Perspectives

Designing Micro- and Nanosystems for a Safer and Healthier Tomorrow

Giovanni De Micheli
EPF Lausanne

In this article, I provide an assessment of the current status and needs of electronic systems, their design, and their evolution. To put this analysis into perspective and to motivate it, I consider the progress of electronics over the past 50 years, from the invention of the transistor to the microprocessor, to the design of complex multiprocessors that we see today within gaming consoles and other appliances. Looking forward to the next 50 years, I want to address how we have affected and will affect society with our inventions and products, from personal computers and communicators to the upcoming networked systems for health and environmental monitoring.

Various challenges confront us today, including our need for more processing power within our electronic systems to support complex software applications. We desire systems that are batteryless or that at least consume less energy to run. We need affordable and competitive manufacturing technologies. But besides purely technical challenges, I want to address the issue of how we can have a deeper impact on society. Who will benefit, as end users and as commercial providers, from the progress in electronic systems? Which market sector will reap the benefits of the new inventions: the semiconductor, system, or service sectors?

We can start addressing these questions by analyzing current products, their design requirements, and their possible evolution. *Electronic design automation* (EDA) provides the enabling technology to design complex chips and is based on principles of formal modeling, analysis, and synthesis. Unfortunately, EDA is entangled in solving many problems related to deep-submicron design, so it has missed opportunities in system-level design and is still mainly a small niche market. Our objective should be to reposition design automation as a central engineering task. We want a broader, more scientifically challenging scope that will attract the best young researchers and create more value for society and the economy.

In the next 50 years, we will increasingly see electronics distributed everywhere—in clothing, cars, homes, offices, the environment, and so forth. We are talking about a global market that affects people's everyday lives. We have some audacious goals, such as breaking language barriers by creating portable devices that are powerful enough to do real-time language translation. We want to eliminate energy dependence for electronic systems and design them to be autonomous, thus eliminating the problems of changing and disposing of batteries. We would like to link up every human on the planet. We want to intelligently support human health, as we all want to live longer and better. At the same time, we would like to monitor and protect the environment of our planet. All these goals require a broader vision on manufacturing and design technologies for electronic systems.

Features and challenges of SoCs

First, I would like to position electronic chip design on the basis of today's requirements. Most systems must be mobile, which requires ultralow power design and eventually low-voltage operation. Many systems have life-critical applications—for example, those for vehicular control and health monitoring. Thus, we need high reliability, which we can achieve via redundancy. Systems must deliver high performance to run complex software applications, which, together with low voltage operation, implies that the systems

