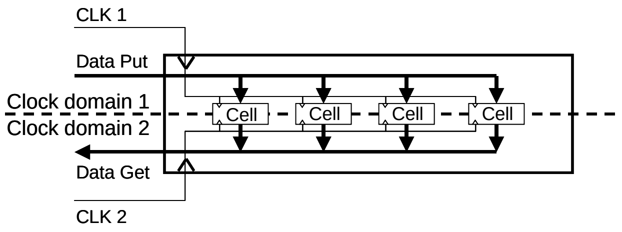

Figure 7. Mixed clock buffer

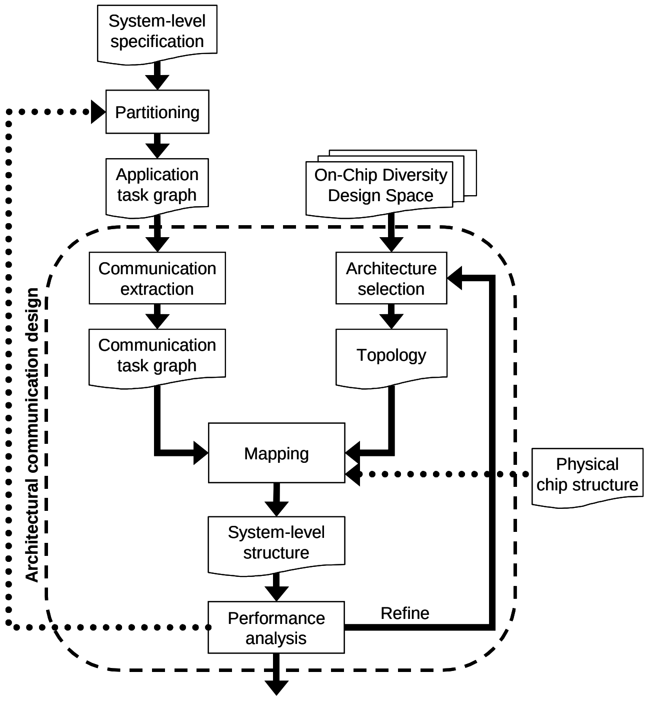

Figure 8. Framework for the communication architecture design with on-chip diversity

In order to select the most appropriate communication architecture for the target application, the design flow needs to take into account the structure of the application and the design choices offered by the on-chip diversity. As shown in Figure 8, a topology is selected for the communication architecture, considering eventually the physical constraints introduced by the technology that is used. The communication task graph needs to be extracted from the application specification and then mapped onto the selected architecture in an efficient way, in order to minimize the communication volume. The output of the mapping step is the system-level structure of the SoC, which can be refined subsequently in order to increase the overall performance.

IV. EXPERIMENTAL RESULTS

To asses qualitatively different communication schemes for on-chip diversity, we have experimented with several architectures using two very diverse driver applications: the first one is a parallel boolean satisfiability (SAT) solver [13] which is a data-intensive application, while the second one is spatial beamforming [16] which is a more control-oriented application. The results obtained in these two very different applications are interesting and worthwhile of discussion. However, due to space limitations, in what follows we report only the results for the beamforming application and examine the ways in which the structure of the application may affect the choice of the communication architecture.

A Spatial beamforming application

Beamforming is an important modern application which is used in a range of domains from the military to medical instruments and consumer electronics. Recent advances in MEMS technology have lead to the creation of high-precision acoustic sensors that can be embedded in a SoC, thus enabling the design and manufacturing of special purpose chips for acoustic beamforming. Zhang et. al. [16] propose a parallel algorithm which uses ultrasound beamforming to perform *3D volume reconstruction*, targeted at medical imaging instrumentation. As for the case of SAT, we believe that using a reliable communication scheme such as the one presented in Section B would lead to a significant performance improvement when used in the context of on-chip diversity.

A beamforming system is made of an array of sensors (similar to Figure 1) that perform data acquisition and a back-end computing architecture responsible for the raw data processing. The incoming sound waves, as well as their multipath components, are received and digitized by the sensor array. A certain segment of the digitized signals is processed by the beamformer, resulting in discrete time series. A digital beamformer is a spatial filter that processes data from the array of sensors in order to enhance the signal received from a certain direction or even to identify the source of the signal. This process results in a great reduction of the background noise.

