Manfred Dietrich
Joachim Haase *Editors*

Process Variations and Probabilistic Integrated Circuit Design

Springer

Process Variations and Probabilistic Integrated Circuit Design

Manfred Dietrich • Joachim Haase
Editors

Process Variations and Probabilistic Integrated Circuit Design

Springer

Editors
Manfred Dietrich
Design Automation Division EAS
Fraunhofer-Institut Integrierte Schaltungen
Zeunerstr. 38, 01069 Dresden
Germany
manfred.dietrich@eas.iis.fraunhofer.de

Joachim Haase
Design Automation Division EAS
Fraunhofer-Institut Integrierte Schaltungen
Zeunerstr. 38, 01069 Dresden
Germany
joachim.haase@eas.iis.fraunhofer.de

ISBN 978-1-4419-6620-9 e-ISBN 978-1-4419-6621-6
DOI 10.1007/978-1-4419-6621-6
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2011940313

Preface

Continued advances in semiconductor technology play a fundamental role in fueling every aspect of innovation in those industries in which electronics is used. In particular, one cannot fail to appreciate the benefits these advances offer in either reducing the dimensions into which an electronic system can be built or increasing the sheer complexity and overall functionality of the individual circuits. In general, industry tends more to take advantage of the opportunity of offering additional features and capability within a given space that reducing the overall size.

Whereas the manufacturing industry has matched the advances in the semiconductor industry so that failure rates during fabrication at each stage have been maintained at the same rate per element, the number of elements has increased astronomically. As a result, unless measures are not taken, the overall failure rates during production will increase dramatically. There are certain factors that will compound this trend, for example the fact that semiconductor technology yields may be a function of factors other than simple manufacturing ability and may become unacceptable as functional density increases.

It is thus essential to investigate which parameters of the various manufacturing processes are the most sensitive in the production failure equation, and to explore how their influence can be reduced.

If one focuses on the integrated circuit itself, one might consider either addressing the parameters associated with the silicon processing, the disciplines involved in the design activity flow, or better still, both! In fact they are combined in a new design approach referred to as *statistical analysis*. This is heralded by many as the next-generation

EDA technology and is currently oriented specifically at addressing timing analysis and power sign-off. Research into this field commenced about five years ago and saw significant activity during the period since that start, although there are indications of reduced interest of late. This decline in activity may be partly due to the fact that the results of the work have been slow to find application. Perhaps the key direction identified during this period has been the need to develop and optimize statistical models for integrated circuit library components, and it is in this

area that effort will probably concentrate in the near future. This book will present some results from research into this area and demonstrate how the manufacturing parameter variations impact the design flow.

On the one hand, it is the objective of this book to provide designers with a qualitative understanding of how process variations influence circuit behavior and to indicate the most dominant parameters. On the other hand, from a practical point of view, it must also acknowledge that designers need appropriate tools and strategies to evaluate process variants and extract the modeling parameters.

It is true that certain modeling methods have been employed over the years and constitute the framework under which submicron integrated circuits have been developed to date. These have concentrated on evaluating a myriad of electrical model parameters and their variation. This has led to an accurate determination of the inter-dependence of these parameters under given conditions and does provide the circuit developer with certain design information. For example, the designer can determine whether the leakage current of a given cell or circuit is greater than a key threshold specification, and similar determinations of power and delay can be made. In fact, this modeling approach can include many parameters of low order effect yet can be defined in such a way that many may be easily monitored and optimized in the fabrication technology.

However, this *specific case* and *corner analysis* cannot assess such key factors as yield and is too pessimistic and still too inaccurate to describe all variation effects, particularly those than involve parameters with non-linear models and non-Gaussian distributions. It is only from an appreciation of these current problems that one can understand that the benefits of advanced technologies can only be realized using an alternative approach such an advanced statistical design. It is an initial insight into these new methods that the editors wish to present in these pages. It is not the objective to look at the ultimate potential that will be achieved using these methods, rather to present information on the research already complete. The start-point is the presentation of key mathematical and physical fundamentals, an essential basis for an appreciation of the subsequent chapters. It is also important that the reader understand the main causes of parameter variations during production and to appreciate that appropriate statistical methods must be accommodated in the design flow.

This discussion leads into an overview of the current statistical methods and methodologies which are presented from the designer's perspective. Thus the text leans towards the forms of analysis and their use rather than a derivation of the underlying algorithms. This discussion is supported by some examples in which the methods are used to improve circuit designs

Above all, through presenting the subject of process variation in the present form, the editors wish to stimulate further discussion and recapture the earlier interest and momentum in academic research. Without such activity, the strides made to date towards developing methods to estimate such factors as yield and quality at the design stage will be lost, and realizing the potential advantages of future technology nodes may escape our grasp. To engender this interest in such a broad field, the core of the book will limit its scope to:

- exploring the impact of production variations from various points of view, including manufacturing, EDA methods and circuit design techniques
- explaining the impact through simple reproducible examples.

Within this framework, the editors aim to present material that emphasizes the problems that arise because of intrinsic parameter variations, illustrates the differences between the various methods used to address the variations, and indicates the direction in which one must set course to find general solutions.

The core material for the book came from many sources – from consultation with many experts in the semiconductor and EDA industries, from research centers, and from university staff. It is only from such a wide canvas that the book could genuinely represent the broad spectrum of views that surround this subject. The heart of the book is thus that of these contributors, experts in the field who have embodied their frustrations and practical experience in each page.

Certain chapters of the book use results obtained during two German research projects which received funding from the German Federal Ministry of Education and Research (BMBF). These projects are entitled "Sigma 65: Technologiebasierte Modellierung und Analyseverfahren unter Bercksichtigung von Streuungen im 65nm-Knoten" (Technology based modeling and analyzing methods considering variations within 65nm technology) and "ENERGIE: Technologien fr energieeffiziente Computing-Plattformen" (Technologies for energy-efficient computing platforms; the subproject is part of the the Leading-Edge Cluster CoolSilicon) [1]. Both projects address technology nodes beyond 65nm.

All contributors would like to thank the Springer Publishing Company for giving them the opportunity to write this book and have it published. Special thanks go to our Editor, Charles Glaser, for his understanding, encouragement, and support during the conception and composition of this book. We also thank very much Elizabeth Dougherty and Pasupathy Rathika for their assistance, efforts and patience during the preparation of the print version.

Last but not least, we cannot close without thanking also the management and our colleagues at the Fraunhofer-Gesellschaft (Design Automation Division of the Institute for Integrated Circuits) without whose support this book would not have been possible. Being able to work within the infrastructure of that organization and the having available a willing staff to prepare illustrations, tables, and overall structure have been invaluable.

Manfred Dietrich
Joachim Haase

[1]These activities were supported by the German Federal Ministry of Education and Research (BMBF). The corresponding content is the sole responsibility of the authors. Funding initials are 01 M 3080 (Sigma65) and 13 N 10183 (ENERGIE).

Contents

Acronyms

ACM	Advanced Compact Model
ADC	Analog-to-Digital Converter
ANOVA	Analysis of variance
ASIC	Application-specific integrated circuit
BSIM	Berkley short-channel IGFET model
CCS	Composite current source model
CD	Critical Dimension
CDF	Cumulative distribution function
CGF	Cumulant generating function
CLM	Channel length modulation
CMC	Compact Model Council
CMCal	Central moment calculation method
CMP	Chemical-mechanical polishing
CMOS	Complementary metal-oxide-semiconductor
CPF	Common power format
CPK	Process capability index
CSM	Current source model
DAC	Digital-to-Analog Converter
DCP	Digital Controlled Potentiometer
DF	Distribution function
DFM	Design for manufacturability
DFY	Design for yield
DIBL	Drain-induced barrier lowering
DNL	Differential Nonlinearity
DoE	Design of Experiments
ECSM	Effective current source model
EKV	Enz–Krummenacher–Vittoz model
FET	Field-effect transistor
FinFET	“Fin” field-effect transistor
GBD	Generalized beta distribution

GEM	Generic Engineering Model
GDS II	Graphic Data System format II
GLD	Generalized lambda distribution
GIDL	Gate-induced drain leakage current
GPD	General Pareto distribution
HiSIM	Hiroshima university STARC IGFET Model
HOS	Higher-order sensitivity
HCI	Hot carrier injection
IC	Integrated Circuit
ICA	Independent component analysis
IGFET	Insulated-gate field-effect transistor
ITRS	International Technology Roadmap for Semiconductors
INL	Integral Nonlinearity
IP core	Intellectual Property Core
JFET	Junction gate field-effect transistor
LHS	Latin hypercube sampling
LOCOS	Local oxidation of silicon
LSM	Least square method
LSB	Least Significant Bit
MOSFET	Metal-oxide-semiconductor field-effect transistor
MPU	Microprocessor unit
MC	Monte Carlo
ML	Maximum likelihood
MSB	Most Significant Bit
NBTI	Negative bias temperature instability
NLDM	Nonlinear Delay Model
NLPM	Nonlinear Power Model
NQS	Non-Quasi Static
OASIS	Open Artwork System Interchange Standard
OPC	Optical Proximity Correction
OCV	On-chip variation
PCA	Principal Component Analysis
PDF	Probability density function
PDK	Process Design Kit
POT	Peak over threshold
PSP	Penn-State Philips CMOS transistor model
PVT	Process-Voltage-Temperature
RDF	Random doping fluctuations
RSM	Response Surface Method
SAE	Society of Automotive Engineers
SAR	Successive Approximation Register (ADC type)
SAIF	Switching Activity Interchange Format
SDF	Standard Delay Format
SOI	Silicon on insulator
SPA	Saddle point approximation

SPE	Surface Potential Equation
SPEF	Standard Parasitic Exchange Format
SPDM	Scalable Polynomial Delay Model
SPICE	Simulation program with integrated circuit emphasis
STA	Static timing analysis
STARC	Semiconductor Technology Academic Research Center
SSTA	Statistical static timing analysis
STI	Shallow Trench Isolation
SVD	Singular value decomposition
UPF	Unified power format
VCD	Value change dump output format
VLSI	Very-large-scale integration
VHDL	Very High Speed Integrated Circuit Hardware Description Language
VHDL-AMS	Very High Speed Integrated Circuit Hardware Description Language – Analog Mixed-Signal

List of Mathematical Symbols

$\mathbb{R}^n$	Euclidean space of dimension n
$\underline{a}$	Vector
$\boldsymbol{A}$	Matrix
k_B	Boltzmann constant
q	Elementary charge
V_th	Threshold voltage
V_fb	Flat-band voltage
V_T	Thermal voltage $k_\mathrm{B}T/q$
I_ds	Drain-source current
I_gs	Gate-source current
V_dd	Supply voltage
V_gs	Gate-source voltage
V_ds	Drain-source voltage
V_sb	Source-bulk voltage
I_leak	Leakage current
I_GIDL	Gate-induced drain leakage current
I_sub	Subthreshold leakage current
I_gate	Gate oxide leakage current
$I_{\mathrm{ds},V_\mathrm{th}}$	Drain-source current at $V_\mathrm{gs} = V_\mathrm{th}$ and $V_\mathrm{ds} = V_\mathrm{dd}$
g_ds	Drain-source conductance
I_dsat	Drain saturation current
T_ox	Oxide thickness
C_ox	Oxide capacitance
μ_eff	Effective mobility
L_eff	Effective channel length
ε_F	Fermi level
k_B	Boltzmann constant
$\boldsymbol{J}$	Jacobian matrix
X	Random variable X (one dimensional)
x	Sample point of a one-dimensional random variable x

$\underline{X}$	Random vector $\underline{X}$
$\underline{x}$	Sample point of a random vector $\underline{x}$
$\boldsymbol{\Sigma}$	Covariance matrix $\boldsymbol{\Sigma}$
$N(\mu, \sigma^2)$	Univariate normal distribution with mean value μ and variance σ
$N(\underline{\mu}, \boldsymbol{\Sigma})$	Multivariate normal distribution with mean $\underline{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
$\boldsymbol{I}_m$	Identity matrix of size m
$\underline{0}$	Null vector or zero vector
$E[X]$	Expected value of the random variable X
$\text{cov}(X, Y)$	Covariance of random variables X and Y
$\text{var}(X)$	Variance of random variable X
$\text{erf}x$	Value $\text{Erf}x$ of Gaussian error function
$\phi(x)$	Value $\phi(x)$ of PDF of the $N(0, 1)$ normal distribution function
$\boldsymbol{\Phi}(x)$	Value $\boldsymbol{\Phi}(x)$ of CDF of the $N(0, 1)$ normal distribution function
$\text{Prob}(X \leq x_{\text{limit}})$	Probability for $X \leq x_{\text{limit}}$
$\rho_{X,Y}$	Correlation coefficient of random variables X and Y

Chapter 1
Introduction

Joachim Haase and Manfred Dietrich

During the last years, the field of microelectronics has been moving to nanoelectronics. This development provides opportunities for new products and applications. However, development is no longer possible by simply downscaling technical parameters as used in the past. Approaching the physical and technological limits of electronic devices, new effects appear and have to be considered in the design process. Due to the extreme miniaturization in microelectronics, even small variations in the manufacturing process may lead to parameter variations which can make a circuit unusable. A new aspect for digital designers is the occurrence of essential variations not only from die to die but also within a die. Therefore, inter-die and intra-die variations have to be taken into account not only in the design of analog circuits as already done, but also in the digital design process. The great challenge is to assure the functionality of high complex digital circuits with respect to physical, technological, and economic boundary conditions. In order to evaluate design solutions within an acceptable time and with acceptable efforts the methods applied in the design process must support the analysis of design solutions as accurate as necessary and as simple as possible. As a result, the expected yield will be achieved and circuits can be manufactured economically. In this context, CMOS technology will remain the most important driving force for microelectronics over the next years and will be responsible for most of the innovations and new applications. For this reason, the subsequent paragraph will focus on this technology. The first chapter provides an introduction to the outlined problems.

J. Haase (✉) • M. Dietrich
Design Automation Division EAS, Fraunhofer-Institut Integrierte Schaltungen, Zeunerstr. 38, 01069 Dresden, Germany
e-mail: joachim.haase@eas.iis.fraunhofer.de; manfred.dietrich@eas.iis.fraunhofer.de

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6_1,

1.1 Development of CMOS Semiconductor Technology

Technology progress in IC design and semiconductor manufacturing has resulted in circuits with more functionality at lower prices for the last decades. The number of components on a chip especially in digital CMOS circuits doubled roughly every 24 months as predicted by Moore's Law. This trend was mostly driven by decreasing the minimum feature sizes used in the fabrication process. The requirements in the context of this development have been summarized in the International Technology Roadmap for Semiconductors (ITRS) for years [1]. For a long time, the progress has been expressed by moving from one technology node to the next. The technology nodes were characterized by the half pitch item of DRAM staggered-contacted metal bit lines as shown in Fig. 1.1. The 2009 ITRS document adds new criteria for further developments. Nevertheless, the half-pitch definition anymore indicates the direction of the expected future progress. In the case of MPUs and ASICs it measures the half-pitch of M1 lines. For flash memories, it is the half-pitch of unconducted polysilicon lines.

In this way, the 130 nm-, 90 nm-, 65 nm-, 45 nm-nodes, and so on were defined. The half-pitch is scaled by a factor $S \approx 0.7 \approx 1/\sqrt{2} = 1/\alpha$ moving from one node to the next. Over two cycles, the scaling factor is 0.5. In accordance with this development, the device parameters, line parameters, and electrical operating conditions were scaled. The result was a decrease of the delay time of digital components with simultaneous decrease of the their sizes. Thus, faster chips with more components could be developed that enabled a higher functionality. For more than 35 years, the fundamental paper by Robert H. Dennard and others [2] could be used as a compass for research and development in this area (see Tables 1.1 and 1.2, $\alpha = \sqrt{2}$).

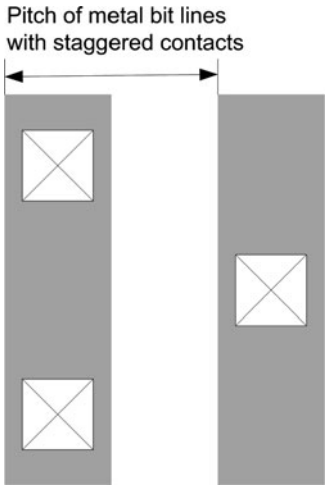


Fig. 1.1 2009 Definition of pitches [1]

Table 1.1 Scaling of device parameters [2, 3]

Parameter	Scaling factor
Channel length L	$1/\alpha$
Channel width W	$1/\alpha$
Oxide thickness t_{ox}	$1/\alpha$
Body doping concentration N_a	α
Threshold voltage V_{th}	$(<)1/\alpha$
Gate capacitance $C_{\mathrm{g}} \sim \frac{\kappa_{\mathrm{ox}}}{t_{\mathrm{ox}}} W \cdot L$	$1/\alpha$

Table 1.2 Scaling for interconnection lines [2, 3]

Parameter	Scaling factor
Wire pitch	$1/\alpha$
Wire spacing s_{w}	$1/\alpha$
Wire width W_{w}	$1/\alpha$
Wire length L_{w}	$1/\sqrt{\alpha}$
Wire thickness t_{w}	$1/\sqrt{\alpha}$
Line resistance $R_{\mathrm{L}} \sim \frac{L_{\mathrm{w}}}{W_{\mathrm{w}} t_{\mathrm{w}}}$	α
Line response time $\sim R_{\mathrm{L}} C$	1
Wire-to-wire capacitance $\sim \frac{\kappa_{\mathrm{isolation}}}{s_{\mathrm{w}}} t_{\mathrm{w}} L_{\mathrm{w}}$	1

Table 1.3 Scaling for circuit performance [2, 3]

Parameter	Scaling factor
Supply voltage	$1/\alpha$
Depending voltages V	$1/\alpha$
Current I	$1/\alpha$
Delay time (of a component)	$1/\alpha$
Power dissipation $\sim VI$ (of a component)	$1/\alpha^2$
Power density $\sim VI/A \sim \frac{1/\alpha^2}{1/\alpha^2}$	1
Normalized voltage drop of lines $\sim IR_{\mathrm{L}}/V$	α
Line current density $\sim \frac{I}{t_{\mathrm{w}} W_{\mathrm{w}}}$	Increasing with α

The scaling rules assured not only the functional progress. Performance was increased while reducing power per circuit components. The power density retained stable (see Table 1.3).