0740-7475/08/$25.00 © 2008 IEEE

Copublished by the IEEE CS and the IEEE CASS

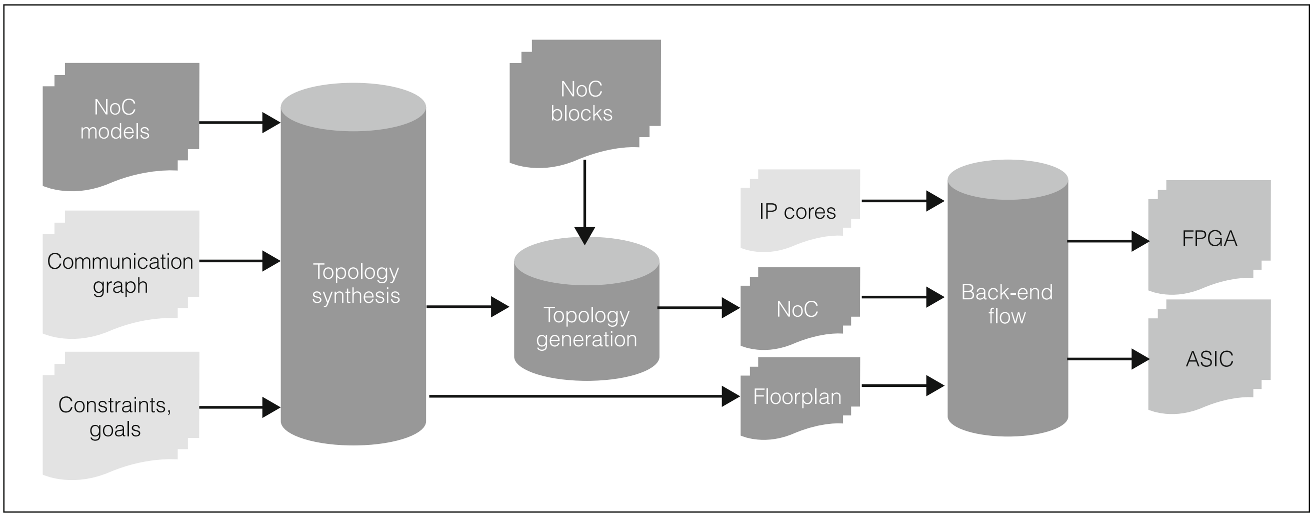

Figure 1. A design flow for network-on-chip (NoC) design. (Courtesy of iNOCs.)

must have parallel architectures. For these reasons, we have witnessed a change from processors to multiprocessors in the past few years. There is technology support for designing multiprocessors, but we also need to rethink the way in which we design software.

Fabrication technology support is also an important issue. We know that CMOS technology scaling is going to slow down and stop soon, but there are many new ideas, such as the use of *silicon nanowires* (SINWs), *carbon nanotubes* (CNTs) , and bistable molecules like *rotaxane*. The important question is whether these new technologies are ready for system design. And if they are, can they be mixed and matched with CMOS? We already have examples of hybridization of technologies, such as using SINWs and CNTs together with CMOS cells to provide interconnections. Nevertheless, these examples are still in the research domain. Another important issue is how to do design with these technologies. Is this just a question of changing the back end of the design flow, or should we rethink the way in which systems are conceived and synthesized? This is a key issue, especially considering that new technologies have higher defect densities and failure rates.

While looking at computational structures within chips, we must consider two important requirements: predictable design in terms of timing and fast design closure. *Crossbar* architectures have received wide attention lately, because geometry and timing properties of interconnections are regular and predictable. This design style—reminiscent of programmable logic arrays (PLAs) in the 1980s—also provides a way of matching nanotechnologies with current lithography-based microtechnologies. For example, crosspoint sites can be personalized to do computation (or storage), and nanotechnologies can serve to assemble specific switching circuits in the sites themselves. Important issues are how we can effectively use such crossbar structures within bigger chips and how can we address the sites with wires that are designed using optical lithography. Related issues include matching dimensions, voltages, and currents. Therefore, from this perspective, the use of multivalued logic, as well as redundancy and encoding to address possible local failures, is important.[1]

Predictable design and fast design closure are also important for the communication fabrics on chips. The paradigm of choice today is the *network on chip* (NoC), where processing and storage elements communicate via packet routes.[2] NoCs provide modular and flexible interconnects as well as reliable on-chip communication. In 2007, Intel designed, fabricated, and tested a large chip having 80 cores interconnected by a NoC.[3] An important issue is how to design NoCs that can be synthesized and optimized. Today, new design flows and tools (as well as emerging start-up companies) make it possible to implement NoCs starting from high-level specifications and to tailor them to the required applications, thus providing higher performance and lower power consumption (Figure 1).

Packaging is playing an important role in SoCs. We see a trend of moving from planar to 3D integration because chips have limited wiring resources and because electrical and manufacturing constraints limit integration of heterogeneous blocks on a plane.

Fortunately, technology comes to the rescue. With *through-silicon vias* (TSVs), we can stack different chips in layers to provide different functions, such as computing arrays, memory arrays, analog and RF circuitry, microantennas, and so on. But how do we realize the interconnection over such 3D chips? In this case, 3D NoCs can provide an effective and reconfigurable means of implementing communication.

Silicon chips not only feature electrical functions. Today, we see increasingly more mechanical parts being mixed and matched with electrical parts, such as *microelectromechanical systems* (MEMS), which, for example, can be used to harvest energy from the environment. Another integratable function is microfluidics on chip to transport fluids, as well as biological or inorganic samples. There are several interfaces between the living world and electronic chips, such as neural interfaces and implants, requiring dedicated electronics.

It is time to think in terms of a broader codesign paradigm, beyond hardware and software codesign, where we design, mix, and match components of different natures. Overall, we want to harmonize the design of complex heterogeneous systems. We also must remember that chips are embedded within an environment. They sense, process, and communicate. This can be done over the body with a body area network or within a geographical area with local, metropolitan, or wider networks. To perform information processing, production, and consumption, SoCs must also have correct and dependable software.

