Stochastic Modeling and Performance Evaluation for Digital Clock and Data Recovery Circuits

Alper Demir Peter Feldmann

{alpdemir, peterf}@research.bell-labs.com

Bell Laboratories
Murray Hill, New Jersey, USA

Abstract

Clock and data recovery circuits are essential components in communication systems. They directly influence the bit-error-rate performance of communication links. It is desirable to predict the rate of occasional detection errors and the loss of synchronization due to the non-ideal operation of such circuits. In high-speed data networks, the bit-error-rate specification on the system can be very stringent, i.e., 10^{-14}. It is not feasible to predict such error rates with straightforward, simulation based, approaches. This work introduces a stochastic model and an efficient, analysis-based, non-Monte-Carlo method for performance evaluation of digital data and clock recovery circuits. The analyzed circuit is modeled as finite state machines with inputs described as functions on a Markov chain state-space. System performance measures, such as probability of bit errors and rate of synchronization loss, can be evaluated through the analysis of a larger resulting Markov system. A dedicated multi-grid method is used to solve the very large associated linear systems. The method is illustrated on a real industrial clock-recovery circuit design.

1 Introduction

High-speed communication systems have extremely tight bit-error-rate (BER) specifications. For SONET/SDH applications it is not uncommon to have BER requirements in the order of 10^{-14}. Such specifications are practically impossible to verify through straightforward simulation because of the extremely long sequence that would need to be simulated in order to get meaningful error statistics. In the absence of an analysis tool, designers rely on the experience of previous designs, intuition, and good luck. This environment discourages innovative solutions and non-incremental applications.

On the other hand, the design process of communication systems would benefit significantly from the existence of a reliable design performance evaluation capability. Such a capability would permit the evaluation of a number of alternative algorithms, architectures, circuit techniques, and technologies in a short time and without the commitment of expensive resources.

A situation that illustrates the need for a reliable evaluation capability of the BER occurred in the design of a SONET-type application at a well-known micro-electronics company. The specification for a multiplexer chip required a BER of 10^{-14}. The prototype implementation, based on the modification of an existing design delivered performance that was more than an order of magnitude bellow the specification. The designers suspected that the main cause for the errors is the interference noise in the PLL-based clock recovery circuit, induced by the rest of the chip's circuitry. A number of circuit, technology, and packaging remedies were proposed, but the designers were frustrated by their inability to predict their effectiveness.

This paper introduces a method for performance evaluation of a digital clock-data recovery circuit design. Clock-data recovery (CDR) circuits implement the most critical function that is performed at the receiver of a synchronous data communication system. Their function is to determine not only the frequency at which the incoming signal needs to be sampled, but also the optimal choice of the sampling instant within each symbol interval.

Our analysis method computes the probability of error directly from the design description, without relying on the simulation of long sequences. The system under evaluation is described as a finite-state machine (FSM) with some of its inputs being random. The random variables describe incoming data, noise, and jitter. The random inputs are modeled as functions on the state-space of Markov chains. It is shown that under these circumstances the entire system can be modeled by a larger Markov chain. The quantities of interest for our system, such as the probability of a sampling error, or the mean time between failures due to sampling errors are thus available from standard Markov chain analysis. The remaining challenge is to perform computations with the extremely large transition probability matrices associated with Markov chains that can easily reach millions of states for moderately complex systems. In this work we employ a specialized multi-grid method which takes advantage of the underlying problem structure and is capable of solving million state problems in less than an hour on a beefed-up workstation.

2 Modeling and Performance Evaluation

Throughout the paper, we will be using the CDR circuit [1, 2] shown in Figure 1 to illustrate the stochastic model and the performance evaluation techniques. The framework we present here is by no means restricted to this particular circuit, and the general model we describe can be used for other discrete-time mixed-signal processing circuits.