However, for instance downscaling the threshold voltage V_{th} and oxide thickness t_{ox} results in higher subthreshold leakage and gate leakage currents resp. [4]. Thus, power consumption became more and more a problem. "Dennard's Law" could no longer be followed [5]. To overcome the limits, new materials, new devices, and new design concepts have been investigated [6]. In parallel, process variations have to be considered in order to predict performance and yield of VLSI designs.

Further trends include, on the one hand, geometrical and equivalent scaling and, on the other hand, a functional diversification. The first trend is announced as "More Moore" while the second is discussed as "More than Moore" [1]. At the end, system-level performance has to be improved [7]. In order to compare different solutions, reliable methods to predict the system behavior are becoming necessary.

1.2 Consequences of Silicon Technology Challenges

Reducing the channel length of the CMOS devices, short-channel effects such as velocity saturation and drain-induced barrier lowering have to be considered. The threshold voltage V_{th} strongly depends on the effective channel length L_{eff} and the operational voltages. These and other effects have to be considered in the transistor models in order to predict performance and power consumption sufficiently exact.

Scaling of the threshold voltage V_{th} leads to a point where the subthreshold leakage current $I_{\mathrm{sub}} \sim \mathrm{e}^{-\frac{V_{\mathrm{th}}}{nV_{\mathrm{T}}}}$ with slope factor $n \approx 1.5$ and thermal voltage $V_{\mathrm{T}} \sim$ *temperature* T becomes a dominant factor for the power consumption of a circuit. Thus, a further scaling of the threshold voltage is difficult. Furthermore, the signal swing given by the difference of gate source voltage V_{GS} and V_{th} cannot be decreased under a critical limit without compromising the robust circuit behavior. This fact furthermore limits the scaling of the supply voltage.

Further contributions to the transistor leakage are the band-to-band-tunneling leakage and the gate leakage current $I_{\mathrm{gate}} \sim \mathrm{e}^{-\frac{t_{\mathrm{ox}}}{\beta_1}}$, where β_1 is a fitting coefficient. The value strongly depends on the gate thickness t_{ox}. The gate leakage results from tunneling of electrons through the gate dielectric [8]. The gate capacitance must be maintained over a limit while shrinking the geometry in order to assure the controllability of the channel current. Thus, shrinking of the gate thickness could be avoided by a gate material with high permittivity known as high-k material. "High" notes that the permittivity is greater than that of silicon oxide SiO_2.

Shrinking the geometry also influences the interconnection of components. The delay of local wires between gates remains constant (see Table 1.2). However, global wires such as busses and clock networks tend to follow the chip dimensions. Wires can be considered as distributed RC lines. The delay depends on the product of line resistance times line capacitance. Thus in order to reduce the delay, interconnect materials with lower resistance and dielectrics with lower permittivity (low-k materials) have been investigated. For instance, a lower resistance can be achieved by using copper instead of aluminium for interconnect lines. A lower permittivity reduces also the parasitic wire-to-wire capacitance. However, it is suspected that modifications of the dielectric material could lead to an inacceptable leakage. Looking at the RC product, it follows that the delay of the interconnect lines increases quadratically with its length. Thus, splitting the long interconnect lines and inserting repeaters is a reasonable strategy to reduce the overall delay [9]. However, this is paid by higher energy costs per transition because of the inserted drivers. There is a tradeoff between speed and energy consumption. Reducing the signal swing is an effective method to save energy. However, the robustness of the signal transmission against supply noise, crosstalk, and variations of the line parameters must be assured.

With scaling also the impact of the variations increases. It can be distinguished between those that are coming from the manufacturing process as, for instance, the lithography and those that are due to fundamental physical limitations as, for

instance, given by energy quantization. The variations are classified into different manners. Front-end variability is variability that impacts the devices. Back-end variability results from steps creating the interconnects. Furthermore, it should be distinguished between variations from die-to-die and variations within a die. They are called inter-die and intra-die variations, respectively. The inter-die variations impact all devices and interconnects of a die in (nearly) the same way. We will try to describe them using correlated random variables, whereas intra-die variations can be described by uncorrelated or weak spatial correlated random variables. Downscaling the CMOS technology intra-die variations become more important. The parameter P can be represented by a sum of its nominal value P_{nom} as well as random variables characterizing the inter-die variation P_{inter} and intra-die variation P_{intra} contributions [10]

$$P = P_{\text{nom}} + P_{\text{inter}} + P_{\text{intra}}. \tag{1.1}$$

Besides these variations, changes of the environment a circuit operates in must also be considered. The temperature, supply voltages, and input signals have an impact on the circuit performance. These variations are called environmental variations. The functionality of a circuit must be guaranteed within specified limits. Last but not least the functionality over time must be assured. Aging effects such as electromigration and negative bias temperature instability are further sources of variations.

Furthermore, shrinking device geometry while scaling device parameters and operating conditions in accordance makes the transistor performance more sensitive to variations. This trend due to short-channel effects can be noticed for leakage currents and speed. For instance, the sensitivity of the I_{on} current that depends on the effective channel length L_{eff}, the supply voltage, and the effective carrier mobility μ_{eff} that depends on the channel doping N_{ch} increases over technology generations [11] and makes the delay times more sensitive against parameter variations.

In order to reduce the consequences of these developments, new technology innovations and device architectures as strained silicon, silicon-on-insulator, very high mobility devices, and for instance trigate transistors have been developed (see more information, for instance, in [6, 9, 12]).

1.3 Impact on the Design Process

1.3.1 An Example Concerning Inter-Die and Intra-Die Variations

Let us discuss the impact of parameter variations on the design process with the help of a simple example. The map between a performance value y and the parameter values x_i shall be given by a function f

$$f : \mathbb{R}^n \to \mathbb{R}, (x_1, x_2, \cdots, x_n) \mapsto y. \tag{1.2}$$

Let X_i be a random variable that describes variations of the ith parameter and Y describes the associated variation of the performance value

$$Y = f(X_1, X_2, \cdots, X_n). \tag{1.3}$$

Knowledge of the map f and the probability distributions of the X_i would allow to determine the probability distribution of Y with the help of Monte Carlo simulation studies. Using a simplified approach, expected tolerances of the performance parameter can be estimated. f is replaced by its first-order Taylor series at the operating point. The parameters shall be Gaussian distributed, where μ_i is the mean or nominal value of the ith parameter and σ_i is its standard deviation. Thus for "small" parameter variations, Y can be approximated by a first-order Taylor series

$$Y \approx y_{\text{nom}} + \sum_{i=1}^{n} a_i \cdot (X_i - \mu_i), \tag{1.4}$$

where y_{nom} is the nominal value of the investigated performance parameter and $a_i = \frac{\mathrm{d}f}{\mathrm{d}x_i}$ are the first-order derivatives or parameter sensitivities at the nominal values of the parameters. Then Y is also Gaussian distributed with mean value y_{nom} and standard deviation σ_Y where $3\sigma_Y$ measures the tolerance. The variance is given by

$$\sigma_Y{}^2 = \sum_{i=1}^{n} a_i{}^2 \sigma_i{}^2 + 2 \cdot \sum_{i=1}^{n} \sum_{j=i+1}^{n} \rho_{i,j} a_i a_j \sigma_i \sigma_j, \tag{1.5}$$

where $\rho_{i,j}$ is the correlation coefficient of the ith and jth parameter.

Let us now built up the sum of n parameters with the same Gaussian distribution $N(\mu, \sigma)$ and defining in this way a special performance variable Y^*:

$$Y^* = \sum_{i=1}^{n} X_i{}^*. \tag{1.6}$$

Y^* can for instance be interpreted as the delay time of a chain of n gates with same delay distribution. If all delay times of the individual gates are strongly correlated (all correlation coefficients $\rho_{i,j}$ equal 1), it follows from (1.5)

$$\sigma_{Y^*,\text{correlated}} = \sqrt{n \cdot \sigma^2 + 2 \cdot \frac{n \cdot (n-1)}{2} \cdot \sigma^2} = n \cdot \sigma. \tag{1.7}$$

If all delay times of the individual gates are strongly uncorrelated (all correlation coefficients $\rho_{i,j}$ equal 0), it follows from (1.5)

$$\sigma_{Y^*,\text{uncorrelated}} = \sqrt{n} \cdot \sigma. \tag{1.8}$$

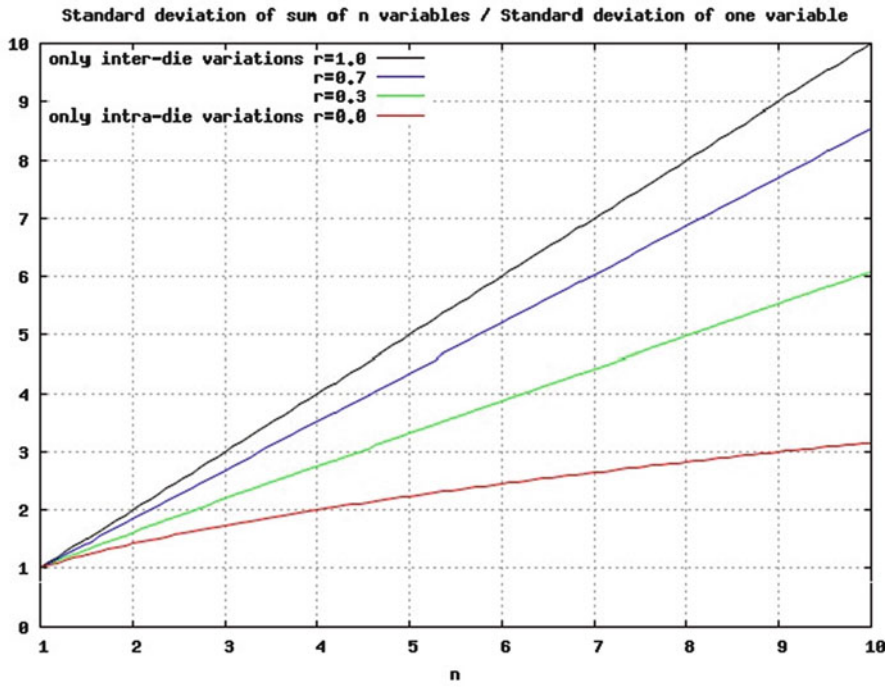


Fig. 1.2 σ_n of a sum of n variables divided by the standard deviation σ of one variable $\frac{\sigma_n(r)}{\sigma}$

Let us now assume that the variations of the parameters result from strongly correlated inter-die variations with variance $r \cdot \sigma^2$ and uncorrelated intra-die variations with variance $(1-r) \cdot \sigma^2$. The intra-die and inter-die variations are also uncorrelated. Thus, the overall variance of an individual parameter retains σ^2. Then we get

$$\sigma_{Y^*,\text{mixed}} = \sigma_n(r) = \sqrt{n^2 \cdot r + n \cdot (1-r)} \cdot \sigma \tag{1.9}$$

Static timing analysis (STA) is widely used in the design flow for timing verification. It assumes a full correlation of process parameters within a die. Thus, it neglects the characteristics of intra-die variations and only considers inter-die variations. The behavior is checked for "corner cases." "Worst case," "typical case," and "best case" are investigated for associated parameter sets of the transistor models [13].

However, Fig. 1.2 shows that, for instance, the delay time may be overestimated in this way. The procedure brings to much pessimism into the design flow. The more intra-die variations have to be taken into account, the more improvements on analysis methods are necessary.

1.3.2 Consequences for Methods to Analyze Designs

Design methods for nanoscale CMOS have to consider the variability and uncertainty of parameters predicting the behavior of a circuit. There is an impact of challenges in nanoscale technology on EDA tool development [14]. A number of methods are available to take variability into consideration.

The compact device models represent a link between the characteristics of the manufacturing process and the methods that shall predict the behavior of a semiconductor circuit. Interactions that are understood can be expressed in a systematic way by deterministic mathematical models. Phenomena that are poorly understood are often described by stochastic models. Thus, the choice of an appropriate model is essential for the subsequent conclusions. The Berkeley Short-channel IGFET Models are state-of-the-art compact MOS models. BSIM3 was a first industry-wide used model. It was extended to the BSIM4 model in order to describe MOSFET physical effects in the sub-100 nm regime. These models are based on threshold voltage formulations. The new PSP model is a surface-potential based model. It promises an accurate description of the moderate inversion region that becomes a larger part of the voltage swing as the supply voltage is scaled down [15]. The behavior in the time domain as well as the leakage behavior must be covered by the models in use.

Several methods have been developed and implemented to extract parameters of compact models either from measurement or based on device simulations [16]. For statistical design methods, the knowledge of the probability characteristics of the parameters is necessary. Various methods have been developed to determine these characteristics of the transistor parameters [17]. Important sources of variations of the transistor behavior in the 65-nm process are, for instance, variations of gate length, threshold voltage, and mobility [18]. The determination may base on TCAD approaches or measurements of process variations using test chips or circuits. Transistor arrays and ring oscillators are typical test structures for this purpose [19]. However, there are only a few publications on real data concerning process variations [20]. For future technology nodes, predictive transistor models have been developed [11,21,22]. They enable to study future developments in a very early stage. To map random process variability onto designer-controllable variables, simple approaches have been investigated [23].

Several mathematical methods can be applied in order to describe the parameter variations. In most cases, it can be and is assumed that the parameters are Gaussian distributed. The dependency of the parameters can be expressed in these cases by correlation matrices. However, if these parameters are not linearly mapped on the performance variables, these variables are in general not Gaussian distributed. This is for instance possible if the map (1.4) cannot be applied. Thus, it is also necessary to consider methods for describing non-Gaussian random variables. Basic relations between parameters and performance variables can be investigated using techniques that analyze variances. In the case of Gaussian distributed parameters, principal component analysis can be used to reduce the number of basic random variables. Correlated non-Gaussian parameters can be transformed to statistically independent

variables using independent component analysis for instance. Furthermore, appropriate methods for describing spatial correlation of parameters of a die must be available if necessary.

A main task consists in mapping the probabilistic characteristics of process or transistor parameters onto performance variables of components and circuits. In principle, this can be done by numerical and analytical methods. Handling the complexity arising in the IC design flow is a major problem. Thus, special methods as, for instance, statistical static timing analysis (SSTA) [24] have been developed. These methods require, on the one hand, an additional effort in a preparation phase – for instance for library characterization. On the other hand, they assume some simplifications as for instance linear mapping in order to handle the complexity. Thus, in order to check their advantages and limitations it is necessary to the check the results of these approaches against a "golden" model at least in the introduction phase. A golden reference can often be established by Monte Carlo studies.

The objectives of the design process are often contradictory. Short delay times, low leakage and dynamic power, high yield, and high robustness are requirements. From the mathematical point of view, this is a multicriteria optimization problem. A cost function built up by a weighted sum delivers only one solution. A set of optimal solutions can be determined as a Pareto frontier [25]. Based on the proposed optimal solution points, it can be decided which one should be preferred.

The following chapter describes fundamentals of transistor modeling and mathematical methods to handle statistical design tasks. Chapter 3 gives a description of the sources of variability and their representations. Chapter 4 demonstrates typical methods used for the investigations of the impact of variations on the performance of a design. In Chap. 5, some application examples will show how to make a good choice under the available methods and apply them for special designs. The chapters give an overview on the current state of the art in the different fields and go into more detail when discussing special experiences of the contributors with some of the presented approaches.

