# On-Chip Self-Calibrating Communication Techniques Robust to Electrical Parameter Variations

**Frédéric Worm, Paolo Ienne, and Patrick Thiran**
Swiss Federal Institute of Technology Lausanne

**Giovanni De Micheli**
Stanford University

> *Editor's note:*
> Dynamic self-calibration holds the promise of overcoming conservative worst-case design techniques needed to combat deep-submicron process and operating variations. This article proposes an on-chip point-to-point interconnect scheme characterized by self-calibration that can operate dynamically to achieve the best energy/performance trade-off.
> *—Soha Hassoun, Tufts University*

**Recent silicon technologies** used for SoCs are increasingly different from original CMOS, which was an almost ideal digital technology. Second-order effects of different sorts, such as capacitive or inductive crosstalk, have become more critical because of a wealth of physical phenomena, including 3D and quantum effects. Similarly, ever-decreasing geometries are widening the spread of electrical parameters linked to imprecision in the manufacturing process. All these factors contribute significantly to design process complexity. Without a dramatic, unforeseen change in circuit and manufacturing technology, this situation can only get worse.

The adopted design methodologies have always been based on worst-case design approaches. Gate and interconnect delays are a typical example. Circuits clock registers only when data is sure to be stable, and designers use worst-case analysis—as determined, for example, by static timing analyzers—to achieve this timing estimate. Designers simply model all sources of deviation from the nominal situation and total them to determine the most conservative estimate of the incurred delay.

Observed trends in worst-case analysis for current design methodologies could invalidate the benefits of faster, scaled-down semiconductor technologies. As a result, large capital investments in deep-submicron silicon fabrication might not return competitive chips. Worst-case design will show diminishing returns in speed as designers scale down devices and supply voltages. The complex interaction of several physical factors will become increasingly harder to model accurately, pushing designers toward ever more conservative assumptions. Although some research aims to improve the accuracy of worst-case static timing estimations,[1] a more radical approach is needed. Otherwise, there will be a heavy price to pay—mostly in terms of energy consumption—even as power savings becomes a primary goal in many SoC applications.

Figure 1 illustrates the point with a simple qualitative example. Recall that accurate knowledge of the delay and voltage relation is key for many optimization techniques, such as transistor sizing and dynamic voltage scaling. The nominal relation between delay and supply voltage is modified by several physical phenomena, whose cumulative effects constitute a worst-case relation. Therefore, at a given supply voltage, $V_{DD}$, a designer will assume the most conservative delay—that is, that the operating point is not, for instance, A but B—and implement the design accordingly. However, at a particular instant, the device is likely to be operating under far more favorable conditions—for instance, with a lower delay indicated by operating point C. This implies a waste of energy because operation at reduced voltage $V_{DD}'$ (B′)

0740-7475/04/$20.00 © 2004 IEEE

Copublished by the IEEE CS and the IEEE CASS

would yield the actual performance the system was designed for. A less conservative operation in B′ rather than B would achieve the same user function in the same time with potentially significant energy savings—roughly proportional to the difference of the square of the supply voltages $V_{DD}$ and $V_{DD}'$.

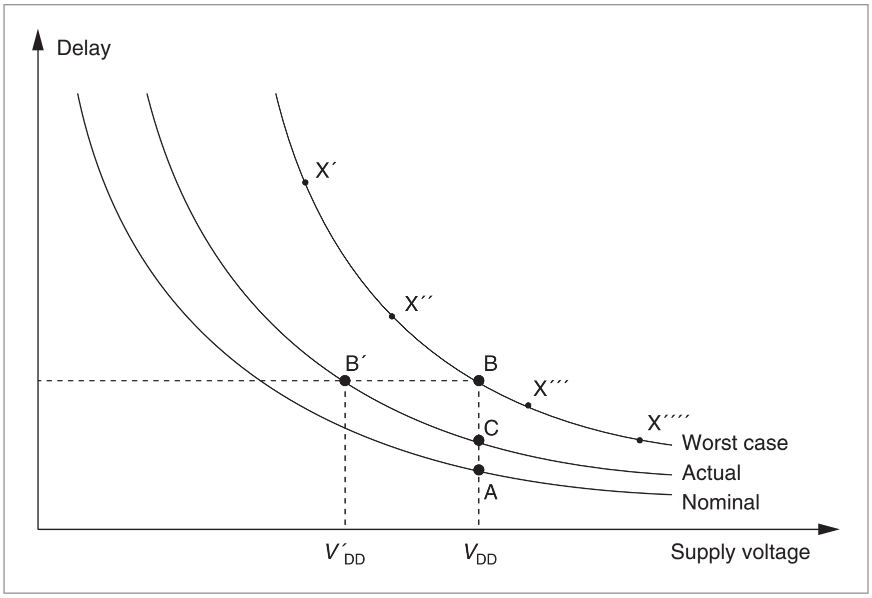


**Figure 1. Delay and voltage relation for nominal, actual, and worst-case design. The worst-case design typically wastes resources—usually silicon area and, more critically, energy. Traditional dynamic voltage-scaling techniques would select only points such as X′ to X′′′′.**

## Self-calibrating circuits

Tolerance for process, voltage, and temperature (PVT) variations is becoming increasingly important.[2] To achieve aggressive circuits that exploit the features of expensive downscaled technologies, we propose designing self-calibrating circuits that break the worst-case barrier. Such circuits determine operational parameters, such as voltage, at runtime to meet overall reliability constraints. In other words, these parameters ensure that the number of data and timing errors is bounded and that most can be corrected. Because all factors that reduce circuit performance could combine to realize the worst-case situation, self-calibrating circuits must still be conservatively overdesigned to withstand this possibility. However, we want to avoid paying the price (typically, in wasted energy) for such conservative designs in the general case.

Of course, using adaptive design techniques in extremely aggressive designs is not new; in some situations it is commonplace. Researchers and practitioners have already gone far. For instance, in a state-of-the-art commercial processor, regional clock skew is adaptively tuned at power-up using relatively complex controllers to compensate for local process variations across a single die.[3] Nonetheless, it is not common to use powerful digital controllers, such as complex finite-state machines, to adjust the operating point of transistors when the overall design might be jeopardized, or while the circuit is operating. Designers generally use tight analog feedback loops (phase-locked loops or delay-locked loops). We believe that digital controllers can be effective in certain limited but important situations.

We see a strong potential for applying small synthesizable digital controllers in applications in which

- it's possible to trade robustness for energy (for example, in Figure 1 an investment of energy guarantees correct operation under all conditions);
- it's possible to check, with low overhead, whether the system is operating correctly and, if not, to operate the circuit under different conditions; and
- the application has some intrinsic tolerance to limited latency deviations (as in modern communication systems and memory hierarchies).

This article describes an on-chip point-to-point interconnect scheme that permits on-line self-calibration to achieve the best energy/performance trade-off. We designed the scheme to recover from the occasional choice of an overly aggressive value for the operating point at which the interconnect, in fact, does not operate correctly—or at all.