The CDR circuit in Figure 1 consists of two coupled feed-

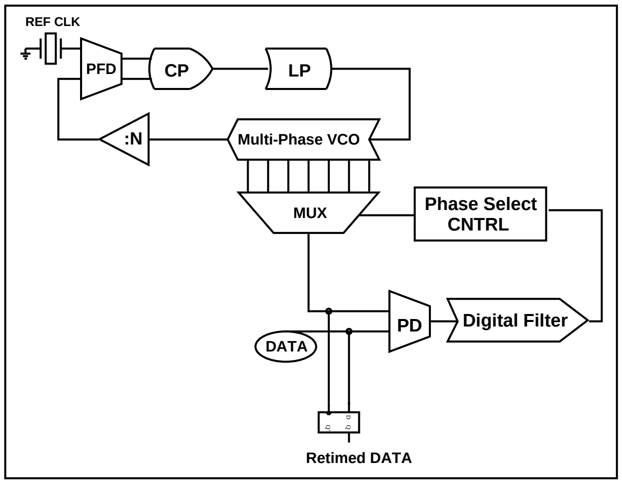


Figure 1: Clock and data recovery circuit

back loops. The first one (upper left) is a traditional "analog" charge-pump phase-locked loop (PLL) with a crystal reference and a voltage-controlled oscillator (VCO) that can generate multi-phase clocks (e.g., a ring-oscillator). The second loop (lower right) is *digital*, and has the purpose of selecting "the best" of the clock phases generated by the first loop in order to retime/align the data. This phase selection is continually updated by the loop. The currently selected phase and the incoming data are "compared" in the phase detector (PD) which produces a *digital* phase error signal. The digital output of the PD is further filtered to produce the actual digital signal that controls the phase selection multiplexer.

For a timing recovery circuit, the BER specification for the retimed data has to be met for a given data characteristics in the presence of jitter. Moreover the phase detector can produce a phase error signal only when a transition occurs in the data signal. The input data stream is usually specified in terms of the longest possible bit sequence with no transitions and a maximal drift in frequency. The input data jitter is specified by eye opening, usually defined as uncorrelated timing jitter from a bit to the next. Sometimes correlated or cumulative jitter, i.e., a random walk, may also be specified. There are also specifications on the recovered clock jitter.

In this paper, we are going to concentrate on the digital phase selection loop. The major jitter source in most CDR applications is the incoming data, but the internally generated clock jitter due to device noise or interference from other circuits can also become significant. Once the internal clock jitter has been characterized using techniques covered elsewhere, it can easily be captured in our models and analysis.

It is virtually *impossible* to simulate the BER for the whole clock recovery system at once using *detailed transistor-level* models for the whole circuit. The size of the problem (in terms of the variables and the number of differential equations that describe it) is simply too large to handle, now or in the foreseeable future. Moreover, we are confronted with a mixed digital-analog circuit with large time constant feedback loops, i.e., stiff. The only use of transient simulation at the circuit level is for connectivity verification, and simple functional verification of the laid-out circuit. Obviously, such a verification is far from ensuring that the design meets the BER specification. To tackle the system level problem, we have to develop intelligent models that simplify the problem, but at the same time, capture the characteristics of the circuit that is essential for its operation.

The components in the digital phase selection loop, such as the phase detector and the digital filter, are highly nonlinear circuits with switching behavior if viewed from a differential equation (DE) perspective. A DE model noise analysis based on linearization (time-invariant or time-varying) is neither useful for, nor applicable to, this problem, because the noise or data jitter is too large for the linearization to remain valid. Moreover, the internal device noise sources (e.g., thermal noise) have little significance. The loop components are "almost" ideally functioning digital circuits and hence can be modeled as *discrete-time* digital systems. On the other hand, jitter and the phase error between the selected clock phase and data are continuous variables.