As shown in [16], beamforming can be implemented in a parallel computational environment. The inputs received from the 2D planar array of sensors can be decomposed into two linear array beamforming steps. The first step includes a linear array beamforming along the X axis, which will be repeated a fixed number of times. Then, a line array beamforming is performed along the Y axis, which will output the final results. This algorithm has the advantage that all line array beamformers can be executed in parallel, resulting in a high degree of coarse-grain parallelism. Between these two phases, however, the data needs to be reorganized in the memories of the processing nodes, which results in an *all-to-all communication* pattern.

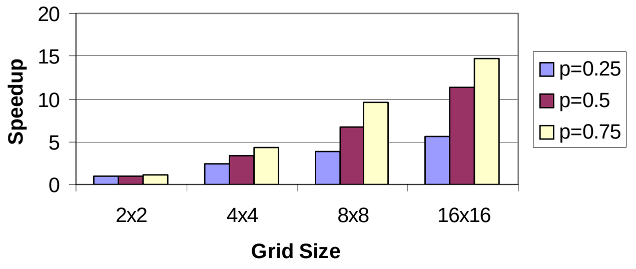

Figure 9. Scalability of the beamforming application

For our experiments, we assume that the signal acquisition is performed by an array of MEMS-based sensors and the beamforming is executed on a network of processing elements (as in Figure 1). The basic communication element of our architecture is the stochastically communicating NoC with different grid sizes. The communication architecture we consider may have several topologies, as shown in Figure 6; later in this section we report on the differences in performance between these configurations. For the experiments involving multiple voltage islands, the interface buffers (see Figure 7), may add extra delay. However, the decision to consider these penalties should be based on whether or not the actual size of the packets makes such effects sizeable.

B Scalability results

Because of the underlying communication strategy that we are using, we expect to see that the beamforming application scales very well with size of the on-chip network. In Figure 9 we have plotted the speedup of the algorithm using various grid sizes, normalized to the time taken by a reference 4x4 grid. We also display the results obtained with a range of transmission probabilities (p = 0.25, p = 0.50, p = 0.75) in order to show the impact of this parameter of the algorithm on the overall performance. The buffer sizes of the nodes and the time to live (TTL) of the messages were chosen to be appropriate for each grid size. As it can be seen in Figure 9, at each step, the number of IPs is increased 4 times, such that the network used in one experiment can be considered as an island of the next experiment (see left side of Figure 6). We can see in Figure 9 that, by increasingly adding more such self-similar islands, we can achieve speedups as high as 15x.

We note, that the increasing trend in the speedup we have obtained in case of the SAT solver, is even stronger than the one we are observing in Figure 9. In Figure 10 we see that the speedup seems to be almost linearly increasing with the number of nodes. This can be explained by the fact that each application has its own traffic patterns and hence the relative performance gains will always be dependent on the nature of the application. In the parallel implementation of the beamforming application, the tasks have a tighter coupling than tasks for the SAT solver (because of data dependencies that occur at every step of the algorithm). This makes the intrinsic parallelism of the beamforming application harder to exploit, while in the case of the SAT solver the nodes are executing sub-tasks that are almost independent. Nevertheless, we can notice that, just as in the case of SAT, our communication scheme scales very well with the number of communicating nodes in the NoC and brings a non-decreasing efficiency to the application.

C Fault-tolerance results

From the fault-tolerance point of view, we have also studied the impact on-chip soft errors have on the behavior of this algorithm. We have measured the latency and number of message transmissions in the network, normalized to the case when there are no errors. In the top