When thinking broadly of all the applications and services that new SoCs can provide, we must realize that the enabling system design technology will be an evolutionary form of the current EDA tools and methods. I like to call it *system-level design technology*, and, as EDA itself, it is based on modeling, analysis, and synthesis. As we explore the future, we need to build on previous experience, but we also need to be bold and look ahead with a broader perspective. Specifically, we must address how to engineer heterogeneous complex systems in all their facets, as scientific and commercial value will stem from the holistic system aspect of the design.

Evolution of VLSI and design technologies

Now, I'd like to address the evolution of VLSI and design technologies through three case studies: the evolution of SoCs toward *labs on chips* (LoCs); the evolution of design automation tools and methods toward bioanalysis and synthesis; and eventually the evolution of interconnect technology toward wireless sensor networks.

From SoCs to LoCs

An LoC integrates chemical and biological manipulation on an intelligent substrate. LoCs are very versatile; for example, they can be used at medical points of care for computer-aided diagnosis, and in environmental networks for pollution control. LoCs promise to revolutionize medical care: this is important both for advanced countries (where the cost of healthcare is skyrocketing) and for developing countries (where it is important to bring medicine at an affordable cost to everyone).

When looking at how LoCs are realized inside, many interesting features surface. LoCs provide the ultimate hybridization of technologies. Microfluidics can handle the sample transport, and sensors can bind to proteins, DNA, and viruses. We also need low-noise electronics and powerful on-chip data-processing algorithms and software. Now, I address the different technological challenges by going through some examples. Biological samples, for instance, can be moved on chip via a magnetic field generated by spirals that eventually are designed on the chip's top metal level (see Figure 2a).[4] We want to transport, and possibly split and merge, droplets over a 2D array. This involves scheduling and routing multiple samples at the same time. Interesting enough, the technology for controlling droplet transport is similar to the technology developed in the EDA community for high-level synthesis.[5]

Figure 2b shows how a DNA strand can bind to a complementary probe. We can fabricate chips with DNA probes, where the matching of DNA to a probe creates a redox reaction, which then can be measured via a sensor under the probe itself. This lets us design chips that do *nonlabeled sensing*—that is, where there is no need for tagging DNA by fluorophores or for using bulky and expensive optical readouts. Thus, we can integrate the sensing with the electronics to do the measurements and achieve lower-cost devices.[6] Moreover, we can array the probes and create 2D matrices that can do parallel sensing. These array detectors are critical for achieving *high-throughput biology* experiments. Since these sensors generate considerable data, the fast and correct interpretation of this data is critical. The end result of a measurement is a signature