The simplest model that captures the essential behavior of the digital phase selection loop in Figure 1 can be expressed with the following difference equation

$$\Phi[k+1] = \Phi[k] - G\,\mathrm{sgn}(\Phi[k] + n_w[k]) + n_r[k] \qquad (1)$$

where Φ is the phase error between the incoming data and the recovered clock. The phase detector is simply modeled as a *memoryless* nonlinear function which produces the *signum* of its input at the output. n_w and n_r are random processes that model the jitter of the incoming data. n_w is a zero-mean *white*, i.e., uncorrelated in time, noise process that is usually Gaussian. n_w models the eye opening of the data and its characteristics can be readily deduced from the system specifications. n_r is usually a *nonzero* mean white noise process. From (1) one can see that if n_r has nonzero mean than the phase error will have a deterministic drift in the absence of the *signum* term which is responsible for phase corrections. One can also observe that the random part of n_r has a cumulative effect on the phase error: If the *signum* term and n_w was not present in (1) than the phase error would be a *random walk* with drift for a nonzero mean white n_r. Almost all jitter specifications on the incoming data can be represented together by n_w and n_r by assigning appropriate amplitude distributions (e.g., Gaussian with certain mean and variance). For instance, one can even "mimic" deterministic sinusoidally varying jitter by assigning the amplitude distribution of n_r appropriately.

The hardware implementation of the phase detector has to operate at the full data speed, hence it needs to be implemented by a relatively simple state machine. The same is true for any digital filtering that might be done at the output of the phase detector. Let us assume that $S[k]$ is a vector representing the state of the finite state machine (FSM) that implements the phase detector and the filter. We will now rewrite/revise (1) in the following more general form which will capture a real implementation

$$\begin{aligned} \Phi[k+1] &= \Phi[k] - f(\Phi[k] + n_w[k], S[k]) + n_r[k] \qquad (2) \\ S[k+1] &= g(\Phi[k] + n_w[k], S[k]) \qquad (3) \end{aligned}$$

Above, the functions f and g specify the phase-detector-filter FSM: g gives the next state of the state machine given its present state and present noisy phase error value. Similarly, f produces a value indicating the phase correction. In the implementation of Figure 1, f takes three possible values 0, G, $-G$ indicating no correction, phase delay or advance respectively. G is the smallest phase increment available from the internal clock.

The combined vector $X[k] = (\Phi[k], S[k])$ represents the state variables of the system described by the *nonlinear* difference equations in (2). Since there are noise sources as inputs to the system, $X[k]$ is best characterized as a stochastic process. We would like to analyze this stochastic process in order to evaluate the various system performance measures.

When the noise sources n_w and n_r are white, i.e., uncorrelated in time, $X[k]$ is a Markov process, that is, given its current state, its future is independent of its past. One way to analyze the system in (2) is using the machinery of discrete-time Markov chains, which requires that we discretize the phase error and also the noise sources to obtain a discrete state-space. The granularity of the discretization of the phase error and the noise sources is dictated by the number of clock phases and the magnitude of the noise source n_r. The discretization grid needs to be fine enough to accurately capture the small jumps in phase error due to n_r.

A Markov chain MC is completely characterized by its transition probability matrix (TPM) $\mathbf{P} = [p_{ij}]$

$$p_{ij} = \mathtt{Pr}\left(X[k+1] = x_j \,\middle|\, X[k] = x_i \right) \tag{4}$$

where the state set $\{x_1, \cdots, x_L\}$ is the reachable state space of the MC, which is a subset of the Cartesian product of the discretized phase values $\{\phi_1, \cdots, \phi_M\}$ and the state set $\{s_1, \cdots, s_N\}$ of the phase detector/filter FSM. The entries of $\mathbf{P}$ are completely specified by the difference equations in (2) and the probabilistic characterization of the discretized noise sources. $\mathbf{P}$ is a very large but highly structured matrix. The structure is induced by the phase detector/filter FSM and the difference equations. It can be constructed using hierarchical Kronecker algebra-like techniques as a composition of smaller components representing building blocks of the system. This representation makes it possible to manipulate and store $\mathbf{P}$ even when the total state space is very large. Figure 2 shows a more detailed compositional model of the clock recovery system of Figure 1 described graphically in the above formalism. This representation can be generalized to networks of FSMs with stochastic inputs to describe various high-speed communication circuits.