References

1. International technology roadmap for semiconductors, executive summary. http://www.itrs.net/Links/2009ITRS/2009Chapters_2009Tables/2009_ExecSum.pdf (2009)
2. Dennard, R., Gaensslen, F., Rideout, V., Bassous, E., LeBlanc, A.: Design of ion-implanted MOSFET's with very small physical dimensions. IEEE Journal of Solid-State Circuits **9**(5), 256–268 (1974)
3. Meer, P., van Staveren, A.R., M., A.H.: Low-Power Deep Sub-Micron CMOS Logic, 1 edn. Springer (2004)
4. Roy, K., Mukhopadhyay, S., Mahmoodi-Meimand, H.: Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits. Proceedings of the IEEE, **91**(2), 305–327 (2003)
5. Bohr, M.: A 30 year retrospective on dennard's MOSFET scaling paper. IEEE Solid-State Circuits Newsletter **12**, 11–13 (2007)
6. Chen, T.C.: Challenges for silicon technology scaling in the nanoscale era. In: Proceedings of the Solid State Device Research Conference, 2009. ESSDERC '09, pp. 1–7 (2009)

7. Dennard, B.: (interview) discussing dram and cmos scaling with inventor bob dennard. IEEE Design & Test of Computers **25**(2), 188–191 (2008)
8. Mukhopadhyay, S., Roy, K.: Modeling and estimation of total leakage current in nano-scaled CMOS devices considering the effect of parameter variation. In: Proceedings of the 2003 International Symposium on Low Power Electronics and Design ISLPED '03, pp. 172–175 (2003)
9. Rabaey, J.: Low Power Design Essentials. Springer, Boston, MA (2009). DOI 10.1007/978-0-387-71713-5
10. Srivastava, A., Blaauw, D., Sylvester, D.: Statistical Analysis and Optimization for VLSI: Timing and Power. Springer Science+Business Media Inc, Boston, MA (2005). DOI 10.1007/b137645
11. Zhao, W., Cao, Y.: New generation of predictive technology model for sub-45 nm early design exploration. IEEE Transactions on Electron Devices **53**(11), 2816–2823 (2006)
12. Haselman, M., Hauck, S.: The future of integrated circuits: A survey of nanoelectronics. Proceedings of the IEEE **98**(1), 11–38 (2010)
13. Chiang, C.C., Kawa, J.: Design for manufacturability and yield for nano-scale CMOS. Series on integrated circuits and systems. Springer, Dordrecht (2007). DOI 10.1007/978-1-4020-5188-3
14. Kawa, J., Chiang, C., Camposano, R.: EDA challenges in nano-scale technology. In: IEEE Custom Integrated Circuits Conference CICC '06, pp. 845–851 (2006). DOI 10.1109/CICC.2006.320844
15. Grabinski, W., Nauwelaers, B., Schreurs, D.e.: Transistor Level Modeling for Analog/RF IC Design, chap. PSP: An Advanced Surface-Potential-Based MOSFET Model, pp. 29–66. Springer, Dordrecht (2006)
16. Sharma, M., Arora, N.: OPTIMA: A nonlinear model parameter extraction program with statistical confidence region algorithms. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **12**(7), 982–987 (1993)
17. Cheng, B., Dideban, D., Moezi, N., Millar, C., Roy, G., Wang, X., Roy, S., Asenov, A.: Benchmarking statistical compact modeling strategies for capturing device intrinsic parameter fluctuations in BSIM4 and PSP. IEEE Design Test of Computers **27**(2), 26–35 (2010). DOI 10.1109/MDT.2010.2
18. Zhao, W., Liu, F., Agarwal, K., Acharyya, D., Nassif, S., Nowka, K., Cao, Y.: Rigorous extraction of process variations for 65-nm CMOS design. IEEE Transactions on Semiconductor Manufacturing **22**(1), 196–203 (2009)
19. Orshansky, M., Nassif, S., Boning, D.: Design for Manufacturability and Statistical Design. Springer (2008)
20. Nassif, S.: Modeling and analysis of manufacturing variations. In: IEEE Conference on Custom Integrated Circuits, pp. 223–228 (2001). DOI 10.1109/CICC.2001.929760
21. Li, X., Zhao, W., Cao, Y., Zhu, Z., Song, J., Bang, D., Wang, C.C., Kang, S., Wang, J., Nowak, M., Yu, N.: Pathfinding for 22nm CMOS designs using predictive technology models. In: IEEE Custom Integrated Circuits Conference CICC '09, pp. 227–230 (2009). DOI 10.1109/CICC.2009.5280845
22. Zhao, W., Li, X., Nowak, M., Cao, Y.: Predictive technology modeling for 32nm low power design. In: 2007 International Semiconductor Device Research Symposium, pp. 1–2 (2007). DOI 10.1109/ISDRS.2007.4422430
23. Wang, V., Agarwal, K., Nassif, S., Nowka, K., Markovic, D.: A simplified design model for random process variability. IEEE Transactions on Semiconductor Manufacturing **22**(1), 12–21 (2009). DOI 10.1109/TSM.2008.2011630
24. Blaauw, D., Chopra, K., Srivastava, A., Scheffer, L.: Statistical timing analysis: From basic principles to state of the art. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **27**(4), 589–607 (2008). DOI 10.1109/TCAD.2007.907047
25. Graeb, H., Mueller, D., Schlichtmann, U.: Pareto optimization of analog circuits considering variability. In: 18th European Conference on Circuit Theory and Design ECCTD 2007, pp. 28–31 (2007). DOI 10.1109/ECCTD.2007.4529528

Chapter 2
Physical and Mathematical Fundamentals

Bernd Lemaitre, Christoph Sohrmann, Lutz Muche, and Joachim Haase

This chapter provides a short overview on the basics of CMOS transistor modeling with respect to deep submicron requirements and mathematical approaches to analyze variations in the design process. Technical terms are going to be defined and explained; physical processes and mathematical theories will be illustrated.

The most important component in today's microelectronics is the transistor. Section 2.1 focuses on the MOSFET transistor and its modeling. The effects of variations in different technology parameters on the transistors behavior will be analyzed. The subsequent chapters build upon this background and deduce the influence of the device level on the circuit level. MOSFET transistors are designed as pMOS and nMOS transistors.

These complementary MOSFET transistors form the foundation of the implementation of low-energy CMOS circuits. Today more than 90% of all digital circuits are designed and manufactured using this technology. The functionality of these transistors will be briefly described in the first section of the chapter. In addition, the effects of different technology parameters on their behavior will be examined and effects of technology progress on the development of transistor modeling approaches will be discussed. Moreover, the relation between technological variations, parameter sensitivities, and variations of model parameters will be investigated.

In addition, this section will outline the impact of variations of transistor parameters on the variations of delay times and energy consumption of a circuit.

Section 2.2 introduces statistical methods for describing and analyzing variations which are important for an understanding of approaches used in the design process.

B. Lemaitre (✉)
MunEDA GmbH, Stefan-George-Ring 29, 81929 Munich, Germany
e-mail: bernd.lemaitre@muneda.com

C. Sohrmann • L. Muche • J. Haase
Fraunhofer Institute for Integrated Circuits IIS, Design Automation Division EAS,
Zeunerstraße 38, 01069 Dresden, Germany
e-mail: christoph.sohrmann@eas.iis.fraunhofer.de; lutz.muche@eas.iis.fraunhofer.de;
joachim.haase@eas.iis.fraunhofer.de

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6_2,

The section is going to explain how to describe univariate and multivariate normal distributed random variables as well as non-Gaussian distributions. Additionally, concepts to determine parameters of non-Gaussian distributions are presented. Furthermore, methods that reduce the complexity of random variables using principal component analysis and singular value decomposition will be shown. There are different ways to transform random variables using analytical and numerical methods. Underlying assumptions, limitations, and application possibilities of these statistical methods will be discussed. Moreover, approaches that allow for analyzing not only linear models but also second-order and special polynomial higher-order models will be introduced. A short outlook on importance sampling as a way to determine small probabilities and on the evaluation of results by statistical tests concludes the chapter.

Bernd Lemaitre and Christoph Sohrmann are the authors of Sect. 2.1. Lutz Muche and Joachim Haase prepared Sect. 2.2.

2.1 Modeling of CMOS Transistors

Physical, manufacturing, environmental, and operational conditions influence strongly the CMOS transistor characteristics. When scaled into the deep submicron regime, their influence on leakage and time domain behavior has to be evaluated anew. The section describes the physical background behind different effects that have to be considered by the digital designer as well as the impact of variations on the behavior. Spatial and temporal correlations of parameters are considered. The main objective is to separate first- and second-order effects that are important for the static and dynamic behavior. The principles that determine the threshold voltage and in this way the subthreshold leakage are discussed. This considers the impact of channel length, drain-induced barrier lowering and body-biasing effect among others. Furthermore, the mechanisms (Fowler–Nordheim and direct-oxide tunneling) that are the source of gate leakage are presented. In order to keep the gate leakage under control, high-κ materials are introduced. It is described how the impact of the velocity-saturation effect on reducing the current drive for a given gate voltage in the DSM regime influences the characteristic of the CMOS transistor. Device and technology innovations such as strained silicon, dual-gated devices, and very high mobility devices are briefly explained. In this chapter, various compact transistor models also will be described with main focus on the BSIM model. An overview of the various leakage mechanisms and an insight view into the leakage modeling of those transistor models will be done. Also, aspects of variability modeling will be discussed.

2.1.1 General Types of MOSFET Models

During the 1970s, the nMOS technology was the major technology for highly complex, digital circuits. Because of the advantages of the CMOS technology, including

low static power consumption, simple scalability laws, and stability of operation, the CMOS technology became the general-purpose technology in the 1980s. The use of electrical simulators, such as SPICE, allows a quick evaluation of the circuit performance before high costly prototypes. However, the quality and accuracy of the simulation results by a simulator depends on the quality and accuracy of the circuit element model. Therefore, the used MOSFET model for circuit simulation plays a crucial role in chip design productivity. One has to differentiate between two main types of device models, the numerical device model and the compact model. Numerical device models are used to study the device physics and to predict the electrical and thermal behavior of a semiconductor device. These models solve a set of partial differential equations, describing the physics of the device. Because of their high computational effort and huge amount of memory, numerical device models are not suited for use in circuit simulators. Compact models describe the terminal properties of the device by using of a simplified set of equations, or by an equivalent subcircuit model. The purpose of a compact model is to obtain simple, fast, and accurate representations of the device behavior. Compact models are suited to evaluate the performance of integrated circuits with large quantity of transistors. In general, compact device models can be divided into three categories:

- Physical models (based on device physics)
- Table lookup models (with tables containing device data for different bias points)
- Empirical models (where the device characteristics are represented by equations that fit the data)

2.1.2 A Brief History of Transistor Models

The first MOSFET model for SPICE-like circuit simulators, the LEVEL 1 model, often called Shichman-Hodges model [1], is a simplified first-order model only for long channel transistors. The simple model describes the current dependence on voltages for a gate voltage greater than the threshold voltage. The subthreshold behavior and current is assumed as zero. The terminal capacitances, which are described by the Meyer model [2], are not charge-conserving. The LEVEL 2 model addresses in addition second-order, small-geometry effects. The subthreshold current is not equal to zero and the capacitive model can be either the Meyer model [2] or the Ward-Dutton model [3], where the charge is conserved. In practice, the Level 2 model is computationally very complex. One of the main drawbacks of Level 2 are the often observed convergence problems during circuit simulation. The drawbacks are extensively discussed in [4]. The LEVEL 3 model is a semi-empirical model that addresses the shortcomings of LEVEL 2. It uses the Ward-Dutton capacitive model and convergence problems are rarely observed. The main drawbacks of the Level 3 model are the non-ideal modeling of the subthreshold current and the failure of correct modeling of the output conductance g_{ds}, which is defined as

$$g_{\mathrm{ds}} = \frac{\partial I_{\mathrm{ds}}}{\partial V_{\mathrm{ds}}} \,. \tag{2.1}$$

Especially the failure of the g_{ds} modeling makes the simulation of analog circuits critical; because g_{ds} is one of the main transistor attributes that affect the gain of an operational amplifier, or in general analog circuits.

The growing demand of the market in the 1980s for CMOS digital and Mixed Signal Chips and the higher pressure on the design groups pushes the development of new model types of the second model generation. Obviously, the Level 1, 2, and 3 models had too many shortcomings in practice to simulate circuits with ever-larger number of transistors and ever smaller dimensions. A different modeling approach compared to the first model generation (LEVEL 1, 2, and 3) had to be chosen to overcome especially the functional complexity and the shortcomings for smaller transistors (short channel effects).

At the University of Berkeley, the so-called BSIM models [5] (Berkeley Short-Channel IGFET Model) were developed with main emphasis on faster and more robust mathematics for circuit simulation, but less effort in the developing physical modeling approach. For analog circuit simulation, the main problems with the first BSIM generation were again a poor and sometimes negative modeling of the output conductance g_{ds}. Also, convergence problems occur within the SPICE simulation. Some of these problems were enhanced by modifications within BSIM2 and a HSPICE version Level28 [6]. During practical use of these models, the main shortcomings of the second model generation were their more empirical modeling approach and therefore the need to implement more fitting parameter without a clear physical meaning [7]. In the 1990s, the third model generation was introduced by BSIM3 and its extension BSIM4, but also with MOS Model 9, that was brought in by Philips into the public domain. By formulation of the third model generation, the modeling groups tried to come back to a more physical-based modeling approach. This should allow a more physical assignment of the model parameter to real physical measured effects and its values. Also, the introduction of smoothing functions especially at the transition between two operation regions of the transistor, which could not be modeled by one continuous equation, helps to prevent the output conductance and convergence problems. All models up to now uses formulations with the Drain-Source voltage as reference. The EKV model [8] uses the Bulk voltage as reference and is therefore full symmetrical formulated related to the Drain and the Source voltage. The mentioned MOSFET models are only the well-known models, which are available in the public domain. A lot of company proprietary models were developed by large semiconductor companies for internal use, which are implemented in popular SPICE like simulators, e.g., [6].

In August 1996, the Compact Model Council (CMC) [9] was formed, by large semiconductor, EDA companies and Foundries. The main purpose of the CMC was the promotion and standardization of compact models, and the implementation into commercial available SPICE-like simulators. The vision of the CMC was to promote the international, nonexclusive standardization of compact model formulations, and the model interfaces. One major push for forming the CMC, as industry-driven organization, was the problem that many proprietary models were in use. The interface between companies in cooperation or the interface working together

with design houses was too complex. Therefore, there was the need to standardize compact models for all major technologies in a way that customer communication and efficiency can be enhanced. Within the CMC, some models were standardized, e.g., BSIM3 and BSIM4 models for use down to 90 nm technologies.

In 2004 after many discussions in the modeling community, there was a widely-agreed-upon understanding that traditional threshold-voltage compact models, as used up to now, have to be replaced by more advanced surface-potential, or inversion charge-based models. Besides the need to rework the short and narrow channel effects, non-uniform lateral and vertical doping, and the introduction of quantum-mechanical corrections for the new technology generations, the new fourth model generation should have one continuous formulation over all regions of MOS operation. In 2004, the CMC calls for the next generation of industrial compact models, useful for 90 nm, 65 nm, 45 nm CMOS Technology nodes and below. Two new modeling approaches were developed were a continuous formulation of the MOS device behavior were described, based on the solution of the surface-potential in the channel ψ_s or the inversion charge Q_{inv}. The University of Berkeley developed the BSIM 5 model [10] with an iterative solution of the inversion charge Q_{inv}. The Pennsylvania State University and Philips developed together the PSP model [12, 13] as a common modeling activity based on Philips MOS 11 (successor of MOS 9) and the PS model from the Pennsylvania State University. The PSP model based on a explicit solution on the surface-potential ψ_s. As third model, the HISIM model [11, 14], which was formulated years before, from the Hiroshima University, was investigated by the CMC for a new modeling standard. The HISIM model bases on an iterative solution of the surface-potential ψ_s (see also Table 2.1).

In 2006, the CMC has standardized, the PSP model for standard CMOS technologies and in 2007 the HISIM model for high-voltage, high power applications.

In Table 2.1, an overview of the main MOSFET models with the technology, nodes, where these models are mainly in practical use, is given. The transition from one model to another, pushed by the introduction of new technology generations could only be done step by step. The new model will be tested and verified on the data of the new technology. To start design activity in the new technology, the technology characterization starts with the old model. If the new model is verified, the new model is introduced in one of the next design packages. Also, for mature technologies, such as 130 nm, Foundries will mostly use BSIM3 models today, because a new re-characterization of the technology on the basis of the latest models is too expensive. Therefore in practice all models of the 3rd generation are more or less in use for active design, depending on the used technology node (Table 2.1).

2.1.3 MOS Physics and Modelling

The current section shall provide a brief introduction to MOS transistor physics and its modeling. It is not supposed to be a comprehensive guide to semiconductor physics, which would require a solid mathematical background. The goal is rather to

Table 2.1 Overview of transistor models and respective technology nodes

Technology node	MOSFET model generation	MOSFET model	CMC standard	Model type based on
$\gg$1 µm	1st generation	Level 1		V_{th}
		Level 2		V_{th}
		Level 3		V_{th}
$\geq$1 µm	2nd generation	BSIM 1		V_{th}
		BSIM 2		V_{th}
350 nm		HSPICE level 28		V_{th}
250 nm	3rd generation	BSIM 3.x	X(1996)	V_{th}
180 nm		BSIM 4.x	X(2000)	V_{th}
		MOS 9, MOS 11		V_{th}
130 nm		EKV		Q_{inv} (iterative)
90 nm	4th generation	BSIM 5		Q_{inv} (iterative)
65 nm		PSP	X(2006)	ψ_s (explicit)
45 nm		HISIM	X(2007)	ψ_s (iterative)
32 nm				

learn some ideas leading to current modeling strategies and additionally learn about some causes of variations. The interested and the expert reader shall be referred to the appropriate literature for further study [3, 15]. Nevertheless, some fundamental concepts will be required for understanding the operation of a transistor. Therefore, we start this section with some basic concepts of semiconductor physics.

It is well known that the most widely used material in microelectronics today is silicon. Unfortunately, the properties of pure silicon are far from adequate for use in cutting edge applications. Therefore, the material requires some radical manipulation before it may be applied. It turns out that introducing impurity atoms into the silicon crystal, a process known as doping, provides such a handle. Doping allows the electronic properties of silicon to be tweaked as desired. The reason for this becomes clearer by revisiting silicon's atomic structure. As a Group IV element, each atom has four valence electrons. In the condensed state, silicon forms a diamond cubic lattice with four covalent bonds at each lattice site. Four electrons per site are involved in these bonds and no carriers are left for contributing to the conduction process. Therefore, pure silicon is an inadequate material for electronic applications. However, doping silicon with impurity atoms having either less or more than the four electrons required for perfect bonding between neighboring atoms introduces additional free carriers into the crystal. The main concept now is that charge will not only be carried by the abundant electrons which are not involved in the bonding process but also by so-called electron holes, a conceptual, positively charged particle, describing the absence of a valence electron in the bonding process. Electron holes are quasiparticles which behave like real particles and which may therefore be modeled similarly. The concept of electrons and holes leads to the following nomenclature. Group III elements are called *p*-type dopants since they

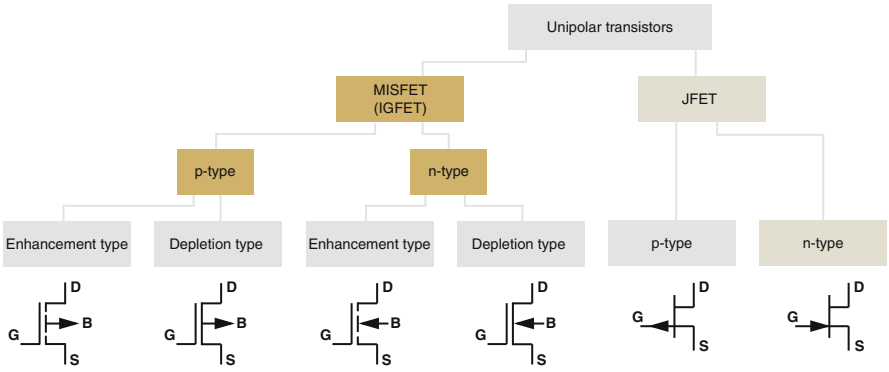


Fig. 2.1 Existing types of MOSFETs

have less than four valence electrons and therefore introduce positively charged holes into the silicon. Group V elements, on the other hand, are called *n*-type dopants for the fact that they add abundant electrons to the lattice. Additionally, the former are called acceptors for they are accepting electrons from the silicon crystal, whereas the latter are called donors, atoms which donate electrons to the crystal. In fact, a doped semiconductor at the same time contains both, electrons and holes. Depending on the ratio of the two species, they are labeled as minority and majority charge carriers. In *n*-type semiconductors, electrons are the majority and holes the minority carriers. Vice versa for *p*-type semiconductors. The number of free carriers in the crystal depends on the concentration of the doping atoms. The same applies to the conductivity.