The idea of operating CMOS devices at voltages below the worst-case characterization point—and thus in subcritical regions where errors might occur—has received little attention. A recent article addressed the possibility of exploiting devices in subcritical regions for DSP.[4] In that case, DSP algorithms compensate for errors arising from subcritical voltages. Our goal is similar, but it concerns communication rather than computation; therefore, we can exploit classic techniques to achieve correct behavior despite occasional errors—as the second condition in our previous list requires.

## On-chip interconnects

The successful design of highly complex SoCs depends on the availability of robust design methodologies that allow a short time to market with low risk. Faced with the need to integrate billions of transistors on a single chip, design technologies are under increasing pressure.

Designing such SoCs is possible only by using complex components such as full subsystems with processors, controllers, and digital signal processors as major predefined building blocks. Therefore, because of the difficulty of global synchronization, we can view modern SoCs as heterogeneous multiprocessing systems with multiple, possibly asynchronous, timing references. Given a library of modular components, designers' main challenge for future SoCs will be to efficiently connect such components into an effective network that implements the desired functionality. On-chip micronetworks, or networks on a chip,[5] will become the central focus of the design process and will inherit techniques such as layered design and packetized communications and methodologies from today's macronetworks.

In our discussion of long-distance on-chip VLSI interconnects (informally called buses), we focus, without loss of generality, on three objectives:

- *Performance requirements.* A bus implementing a communication link should provide enough bandwidth to support the required communication demand. This demand might not be precisely known in the early design stages. Additionally, we must recognize that a bus' workload can change dynamically, meaning its bandwidth needn't always be kept at its peak. Therefore, dynamically adjusting bus bandwidth can greatly enhance design versatility.
- *Energy consumption.* Studies have shown that wires account for a significant portion of total energy consumption (40% to 50%).[6] A large share of this consumption results from long, high-capacity wires crossing the die and connecting different subsystems. With larger dies and more subsystems on a chip, the proportion of power consumed by communication can only grow. Obviously, we need techniques to reduce the energy consumed by on-chip communication.
- *Reliability and noise sensitivity.* We already mentioned that many technological factors challenge the traditional robustness of digital CMOS design, and functionality depends on phenomena that are increasingly more difficult to model. This conflicts dramatically with the fact that the best way to achieve low-energy communication is to use small voltage swings, but at the cost of further decreasing a circuit's noise immunity. Design methodologies for interconnects must account for growing noise sensitivity and indeterminacy.

A common technique for minimizing power consumption on buses is to choose an appropriate encoding scheme that reduces switching activity without affecting the signal information content. This approach accounts for interwire capacitances[7] and has recently been extended to address reliability issues.[8] Bus encoding techniques have proved effective at reducing power consumption, although best results are generally obtained in specific devices, such as address buses. In fact, energy-efficient encoding complements our scheme.

As Figure 1 already suggests, the classic way to reduce power consumption is to use a lower supply voltage, and for interconnects and buses in particular, low-swing signaling techniques.[9,10] Although very effective on the power side, these techniques alone significantly compromise a design's robustness. Instead of helping designers address new deep-submicron effects, they further complicate the design process. Our proposed scheme uses low-swing communication judiciously while ensuring that the system's overall reliability does not decline but, on the contrary, increases.

Like low-swing techniques, the well-established and effective technique called dynamic voltage scaling (DVS) reduces power consumption in systems under given performance constraints.[11] Its most common application is for dynamically adapting mobile-processor speeds to current computational requirements, and several commercial processors (Intel XScale, Mobile Pentium, and Transmeta Crusoe) now support it. DVS is based on the characterization of devices at several different working points (pairs of supply voltages and operating frequencies). These pairs correspond to a set of safe operating conditions computed or measured while accounting for all worst-case parameters—for instance, points X′ to X′′′′ in Figure 1.

Shang et al. introduced a transmission scheme applying DVS to chip-to-chip interconnection networks.[12] Such a system is a direct extension of processor voltage scaling and assumes the knowledge of a fixed relation between voltage and frequency for safe operation. Our communication scheme similarly extends the idea of DVS to on-chip communications in the form of variable voltage-swing signaling, but in the spirit of our self-calibration idea, it doesn't rely on prior knowledge of robust working points.

## Self-calibrating architecture

For simplicity, we focus on a typical unidirectional point-to-point interconnect between subsystems. Figure 2a shows a qualitative view of the classic interconnect. At the producer end, a FIFO or similar buffer decouples the two subsystems, which might operate at different frequencies, and a large driver (typically a chain of appropriately sized inverters) charges or discharges the large capacitance represented by the interconnecting wires. A receiver (typically a CMOS gate) compares the voltage level of the line to a threshold.

As Figure 2b shows, we add a few elements to the classic scheme. To reduce the energy consumed per bit, we apply a form of DVS to the interconnect by dynamically controlling the driver swing and the corresponding receiver threshold. There are well-known electrical schemes to reduce the interconnect's voltage swing. Of course, the variable voltage swing affects the speed at which the interconnect driver can charge or discharge the load capacitance; thus, lower swings reduce the maximum reliable operating frequency. Hence, we need to adapt the communication speed, too, as in traditional DVS techniques.

Our architecture is seamlessly applicable to segmented buses. In such cases, we can use the same voltage swing along all segments because every repeater consists only of an inverter supplied at voltage $V_{ch}$. Later in this article, we report conservative results that consider the energy spent only on the interconnect wires. In reality, the repeaters draw additional energy, which also scales down with our technique.

Operating with lower voltage swings makes our communication more sensitive to several noise sources. To cancel this effect, we introduce error detection encoding at the word level on the source side, and we implement a typical automatic repeat request (ARQ) strategy, namely Go-Back-N.[13] The ARQ strategy entails small latency variations. Although hard real-time applications might suffer from these variations, many practical embedded systems can tolerate them because of their softer real-time constraints. Finally, our scheme requires a self-calibration controller that decides on the operating frequency and voltage swing. This controller must choose voltage/frequency pairs from a set of safe operating points and as a function of the requested bandwidth. It must also explore the design space to discover safe and lowest-power operating points. Therefore, it needs as an input some information on both bandwidth requirements and link reliability.

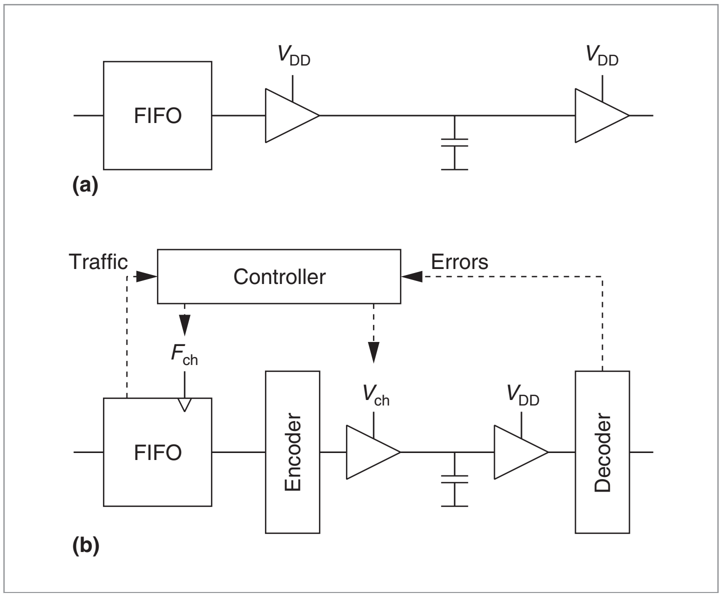


**Figure 2. The basic idea of a self-calibrating, point-to-point, unidirectional on-chip interconnect: the classic static scheme, with a FIFO buffer to decouple the two subsystems (a); the proposed self-calibrating scheme, with the elements needed to achieve the desired goals (b).**

In summary, our system uses variable frequency and voltage swing to trade off speed for energy, implements error detection and ARQ to guarantee reliable communication, and exploits a variable relation between operating frequency and voltage swing to find the best safe operating point under current environmental conditions by monitoring the error rate.