Now that we described our system in the MC formalism, we can compute various quantities that characterize the state of the system as a stochastic process. For the clock recovery system, whenever the phase error plus the data jitter, i.e., $\Phi[k] + n_w[k]$ in (2), becomes larger/smaller than half a clock cycle, the system might potentially produce bit errors. It would be highly desirable to compute the probability of this event happening. This probability can be directly obtained form the steady-state probability distribution of reachable states, which is the most basic analysis for MCs. This involves computing the eigenvector corresponding to the eigenvalue 1 of the stochastic matrix $\mathbf{P}$ [3]. Another measure of performance for CDR circuits is the average time between cycle slips. This translates into the computation of mean transition times between certain sets of MC states, which is another standard computation in MC analysis. It involves solving a linear system with the (modified) TPM.

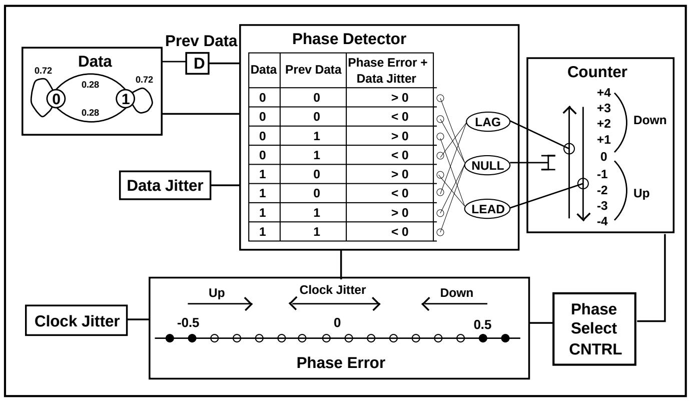


Figure 2: Model of clock and data recovery circuit

3 Numerical Methods

The TPM, $\mathbf{P}$, of a MC is commonly called a stochastic matrix [3]: From its definition, we immediately deduce that it has non-negative entries (they are all probabilities), and its row sums are equal to 1 (a row expresses all the possibilities in a given state), i.e.,

$$\mathbf{P}\,[1,1,\ldots,1]^T = [1,1,\ldots,1]^T$$

It follows that $\mathbf{P}$ has $\xi = [1,1,\ldots,1]^T$ as a right-eigenvector corresponding to the eigenvalue 1. Let us denote the stationary state probabilities with $\eta_{[i]} = \mathtt{Pr}\,(z[n] = \sigma_i)$ as the entries of the $1 \times L$ row vector $\eta = [\eta_{[i]}]$. η satisfies [3]

$$\eta = \eta\,\mathbf{P} \tag{5}$$

Being a probability distribution, the vector η has nonnegative entries and the sum of its entries is equal to 1, and it is a left-eigenvector of $\mathbf{P}$ corresponding to the same eigenvalue 1. The computation of η is the most basic analysis for MCs. The information in η already makes it possible to compute some performance measures for the modeled system as discussed in Section 2. Moreover, computation of η is the prerequisite for computing other performance quantities such as the autocorrelation of a function defined on the states of the MC. Hence, we concentrate on methods for computing η, which can be posed either as an eigenvalue problem through (5), or as the solution of the following *homogeneous* linear system

$$(\mathbf{P}^T - \mathbf{I})\,\eta^T = 0 \tag{6}$$

with the normalization

$$\eta\,\xi = 1 \tag{7}$$

A variety of standard iterative techniques can be used to solve these problems. These techniques, however, do not exploit the properties of MCs.