Depending on those physical properties, transistors may be categorized into a variety of classes. The most fundamental two classes are the *unipolar* and the *bipolar* devices. As the name suggests, in the former case, only one kind of carriers contributes to the transport, whereas in the latter case both species may participate. In the context of CMOS design, one is mainly concerned with unipolar devices, also called *field-effect* transistors (FETs). Within the class of FETs, there are again two main categories, the *insulating gate* type (IGFET) and the *junction gate* type (JFET). The former type is most widely used and best known as its prominent representative, the *metal-oxide-semiconductor* type (MOSFET). Transistors are further distinguished by the type of terminal doping which can be either *n-type* or *p-type*, as explained above. Depending on the doping type, a transistor can now be either conducting or insulating at zero voltage between gate and source, called bias. For the case of JFETs, both types are conducting. Applying a gate-source voltage to either of the two suppresses possible current flow in the channel. In case of MOSFETs, those two types are denoted as *depletion* or *enhancement* type, depending on whether applying voltage between gate and source enhances or suppresses the current in the channel, respectively. A schematic summary of FET-types is given in Fig. 2.1.

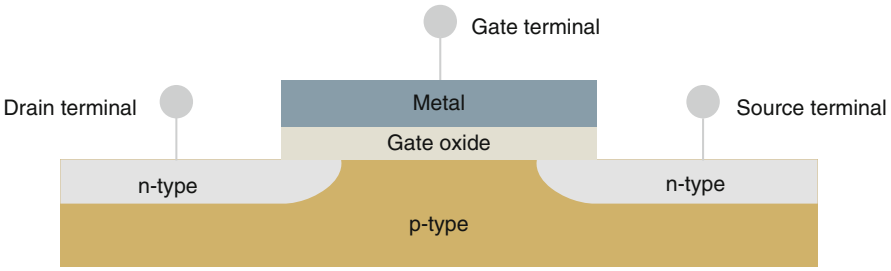


Fig. 2.2 Schematic cross-sectional cut of an *n*-type MOSFET structure

After having introduced some fundamental physical ideas and types of transistors, the following explanations focus on the structure and operation of MOSFETs. Figure 2.2 schematically shows a cross-sectional view of a *n*-type MOSFET. The base material, i.e., the substrate, is a slightly *p*-doped silicon crystal. Two heavily *n*-doped regions are implanted as the source and the drain electrodes. Since substrate remains in between, this forms an NPN-structure. Thus, no conduction is possible in the off-state. The remaining *p*-substrate forms the channel of the transistor. An insulating dielectric layer of silicon dioxide (SiO_2) is then deposited right above the channel, which separates the channel from the gate electrode. The gate material is *n*- or *p*-doped polysilicon. The stacking of bulk, dielectric, and gate forms a capacitor, which is loaded upon applying a voltage difference between bulk and gate. As the naming MOSFET implies, the current-voltage-characteristics of the channel can be manipulated by the electric field in the "bulk-gate-capacitor." For an *n*-type MOSFET, the source terminal is in general connected to the bulk and is used as the voltage reference point. Therefore, the quantity V_{gs} equally refers to the gate-bulk-voltage. Depending on the applied V_{gs} and V_{ds}, three modes of operation can be distinguished:

- Subthreshold or weak inversion regime: $V_{gs} < V_{th}$
- Linear regime: $V_{gs} > V_{th}$ and $V_{ds} < (V_{gs} - V_{th})$
- Saturation or strong-inversion regime: $V_{gs} > V_{th}$ and $V_{ds} > (V_{gs} - V_{th})$

The operation of the transistor is best understood by first letting $V_{ds} = 0$ and slowly raising V_{gs}. This process will be exemplarily described in the following using the already discussed *n*-type device. By applying a positive voltage to the gate, $V_{gs} > 0$, and entering the *subthreshold or weak inversion regime*, the majority carriers within the *p*-type substrate will be repelled from the insulating SiO_2 layer, thereby forming a region depleted of majority carriers – *the depletion region*. This region contains less positive carriers than the remaining bulk and thus is less positively charged. Further increasing V_{gs} leads to fully majority-carriers-deprived SiO_2 surface, leaving a neutrally charged layer close to the insulator. This is observed when $V_{gs} = V_{th}$. Here, the transistor switches to the *linear regime*. Beyond this point, a layer of negatively charged carriers begins to accumulate at the insulator surface, forming an oppositely charged layer within the positive *p*-type bulk background – the so-called *inversion layer*.

Table 2.2 Classification of transistor models

$\mathbf{V}_{\text{th}}$-based models	Charge-based models	Surface potential-based models
LEVEL 1–3	EKV	PSP
BSIM1–4	ACM	HiSIM
Philips-MM9	BSIM5	Philips MM11

In the subthreshold regime, where V_{gs} lies in between flat-band-voltage and V_{th}, the channel between source and drain is said to be in weak inversion. There are very few free carriers available for charge transport. The current flows mainly by diffusion rather than drift. As the name suggest, this regime is often made use of in analog circuits. The source-drain current behaves similarly to the collector-emitter current of a bipolar transistor. Below threshold, there is an exponential dependence between drain-source current and gate-source voltage. This is the reason why the subthreshold regime is important for low-voltage and low-power analog circuits.

$$I_{\text{ds}} \propto \mathrm{e}^{\frac{V_{\text{gs}}-V_{\text{th}}}{nV_{\text{T}}}}. \tag{2.2}$$

For a few years now, this technique is more and more applied to digital circuits as well [17]. Objectives are a low power-consumption, e.g., in sensor networks, or high performance by achieving very low delays. However, in the subthreshold regime, parameter variations are a much more severe challenge for the design because of the strongly nonlinear behavior of delays and current as a function of input slew or load capacitance.

The available transistor models may be classified by how the integral for the drain current is evaluated. There are three common approaches: $\mathbf{V}_{\text{th}}$ based, charge based, and surface potential based. This classification and its realization in transistor models is shown in Table 2.2.

2.1.4 Physical Effects in Transistor Models

The growth in integrated circuit density and speed is the heart of the rapid growth of the semiconductor industry. The transistor saturation current is an important parameter because the transistor current determines the time needed to charge and discharge the capacitive loads on a chip, and thus impacts the product speed more than any other transistor parameter. The goal of MOSFET scaling could be understood by two general topics.

First, the increase of transistor current (speed) for charging and discharging parasitic capacitances and second the reduced size (density). The increased transistor current requires a short channel and high gate oxide field because the inversion layer charge density is proportional to the oxide field. The reduced size of the device requires a short channel length and smaller channel width. Therefore, the planar CMOS devices were scaled in the past mainly by shrinking the dimensions and the voltages [80] (Fig. 2.3).

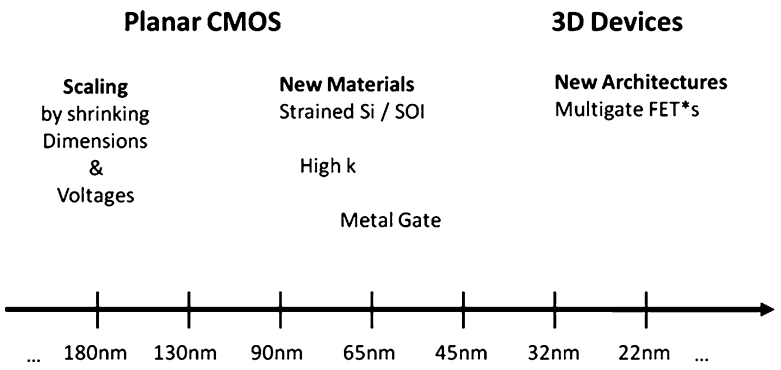


Fig. 2.3 Main trends in CMOS scaling

Table 2.3 Major effects to be modeled within 4th generation CMOS models for simulation down to 45/32 nm

- Gate to body, gate to inversion, gate to S/D oxide tunneling currents
- Stress effect as a function of layout
- Impact Ionization current
- Flicker Noise and thermal noise at all terminals, all biases, all temperatures
- Nonuniform vertical doping
- Nonuniform lateral doping
- Short channel effect
- Drain-induced barrier lowering (DIBL)
- Channel length modulation
- Substrate current induced body effect
- Velocity saturation including velocity overshoot, source end velocity limit
- Well proximity effect on Vth
- Poly gate depletion
- Narrow-width effect
- Bulk-charge effect
- Field-dependent mobility
- Finite inversion layer thickness (quantum mechanical effect)
- Non-quasi-static (NQS) effect
- Diode IV forward and reverse model
- Diode reverse breakdown
- Diode CV forward and reverse, including temperature
- Gate resistance model
- Substrate resistance network
- GIDL/GISL
- Asymmetric and bias-dependent source/drain resistance

As process technology scales beyond 100-nm feature sizes, for functional and high-yielding silicon the traditional design approach needs to be modified to cope with the increased process variation, interconnect processing difficulties, and other novel physical effects [81] (Table 2.3).

The scaling of gate oxide in the nano-CMOS regime results in a significant increase in gate direct tunneling current. The subthreshold leakage and gate direct tunneling current are no longer second-order effects. The effect of gate-induced drain leakage (GIDL/GISL) will be felt in designs, such as DRAM and low-power SRAM, where the gate voltage is driven negative with respect to the source. Scaling planar CMOS will face significant challenges. Introduction of new material systems, e.g., strained Si/SOI, high-κ and metal gates were used to scale devices down to

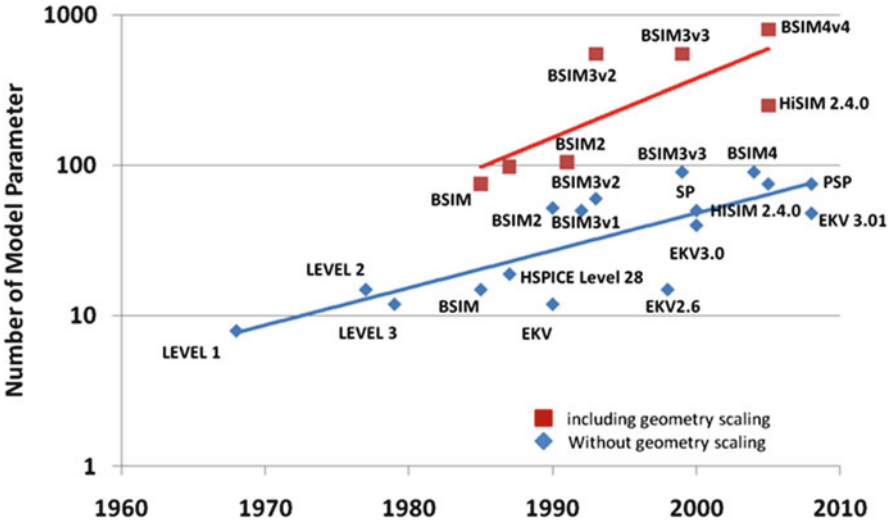


Fig. 2.4 Evaluation of the numbers of model parameter with the model complexity [79]

32 nm and 22 nm. In addition, new device architectures, e.g., multigates and 3D devices were needed to break the scaling barriers in future beyond 22 nm technology range.

The term high-κ dielectric refers to a material with a high dielectric constant κ (as compared to silicon dioxide) used in semiconductor manufacturing processes, which replaces the silicon dioxide gate dielectric.

As the thickness scales below 2 nm, leakage currents due to tunneling increase drastically, leading to unwieldy power consumption and reduced device reliability. Replacing the silicon dioxide gate dielectric with a high-κ material allows increased gate capacitance without the unwanted leakage effects [82].

Strained silicon and strain engineering refers to a strategy employed in semiconductor manufacturing to enhance device performance. Performance benefits are achieved by modulating strain in the transistor channel, which enhances electron mobility (or hole mobility) and thereby conductivity through the channel [83].

In order to shrink down beyond 22 nm (see Fig. 2.4), 3D devices or multigate devices which incorporate more than one gate into a single device are in development. The multiple gates may be controlled by a single gate electrode, wherein the multiple gate surfaces act electrically as a single gate, or as independent gate electrodes. Multigate transistors are one of several strategies being developed by CMOS semiconductor manufacturers to create ever-smaller microprocessors and memory cells, colloquially referred to as extending Moore's Law [84, 85].

Compact models describe the terminal properties of the scaled devices by using a simplified set of equations, or by an equivalent subcircuit model. As a consequence of the ongoing scaling activities and changing of device architecture, the compact models have to follow by including the main new effects into the model equations.

Also approximations needed for simplified modeling have to be adjusted. Second-order effects in today's technologies could change to first-order effects in the next technology node, e.g., the subthreshold currents and the effect of gate-induced drain leakage (GIDL/GISL).

Unfortunately for the first chip designs in a new technology, the designer has to cope with available models, which were developed for older technology nodes.

The modeling and the availability of new models within commercial circuit simulators, including all novel effects, will follow the technology development and ramp up in a Manufacturing Line approximately 1–3 years later.

Also, the complexity of the models and the number of parameter will increase with time, following the changes and the complexity of the scaled technologies. The number of parameters of the most commonly used circuit simulation models achieves an order of 1,000 parameter (see Fig. 2.4). Starting within the 1980, some models were developed including geometry scaling models that increases the number of model parameter extensively.

2.1.5 Impact of Variations and Model Sensitivity

After the previous short summary of various nominal effects occurring in today's technology, this section focuses on how process variations affect performance upon the continuing scaling.

2.1.5.1 Variations and Scaling

During the last decades, MOS technology was constantly scaled down with a rate approximately predicted by Moore's law as easly as in 1965 [18]. The rate at which the integration density increased over the years remained surprisingly constant. This can be mainly attributed to the concept of *constant field scaling*, where transistor parameters are scaled down such that the internal electric field remains constant and thus the physical behavior is preserved. This has first been proposed in the seminal work by Dennard et al. [19]. However, in order to maintain or even increase circuit performance, the device threshold voltages need to be scaled in proportion to the supply voltage [4]. This in turn has a severe side-effect on the subthreshold leakage current, which depends exponentially on the difference between V_{gs} and V_{th}. Therefore not only will nominal leakage increase drastically, but also the sensitivity to threshold voltage fluctuations increases exponentially. Since V_{th}-fluctuations are easily seen to increase with shrinking device dimensions and decreasing dopant number, old-fashioned shrinking by scaling soon crosses a point where reliability becomes a serious issue. Ever since, process variations occurred within semiconductor fabrication. However, this point marked a new kind of hurdle to be taken and its disturbing arrival in technology was anticipated long before. Fortunately, the topic of process variations became strongly popular and

much research had been done in order to prevent the sudden death of Moore's Law. Eventually, new strategies were introduced such as high-κ dielectrica, metal gates, strained silicon, fully depleted SOI (FD-SOI), or multi-gate devices, coming to the rescue of Moore's prediction.

On the other hand, there is a constantly increasing variety of effects leading to an increase of variability with the continuation of scaling. Kenyon et al. [21] recently provided an excellent summary of challenges in terms of variability for the 45 nm technology. One can summarize the most important sources of fluctuations, which sooner or later require adequate modeling:

- Random dopant fluctuations [22]
- Line-edge roughness [23]
- Variations of oxide thickness [24]
- Nonuniform threshold voltage by fixed charge [25]
- Defects and traps [26]
- Patterning proximity effects [27]
- Polish [28]
- Strain-induced variation [29]
- Variations in implant and anneal processes [30]
- Variation of temperature in operation
- Ageing and wear-out
- Signal coupling and cross-talk
- Supply voltage and package noise

These unwieldy and mostly nonlinear effects need to be tackled and controlled by process engineers and designers currently but even more so in the years to come.

2.1.5.2 Parameter Correlations

Generally, all fluctuations across devices, circuits, dies, wafers, or wafer lots are correlated in a certain way. Only the correlation strength varies depending on the source of the variation (which process step, environmental influences, and so on). Considering a single parameter fluctuation, one may think of a temporal or spatial *correlation length* within the manufacturing process. This length determines whether the variation of a parameter can be modeled independently across different entities or whether the coherence needs to be taken into account. For simplicity, engineers usually take the binary approach by setting the correlation coefficient to either zero or one, respectively. Although this is far from realistic, the usual lack of detailed measurements renders any attempt of a more detailed modeling pointless. One therefore retracts to such a simplified description which additionally offers two major simplifications: First, each varying parameter can be statistically described by as little as two numbers. Secondly, the binary correlation approach gives rise to the appealingly simple notion of *local and global fluctuations*. Local fluctuations are also known to analog designers as *mismatch*, the single most important statistical design parameter for matching pair transistors. Since effects resulting from global

fluctuations are usually taken into account by a corner-based analysis, they are of lesser importance. However, as the number of corners increases exponentially with each technology node, such guardbanding leads to a significant design pessimism, strongly diminishing yield. The binary correlation description may be sufficient for most modeling purposes. However, it is important to realize that a truly realistic and physical variability model on circuit or system level requires a fully multivariate stochastic description. This is rather involved since any fluctuating parameter needs to be included as an additional dimension of a multivariate distribution function. However, such methodology becomes especially important in hierarchical system modeling with uncertain input [31–33].