## Challenges of self-calibration

Making the system robust under the expected extreme conditions entails several challenges. The main point is that we are not trying to screen out and remove some relatively infrequent errors, as error detection codes and ARQ protocols do. On the contrary, we try to operate the system as close as possible to the point at which it becomes nonoperational. In a sense, we push our system to explore the operating space, so that at times it actually becomes nonoperational.

Figure 3 shows a more practical view of our system. It represents in greater detail the idea illustrated in Figure 2b, with the addition of some necessary components.

### Channel bit-error modeling

Worm et al. have discussed several system modeling

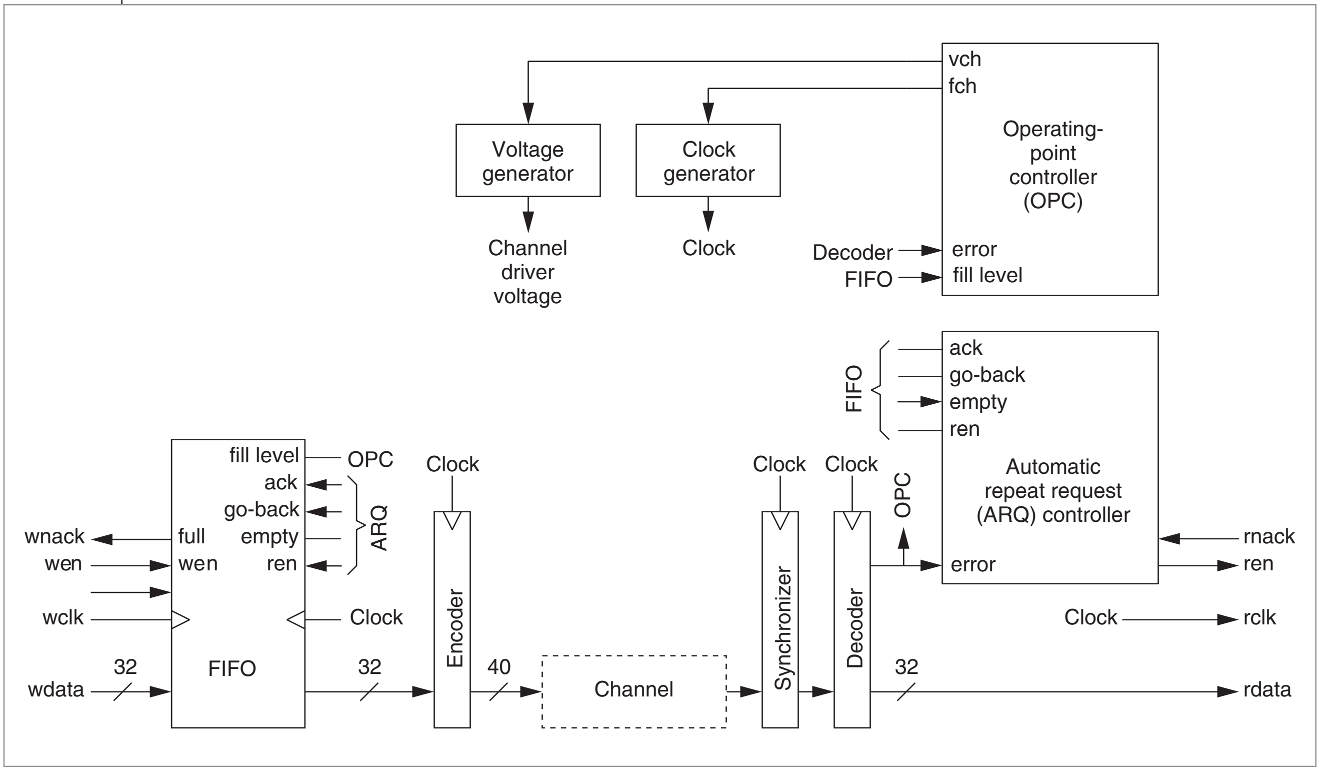


**Figure 3. Possible architecture for the self-calibrating, point-to-point, unidirectional on-chip interconnect.**

issues.[14] The issue most relevant to achieving self-calibration is the availability of a reliable error model as a function of the voltage swing and the transmission frequency.

We consider two possible sources of errors, or noise. The first is an additive white Gaussian noise, modeling external disturbances. The second noise source captures the variability of the channel cutoff frequency around its nominal value, representing the effects of temperature, manufacturing conditions, and so forth, on the propagation delay through the interconnect. We assume these two noise sources are uncorrelated. We further assume that an error occurs if the operating frequency exceeds the channel cutoff frequency or the additive noise exceeds half the voltage swing. Although external disturbances are more accurately modeled as burst noise, a white-noise model suffices to prove our concept. Note that the operating-point control policy doesn't rely on any assumption about the noise model, which serves only to generate random bit errors in our experiments.

With these assumptions, we can derive a relation to express the probability of errors on a single line as a function of the voltage swing and the transmission frequency. At a given voltage, for transmission frequencies below the channel cutoff frequency, the probability of error is not 0 but extremely small. Conversely, for very high transmission frequencies, the same probability is practically 1.

The bit-error probability doesn't express a bit-flipping probability. Because we model the charging and discharging of interconnect bit lines—including timing errors such as those induced by crosstalk—the bit-error probability models approximately the probability that a line is sampled before having time to change to its new state (see Figure 4). That is, we can assume that if the operating frequency is too high, the word read on the interconnect is simply the previous one, because there wasn't enough time for the lines to transition to their final state. This has important consequences for our choice of encoding.

## Delay-insensitive encoding

Simple spatial encoding (such as adding parity bits to the data word) is insufficient. Such encoding would effectively detect, for instance, that because of crosstalk a single bit hasn't yet transitioned. Yet, if our clock is so fast that the entire previous word is still present on the interconnect (for example, when the sampling process is like (2) in Figure 4), a pure spatial encoding would see the result as correct and would not detect that the new word is simply not ready. Instead of more-classical delay-insensitive encodings, such as 1-of-*N* schemes, we use the simpler scheme shown in Figure 5. Our error detection scheme works by generating one additional bit, alternatively a 0 and a 1, that is not transmitted but is produced independently at the source and destination, and by computing and transmitting an 8-bit cycle redundancy check code (CRC-8) using the generator polynomial $x^8 + x^2 + x + 1$ on the data word (for example, 32 bits) padded with the generated bit.[13] This bit ensures that no two successive identical data words can have the same encoding; hence, two successive 40-bit encoded words on the channel can be identical only if an error occurs. We have verified that for independent uniformly distributed input data, the redundant bit lines have the same switching activity as nonencoded data lines, transitioning an average of once every two cycles (half the switching rate of a clock).