A family of iterative techniques, specific to MC problems, are *aggregation/disaggregation*-type methods [4]. These techniques arise from and are related to the *lumpability* and *decomposability* concepts in MCs. Here we discuss only lumpability.

Assume that we are given an N-state MC. We partition these N states into n disjoint sets with $n < N$, and form a new stochastic process by defining new states corresponding to the n sets. The value of the new stochastic process at time k is the new state that corresponds to the set that contains the state of the original chain at time k. This procedure could be used to reduce a MC with a very large number of states to a process with a smaller number of states, called the *lumped* process. It is often the case that we are only interested in these *coarser* states. For example, in the model of the clock recovery circuit, we are interested in the phase error which is only a component of the state vector. There are multiple states which correspond to the same phase error value. With the above procedure, we can define a process which is exactly equal to the phase error. However, the crucial question is, whether the newly defined process is *Markov* for *any* initial probability distribution for the states of the original MC. If so, we can treat the new process with MC methods and hence reduce the size of the problem. Unfortunately, the answer to this question in most cases is no, otherwise the model we originally developed was redundant and could have been simplified. If we loosen the condition for the newly defined process to be Markov from *any* to *some* initial probability distribution, and if such an initial distribution exists, the MC is called *weakly lumpable*. In this case, the computation of the TPM for the reduced MC requires both the TPM of the original MC and the initial probability distribution [3]. This is basically the starting point for aggregation-disaggregation techniques for MCs that are used to accelerate the convergence of basic iterative methods such as Jacobi and Gauss-Seidel and possibly the Krylov subspace methods. For instance, let Jacobi be the iterative method. After performing a number of steps of Jacobi, the current iterate for the stationary vector and the TPM for the MC is used to compute a reduced stationary vector and a reduced TPM for the weakly lumped chain. Then, the reduced iterate vector is used as the initial guess to solve the weakly lumped chain exactly. Next, the solution of the lumped problem together with the initial guess is used to produce a correction to the *finer* level iterate. These steps are repeated till convergence [4]. This technique was generalized to more than two *lumping levels* by Horton and Leutenegger [5]. The multilevel method utilizes a set of recursively lumped versions of the original MC to achieve accelerated convergence. It can be interpreted as an algebraic multi-grid method.

The multi-level algorithm can achieve much better performance if the special structure in the MC or the underlying model composed of finite-state machines is exploited to develop a *coarsening* or *lumping* strategy. For the model of the clock recovery circuit in Figure 2, we employed a coarsening strategy which lumps the two states corresponding to consecutive discretized phase error values. In this way, the lumped problems resemble the original problem but with coarser phase error discretization. However, the coarsened problems do not capture all the behavior of the original model. For instance, for some of the problems, the phase error grid is too coarse for the effect of the small noise n_r in (2) to be represented accurately. Nevertheless, the coarse problems retain enough characteristics of the fine problem so as to help accelerate the convergence. In our current implementation, the lumping and expanding steps are interleaved with simple Gauss-Jacobi iterations and the coarsest problem is solved exactly with a direct method. For now, we use explicit sparse storage for both the fine and the coarse problems, which allows solving models of practical clock recovery circuits with $\approx 10^6$ states. For solving more complex models, we are looking into using hierarchical generalized Kronecker-algebra [6, 7] and/or probability decision diagram/tree/graph [8] representations. Figure 3 shows the nonzero pattern for the transition probability matrix of the clock recovery circuit model, where one can observe the compositional structure of the problem.