2.1.5.3 Local Fluctuations and Pelgrom's Mismatch Model

The importance of local parameter fluctuations has been emphasized previously. Although analog design suffers strongly under local fluctuations, any circuit involving matching transistor pairs, for instance an SRAM cell [34], is severely affected. Apart from the transistor mismatch, nonuniform behavior throughout the circuit is a result from local fluctuations. The first work to describe a model for local fluctuations was the seminal paper by Pelgrom et al. [35]. Their mismatch model was based on very few but fundamental assumptions and has general validity. The model proposes a spatially varying parameter $p(x,y)$ and defines the mismatch as the difference of this parameter over two different rectangles, Ω_1 and Ω_2, representing the areas of two devices. The model shall predict the difference of the expected value of the parameter for each area, which reads as

$$\Delta p_{\Omega_1\Omega_2} = \langle p(x,y)\rangle_{\Omega_1} - \langle p(x,y)\rangle_{\Omega_2}\,. \tag{2.3}$$

After parameterizing the distance by D and setting the area to $|\Omega| = W \cdot L$, the variance of Δp can be computed, yielding

$$\sigma^2_{\Delta p} = \frac{A_{\Delta p}}{WL} + S_{\Delta p} D^2, \tag{2.4}$$

where $A_{\Delta p}$ and $S_{\Delta p}$ are process- and device-dependent parameters. This model predicts in a simple form the local fluctuations under scaling and may be applied to any parameter such as V_{th} or T_{ox}. The fluctuation, i.e., standard deviation, of a parameter as a function of device area can thus be estimated as

$$\Delta^{\mathrm{loc}} p \propto \frac{1}{\sqrt{|\Omega|}}. \tag{2.5}$$

Fluctuations of device dimensions scale according to the equations $\Delta^{\mathrm{loc}} L \propto 1/\sqrt{W}$ and $\Delta^{\mathrm{loc}} W \propto 1/\sqrt{L}$. Pelgrom's law is commonly used in analog design, where increasing the device area is a well-established means of reducing local fluctuations.

2.1.5.4 Mathematical Description of Fluctuations

This section looks at some mathematical aspects regarding the modeling of fluctuations in circuit simulation. Any strategy leading to a continuation of the shrinking process whilst improving performance must necessarily involve one of the two following techniques: Reduction of the fluctuations themselves, for instance ΔV_{th}, ΔT_{ox}, or ΔL_{eff}, or reduction of the responsiveness of the circuit to fluctuations. Strong fluctuations alone do not necessarily imply a strong impact on the circuit's performance. If the circuit is insensitive to or even independent from this very parameter, the range of variation becomes less significant for the design process. The amount of impact is called *sensitivity*. Thus, the effect of a variation on a performance quantity, in the following called y, depends equally on the parameter fluctuation and the sensitivity of the measured quantity on the parameter. This fact is also easily derived from a mathematical viewpoint and can be motivated by a Taylor expansion of the usually complicated and unknown parameter dependence, $y = y(p)$, which up to first order reads as[1]

$$y \approx y_0 + \frac{\partial y}{\partial p}(p - p_0). \tag{2.6}$$

The parameter p usually varies according to a distribution within a range characterized by its variance, σ^2. The performance deviation from its nominal value y_0 is henceforth called Δy. Since this quantity is dimensionful, one usually introduces a normalization. Care has to be taken which normalization is used when reusing sensitivity information from commercial simulators in an external context. In the following, we define the sensitivity as

$$\frac{\Delta y}{y} = \frac{\partial y}{\partial p}\frac{\Delta p}{y}, \tag{2.7}$$

where Δp is the deviation of the parameter from its nominal value. Although being just a low-order polynomial approximation, this model has quite a compelling advantage: Even with a single sensitivity value in a single point of parameter space, the device behavior with respect to many effects such as process variations, degradation and aging can be approximated albeit in a qualitative fashion. This model has proven to work remarkably well in practice. Additionally, the sensitivity coefficient is a very handy ballpark number for a quick-and-dirty estimation of how much the performance of a design is affected by the parameter. In many cases, the sensitivity is readily determined by a finite-difference approach. If such a naive approach is computationally too expensive, there is a great deal of literature dealing with efficiently solving this problem [36–38]. At this point, however, we will not go further into detail and investigate some model sensitivities analytically instead.

[1] Whether the symbol y refers to the value or to the function should be evident from the context.

In summary, the overall variability of a device, circuit, or system, is equally affected by both, the variance of the parameters and the sensitivity against this parameter. For physical parameters, the former is usually a technological artifact and solely determined by the fabrication precision. The latter, however, may be adjusted through circuit and system design. In the following, we will primarily focus on the sensitivity of MOS transistors with respect to fluctuating parameters by reviewing established leakage and timing models.

2.1.5.5 Delay Sensitivity

There are a number of analytically tractable models attempting to predict the cell delay [16, 39–41] by approximating the drain current behavior. Such models are usually of low complexity compared to full transistor models and contain many assumption, for instance, regarding saturation velocity or gate voltage. The simplification is achieved by introducing fit parameters which strongly depend on the dimensions and technology used. However, this also means that they do not predict the current correctly in all regimes. But such accuracy is not required for a first-order manual analysis. Especially the *Alpha Power Law Model* by Sakurai et al. [39] focuses only on the correct description of the cell delay. The authors approximate the delay as a sum of two components: An input slope dependent part and the time to charge or discharge the following cell. The input slope-dependent component becomes less significant with enhanced velocity saturation. Under this assumption, the estimated cell delay reads as

$$\tau_{\text{cell}} \propto \frac{C_{\text{L}} V_{\text{dd}}}{I_{\text{dsat}}}, \tag{2.8}$$

where C_{L} is the output capacitance to be driven by the cell. The saturation current is defined by the α-model as

$$I_{\text{dsat}} = \frac{W}{2L} \mu C_{\text{ox}} \left(V_{\text{gs}} - V_{\text{th}}\right)^{\alpha}, \tag{2.9}$$

where α lies in between 1 and 1.5. In the constant field scaling picture with a scaling factor S, the parameter scale according to $W, L, V_{\text{dd}}, V_{\text{th}}, C_{\text{L}} \propto 1/S$ and $C_{\text{ox}} \propto S$, and V_{gs} at saturation should be proportional to V_{dd}. For constant mobility, we conclude that

$$I_{\text{dsat}} \propto \frac{1}{S^{\alpha-1}}, \tag{2.10}$$

which decreases with increasing S. The cell delay can thus be assumed to scale as

$$\tau_{\text{cell}} \propto S^{\alpha-3} \approx \frac{1}{S^2}. \tag{2.11}$$

The same analysis can be carried out for the sensitivity. By applying the chain rule to the delay dependence on V_{th}, we find

$$\frac{\Delta\tau_{\mathrm{cell}}}{\tau_{\mathrm{cell}}} = \frac{\partial\tau_{\mathrm{cell}}}{\partial V_{\mathrm{th}}}\frac{\Delta V_{\mathrm{th}}}{\tau_{\mathrm{cell}}} = \frac{\partial\tau_{\mathrm{cell}}}{\partial I_{\mathrm{dsat}}}\frac{\partial I_{\mathrm{dsat}}}{\partial V_{\mathrm{th}}}\frac{\Delta V_{\mathrm{th}}}{\tau_{\mathrm{cell}}} = \alpha\frac{\Delta V_{\mathrm{th}}}{V_{\mathrm{gs}} - V_{\mathrm{th}}}. \tag{2.12}$$

Ignoring the scaling of ΔV_{th}, this essentially leads to a scaling of the relative delay fluctuations proportional to S. Thus, this simple model has shown that the impact of fluctuations on the delay becomes worse in a simple scaling scenario. Therefore, new technological concepts and devices are required in order to push Moore's law further. An equivalent reduction can only be achieved by introducing entirely new models or by a reduction of the fluctuation itself, ΔV_{th}. Until today, simulations concluded that ΔV_{th} remained almost a constant [42, 43]. However, this picture will not hold for sub-45 nm technologies where a strong increase in variability is predicted. The above analysis is meant as an exemplary calculation and may also be carried out for more realistic drain current and timing models as well for other parameters such as L, V_{dd}, or C_{ox}.

2.1.5.6 Leakage Current Sensitivity

Modern MOS transistors exhibit a variety of different leakage mechanisms. A comprehensive analysis of their nominal and sensitivity behavior is an involved task. In order to demonstrate the multitude of leakage effects, we have summarized significant sources of static leakage currents [44]:

- Reverse-bias *pn* junction leakage
- Subthreshold leakage current
- Gate oxide leakage current
- Gate induced drain/source leakage
- Gate current due to hot carrier injection
- Channel punchthrough current [45]
- Dielectric breakdown [16]

Additionally, there is a dynamic leakage component resulting from cell switching and shorting the supply to the ground. The most pronounced leakage currents, such as subthreshold leakage, shows a very strong sensitivity to V_{th}. In general, the threshold voltage is one of the most important parameters regarding variability. Much work has been done to compute scaling properties of V_{th}-fluctuations [42, 43]. The formula for the threshold voltage can be written as [46]

$$V_{\mathrm{th}} = V_{\mathrm{fb}} + |2\phi_{\mathrm{p}}| + \frac{\lambda_{\mathrm{b}}}{C_{\mathrm{ox}}}\sqrt{2qN_{\mathrm{ch}}\varepsilon_{\mathrm{S}}(|2\phi_{\mathrm{p}}| + V_{\mathrm{sb}})} - \lambda_{\mathrm{d}}V_{\mathrm{ds}}, \tag{2.13}$$

where λ_b and λ_d are parameters for the drain-induced barrier lowering (DIBL) and body-biasing effect. Then one can write the subthreshold leakage (drain leakage) [46] as

$$I_{sub} = \mu_{eff} C_{ox} \frac{W}{L} (m-1) V_T^2 e^{\frac{V_{gs}-V_{th}}{mV_T}} \left(1 - e^{-\frac{V_{ds}}{V_T}}\right), \tag{2.14}$$

the gate oxide leakage [46] as

$$I_{gate} = WA_g \left(\frac{V_{dd}}{T_{ox}}\right)^2 \exp\left(-B_g \frac{T_{ox}}{V_{dd}}\right), \tag{2.15}$$

and the GIDL (junction leakage) [47, 48] as

$$I_{GIDL} = AW\Delta L \frac{\varepsilon_{Si}}{E_0 N_0} E_{Si}^4 \exp\left(-\frac{E_0}{E_{Si}}\right). \tag{2.16}$$

The GIDL is usually orders of magnitude larger in NMOS than in PMOS devices, but overall negligible compared to other leakage mechanisms [4]. In deriving a sensitivity expression, we therefore focus on subthreshold leakage variations under varying V_{th} and compute the derivative of (2.14). One thus obtains

$$\frac{\Delta I_{sub}}{I_{sub}} = \frac{\partial I_{sub}}{\partial V_{th}} \frac{\Delta V_{th}}{I_{sub}} = -\frac{I_{sub}}{mV_T} \frac{\Delta V_{th}}{I_{sub}} = -\frac{\Delta V_{th}}{mV_T}. \tag{2.17}$$

Regarding the exponential dependence between the nominal current and the nominal voltage, this result might come somewhat as a surprising. However, one has to bear in mind that the nominal leakage current still grows exponentially with a V_{th}-swing. The above sensitivity is only relative to this nominal dependence. In summary, when scaling the technology, relative variations in the subthreshold leakage scale in a similar fashion as ΔV_{th}.

2.2 Methods to Describe and Analyse Parameter Variations

2.2.1 Introduction

We are interested in the statistical behavior of characteristics describing the quality of a micro- or nanoelectronic structural element, such as leakage, delay or transition time, say performance, output or response characteristics, denoted by y.

In particular, the values of these characteristics depend on numerous process parameters (input variables) such as threshold voltage, oxide layer thickness or gate lengths, e.g., $V_{th}, T_{ox}, \Delta L, \ldots$, denoted by $X_1, X_2, \ldots, X_m$ in the following. In other terms, the performance characteristics y can be described by functional relationships $y = h(x_1, x_2, \ldots, x_m)$. Call $\underline{X}$ the vector of process parameters $\underline{X}^T = \{X_1, X_2, \ldots, X_m\}$ of length m. Call Y a performance parameter. $X_1, X_2, \ldots, X_m$ as well as Y are real valued random variables. This is represented in Fig. 2.5.

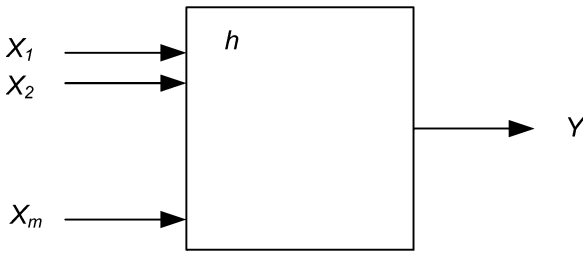


Fig. 2.5 Relation between random input variables $\underline{X}$ and random output performance variable Y

In the following, we will consider several mathematical methods that are provided to analyze the interaction between input and performance variables. This gives an impression of the power and the limitations of the different approaches and the related problems.

We assume the knowledge of the joint probability density function (PDF) $f_{\underline{X}}(\underline{x})$ of the process parameter vector $\underline{X}$, or the PDF's $f_{X_1}(x_1)$, $f_{X_2}(x_2)$, ..., $f_{X_m}(x_m)$ of the process parameters $X_1, X_2, \ldots, X_m$, in particular.

Our goal is the evaluation of the cumulative distribution function (CDF) $F_Y(y)$, PDF $f_Y(y)$, and the moments EY^k of the performance characteristic Y.

Starting point are the different possibilities to characterize random variables. The process parameters can often be characterized by a normal distribution. However, in a lot of cases nonnormal distributions are also of interest. This may concern the description of process parameters but more often the performance variables. Handling the variability of a huge number of parameters methods to reduce the complexity is required. More details will follow in Sects. 2.2.3–2.2.6.

The characteristics of the performance variable can be investigated by analytical methods if special requirements are fulfilled for the random characteristics of the input variables and the function h. This is the content of Sect. 2.2.7. In general, the dependency can be investigated by numerical methods. The interesting task is to reduce the computational effort in this case. Related problems will be discussed in Sect. 2.2.8.

At the end, we have to check the results by appropriate methods. This will be discussed in Sect. 2.2.9.

2.2.2 Characterization of Random Variables

We want to refresh some terms that will be used to describe real-valued continuous random variables. To simplify the representation, it is restricted to the one-dimensional case. Generalizations for the multivariate case can be carried out.

The expected value of a random variable X with probability density function $f_X(x)$ is given by

$$E[X] = \int_{-\infty}^{\infty} x \cdot f_X(x) \mathrm{d}x = \mu_X. \tag{2.18}$$

The moment of order k is defined by

$$E\left[X^k\right] = \int_{-\infty}^{\infty} x^k \cdot f_X(x)\mathrm{d}x. \tag{2.19}$$

The central moment of order k is defined by

$$E\left[(X-\mu_X)^k\right] = \int_{-\infty}^{\infty} (x-\mu_X)^k \cdot f_X(x)\mathrm{d}x. \tag{2.20}$$

The second-order central moment is called variance

$$var(X) = E\left[(X-\mu_X)^2\right] = \int_{-\infty}^{\infty} (x-\mu_X)^2 \cdot f_X(x)\mathrm{d}x = {\sigma_X}^2, \tag{2.21}$$

where the square root σ_X of the variance is the standard deviation of the random variable. Let g be a mapping $g : \mathbb{R} \to \mathbb{R}$ then we can generalize and determine the expected value of the random variable given by $g(X)$

$$E\left[g(X)\right] = \int_{-\infty}^{\infty} g(x) \cdot f_X(x)\mathrm{d}x. \tag{2.22}$$

Some general rules for handling the expected values can be derived. The expected value of the random variable $a \cdot X$, where a is a constant is

$$E\left[a \cdot X\right] = a \cdot E\left[X\right]. \tag{2.23}$$

Its variance is given by

$$var(a \cdot X) = E\left[a^2 \cdot (X-\mu_X)^2\right] = a^2 \cdot var(X). \tag{2.24}$$

The expected value of the sum of two random variables X and Y is always the sum of the expected values

$$E\left[X+Y\right] = E\left[X\right] + E\left[Y\right]. \tag{2.25}$$

If two random variables X and Y are given, their covariance can be defined as follows

$$\begin{aligned} cov(X,Y) &= E\left[(X-\mu_X)\cdot(Y-\mu_Y)\right] = E\left[X\cdot Y\right] - \mu_X \cdot \mu_Y \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x-\mu_X)\cdot(y-\mu_Y)\cdot f_X(x)\cdot f_Y(y)\mathrm{d}x\mathrm{d}y. \end{aligned} \tag{2.26}$$

The correlation coefficient $\rho_{X,Y}$ of two random variables is a normalized covariance with values between -1 and 1

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \cdot \sigma_Y} \tag{2.27}$$

This correlation coefficient is also known as Pearson's correlation coefficient. Because of (2.27), the covariance of two random variables can be expressed by $\text{cov}(X,Y) = \rho \cdot \sigma_X \cdot \sigma_Y$. Two random variables are uncorrelated if their correlation coefficient equals 0. Two random variables X and Y are independent if for all $\mathbb{R} \to \mathbb{R}$ maps g and h

$$E\left[g(X) \cdot h(Y)\right] = E\left[g(X)\right] \cdot E\left[h(Y)\right]. \tag{2.28}$$

It follows from (2.28) that for independent random variables X and Y and constants m and n the following relations are correct: $E\left[X^m \cdot Y^n\right] = E\left[X^m\right] \cdot E\left[Y^n\right]$ and also $E\left[(X-\mu_X)^m \cdot (Y-\mu_Y)^n\right] = E\left[(X-\mu_x)^m\right] \cdot E\left[(Y-\mu_Y)^n\right]$. Thus, independent random variables are always uncorrelated. The opposite conclusion is in general not right. Further conditions have to be fulfilled. If X and Y are jointly normal distributed (see the following section) and uncorrelated, then they are also independent and (2.28) can be applied.