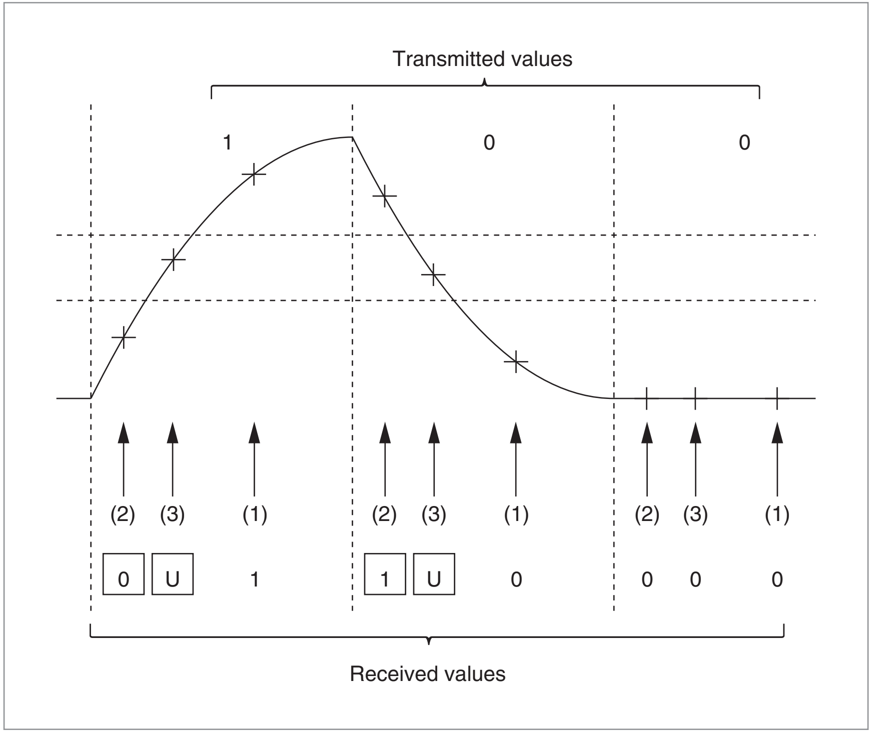


**Figure 4. A qualitative view of the error sources in a self-calibrated interconnect operating under delay/voltage conditions that are too aggressive: correct operation after a sufficient delay (1); bit errors resulting from the sampling after a largely insufficient delay (2); and risk of metastability in the receiver for sampling times that are slightly too aggressive (3). (The figure is simplistic because a new symbol would normally be emitted at the same time the line is sampled.) ´The two horizontal lines represent the receiver thresholds. U indicates an undefined received value because the sampled value is between the two thresholds.**

This scheme combines a flipping bit and a CRC-8. However, analytically assessing the scheme's robustness is beyond the scope of this article.[15] We performed simulations in VHDL with a functional model of the channel that approximates the analytical bit-error-rate model. We transferred 0.32 × $10^9$ random bits and observed no residual undetected error for raw bit-error rates up to $10^{-3}$. Figure 6 shows the residual bit errors as a function of high raw bit-error rates.

Although by no means specific to this encoding, it's worth noting that as the bit-error rate approaches 1, the absolute number of undetected errors increases dramatically. This is of no con-

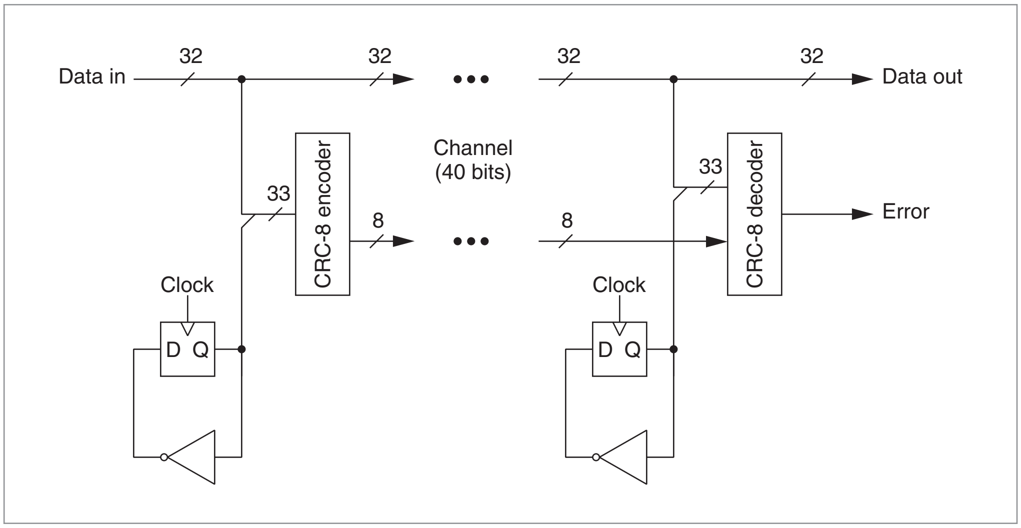


**Figure 5. Possible delay-insensitive encoding scheme. The error signal also detects whether the sampled word is still the last word correctly sent across the channel.**