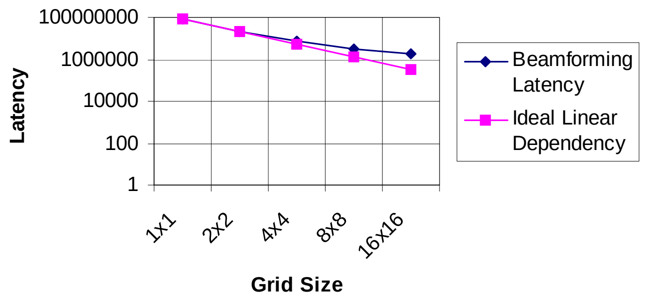

Figure 10. Improvement of the latency with the grid size

part of Figure 11 we can see that the latency increases under the occurrence of data upsets. This can be understood by looking at the way stochastic communication handles this type of errors. Our implementation uses an error detection mechanism to check for data corruption, and when such a corruption is detected the network packet is discarded (see Section B). Because of this reason, when the soft error rate increases, more packets will be discarded and the system will suffer a slowdown. However, we can see that the beamforming application manages to finalize its task because of the inherent fault-tolerance of the communication strategy. Soft errors often cause us to loose packets, but each message is duplicated in many packets and message loss is extremely unlikely. Furthermore, as shown in the top graph in Figure 11, the latency increase does not seem to depend heavily on the grid size.

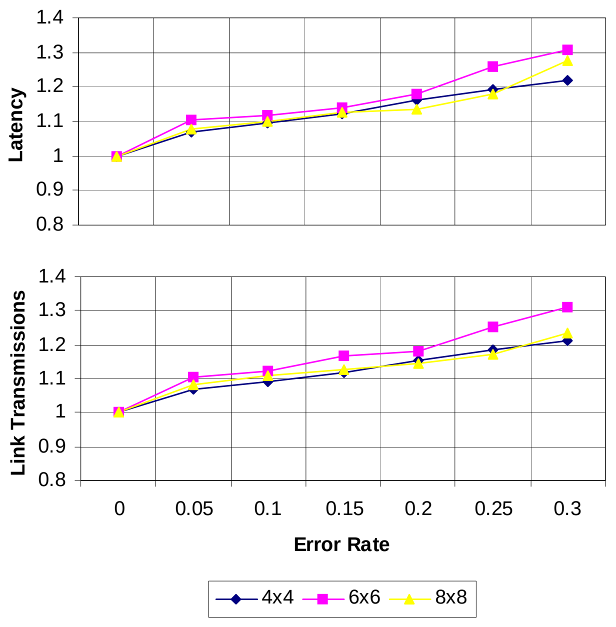

Figure 11. Performance of the beamforming application

In the lower part of Figure 11, we can observe that the number of link transmissions also increases under the presence of failures, which leads to an increase of the energy dissipated during the communication process. This is due to the increased running times of the application which cause more messages to be generated in order to recover from the errors.

D Diversity results

As we want to asses the impact of on-chip diversity on the beamforming application, we have experimented with four different configurations of the on-chip communication architecture. The first configuration is a flat 8×8 NoC, running the basic stochastic communication algorithm described in Section B. In the second configuration, we have organized our IPs in a hierarchical NoC, with four 4×4 regions connected by a higher-level stochastically communicating network (see the left side of Figure 6). In the third experiment we have connected the four regions with a shared bus, as in the middle of Figure 6, while in the fourth we have organized the 64 IPs into two voltage islands, with half of the chip running at a lower frequency than the other half (as shown in Figure 2).