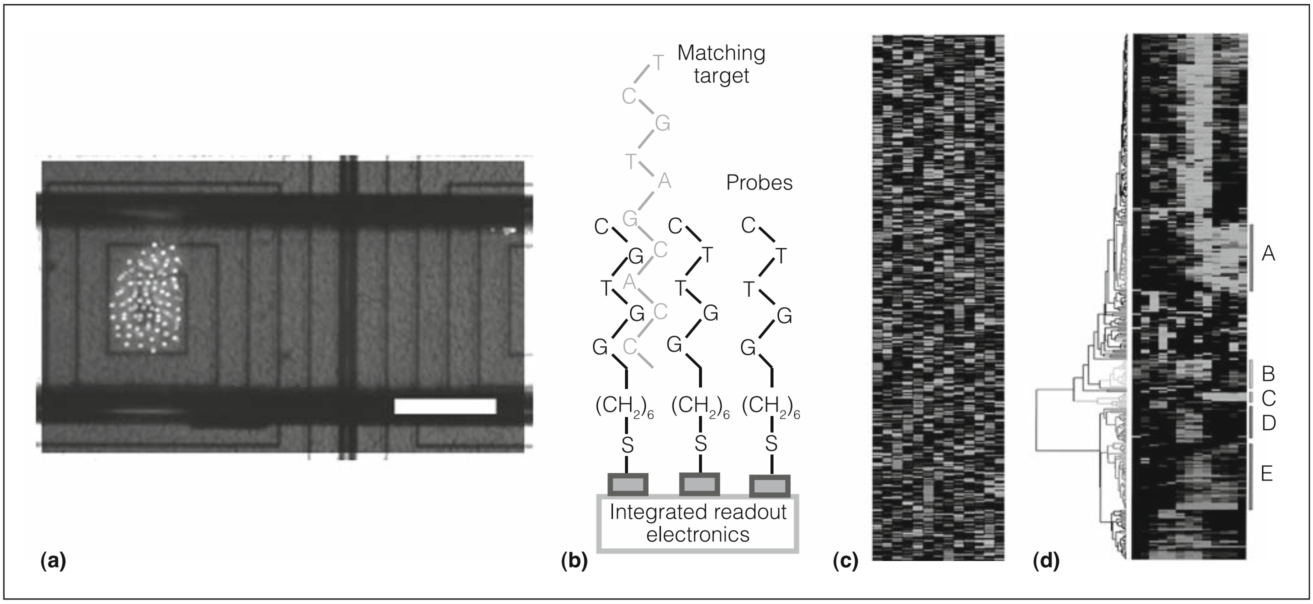

Figure 2. Various aspects of LoCs: droplet transfer by magnetic fields[4] (a), integrated DNA sensing (letters represent bases and chemical compounds) (b), graphic rendering of gene expression levels (c), and clusters of expression levels (represented by capital letters on the side) (d).

of the presence of a virus, disease, or specific compound.

In this case, data mining and interpretation can leverage some techniques based on *clustering* that are reminiscent of methods used within EDA. Figure 2c shows a graphic rendering of the results obtained on a microarray. The shading level (gray color intensity) at each crosspoint represents a specific gene's expression level (for example, presence or absence), and the array is organized in terms of rows that represent genes and columns that represent samples.

Figure 2d shows a matrix row and column permutation that clusters areas with similar shading. Roughly speaking, this lets us relate specific conditions, given the presence or absence of specific genes. Now, the clustering, or in this case the biclustering, of the data can be done efficiently using techniques based on *binary decision diagrams* (BDDs) and *zero-suppressed decision diagrams* (ZDDs). Again, these technologies, developed within the EDA community, have been successfully ported to the bioinformatics community.[7]

Overall, there are several objectives for LoCs. One is *biodiscovery* (that is, finding new biological mechanisms). Another is to help medical doctors by providing better diagnosis tools—for example, linking genetic data to clinical traits and databases. We can use LoCs to do microchemistry, creating compounds via microreactions and supporting experiments in the field. As in the case of *field-programmable gate arrays* (FPGAs), generic LoCs can be programmed to do a specific experiment, such as looking for particular compounds in water. This field-programmable LoC could then be programmed according to the area in the world where we want to use it and the corresponding test.

Bioanalysis and synthesis

By *analysis*, I mean understanding the underlining mechanisms—namely, understanding the full meaning of the ''-omics,'' such as genomics, proteinomics, and so on. By *synthesis*, I mean modifying or creating new realities. Examples include synthesizing drugs that alter specific genetic or metabolic pathways, and synthesizing biological compounds that support computation. The latter, called *synthetic biology*, is attracting a lot of attention today.[8]

For both analysis and synthesis, we need multiple abstractions of the biological materials and reactions. Figure 3 shows examples of abstraction layers. There is of course the abstraction where we model biochemical reactions, with their own event timing, and where the appropriate model is in terms of differential equations. There is also the logic-level abstraction, which is the *zero-delay model*, where we see transitions among states and where we just care about how we go from one state to another, independently of the time it