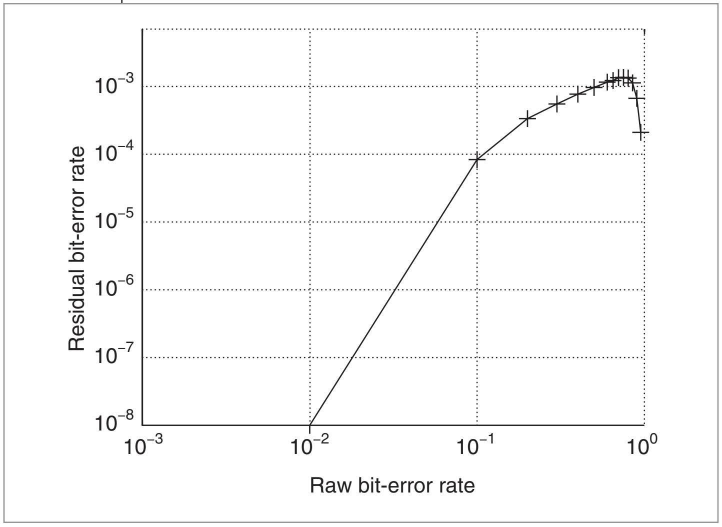


**Figure 6. Residual bit-error rate as a function of the raw bit-error rate.**

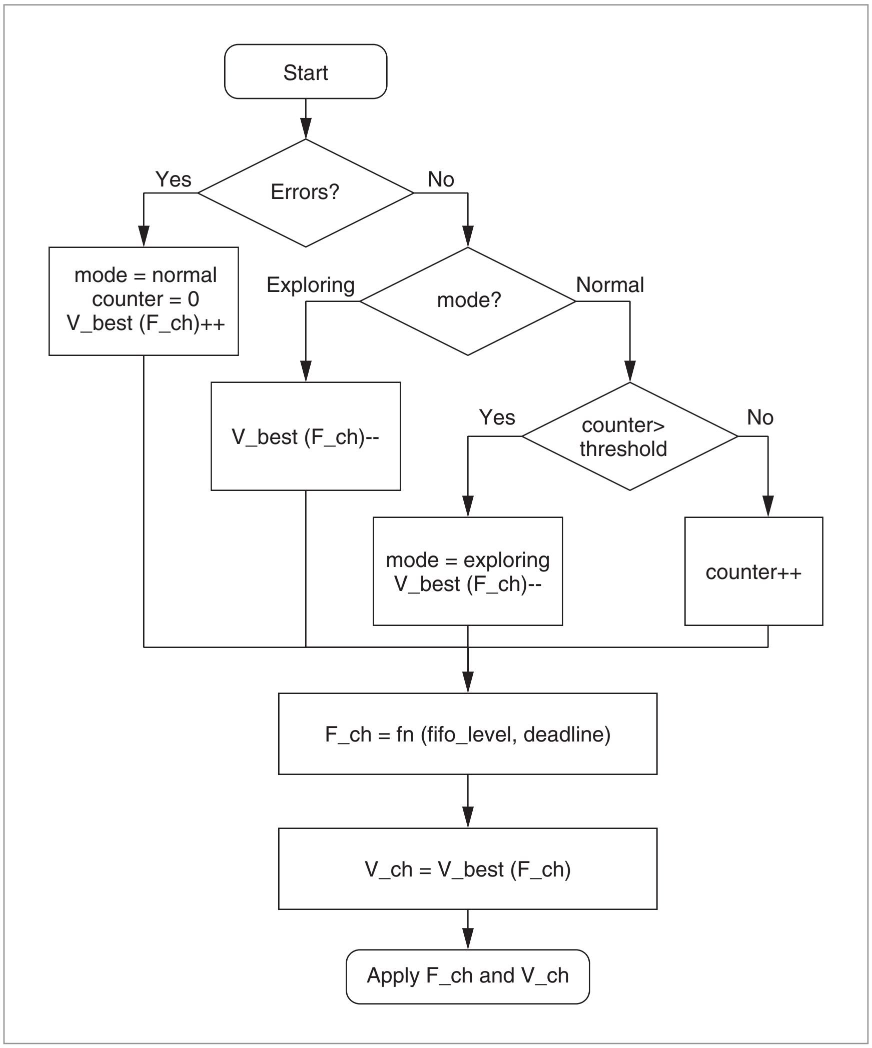


**Figure 7. Simplified operating-point control policy.**

cern in typical applications of error-correcting codes, where we can assume that the error rate is always small, but a self-calibrating system might operate briefly in regions of extremely high bit-error rates. Contrary to the classic CRC-8 and thanks to the flipping bit, our encoding scheme detects errors when the raw bit-error rate approaches 1. Encodings with an even stronger detection probability under our error model are an active subject of research.

## Operating-point control policy

As Figure 3 shows, it's possible to completely separate an ARQ controller from a controller devoted to choosing the operating point. The former's sole task is to push all data words through the channel until they're communicated to the receiver without error, ignoring the channel parameters. In other words, the ARQ controller decides only *which* words to push through the channel. The operating-point controller, on the other hand, selects the lowest frequency and voltage swing required to meet some communication constraint, such as an average delay. It decides how to communicate and determines whether the choice is appropriate, but it ignores *what* is going through the channel.

Figure 7 shows a simplified control algorithm for the operating-point controller, which memorizes the best operating point for each possible frequency. The controller performs three tasks independently:

- It records the location of the best voltage/frequency points (that is, for each possible frequency, it discovers the lowest usable voltage swing). It does this on the basis of experienced errors and periodic attempts to explore more-aggressive operating regions.
- It chooses a frequency on the basis of the delay constraint.
- It chooses the estimated best point's voltage swing at the selected frequency.

We assume the delay constraint is known, which is often the case with multimedia data transfers (see the "Simulation results" section). Figure 8a shows how the controller selects the operating point among a set of possibilities (one point per frequency); the recorded points are an estimation of Pareto operating points. The controller chooses the most appropriate operating point as a function of the observed traffic and the delay constraint. Figure 8b illustrates the effect of the estimation process: Errors immediately push the system to become more conservative (that is, to increase the voltage swing associated with a given frequency). To ensure the most aggressive operation, whenever the system works satisfactorily for a given number of cycles (a threshold value of, say, 500 to 1,000 cycles), it briefly attempts to reduce the voltage at a constant frequency. If errors aren't observed for a few cycles (say, 50), the controller records the new point as the best point at that frequency.

Figure 7's control policy deserves a brief mention here. In particular, we're interested in comparing our policy with that of an optimal controller that already knows the Pareto voltage that should have been used for every frequency. It turns out that the two controllers perform similarly. We observed no difference in terms of transfer delay and residual word errors (none in either case), but the optimal controller saved approximately 1 percentage point more power.

We are also interested in sensitivity to the empirical threshold parameter that dictates how often the controller tries to reduce voltage. Figure 9 reveals that our policy results in correct behavior for a wide range of values.

Although implementing this control algorithm entails considerably more than Figure 7 shows, hardware complexity is still relatively low and requires an area equivalent to a few thousand two-input NAND gates.

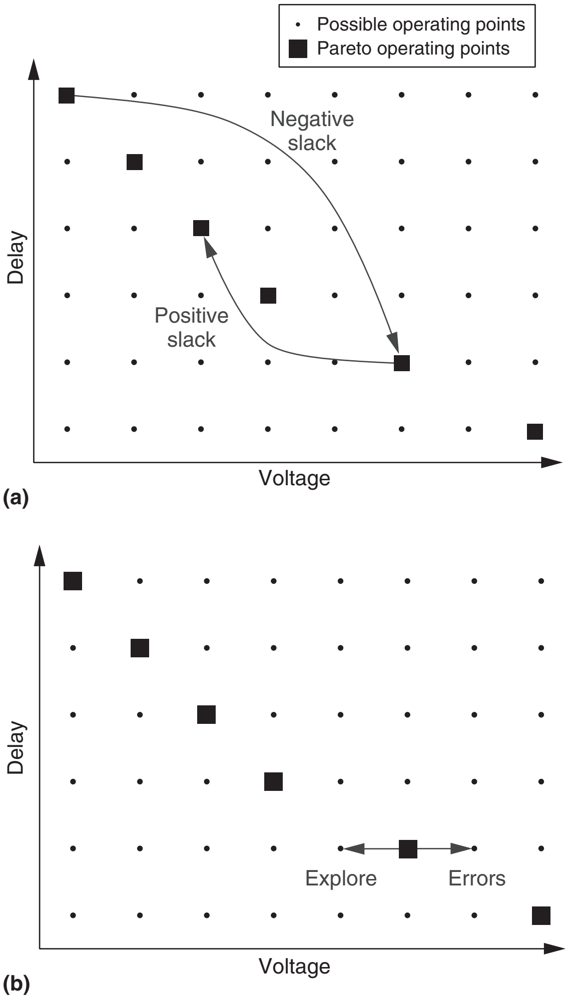


**Figure 8. Use and estimation of best operating points. The control policy fixes the operating frequency as a function of the delay constraint; it sets the operating voltage to the minimum value that has experienced error-free transmission (a). When the system experiences errors, the controller raises the best voltage associated with a given frequency; otherwise, every few cycles it tries to reduce the voltage to ensure the most aggressive operation (b).**

## Simulation results

We synthesized and simulated a self-calibrating 32-bit interconnect system and compared it with a classic fixed-swing system. We modeled typical 0.13-μm CMOS technology and noise sources as follows:

- nominal supply voltage, 1.5 V;
- device threshold voltage, 0.3 V;
- additive noise standard deviation, 0.1 V; and
- average cutoff frequency and standard deviation, 500 MHz and 36 MHz, respectively.