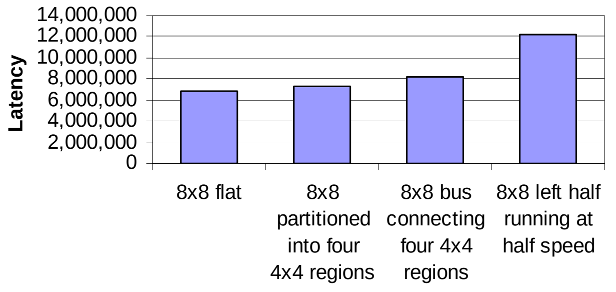

Figure 12. On-chip diversity: latency

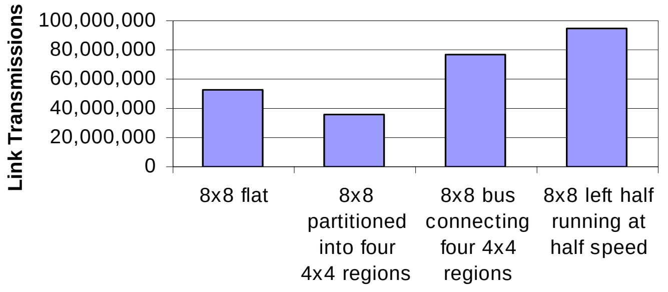

Figure 13. On-chip diversity: link transmissions

Figure 12 shows the average latency obtained with these four communication architectures, while the number of message transmissions is displayed in Figure 13. We can see that the hierarchical NoC has the lowest number of message transmissions, leading therefore to the lowest power consumption, while the flat NoC has a slightly better latency than the other solutions. The voltage island-based approach did not succeed to lower the power consumption in this case because of the inherent symmetry of the beamforming application, which does not allow for any of its modules to be run at a lower voltage and frequency without incurring a performance and energy penalty. In case of the SAT solver, however, the voltage island implementation did result in significant energy savings. The use of a partitioned network in Figure 13 gives the lowest number of link transmissions (and thus the lowest power consumption), if the slight performance disadvantage compared to a flat topology is acceptable.

V. CONCLUSION

In this paper, we have introduced the on-chip diversity and we have detailed a supporting design methodology based on stochastic communication. Our experimental results indicate the advantages of this hierarchical approach over flat designs for two very different applications. Combining heterogeneous architectures and technologies in a multiple voltage/frequency island environment allows circuits to achieve the highest levels of performance. It also introduces a new dimension of flexibility in SoC design. Because the design is partially based on existing CAD tools and design practices, the transition to these novel structures would certainly be smoother than a sudden paradigm shift to a new technology.

VI. REFERENCES

[1] Semiconductor Association. The International Technology Roadmap for Semiconductors (ITRS), 2001.
[2] Karnik, T. et. al. Sub-90nm Technologies--Challenges and Opportunities for CAD. In Proc. ICCAD, 2002
[3] W. Dally and B. Towles. Route packets, not wires: On-chip interconnection networks. In Proc. of Design Automation Conference, June 2001.
[4] Bertozzi, D., Benini, L. and De Micheli, G. Low power error resilient encoding for on-chip data buses. In Proc. of DATE, 2002.
[5] Constantinescu, C. Impact of Deep Submicron Technology on Dependability of VLSI Circuits. In Proc. DSN, 2002.
[6] Chapiro, D. M. Globally Asynchronous Locally Synchronous Systems. PhD thesis, Stanford University, 1984.
[7] Chelcea, T. and Nowick, S. Robust Interfaces for Mixed-Timing Systems with Application to Latency-Insensitive Protocols. In Proc. DAC, 2001.
[8] Kao, J. et. al. Subthreshold Leakage Modelling and Reduction Techniques. In Proc. ICCAD, 2002
[9] Ziegler, M. and Stan, M. A Case for CMOS/nano Co-design. In Proc. ICCAD, 2002.
[10] Lackey, D. et. al. Managing Power and Performance for System-on-Chip Designs using Voltage Islands. In Proceedings of the ICCAD, 2002.
[11] Leon-Garcia, A. and Widjaja, I. Communication Networks. McGraw-Hill, 2000.
[12] T. Valtonen et. al. Interconnection of autonomous error-tolerant cells. In Proc. ISCAS, 2002.
[13] M. R. Garey, D. S. Johnson. Computers and Intractability, A Guide to the Theory of NP completeness. Freeman, San Franscisco, Cal., 1979.
[14] Sylvester, D. and Keutzer, K. Rethinking deep-submicron circuit design. IEEE Computer, Nov. 1999.
[15] Dumitraş, T., Kerner, S. and Mărculescu, R. Towards on-chip fault-tolerant communication. In Proc. ASP-DAC, 2003.
[16] Zhang, F. et. al. Parallelization and performance of 3D ultrasound imaging beamforming algorithms on modern clusters. In Proc. of the Int. Conf. on Supercomputing, 2002.
[17] De Micheli, G. Designing Robust Systems with Uncertain Information. ASP-DAC 2003 Keynote Speech, January 2003.