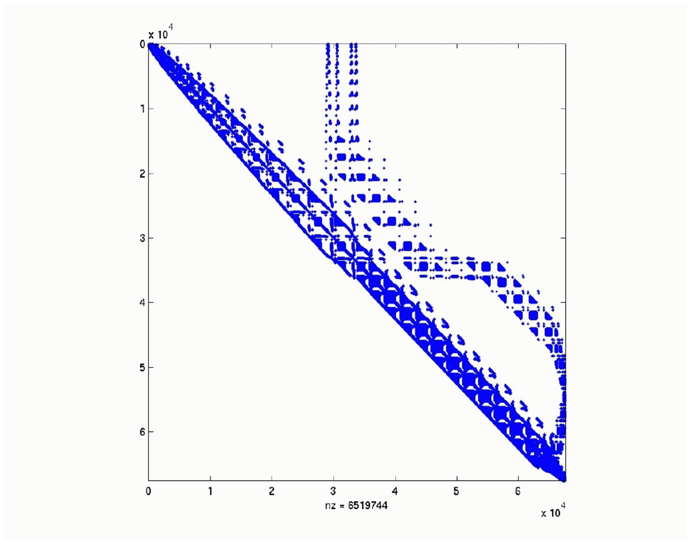
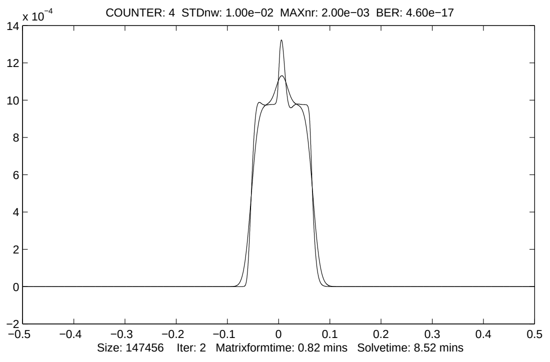


Figure 3: Nonzero pattern for the transition probability matrix

4 Examples

We built a compositional model of the clock recovery circuit in Figure 1. It consists of four interacting FSMs with stochastic inputs. The first FSM models the data statistics taken from SONET system specifications. The second one is the model of the phase detector and has present data, previous data and the noise source n_w (model of the eye opening) as its inputs. It produces a three-valued output: LAG, LEAD and NULL. Its output is the input to an up-down counter FSM that models the loop filter. The counter produces an UP-DOWN signal when it overflows, which is one of the two inputs to the FSM that has the phase error as its state. The other input to the phase error FSM is the noise source n_r.

All figures show the stationary probability density functions of the phase error Φ and the input to the phase detector, i.e, $\Phi + n_w$. The line above the density plots shows the counter length, the standard deviation of the stationary zero-mean white Gaussian noise n_w, the maximum value of the stationary white noise n_r (with a non-zero mean, non-Gaussian distribution with probability density function chosen to reflect SONET system specifications), and the BER computed by integrating the tails of the distribution computed using MC analysis. The line below the density plots shows the size of the state space for the MC generated from the model, the number of multi-grid cycles required for convergence, the CPU time for generating the