This technology data applies only to bit-error simulation; the controller is completely technology independent. Table 1 shows the systems' operating ranges.

The classic system does not implement an error detection scheme, whereas our system contains the encoder and decoder illustrated in Figure 5. We present our results with delays and frequencies relative to the classic system. To calculate the self-calibrating system's energy advantage, we account for the main sources of inefficiency—namely, the need to communicate 25% more bits for the error-detecting code and the need to occasionally resend some pieces of data because of errors. However, we disregard the small amount of energy spent in the ARQ and the operating-point controllers and in the encoder and decoder. The ARQ controller

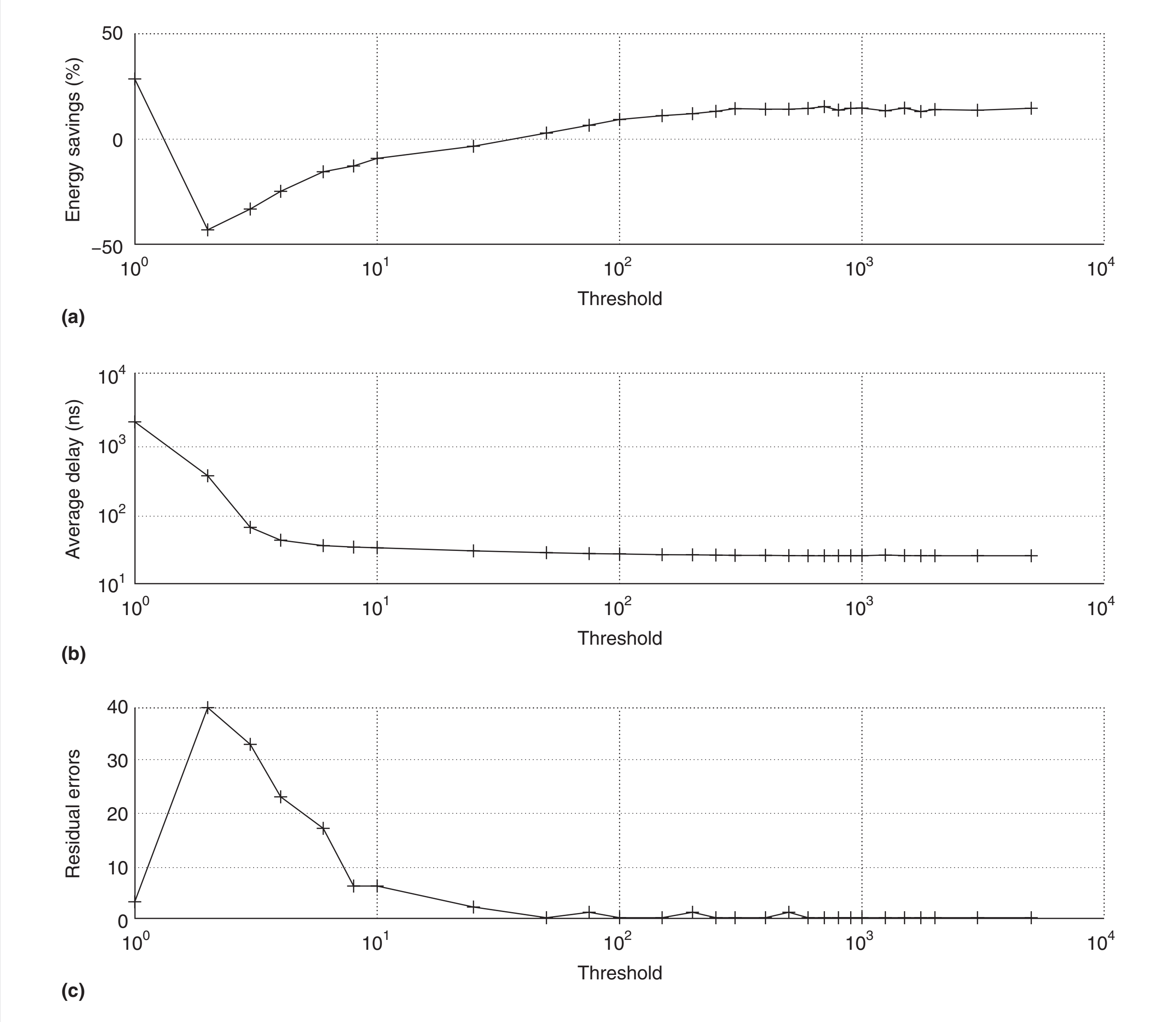


**Figure 9. Sensitivity of various metrics to the threshold appearing in the description of the controller algorithm. Values that are too small cause frequent word retransmissions, negatively affecting energy savings (a), word transfer delay (b), and residual word errors (c).**

**Table 1. Operating ranges for comparing the classic and self-calibrating systems.**

| Operating parameter | Classic | Self-calibrating |
|---|---|---|
| Voltage swing | 1.5 V | 0.5-1.6 V |
| Frequency | 250 MHz | 50-500 MHz |

has roughly 100 gates. A study of the encoding/decoding circuitry shows that the incurred logic overhead doesn't significantly affect the energy balance. We can expect that current high-end systems and future systems in general will already contain an encoder and a decoder.[16] Because we neglect the control logic energy overhead, the ratio of energy consumed by the self-calibrating system to that consumed by the classic system doesn't depend on parameters such as bus length or capacitive load. Therefore, we don't have to specify their actual value in the results.