2.2.3 Normal Distribution

2.2.3.1 Univariate Normal Distribution

The normal or Gaussian distribution often characterizes simple random variables that are given around a mean μ. The samples of the random variable are real numbers. Its special importance results from the central limit theorem. It indicates that the sum of a sufficiently large number of independent and identically distributed random variables with finite mean and variance will be approximately normally distributed. The graph of the density function describes a bell-shaped curve. The PDF of a normal distribution $N(\mu,\sigma^2)$ with mean μ and variance σ^2, $(\sigma > 0)$ is given by

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right). \tag{2.29}$$

The CDF describes the probability that the random variable X is less or equal to x. The subsequent formula describes the CDF of the normal distribution $N(\mu,\sigma^2)$

$$F_X(x) = \text{Prob}(X \le x) = \int_{-\infty}^{x} f_x(t)\mathrm{d}t = \frac{1}{2}\left(1 + \text{erf}\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)\right) \tag{2.30}$$

Table 2.4 Interval limits and probabilities of $N(\mu,\sigma^2)$ distribution

Factor c	Probability Prob$(\mu - c\cdot\sigma < X \leq \mu + c\cdot\sigma)$ in %
1.0	68.26894921
2.0	95.44997361
3.0	99.73002039
6.0	99.99999980
1.6448536270	90.0
1.9599639845	95.0
2.5758293035	99.0
3.2905267304	99.9

with the Gauss error function $\text{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x \text{e}^{-t^2}\text{d}t$. Thus, the probability that the simple $N(\mu,\sigma^2)$ distributed random variable X belongs to the interval $(\mu - c\cdot\sigma, \mu + c\cdot\sigma]$ equals

$$\text{Prob}\left(\frac{(X-\mu)\cdot(X-\mu)}{\sigma^2} \leq c^2\right) = erf\left(\frac{c}{\sqrt{2}}\right) \tag{2.31}$$

Some typical values are summarized in Table 2.4

2.2.3.2 Standard Normal Distribution

The special case $\mu = 0, \sigma = 1$ is called standard normal distribution $N(0,1)$. Its PDF is denoted by

$$\varphi(x) = \frac{1}{\sqrt{2\pi}}\exp\left(\frac{-x^2}{2}\right) \tag{2.32}$$

considering (2.29) and using (2.30) its CDF by

$$\Phi(x) = \int_{-\infty}^{x} \varphi(t)\text{d}t = \frac{1}{2}\left(1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right). \tag{2.33}$$

Generation of Normal Distributed Random Variables

A $N(\mu,\sigma^2)$ normally distributed random variable X can be constructed by

$$X = \mu + \sigma\cdot Z, \tag{2.34}$$

where $Z \sim N(0,1)$. This relation can be applied to create normal distributed random numbers that are used, for instance, in Monte Carlo simulations. Standard normal

distributed random number generators can be derived from uniform $U(0,1)$ random variables using the Box–Muller transformation or are available in appropriate simulation tools.

Assuming that a random variable X is described by a normal distribution, the parameters μ and σ^2 can be estimated based on independently distributed observed values $x_1, x_2, \cdots, x_n$. Maximum-likelihood estimation is an estimation method that determines the parameters of the distribution function in such a way that the sample values have the greatest joint likelihood [49]. Maximum-likelihood estimators are

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{and} \qquad s^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu})^2. \tag{2.35}$$

Because the estimated mean value $\hat{\mu}$ is used in (2.35) to estimate the variance, the estimation of the variance is not unbiased. An unbiased estimator of the variance based on Bessel's correction is

$$\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \hat{\mu})^2 = \frac{n}{n-1}\cdot s^2. \tag{2.36}$$

This version is more frequently used.

2.2.3.3 Multivariate Normal Distribution

Let us now consider a m dimensional random vector $\underline{X}^T = (X_1, X_2, \cdots, X_m)$ instead of a simple random variable X. The components X_i of the random vector $\underline{X}$ are simple random variables. We will discuss the case where the vector $\underline{X}$ is jointly normally distributed. That means, each component X_i is normally distributed and (!) arbitrary linear combinations of its components are also normally distributed. However, the components are in general not independent. The vector of the mean values of the components is $\underline{\mu}^T = (\mu_1, \mu_2, \cdots, \mu_m)$. The dependency of the random components is described by the covariance matrix Σ. The elements of the covariance matrix are

$$\Sigma_{i,j} = cov(X_i, X_j) = E[(X_i - \mu_i)(X_j - \mu_j)] = \rho_{i,j}\cdot\sigma_i\cdot\sigma_j, \tag{2.37}$$

where E determines the expected value of a random variable, σ_i is the standard deviation of X_i, σ_j is the standard deviation of X_j. $\rho_{i,j}$ $(-1 \le \rho_{i,j} \le 1)$ is the correlation coefficient of the random variables X_i and X_j. Thus, the PDF of the m dimensional multivariate normal distribution is given by

$$f_{\underline{X}}(\underline{x}) = \frac{1}{(\sqrt{2\pi})^m\sqrt{\det(\Sigma)}}\exp\left(-\frac{1}{2}\cdot(\underline{x}-\underline{\mu})^T\Sigma^{-1}(\underline{x}-\underline{\mu})\right). \tag{2.38}$$

It is obvious that (2.29) is in accordance with (2.38) for the one-dimensional case.

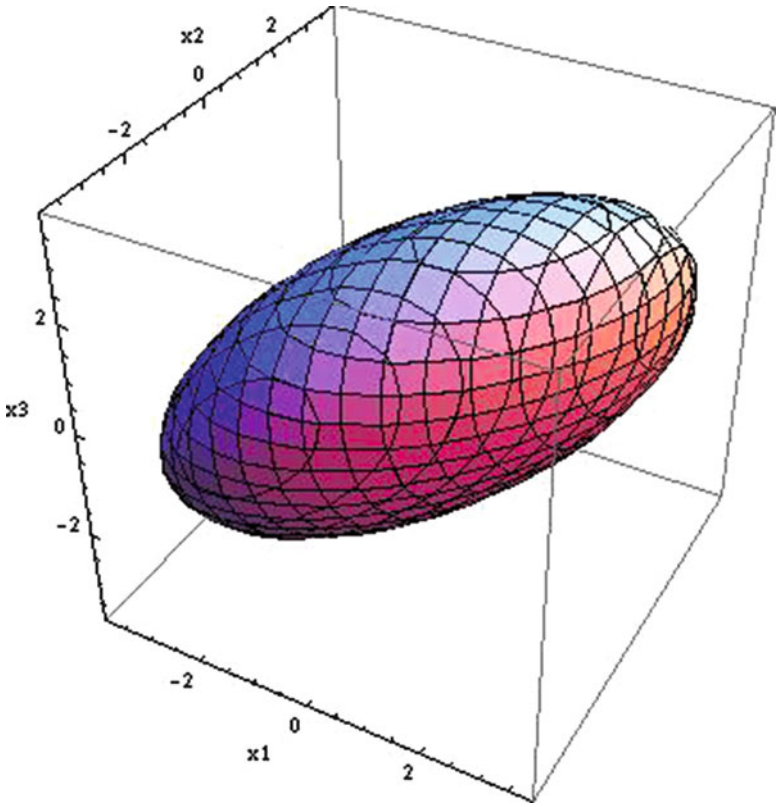


Fig. 2.6 Contour of a 3-ellipsoid with $\underline{\mu}^T = (0,0,0)$, $\sigma_1 = \sigma_2 = \sigma_3 = 1$, $\rho_{1,2} = 0.3$, $\rho_{1,3} = 0.5$ $\rho_{2,3} = -0.2$ and $c = 3.7625$

In the case of the univariate normal distribution, we were interested in the probability that the samples of the random variable belong to the interval $(\mu - c \cdot \sigma, \mu + c \cdot \sigma]$. The equivalent question in the multivariate case consists in determining the probability that the samples of the random vector $\underline{X}$ belong to the m-ellipsoid with the contour $(\underline{x} - \underline{\mu})^T \Sigma^{-1} (\underline{x} - \underline{\mu}) = c^2$. It should be mentioned that considering (2.38) the probability density of all points of this contour is the same (Fig. 2.6).

The probability that the $N_m(\underline{\mu}, \Sigma)$ multivariate distributed vector $\underline{X}$ belongs to the m-ellipsoid described above is given by [49, 50]

$$\text{Prob}(\underline{X} - \underline{\mu})^T \Sigma^{-1} (\underline{X} - \underline{\mu}) \leq c^2 = F_{\chi^2_m}(c^2) = \frac{\gamma\left(\frac{m}{2}, \frac{c^2}{2}\right)}{\Gamma\left(\frac{m}{2}\right)} = \mathscr{P}\left(\frac{m}{2}, \frac{c^2}{2}\right), \quad (2.39)$$

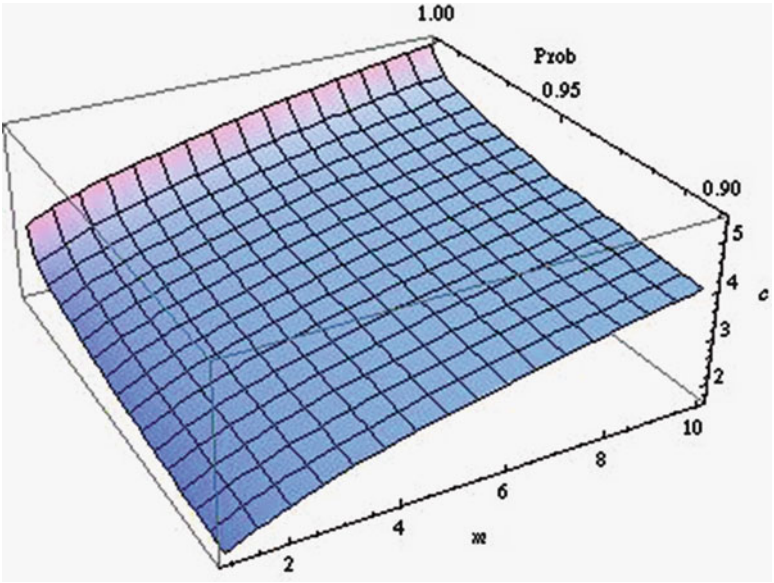


Fig. 2.7 c depending on dimension $m = 1, \cdots, 10$ and Probability given by (2.39)

where $F_{\chi^2_m}$ is the CDF of the chi-square distribution with m independent variables, γ and Γ denote the lower incomplete gamma and gamma functions resp. and $\mathscr{P}$ is known as lower regularized gamma function [51]. Equation (2.39) corresponds to (2.31) in the one-dimensional case.[2]

The dependency of c, dimension m and probability *Prob* used in (2.39) is represented in Fig. 2.7. For $m = 1$, the associated c and *Prob* values are certainly in accordance with Table 2.4. Furthermore, if for instance parameters are jointly normally distributed and a performance value is within its specification limits for all parameter samples inside of an ellipsoid, around the nominal values then this specification is fulfilled at least with the probability given by (2.39). The greater the distance c between contour and mean of the ellipsoid for a fixed m the greater is this probability. This circumstance and its consequences will be discussed in more detail in Sect. 4.6 on yield analysis methods.

A $N_m(\underline{\mu}, \Sigma)$ multivariate distributed random vector can be constructed by

$$\underline{X} = \underline{\mu} + \boldsymbol{G} \cdot \underline{Z}, \tag{2.40}$$

where the covariance matrix is expressed by $\Sigma = \boldsymbol{G} \cdot \boldsymbol{G}^T$ and $\underline{Z}$ consists of uncorrelated $N(0,1)$ distributed normal random variables. That means $\underline{Z}$ is $N_m(\underline{0}, \boldsymbol{I}_m)$

[2] $\frac{1}{\sigma^2} \sim \Sigma^{-1}$ and $\text{erf}\left(\frac{c}{\sqrt{2}}\right) = \mathscr{P}\left(\frac{1}{2}, \frac{c^2}{2}\right)$.

distributed. Equation (2.40) is the multidimensional version of (2.34). The covariance matrix Σ ($\det(\Sigma) \neq 0$) is a symmetric positive definite matrix. Thus, Cholesky decomposition can be used to determine the matrix elements of $\boldsymbol{G}$. $\boldsymbol{G}$ is a lower triangular matrix.

2.2.3.4 Bivariate Normal Distributed Random Numbers

For instance, in the bivariate case where $\underline{X}$ is $N_2(\underline{\mu}, \Sigma)$ are jointly distributed with

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix} \tag{2.41}$$

using (2.38) we get the joint PDF

$$f_{\underline{X}}(\underline{x}) = f_{X_1,X_2}(x_1,x_2) = \frac{1}{2\pi\sigma_1,\sigma_2\sqrt{1-\rho^2}}$$

$$\times \exp\left(-\frac{1}{2(1-\rho^2)}\left(\frac{(x_1-\mu_1)^2}{\sigma_1^2} - \frac{2\rho(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1\sigma_2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2}\right)\right). \tag{2.42}$$

Because in this case $\Sigma = \begin{pmatrix} \sigma_1 & 0 \\ \rho\sigma_2 & \sqrt{1-\rho^2}\sigma_2 \end{pmatrix} \cdot \begin{pmatrix} \sigma_1 & 0 \\ \rho\sigma_2 & \sqrt{1-\rho^2}\sigma_2 \end{pmatrix}^T$ the bivariate normally distributed correlated random variables can be expressed as follows

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} + \begin{pmatrix} \sigma_1 & 0 \\ \rho\sigma_2 & \sqrt{1-\rho^2}\sigma_2 \end{pmatrix} \cdot \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix}, \tag{2.43}$$

where Z_1 and Z_2 are $N(0,1)$ distributed uncorrelated random numbers.

We mention that using this approach in general m uncorrelated random numbers are necessary to describe a m dimensional multivariate random vector. To avoid problems when m is a huge number, methods as principal component analysis (PCA) can be applied to decrease the complexity.

As in the case of the univariate normal distribution, the vector of the mean values and the covariance matrix can be estimated based on observed samples. The formulas correspond to (2.35) and (2.36). However, it should be mentioned that if each variable X_i in $\underline{X}$ is univariate normal, it can happen that the joint distribution is not a multivariate normal distribution [52].

2.2.4 Nonnormal Distributions

2.2.4.1 Moments of the Distribution of a Random Variable

The approximation of a given sample $\{y_1, y_2, \ldots y_n\}$ to a PDF $f_Y(y)$ is a well-studied matter. Most of the methods base on the knowledge of the empirical moments $EY, EY^2, EY^3, \ldots$, (also the notation $m_1(Y), m_2(Y), m_3(Y), \ldots$ or even $m_1, m_2, m_3, \ldots$ is usual) or the central moments $\mu_2(Y), \mu_3(Y), \ldots$, likewise.

Further important characteristics used for an evaluation are the skewness γ_1 and the kurtosis γ_2 (also known as excess kurtosis)

$$\gamma_1 = \frac{\mu_3}{\mu_2^{3/2}}, \quad \gamma_2 = \frac{\mu_4}{\mu_2^2} - 3. \tag{2.44}$$

Special location measures of CDFs are called quantiles. A p-quantile Q_p $(0 < p < 1)$ gives value, where the cumulative distribution function of a random variable Y equals p. Special quantiles are the quartiles $Q_{0.25}$ (lower quartile or first quartile), $Q_{0.50}$ (middle quartile, also called median or second quartile), $Q_{0.75}$ (upper quartile or third quartile), where the distribution takes the values 0.25, 0.50, and 0.75, resp. Further quantiles are $Q_{0.95}$, $Q_{0.99}$, where $p = 0.95$, $p = 0.99$, respectively. For instance $y = Q_{0.95}$ means the probability for Y not to exceed the threshold y is 0.95, $P(Y < y) = 0.95$. Instead of the quantiles, also percentiles may be used where p is given by a percentage.

Quantile quantile plots (QQ plots) are standard tools in the explorative data analysis. They allow to compare samples of data jointly or a sample with a theoretical distribution. An important special case is the normal quantile plot, where the values of a normal distribution are drawn on the abscissa. If both distributions coincide, the graph is approximately on the bisecting line. An S-like curve indicates a distribution with a stronger kurtosis, an inverse S-like curve a distribution with smaller kurtosis. A right skew (left skew) distribution generates a concave (convex) curve, resp.

2.2.4.2 Parameterization of Special Distributions Based on Moments

An approximation of the PDF of a performance characteristic Y of a sample can be made by parameter fitting if a type of distribution function is assumed.

This method requires to specify a type of CDF $F_Y(y)$, which can be a more or less suited approximation only. The analytical known moments of $F_Y(y)$ and these obtained from the sample are identified, which allows an evaluation of the parameters of $F_Y(y)$.