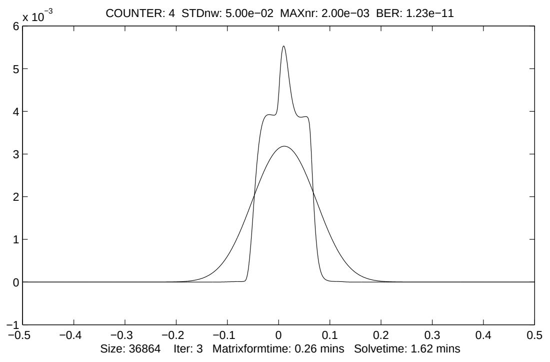


Figure 4: Phase error probability density, and BER

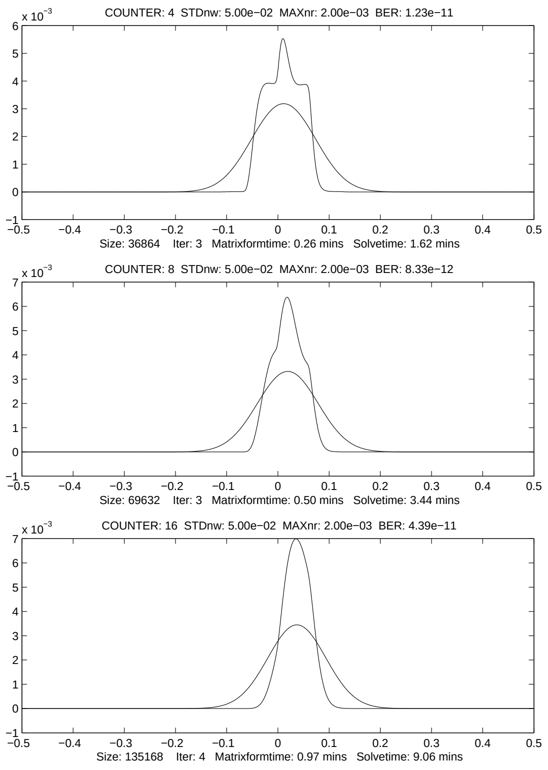


Figure 5: Effect of counter length on BER performance

TPM for the MC and the CPU time spent for the stationary distribution computation. In Figure 4, in the top plot, the noise levels are so small that the CDR system has negligible BER. When the standard deviation of the noise source n_w that models the eye data opening is increased 5 times, the BER increases to 1.23×10^{-11}, as seen in the bottom plot in Figure 4.

In Figure 5, we study the effect of the counter overflow length on the BER performance, all noise levels being held constant. We set it to 4, 8 and 16. We observe that the best BER performance is obtained when counter length is set to 8, BER performance is 1.5 times worse with counter length 4, and 5 times worse with counter length 16.

When the length is set to 4 the loop has high bandwidth. The system tends to follow the dominant noise source, n_w and as a consequence detection errors occur. When the length is set to 16, the effect of the noise source n_r becomes predominant: the loop response becomes too slow to follow the drift caused by n_r and, again, bit errors occur. The length 8 is a good compromise, where both noise sources contribute to the BER. Hence, there is an optimal counter length for given levels of noise, the computation of which is enabled by the accurate and efficient analysis method described in the paper.

5 Conclusions

This paper introduced a new, non-Monte-Carlo analysis method, for the stochastic analysis of data-clock recovery circuits essential components of synchronous data communication systems. The analysis is based on the modeling of the underlying system as a combination of finite state machines and Markov chains. The relevant system performance measures are derived from computations that involve the transition probability matrix of a large resulting Markov chain. Through the use of a specialized multi-grid method, very large systems can be solved in reasonable time on a powerful workstation. The usefulness of the analysis was illustrated through a real industrial design.

References

[1] J. Sonntag and R. Leonowich. A monolithic CMOS 10 MHz DPLL for burst-mode data retiming. In *IEEE International Solid-State Circuits Conference*, 1990.

[2] P. Larsson. A 2-1600 MHz 1.2-2.5V CMOS clock-recovery PLL with feedback phase-selection and averaging phase-interpolation for jitter reduction. In *IEEE International Solid-State Circuits Conference*, 1999.

[3] J.G. Kemeny and J.L. Snell. *Finite Markov Chains.* Springer-Verlag, 1976.

[4] W.J. Stewart. *Introduction to the Numerical Solution of Markov Chains.* Princeton University Press, 1994.

[5] G. Horton and S. Leutenegger. A multi-level solution algorithm for steady-state Markov chains. *ACM Performance Evaluation Review*, 22:191–200, 1996.

[6] B. Plateau. On the stochastic structure of parallelism and synchronization models for distributed algorithms. *Performance Evaluation Review*, August 1985.

[7] P. Buchholz. An adaptive aggregation/disaggregation algorithm for hierarchical Markovian models. *European Journal of Operational Research*, 116(3):85–104, 1999.

[8] M. Bozga, O. Maler. On the Representation of Probabilities over Structured Domains. In *Proc. CAV'99.* , Springer, 1999.