In Figures 10–12, we present results from three experiments. The first, Figure 10, shows the energy advantage of dynamic bandwidth adaptation on a realistic MPEG-based workload. The second example, Figure 11, shows the energy advantage of dynamically tuning the oper-

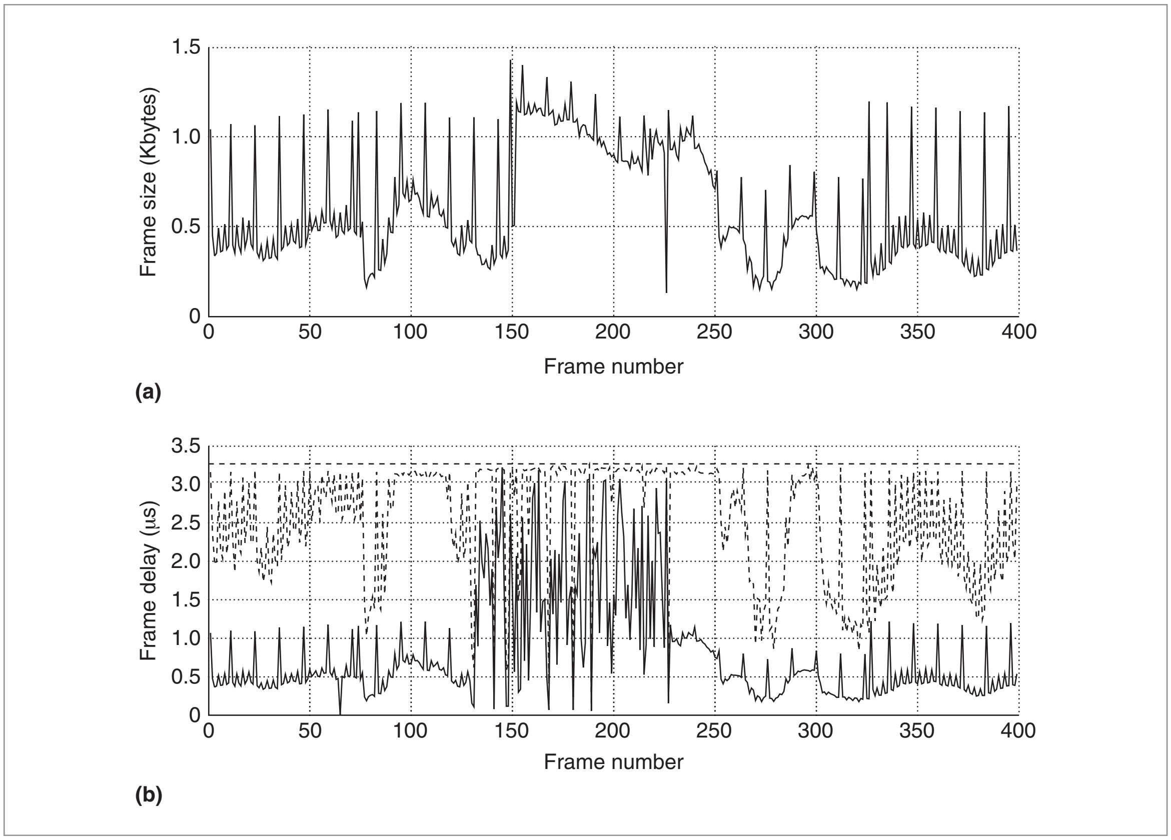


**Figure 10. Transmission of a variable workload: workload variation in time (a); incurred frame delay in the classic system (low delay—solid line at bottom) and in the self-calibrating interconnect (delay as close as possible to the imposed constraint—dashed line at top) (b).**

ating point to actual technology variations. The third, Figure 12, illustrates our system's robustness to unpredictable noise sources.

Modern multimedia algorithms have dynamically varying requirements. Figure 10b shows how the self-calibrating system takes advantage of a time-varying MPEG workload, shown in Figure 10a. The adaptive system tries to exactly match the bandwidth to the current needs. It slows down the communication link to send every MPEG frame exactly in the allotted time and, ideally, not any faster. Operating at a lower frequency grants a substantial reduction in average energy consumption: The whole trace, consisting of 400 frames of several kilobytes each, requires 53% less energy with a dynamically self-calibrating system than with a classic system.

Figure 11 illustrates the effect of technology on the choice of control points. On a wafer whose electrical parameters are poor, simulated with an average cutoff frequency of 430 MHz, the controller chooses mainly Pareto points relatively close to the worst-case delay line. On a good wafer, simulated with an average cutoff frequency of 570 MHz, the points chosen are mostly along a more aggressive delay/voltage line and reflect the lowest delays that the system experiences. (In both cases, the cutoff frequency standard deviation has been decreased to 15 MHz to account for the lower indeterminacy.) On the poor wafer, the self-calibrating system provides an energy savings of 17%, compared with the classic system. The energy savings rises to 38% on the good wafer. The simulated traffic is an artificial workload of 100,000 words, with arrival times following a Poisson process. Average latency through the communication system increases in the self-calibrating system by 14% for the good wafer and 26% for the poor wafer.

Figure 12 illustrates the effect of design hypotheses that turned out to be too optimistic. To simulate the self-calibrating system with more noise, we raised the standard deviation of the additive noise from 0.1 V to 0.15 V and the cutoff frequency standard deviation from 36 MHz to 55 MHz. The classic system will probably not work anymore under these conditions. Overlooking or

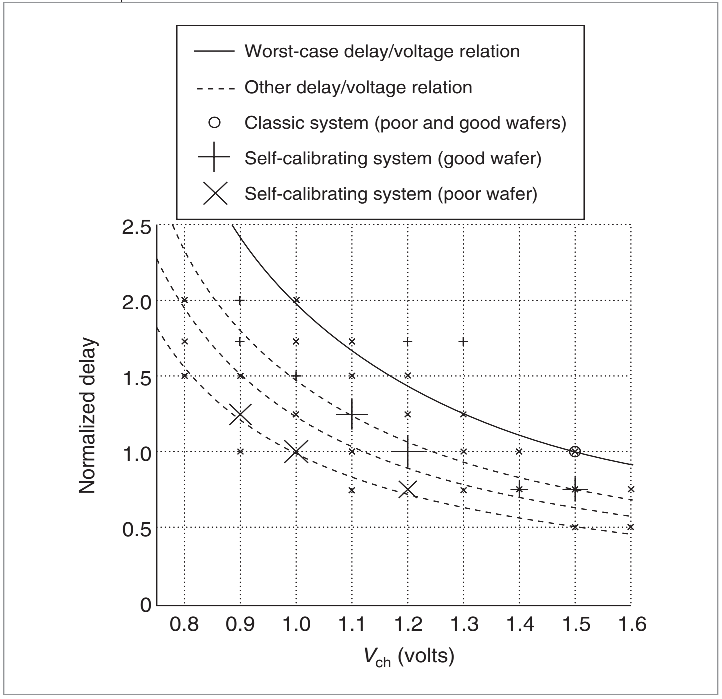


**Figure 11. The operating points used depend on technology variations. The operating points are drawn with a size proportional to usage.**