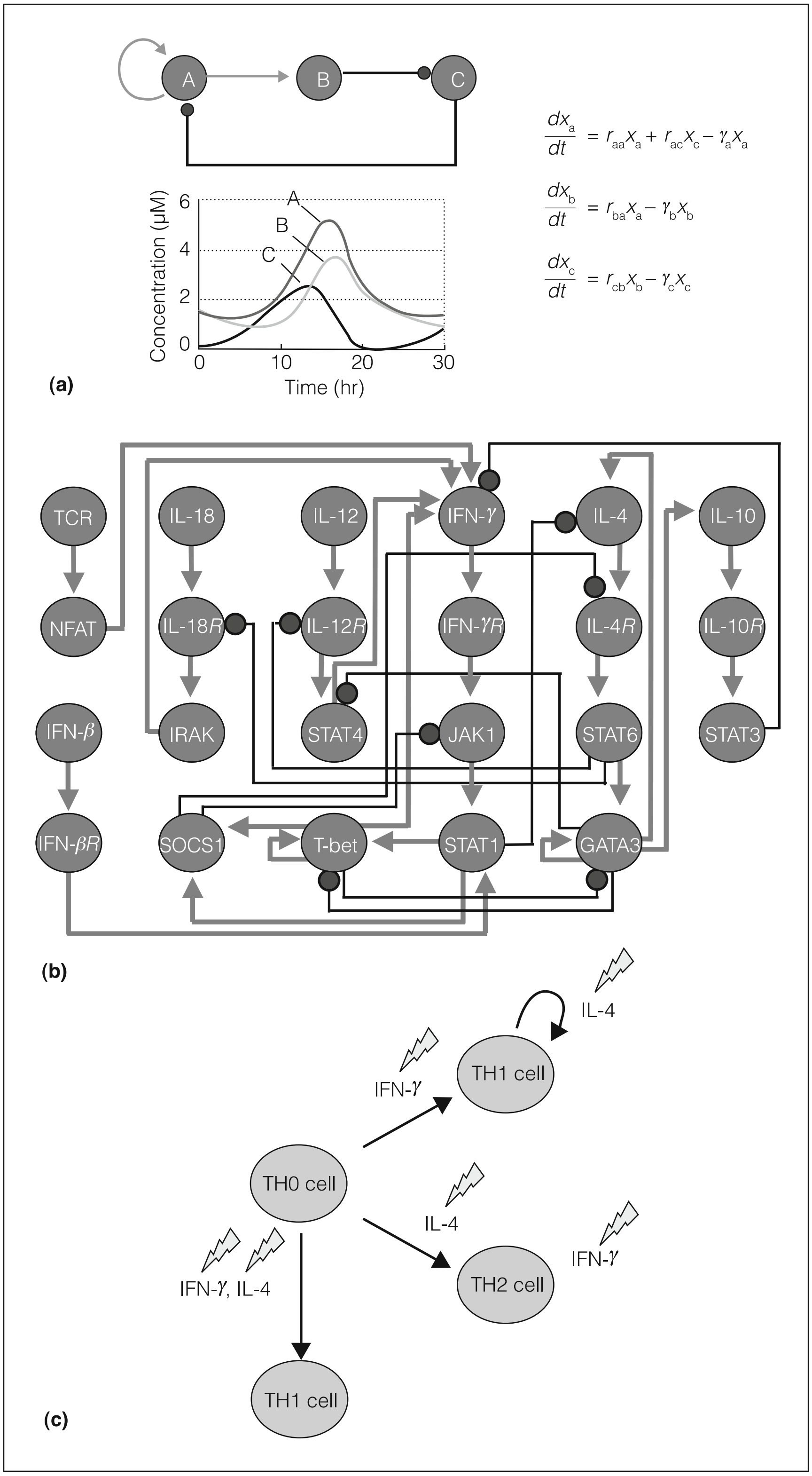

Figure 3. Abstraction levels in biology: biochemical model (example of expression-level variation) (a), zero-delay model (nodes represent specific signaling proteins) (b), and functional model (c).

takes. We can have synchronous or asynchronous models, but overall these models are reminiscent of finite-state systems. We then have *functional abstractions*, where we have a biological function and care about the input-output relation.

For an example with biological significance, consider the T-helper (TH) cells that perform an important role within our immune system. Figure 3c models the TH0 cell, which can evolve into TH1 and TH2, according to the presence or absence of some specific compounds. Figure 3b shows how this model is refined into another showing state transitions. States are denoted by specific compounds. Transitions are stimulated by edges whose head is an arrow, and inhibited by edges whose head is a circle.[9]

We can see how close this abstraction is to the *finite-state machine* (FSM) model we use for circuit design. With this abstraction, we can leverage the *orthogonalization of concerns*[10]—that is, we can focus on logic behavior independently of timing. This technique has been developed for circuit verification and has been recently applied with success to biological-system analysis.