Example. We assume the sample is Pearson type V distributed (inversed gamma distribution). The PDF is

$$f_Y(y) = \frac{p^q}{\Gamma(q)\, y^{q+1}} \exp\left(-\frac{p}{y}\right), \quad y \geq 0 \tag{2.45}$$

with

$$\mathrm{E}Y = \frac{p}{q-1}, \quad (q>1), \quad \mu_2(Y) = \frac{p^2}{(q-1)^2(q-2)}, \quad (q>2) \tag{2.46}$$

which immediately gives the real parameters p and q

$$p = \mathrm{E}Y\left(1+\frac{(EY)^2}{\mu_2(Y)}\right), \quad q = 2+\frac{(EY)^2}{\mu_2(Y)}, \tag{2.47}$$

where EY and $\mu_2(Y)$ can be estimated from the sample

$$\widehat{EY} = \overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i, \quad \widehat{\mu_2(Y)} = \frac{1}{n-1}\sum_{i=1}^{n}(y_i-\overline{y})^2. \tag{2.48}$$

The so-called method of moments is only based on the sample moments. It does not consider the sample elements in particular.

Another approach of parameter fitting taking into account all sample values $y_1, y_2, \ldots, y_n$ is the maximum likelihood (ML) approach. To estimate the wanted parameters, the maximum likelihood function

$$l = \sum_{i=1}^{n} \log(f_Y(y_i)) \tag{2.49}$$

is to maximize, which implies the disappearance of the partial derivatives of l with respect to parameters of $f_Y(y)$. For the above example (inverted gamma distribution), the conditions

$$\frac{\partial l}{\partial p} = \frac{\partial l}{\partial q} = 0 \tag{2.50}$$

lead to

$$\log p = \frac{1}{n}\sum_{i=1}^{n}\log y_i + \frac{\partial}{\partial q}(\log\Gamma(q)), \quad q = \frac{p}{n}\sum_{i=1}^{n}\frac{1}{y_i}. \tag{2.51}$$

In general, the maximum likelihood approach leads to equation systems, which can be solved by an iteration procedure, but not explicitly.

Suitable standard distributions to approximate of the distributional behavior of performance characteristics such as leakage or delay times are unimodal right skewed distributions defined for $y>0$. Good candidates are lognormal, Weibull, skew t, skew normal and the wide class of Pearson distributions.

The Pearson distributions represents a wide class of distribution functions, introduced by Pearson around 1895. All of them represent solutions of an ordinary differential equation with seven real coefficients. According to the sizes of these coefficients, the solutions are among others beta, Cauchy, χ^2, inverse χ^2, exponential, Fisher, gamma, inverse gamma, normal, Pareto, t (Student), and uniform distribution.

The best approximation for a particular sample can be selected according to the skewness γ_1 and the kurtosis γ_2, see [53].

2.2.4.3 Relationship Between Normal and Lognormal Distribution

The close relationships between normal and lognormal distribution can be applied to investigate the logarithm $\log(Y)$ instead of the performance Y itself. Since Y is lognormal distributed with PDF

$$f_Y(y) = \frac{1}{\sqrt{2\pi}\sigma y} \exp\left(-\frac{(\log y - \mu)^2}{2\sigma^2}\right), \tag{2.52}$$

$(y \geq 0, \sigma > 0)$, $Y \sim LN(\mu_{LN}, \sigma_{LN}^2)$, $\log(Y) \sim N(\mu_N, \sigma_N^2)$ with the relationships

$$\mu_N = \log\left(\frac{\mu_{LN}^2}{\sqrt{\mu_{LN}^2 + \sigma_{LN}^2}}\right) \tag{2.53}$$

$$\sigma_N^2 = \log\left(\frac{\mu_{LN}^2 + \sigma_{LN}^2}{\mu_{LN}^2}\right) \tag{2.54}$$

and conversely

$$\mu_{LN} = \exp\left(\mu_N + \frac{\sigma_N^2}{2}\right) \tag{2.55}$$

$$\sigma_{LN}^2 = \exp\left(2\mu_N + \sigma_N^2\right)\left(\exp\sigma_N^2 - 1\right). \tag{2.56}$$

2.2.4.4 The Skew Normal Distribution

A further promising candidate to describe performance parameters is the skew normal distribution. Its PDF is given by

$$\begin{aligned} f_Y(y) &= \frac{2}{c}\varphi\left(\frac{y-b}{c}\right)\Phi\left(a\left(\frac{y-b}{c}\right)\right) \\ &= \frac{1}{c\sqrt{2\pi}}\exp\left(-\frac{(y-b)^2}{2c^2}\right)\left(1 + erf\left(a\frac{y-b}{\sqrt{2}c}\right)\right), \end{aligned} \tag{2.57}$$

with $-\infty < y < +\infty, c > 0$, see Figs. 2.8–2.10. Its moments are given in Table 2.5.

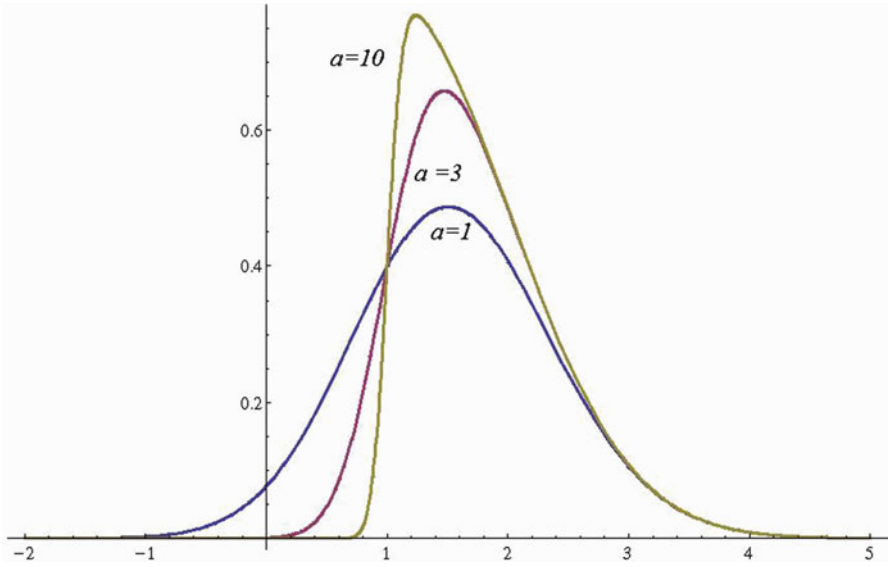


Fig. 2.8 The probability density function of skew normal distribution for $b = c = 1$ and various a

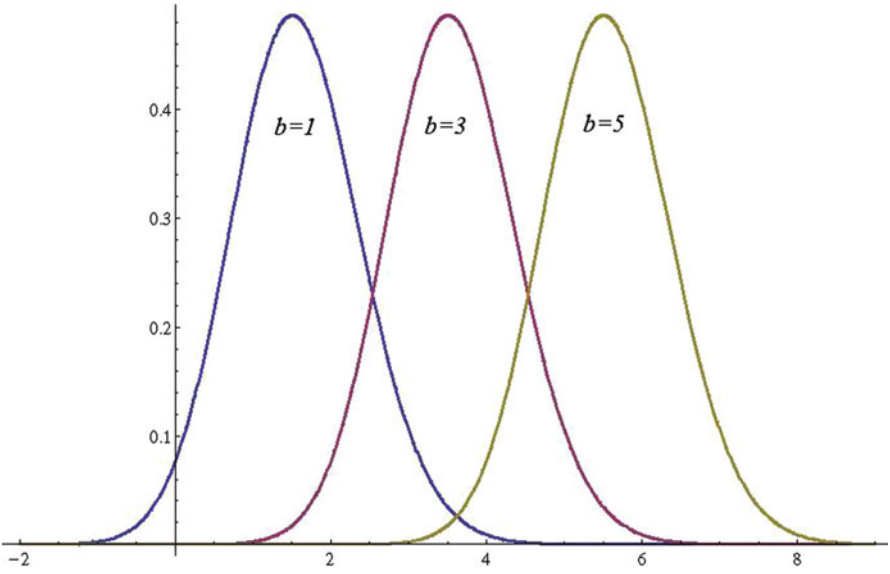


Fig. 2.9 The probability density function of skew normal distribution for $a = c = 1$ and various b

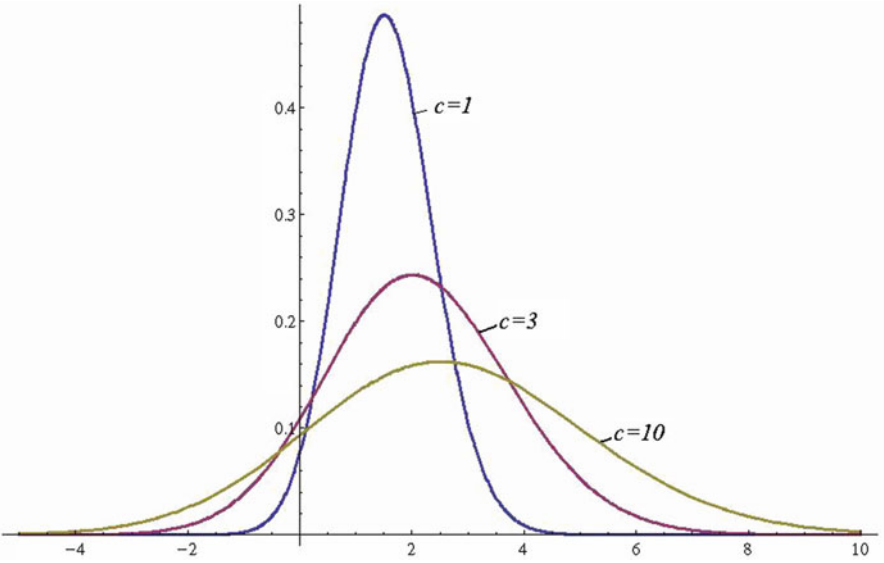


Fig. 2.10 The probability density function of skew normal distribution for $a = b = 1$ and various c

2.2.4.5 General Approximations of the Distribution Based on Moments

Further suitable families of distribution functions are the generalized λ-distribution (GLD) and the generalized β-distribution (GBD), cf. [54]. They are characterized by four parameters. An approximation requires no further assumptions than the knowledge of the first four moments (Table 2.6).

For the GLD, the probability density function is

$$f_Y(z) = \frac{\lambda_2}{\lambda_3 y^{\lambda_3 - 1} + \lambda_4 (1-y)^{\lambda_4 - 1}}, \quad \text{at } z = Q(y), \tag{2.58}$$

for $\lambda_3, \lambda_4 > -\frac{1}{4}$, where $Q(y)$ denotes the quantile function

$$Q(y) = \lambda_1 + \frac{y^{\lambda_3} - (1-y)^{\lambda_4}}{\lambda_2}, \quad 0 \le y \le 1, \tag{2.59}$$

that means $f_Y(z)$ is not given explicitly. Iteratively solving a nonlinear equation system containing integral expressions of Euler's beta function yields an approximation for the $GLD(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$.

Its broad variety is the essential advantage of the GLD that can approximate many different distributions.

Disadvantages of the GLD consists in the disability of solutions for

$$\lambda_3, \lambda_4 < -\frac{1}{4} \quad \text{and} \quad 1 + \gamma_1^2 \le \gamma_2 + 3 \le 1.8(1 + \gamma_1^2). \tag{2.60}$$

Table 2.5 Characteristics of normal, lognormal, and skew normal distribution, absolute moments $m_1, \ldots, m_4$, central moments $\mu_2, \ldots, \mu_4$, skewness γ_1 and kurtosis γ_2

Char.	normal(μ, σ^2)	lognormal(μ, σ^2)	Skew normal (2.57)
EY	μ	$\mathrm{e}^{\mu+\frac{1}{2}\sigma^2}$	$b+\sqrt{\frac{2}{\pi}}\frac{ac}{\sqrt{1+a^2}}$
EY^2	$\mu^2+\sigma^2$	$\mathrm{e}^{2\mu+2\sigma^2}$	$b^2+\sqrt{\frac{8}{\pi}}\frac{abc}{\sqrt{1+a^2}}+c^2$
EY^3	$\mu^3+3\mu\sigma^2$	$\mathrm{e}^{3\mu+\frac{9}{2}\sigma^2}$	$b^3+3bc^2+\sqrt{\frac{2}{\pi}}\frac{ac}{\sqrt{1+a^2}}\left(3b^2+3c^2-\frac{a^2c^2}{1+a^2}\right)$
EY^4	$\mu^4+6\mu^2\sigma^2+3\sigma^4$	$\mathrm{e}^{4\mu+8\sigma^2}$	$\begin{cases} b^4+6b^2c^2+3c^4+ \\ +4\sqrt{\frac{2}{\pi}}\frac{abc}{\sqrt{1+a^2}}\left(b^2-\frac{a^2c^2}{1+a^2}+3c^2\right) \end{cases}$
μ_2	σ^2	$(\mathrm{e}^{\sigma^2}-1)\mathrm{e}^{2\mu+\sigma^2}$	$c^2\left(1-\frac{2}{\pi}\frac{a^2}{1+a^2}\right)$
μ_3	0	$\mathrm{e}^{3\mu+\frac{3}{2}\sigma^2}\left(\mathrm{e}^{3\sigma^2}-3\mathrm{e}^{\sigma^2}+2\right)$	$\sqrt{\frac{2}{\pi}}\frac{4-\pi}{\pi}\frac{a^3c^3}{(1+a^2)^{3/2}}$
μ_4	$3\sigma^4$	$\mathrm{e}^{4\mu+2\sigma^2}\left(\mathrm{e}^{6\sigma^2}-4\mathrm{e}^{3\sigma^2}+6\mathrm{e}^{\sigma^2}-3\right)$	$\begin{cases} \left(\frac{2}{\pi}\frac{c^2}{1+a^2}\right)^2 \\ \times\left(2(\pi-3)a^4+3\left(\frac{\pi}{2}(1+a^2)-a^2\right)^2\right) \end{cases}$
γ_1	0	$\left(\mathrm{e}^{\sigma^2}+2\right)\sqrt{\mathrm{e}^{\sigma^2}-1}$	$\frac{4-\pi}{2}\frac{a^3}{\left(\frac{\pi}{2}(1+a^2)-a^2\right)^{3/2}}$
γ_2	0	$\mathrm{e}^{4\sigma^2}+2\mathrm{e}^{3\sigma^2}+3\mathrm{e}^{2\sigma^2}-6$	$2(\pi-3)\frac{a^4}{\left(\frac{\pi}{2}(1+a^2)-a^2\right)^2}$

Table 2.6 Several standard distribution functions approximated by the GLD, cf. [54]

Distribution	λ_1	λ_2	λ_3	λ_4
Normal $N(0,1)$	0	0.1975	0.1349	0.1349
Lognormal, $\mu = 0, \sigma = \frac{1}{3}$	0.8451	0.1085	0.01017	0.03422
$\chi^2, \theta = 3$	0.8596	0.0095443	0.002058	0.02300
Exponential, $\theta = 1$	0.006862	−0.0010805	$-4.072 \cdot 10^{-5}$	−0.001076

A further suitable class of distributions is the generalized Beta distribution ($GBD(\beta_1, \beta_2, \beta_3, \beta_4)$). Its PDF is given by

$$f_Y(y) = \begin{cases} \dfrac{(y-\beta_1)^{\beta_3}(\beta_1+\beta_2-y)^{\beta_4}}{\beta(\beta_3+1,\beta_4+1)\beta_2^{\beta_3+\beta_4+1}}, & \beta_1 \le y \le \beta_1+\beta_2 \\ 0, & \text{otherwise,} \end{cases} \tag{2.61}$$

where $\beta(a,b)$ denotes Euler's beta function.

Knowing the sample moments $\mathrm{E}Y$, $\mu_2(Y)$, $\mu_3(Y)$, $\mu_4(Y)$, the parameters $\beta_1, \ldots, \beta_4$ are obtained by iterative solving a nonlinear equation system.

The introduced methods describe the distributional behavior of the performance parameters only, they do not incorporate the importance of the process parameters.

2.2.5 Methods to Reduce the Complexity

2.2.5.1 Principal Component Analysis (PCA)

The principal component analysis (PCA) is traditionally based on the spectral decomposition of the covariance matrix Σ of a random vector. The objective of the PCA is to transform a number of possibly correlated random variables into a smaller number of uncorrelated random variables. These uncorrelated variables are called principal components. This shall be shortly figured out.

We assume that an m-dimensional random vector $\underline{X}$ is given with mean value $E[\underline{X}] = \underline{\mu}$ and the symmetric covariance matrix

$$\Sigma = E\left[\left(\underline{X} - \underline{\mu}\right) \cdot \left(\underline{X} - \underline{\mu}\right)^T\right]. \tag{2.62}$$

Let $\lambda_1, \lambda_2, \cdots, \lambda_m$ denote the eigenvalues of Σ. In general, Σ is positive semidefinite, that means, the λ_i, $i = 1, 2, \ldots, m$ are nonnegative real numbers. Σ can be decomposed by use of eigendecomposition as follows

$$\Sigma = \boldsymbol{U} \cdot \boldsymbol{\Lambda} \cdot \boldsymbol{U}^T, \tag{2.63}$$

where $\boldsymbol{U}$ is the square matrix of orthonormalized eigenvectors of Σ with $\boldsymbol{U} \cdot \boldsymbol{U}^T = \boldsymbol{I}_m$ and $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_m)$ is a diagonal matrix with positive eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m \geq 0$ of the covariance matrix. Equal values $\lambda_i = \lambda_{i+1}$ and zeros $\lambda_j = 0$ are theoretically possible.

Thus, the random vector $\underline{X}$ can be represented by

$$\underline{X} = \underline{\mu} + \boldsymbol{U} \cdot \boldsymbol{\Lambda}^{\frac{1}{2}} \cdot \underline{Z}, \tag{2.64}$$

where $\underline{Z} = (Z_1, Z_2 \cdots, Z_m)^T$ is an m-dimensional random vector that is built up by uncorrelated random variables Z_i with mean value 0 and variance 1 and the matrix $\boldsymbol{\Lambda}^{\frac{1}{2}} = \text{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \cdots, \sqrt{\lambda_m})$. It can easily be shown that mean value and applying (2.62) the covariance matrix of the random vector given by (2.64) equal $\underline{\mu}$ and Σ, resp.