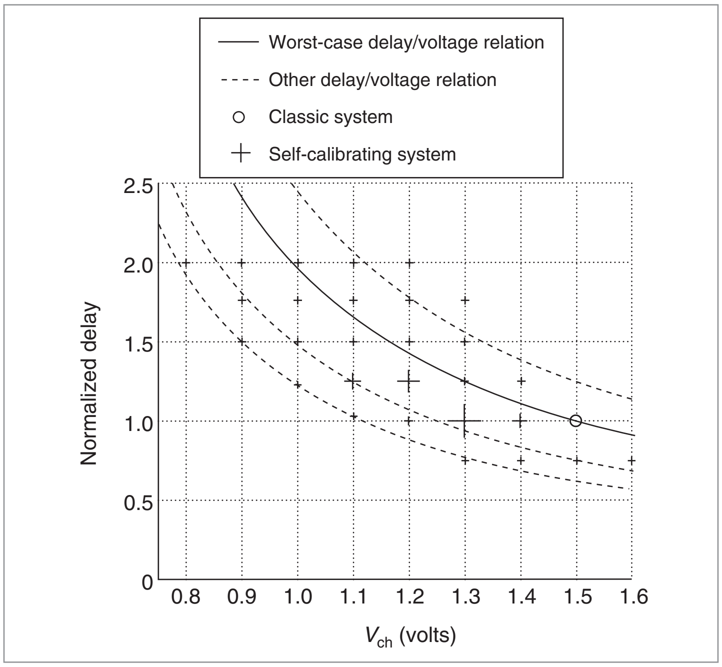


**Figure 12. Operating points used by the self-calibrating system in the presence of strong noise. The classic system has a reduced yield under these conditions, while the self-calibrating system moves to more energy-consuming, but safer, operating points. The operating points are drawn with a size proportional to usage.**

underestimating any error source—such as crosstalk or other deep-submicron second-order effects—in the normal design flow might prevent the manufactured chips from working or result in a very limited yield. As the figure shows, the self-calibrating system adapts to the strong noise by choosing less-aggressive operating points and by trading energy for robustness. Energy savings shrinks to 14% and the average latency grows by 34%, compared with the desired behavior of the classic system. However, the interconnect operates correctly and avoids the yield reductions incurred by the classic system.

**OUR NOVEL DESIGN PARADIGM** for tolerating electrical parameter variations offers much-needed advantages because a wider spread of the electrical parameter will be unavoidable as technologies shrink further. That is, worst-case design assumptions may very well cancel the benefits of technology investments. Therefore, designers will need dynamically self-calibrating techniques to exploit fully the potentials of future nanometric CMOS technologies and overcome manufacturing limitations. ■

## Acknowledgments

The MARCO Consortium and the Gigascale System Research Center partly supported this work.

## ■ References

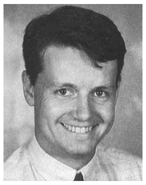
1. M. Orshansky and K. Keutzer, “A General Probabilistic Framework for Worst Case Timing Analysis,” *Proc. 39th Design Automation Conf.* (DAC 39), ACM Press, 2002, pp. 556-561.
2. S. Borkar et al., “Parameter Variations and Impact on Circuits and Microarchitecture,” *Proc. 40th Design Automation Conf.* (DAC 03), ACM Press, 2003, pp. 338-342.
3. S. Tam et al., “Clock Generation and Distribution for the First IA-64 Microprocessor,” *IEEE J. Solid-State Circuits*, vol. 35, no. 11, Nov. 2000, pp. 1545-1552.
4. R. Hegde and N.R. Shanbhag, “Soft Digital Signal Processing,” *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 6, Dec. 2001, pp. 813-823.
5. L. Benini and G. De Micheli, “Networks on Chips: A New SoC Paradigm,” *Computer*, vol. 35, no. 1, Jan. 2002, pp. 70-78.
6. D. Liu and C. Svensson, “Power Consumption Estimation in CMOS VLSI Chips,” *IEEE J. Solid-State Circuits*, vol. 29, no. 6, June 1994, pp. 663-670.
7. P.P. Sotiriadis and A. Chandrakasan, “Low Power Bus Coding Techniques Considering Inter-wire

Capacitances," *Proc. IEEE Custom Integrated Circuits Conf.* (CICC 2000), IEEE Press, 2000, pp. 507-510.

8. D. Bertozzi, L. Benini, and G. De Micheli, "Low Power Error Resilient Encoding for On-Chip Data Buses," *Proc. Design, Automation and Test in Europe* (DATE 02), IEEE CS Press, 2002, pp. 102-109.
9. H. Zhang, V. George, and J.M. Rabaey, "Low-Swing On-Chip Signaling Techniques: Effectiveness and Robustness," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 3, June 2000, pp. 264-272.
10. C. Svensson, "Optimum Voltage Swing on On-Chip and Off-Chip Interconnects," *IEEE J. Solid-State Circuits*, vol. 36, no. 7, July 2001, pp. 1108-1112.
11. T. Pering, T. Burd, and R. Brodersen, "The Simulation and Evaluation of Dynamic Voltage Scaling Algorithms," *Proc. Int'l Symp. Low Power Electronics and Design* (ISLPED 98), ACM Press, 1998, pp. 76-81.
12. L. Shang, L.-S. Peh, and N.K. Jha, "Dynamic Voltage Scaling with Links for Power Optimization of Interconnection Networks," *Proc. 9th Int'l Symp. High-Performance Computer Architecture* (HPCA 03), IEEE CS Press, 2003, pp. 91-102.
13. J. Walrand and P. Varaiya, *High-Performance Communication Networks*, 2nd ed., Morgan Kaufmann, 2000.
14. F. Worm et al., "An Adaptive Low-Power Transmission Scheme for On-Chip Networks," *Proc. 15th Int'l Symp. System Synthesis* (ISSS 02), ACM Press, 2002, pp. 92-100.
15. F. Worm et al., "Soft Self-Synchronizing Codes for Self-Calibrating Communication," to be published in *Proc. Int'l Conf. Computer-Aided Design* (ICCAD 04), IEEE CS Press, 2004.
16. C. McNairy and D. Soltis, "Itanium 2 Processor Microarchitecture," *IEEE Micro*, vol. 23, no. 2, Mar./Apr. 2003, pp. 44-55.

**Frédéric Worm** is a PhD student in the School of Computer and Communication Sciences at the Swiss Federal Institute of Technology Lausanne (EPFL), Switzerland. His research interests include self-calibration techniques for networks on chips. Worm has an MS in communication systems from the Swiss Federal Institute of Technology Lausanne and a DEA (diploma for advanced studies) in networking and distributed systems from the University of Nice-Sophia Antipolis, France.

**Paolo Ienne** is a professor at the School of Computer and Communication Sciences at the Swiss Federal Institute of Technology Lausanne (EPFL), Switzerland, where he heads the Processor Architecture Laboratory. His current research interests cover aspects of advanced SoC design, including automatic processor specialization, programming abstractions for reconfigurable computing, and self-calibrating design methodologies. Ienne has an MS in electrical engineering from Politecnico di Milano and a PhD in computer science from the Swiss Federal Institute of Technology Lausanne. He is a member of the IEEE and the IEEE Computer Society.



**Patrick Thiran** is a professor in the School of Computer and Communication Sciences at the Swiss Federal Institute of Technology Lausanne, Switzerland. His research interests include communication networks and dynamic systems. Thiran has an electrical engineering diploma from the Université Catholique de Louvain, Belgium; an MS from the University of California at Berkeley; and a PhD from the Swiss Federal Institute of Technology Lausanne, both in electrical engineering. He is a member of the IEEE and the ACM.

**Giovanni De Micheli** is a professor of electrical engineering and computer science at Stanford University. His research interests include several aspects of design technologies for ICs and systems. He has an MS and a PhD in electrical engineering and computer science from the University of California at Berkeley. De Micheli is a Fellow of the IEEE and the ACM and a recipient of the IEEE Emanuel R. Piore Award for contributions to synthesis technology.

■ Direct questions and comments about this article to Frédéric Worm, Processor Architecture Laboratory, EPFL, Lausanne, Switzerland; frederic.worm@epfl.ch.