For various reasons, we must distinguish between simulation and traversal of the state space. For example, sometimes the final steady state is the objective of analysis, and we can effectively reach this objective by using traversal methods. We know very well that implicit methods can help us handle large amounts of data. Moreover, we can modify systems by perturbation and do experiments in silico—for example, the knock-out experiment in which we silence a gene by zeroing its expression level. Interesting enough, knock-out is the equivalent of a stuck-at-zero, which is a well-known concept in the testing community.

Overall, one objective of bioanalysis and synthesis is the development of *pharmacogenomics*, focusing on drug therapies that are tailored to the patients' genotypes. As a result, we can construct drugs that directly affect genetic or metabolic pathways. Another objective is *synthetic biology*, in which engineering systems are based on biological components. Important ingredients are the abstractions, such as libraries, and the synthesis process that enables combining components to perform a specific function. On a similar note, biologically driven computation is important, such as using DNA as a way to support computation. Conversely, we can use DNA as a way of creating scaffolds to build microstructures and nanostructures on silicon or on other materials. All these are extremely challenging objectives, and the corresponding software infrastructure has strong links to electronic design technologies.

Large-scale sensor networks

We are embedded in the natural environment, and so we must live with many inconvenient realities—from avalanches to tsunamis, to volcanic explosions, to earthquakes. We know that we can use wireless sensor networks to monitor and control the environment, but we still have to face many challenges today, such as the massive amount of data that networks must process and transmit. We also have issues related to how we distribute sensing nodes, how we power them up, and how we provide redundancy to tolerate local failures.

Overall, when we think of engineering environmental systems, we must consider integrated sensing, computation, communication, and embedded software. An important related issue is how we partition local versus global data processing and communication, and as a result, data abstraction is extremely important. Indeed, we have large amounts of data that we need to reduce locally before transmission.

At the same time, we must perform data interpolation or extrapolation to fill in the voids for missing data samples. Therefore, we need a different paradigm of computation, somehow similar to what ants and bees use: *distributed intelligence*. We want to be able to reason and act locally with some global information. This is a new paradigm for computation that is radically different from what has been used in the past while gathering large amounts of data and then using powerful supercomputers. Here, we want to distribute the computation and do it in part on reasonably small amounts of data.

The quest for energy efficiency is also extremely important, especially in view of the sky-rocketing cost of nonrenewable natural resources. For example, distributed wireless systems eventually need to be autonomous. Energy must be harvested from the environment, for both mobile and fixed applications. For this reason, there is already much interest in the field of energy harvesting, which can be seen as the conversion of unused (or degraded) energy into information.

But there is also a dual problem that is just as important: energy distribution must be efficient. When looking at the design of smart homes, buildings, factories, and electrical grids, we strive to use local information to optimize energy consumption and/or distribution. In this case, we convert information into energy savings. Overall, we see a mutual interaction between energy and information. Therefore, policies for runtime energy and information management are extremely important now and in the future, and we can see this as the ultimate evolution of policies for power management.[11]

Another important issue is how we physically interact with the environment—for example, in the cases of computer-assisted driving and aids to assist the visually impaired. Embedded electronic systems are instrumental to providing these services. A related issue is how we socially navigate the environment—for example, how we find information in the geographic area around us, and how we can meet within virtual worlds. As an additional example, think of the future of technical meetings—such as the Design, Automation and Test in Europe (DATE) Conference circa 2058—which may be a virtual rendezvous located at an Internet address, where delegates communicate by using avatars and software to convey information. The challenge in creating this reality is how to design embedded environments where the users interact and are fully immersed.

Cooperative engineering is a key factor in achieving this vision. We need to be able to bring together engineers, scientists, and doctors with different skills. We must find ways to translate specific technical idioms and provide a means for people with different backgrounds to communicate. Once again, information abstraction and modularity will be extremely important, as well as the creation of collaborative workspaces.

There are some examples of cooperative and multidisciplinary research activities that leverage the technological growth of SoCs. Among these, the *nano-tera.ch* (http://www.nano-tera.ch) program has the objective of developing micro-, nano-, and information technologies to help design and manage distributed embedded systems. Application domains range from bettering human health by developing diagnostic means and wearable sensor networks to monitoring the environment to prevent disasters and thus provide individuals and communities with better security. As a second example, the *humanitarian technology challenge (HTC)* is a new partnership between the IEEE and the United Nations, with the objective of identifying the technologies in the health and environment domains that can benefit developing countries. Examples include food, water, and health monitoring. Both initiatives have broad and altruistic objectives, that hopefully can raise enthusiasm among engineers and bring young people to this profession.

The road ahead has both challenges and rewards. It is extremely important to expand our horizon beyond SoCs dedicated to computation, because this is key to scientific viability as well as commercial profitability. To construct global systems, we need both heterogeneous hardware and the corresponding software infrastructure. Product and system design is an extremely complex task because there are many aspects of design and many technologies that need to be made compatible. Finally, we need system-level design technologies, which are crucial for the design and runtime management of complex systems. ■

Acknowledgments

This article is an edited transcript of the opening keynote speech at DATE 2008 (See http://si2.epfl.ch/~demichel/misc/DATE2008 for slides and an audio feed.)

■ References

1. H. Ben Jamaa et al., "Fault-Tolerant Multi-Level Logic Decoder for Nanoscale Crossbar Memory Arrays," *Proc. Int'l Conf. Computer-Aided Design* (ICCAD 07), IEEE CS Press, 2007, pp. 765-772.
2. G. De Micheli and L. Benini, *Networks on Chips*, Morgan Kaufmann, 2006.
3. S. Vangal et al., "An 80-Tile 1.28TFLOPS Network-on-Chip in 65 nm CMOS," *Proc. Int'l Solid States Circuits Conf.* (ISSCC 07), IEEE Press, 2007, pp. 98-99.
4. U. Lehmann et al., "A CMOS Microsystem Combining Magnetic Actuation and In-Situ Optical Detection of Microparticles," *Proc. 14th Int'l Solid-State Sensors, Actuators and Microsystems Conf.*, IEEE Press, 2007, pp. 2493-2496.
5. J. Ding, K. Chakrabarty, and R. Fair, "Scheduling of Microfluidic Operations for Reconfigurable Two Dimensional Electrowetting Arrays," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 12, Dec. 2001, pp. 1463-1468.
6. C. Stagni et al., "Fully Electronic CMOS DNA Detection Array Based on Capacitance Measurement with on-Chip Analog to Digital Conversion," *Proc. Int'l Solid States Circuits Conf.* (ISSCC 06), IEEE Press, 2006, pp. 69-68.
7. S. Yoon et al., "Discovering Coherent Biclusters from Gene Expression Data Using Zero-Suppressed Binary Decision Diagrams," *IEEE Trans. Computational Biology and Bioinformatics*, vol. 2, no. 4, July-Sept. 2005, pp. 339-354.
8. D. Bake et al., "Engineering Life: Building a Fab for Biology," *Scientific American*, vol. 294, no. 6, June 2006, pp. 44-51.
9. A. Garg et al., "An Efficient Method for Dynamic Analysis of Gene Regulatory Networks and in Silico Gene Perturbation Experiments," *Proc. 11th Ann. Int'l Conf. Research in Computational Molecular Biology* (RECOMB 07), LNCS 4453, Springer, 2007, pp. 62-76.
10. K. Keutzer et al., "System-Level Design: Orthogonalization of Concerns and Platform-Based Design." *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 12, Dec. 2000, pp. 1523-1543.
11. L. Benini, A. Bogliolo, and G. De Micheli, "A Survey of Design Techniques for System-Level Dynamic Power Management," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 3, June 2000, pp. 299-316.

Giovanni De Micheli is a professor and the director of the Institute of Electrical Engineering and of the Integrated Systems Center at EPF Lausanne, Switzerland. His research interests include several aspects of design technologies for integrated circuits and systems, such as synthesis, networks on chips, and low-power design, as well as systems on heterogeneous platforms. He has a PhD in electrical engineering and computer science from the University of California, Berkeley. He is a Fellow of the IEEE and the ACM.

■ Direct questions and comments about this article to Giovanni De Micheli, LSI, INF 341, Station 14, EPF Lausanne, CH-1015 Switzerland; giovanni.demicheli@epfl.ch.