If we now only consider the first dominant m' eigenvalues, we can approximate $\underline{X}$ by

$$\underline{X} \approx \underline{\tilde{X}} = \underline{\mu} + \tilde{\boldsymbol{U}} \cdot \tilde{\boldsymbol{\Lambda}}^{\frac{1}{2}} \cdot \underline{\tilde{Z}}, \tag{2.65}$$

where $\underline{\tilde{Z}} = (\tilde{Z}_1, \tilde{Z}_2 \cdots, \tilde{Z}_{m'})^T$ is an m'-dimensional random vector that is built up by uncorrelated random variables $\tilde{Z}_i$ with mean value 0 and variance 1. $\tilde{\boldsymbol{\Lambda}}^{\frac{1}{2}} = \text{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \cdots, \sqrt{\lambda_{m'}})$ is a diagonal matrix and $\tilde{\boldsymbol{U}}$ the matrix of the associated eigenvectors with m rows and m' columns. Thus, we approximate $\underline{X}$ with a fewer number m' of random variables. We only consider the principal contributors to the variances of the components of $\underline{X}$. However, the variance of the components of $\underline{X}$ is nearly the same as the variance of the components of $\underline{\tilde{X}}$ depending on the number m' of eigenvalues that are taken into consideration.

That means, the PCA is a method to project an m-dimensional space to a smaller m'-dimensional one, $m' < m$. The vector

$$\underline{Y} = \boldsymbol{\Lambda}^{\frac{1}{2}} \cdot \underline{Z} \tag{2.66}$$

in (2.64) forms the principal components. Thus, knowing the matrix $\boldsymbol{U}$ of normalized eigenvectors of the covariance matrix Σ, the components Y_i of the transformed random variable $\underline{Y} = \boldsymbol{U}^T \cdot (\underline{X} - \underline{\mu})$ are denoted as principal components. In other terms, the principal components are linear combinations of the original random variables, for instance the process parameters. Based on the properties of the Euclidian norm, it can be shown that the total variance of the original and transformed variables are equal,

$$\sum_{i=1}^{m} \sigma^2(Y_i) = \sum_{i=1}^{m} \sigma^2(X_i). \tag{2.67}$$

Considering (2.66), the eigenvalues of the covariance matrix of the original variables indicate the contribution of the principal component Y_i to the total variance, e.g.,

$$\sigma^2(Y_i) = \frac{\lambda_i}{\sum_{j=1}^{m} \lambda_j} \sum_{j=1}^{m} \sigma^2(X_j). \tag{2.68}$$

The principal components are orthogonal among themselves, that means, they are uncorrelated at all. There is no determination regarding the number m' of principal components, it can be chosen individually. The signal-to-noise ratio is given by :

$$SNR = \frac{\sigma^2_{\text{signal}}}{\sigma^2_{\text{noise}}} = \frac{\sigma^2(Y_1) + \ldots + \sigma^2(Y_{m'})}{\sigma^2(Y_{m'+1}) + \ldots + \sigma^2(Y_m)}, \tag{2.69}$$

where $Y_1, \ldots, Y_{m'}$ denotes the significant principal components and $Y_{m'+1}, \ldots, Y_m$ the neglected. A great ratio $SNR \gg 1$ means a good accuracy. Thus, the PCA is a simple method to reduce the number of relevant dimensions of a relationship with a minimum loss of information, in other terms, a simplification by reducing the number of variables. The Cholesky decomposition of the covariance matrix does not offer this opportunity.

If the variances of the components of the random vector $\underline{X}$ differ much, the PCA should be carried based on the standardized random vector $\underline{X'}$. Its components are standardized to means of 0 and standard deviations of 1 at first. This can be done by the transformation

$$\underline{X'} = \boldsymbol{D}^{-1} \cdot \left(\underline{X} - \underline{\mu} \right), \tag{2.70}$$

where the matrix $\boldsymbol{D}$ is a diagonal matrix that contains the standard deviations of the components of $\underline{X}$. The covariance matrix of the standardized random vector $\underline{X'}$ equals its correlation matrix and is the same as the correlation matrix $\boldsymbol{P}$ of $\underline{X}$. Therefore, there is the following relation between the correlation and the covariance matrix of $\underline{X}$

$$\boldsymbol{P} = \boldsymbol{D}^{-1} \cdot \Sigma \cdot \boldsymbol{D}^{-1}. \tag{2.71}$$

If the PCA is based on the standardized random variables, a scaling or shifting of the process parameters does not change the results of the PCA. Therefore, PCA based on the correlation matrix is sometimes preferred.

For a simple introduction to PCA, see [55] for further details [56, 57].

It has to be mentioned that the PCA by eigendecomposition and singular value decomposition (SVD) provides (only) a representation by uncorrelated random variables. If $\underline{X}$ is a multivariate normal distributed random vector, this is also a decomposition into independent random variables. In this case, the components of the random vectors $\underline{\tilde{Z}}$ in (2.65) and (2.75) are uncorrelated (and independent) $N(0,1)$ distributed random variables. A decomposition of a random vector where several

components are far away from a normal distribution into independent random variables can be carried out by Independent Component Analysis (ICA) [58].

2.2.5.2 Complexity Reduction Based on Samples

We assume that n samples of an m-dimensional random vector are given by $\{(x_{i1}, x_{i2}, \cdots, x_{im})^T\}, i = 1, \cdots, n$. Using the estimated mean values $\hat{\mu}_i = \frac{1}{n}\sum_{j=1}^{n} x_{ij}$, we can built up a data matrix with m rows and n columns that contains the zero centered samples corrected by their mean values.

$$\boldsymbol{M} = \begin{pmatrix} x_{11} - \hat{\mu}_1 & x_{21} - \hat{\mu}_1 & \cdots & x_{n1} - \hat{\mu}_1 \\ x_{12} - \hat{\mu}_2 & x_{22} - \hat{\mu}_2 & \cdots & x_{n2} - \hat{\mu}_2 \\ \vdots & \vdots & \ddots & \vdots \\ x_{1m} - \hat{\mu}_m & x_{2m} - \hat{\mu}_m & \cdots & x_{nm} - \hat{\mu}_m \end{pmatrix}. \tag{2.72}$$

The matrix $\boldsymbol{M}$ can be used to estimate the covariance matrix of the associated random vector $\underline{X}$

$$\Sigma \approx \widehat{\Sigma} = \frac{1}{n-1} \cdot \boldsymbol{M} \cdot \boldsymbol{M}^T. \tag{2.73}$$

Two disadvantages are

- The occurrence of outliers, which can distort the results,
- Nonlinear relationships, which often cannot be identified.

Furthermore, for higher dimensions m, it can be difficult to calculate and store an eigendecomposition for $\widehat{\Sigma}$. Moving over to normalized components based on a division of the components of $\boldsymbol{M}$ by the associated standard deviation may also be recommended.

2.2.5.3 The Singular Value Decomposition (SVD)

A method in close relationship to the PCAbased on the eigendecomposition is the SVD [59]. The SVD is based on a segmentation of the $m \times n$ matrix $\boldsymbol{A} = \boldsymbol{U} \cdot \boldsymbol{S} \cdot \boldsymbol{V}^T$, where $\boldsymbol{U}$ is a $m \times m$ matrix with $\boldsymbol{U} \cdot \boldsymbol{U}^T = \boldsymbol{I}_m$, $\boldsymbol{V}$ a $n \times n$ matrix with $\boldsymbol{V} \cdot \boldsymbol{V}^T = \boldsymbol{I}_n$ and $\boldsymbol{S}$ a rectangular matrix with the same dimension as $\boldsymbol{A}$. Only the diagonal entries s_{ii} ($i \leq min(m,n)$) of $\boldsymbol{S}$ may be nonzero elements and they can be arranged in an order of decreasing magnitude $s_{11} \geq s_{22}, \ldots$. The positive diagonal elements are called singular values. Let us now decompose

$$\boldsymbol{A} = \frac{1}{\sqrt{n-1}} \cdot \boldsymbol{M} = \boldsymbol{U} \cdot \boldsymbol{S} \cdot \boldsymbol{V}^T. \tag{2.74}$$

Then we can approximate the random vector $\underline{X}$ by using only the greatest m' singular values by

$$\underline{X} \approx \underline{\hat{\mu}} + \tilde{\boldsymbol{U}} \cdot \tilde{\boldsymbol{S}} \cdot \underline{\tilde{Z}}, \tag{2.75}$$

where $\underline{\mu} = (\hat{\mu}_1, \hat{\mu}_2, \cdots, \hat{\mu}_m)^T$, $\tilde{\boldsymbol{U}}$ is a $m \times m'$ matrix, $\tilde{\boldsymbol{S}}$ is a $m' \times m'$ matrix, and $\underline{\tilde{Z}} = (\tilde{Z}_1, \tilde{Z}_2 \cdots, \tilde{Z}_{m'})^T$ is an m'-dimensional random vector that is built up by uncorrelated random variables $\tilde{Z}_i$ with mean value 0 and variance 1. For details to SVD see [57].

2.2.6 Special Problems Describing Random Variables

2.2.6.1 Inter-Die and Intra-Die Variations

We distinguish between inter-die variations and intra-die variations. Inter-die variations are constant inside a die, but variable from die to die. Intra-die variations are variable inside a die.

Inter-die variations cause a shifting of the mean values, whereas intra-die variations are spatial correlated random variables or even location invariant random variables on a die.

Differences of process parameters are caused in the fabrication process (pollution, material, and lithography defects), the environment (changes in temperature and power supply) as well as physical effects (local focused temperature fluctuations, electromigration).

With more and more shrinking structures, the intra-die variations increase.

The inter-die variation of a given process parameter can be described by

$$X_i = x_{\text{nom}} + \Delta X_{\text{inter}}, \tag{2.76}$$

where x_{nom} denotes the nominal value and ΔX is a random variation being constant for all elements of this die.

The intra-die variation of a process parameter is

$$X_i = x_{\text{nom}} + \Delta X_{\text{inter}} + \Delta X(\xi, \eta), \tag{2.77}$$

where (ξ, η) means the spatial position on the die.

An easy mathematical model is given by [60]. The intra-die variation is an uncorrelated random variable plus a spatial correlated random variable (both of them are normal distributed).

Rao et al. [71] suggest a model for leakage estimation accounting for both inter- and intra-die variations, basing on relationships (exponential function like) between leakage current and gate length. The gate length of device can be described by

$$L_{\text{total},i} = L_{\text{nom}} + \Delta L_{\text{inter}} + \Delta L_{\text{intra},j}, \tag{2.78}$$

where the leakage is given by a sum of lognormal distributed random variables. This relationship allows a split-up into the fractions of inter-die and intra-die variations. In particular, increasing intra-die variations effects a strong increasing leakage.

2.2.7 Transformation of Random Variables by Analytical Methods

2.2.7.1 Response Surface Methods

The Response Surface Method (RSM) was introduced by Box and Wilson [61]. A response surface can be thought to be a description of a physical relationship by a mathematical model. Aim of response surface techniques is to find a good approximation to describe dependencies $y \approx h(\underline{x})$. To construct such functions h in an optimal way, a sequence of so-called designed experiments has to be carried out. Also, simulation runs are of that kind.

To investigate and visualize response surfaces, so-called contour diagrams are suitable (parts of response surfaces, where one or two parameters are changed and all other are kept constant). Kinds of applications of response surfaces are

- Approximate mapping within a limited region
- Choice of operating conditions to achieve desired specifications
- Search for optimal conditions.

Candidates for response surface models are any analytical functions as for instance polynomial approximations. For a first step, linear relationships (first-order designs) often are satisfactory, for more detailed studies higher order polynomial (higher order designs), transcendental, or other special approaches can be made.

As mentioned before, model building is based on experiments in order to approximate the mapping between input and output variables of a system. Output values are determined for special sets of input values. In our case, outputs may be performance values as delay, leakage currents, and power dissipations, whereas inputs may be process parameters or temperature and supply voltages. We would like to briefly discuss the procedure how to set up and evaluate these experiments.

The objective of a designed experiment is to describe changes of the performance parameter y in relationship to changes of the process parameter vector $\underline{x}$. To investigate those relationships, all process parameters must be varied, otherwise relationships can be distorted by unrecorded hidden process parameters. A norming of input parameters guarantees an equal treatment of all process parameters. Transformations of the process parameters have the effect to changing the scale, expanding it on one part, contracting it on the other. They cause changes of the variances and further characteristics, which can be compensated by adequate weightings.

A factorial design calls a design, which is running over all combinations. With n_i levels of values of x_i, $i = 1,2,\ldots,m$, the number of test runs of the factorial design is $n_1 \cdot n_2 \cdots n_m$. In general, factorial designs estimate the response surface with great accuracy with a minimum of residual variance. Optimal designs are described for multiple response surface approaches. In many cases, an optimal design is a regular rotatable (rotational symmetric) operating design, e.g., for a first-order design (linear function) a regular m-dimensional simplex with $m+1$ vertices and for a second order design (quadratic function) an m-dimensional cube with 2^m vertices.

With increasing number m of input parameters response surface approaches will be more and more expensive. A possibility of simplification is the reduction of the number of process parameters that considerably influence the result by a correlation analysis. Let $\rho(Z_1,Z_2)$ be the linear correlation coefficient between two random variables Z_1 and Z_2, all those process parameters x_i can be omitted, where the absolute value of $\rho(X_i,Y)$ does not exceed a given threshold value.

In order to make all process parameters X_i equitable, a standardization is useful:

$$X_i' = \frac{X_i - \mu_i}{\sigma_i}, \tag{2.79}$$

where μ_i and σ_i are mean and standard deviation of the original unstandardized random variable X_i. From now, all X_i' are of mean $\mu_i' = 0$ and standard deviation $\sigma_i' = 1$. If they are normally distributed, 99.73% of X_i are inside the interval $[-3,+3]$, called the 3σ-limit.

To compare different response surface approaches, an iterative approach is suggested. It helps to select an appropriate approximation – linear functions, nonlinear with or without coupled terms, or more complicated analytical functions. The coefficients are determined via the least square method by minimizing the quadratical errors. The empirical residual variance

$$\sigma_{\text{Res}}^2 = \frac{1}{n-m-1} \sum_{i=1}^{n} \left(y_i - h(\underline{x_i})\right)^2 \tag{2.80}$$

shows the goodness of different approaches. The relationship with the smallest σ_{res} will be the best.

Extensions and special cases are studied in many papers.

The so-called "black-box model," its meaning and working techniques for polynomial approaches with noncoupled or coupled terms are introduced in [62].

A weighted least square method, which calculates sensitivities additionally to the response surface is studied in [63].

An optimal design via response surface analysis by discussion of coupled terms, orthogonality and rotatability is introduced in [64].

An extension to nonnormally distributed and coupled characteristics, including the calculation of higher order moments is made in [65].

Response surface techniques are basic tools to investigate the statistical behavior of any performance parameter y.

2.2.7.2 Linear Models

Let us assume that the RSM delivers us a linear (more exact affine) dependency between the input parameters $\underline{x}$ and the performance value y. That means,

$$h : \mathscr{D} \subset \mathbb{R}^m \longrightarrow \mathbb{R}, \qquad \underline{x} \mapsto y = y_{\text{nom}} + \underline{a}^T \cdot (\underline{x} - \underline{\mu}) = y_{\text{nom}} + \sum_{i=1}^{m} a_i \cdot (x_i - \mu_i) \,. \tag{2.81}$$

Equation (2.81) is similar to a Taylor series expansion around $\underline{x}_{\text{nom}} = \underline{\mu} \in \mathscr{D} \subset \mathbb{R}^m$ with the function value y_{nom}. The components of $\underline{a}$ can be determined by the parameter sensitivities at the operating point. However, it might often be better to determine them via the difference of the performance values for different $\underline{x}$ values. This might give a better approximation for the whole domain $\mathscr{D}$ of the function h.

The linear approach (2.81) offers the opportunity to study the random characteristics of the performance variable $\underline{Y} = h(\underline{X})$ in special cases by analytical methods. We assume that $\underline{X}$ is $N(\underline{\mu}, \Sigma)$ multivariate normally distributed as described by (2.38). Thus, it follows

$$E[Y] = y_{\text{nom}} = \underline{a}^T \cdot \underline{\mu}. \tag{2.82}$$

If we use the representation $\underline{X} = \underline{\mu} + \boldsymbol{G} \cdot \underline{Z}$, where $\Sigma = \boldsymbol{G} \cdot \boldsymbol{G}^T$ is segmented by a Cholesky decomposition and $\underline{Z}$ is $N(\underline{0}, \boldsymbol{I}_m)$ distributed (see (2.40)) we get

$$E\left[(Y - y_{\text{nom}})^2\right] = \sigma_Y^2 = E\left[\left(\underline{a}^T \cdot \boldsymbol{G} \cdot \underline{Z}\right)^2\right] = \underline{a}^T \cdot \boldsymbol{G} \cdot \boldsymbol{I}_m \cdot \boldsymbol{G}^T \cdot \underline{a} = \underline{a}^T \Sigma \underline{a}. \tag{2.83}$$

That means, Y is $N\left(y_{\text{nom}}, \underline{a}^T \Sigma \underline{a}\right)$ distributed.

Sum of n Uncorrelated Normal Distributed Random Variables

The last relation was already used in the Sect. 1.3.1 in order to discuss the consequences of inter-die and intra-die variations. Let $\underline{X}$ be a n dimensional $N((\mu, \mu, \cdots, \mu)^T, \sigma^2 \cdot \boldsymbol{I}_n)$ distributed random vector. Its n components are uncorrelated and $N(\mu, \sigma^2)$ distributed. Using (2.82) and (2.83) and $\underline{a}^T = (1, 1, \cdots, 1)^2$, we see that the sum of n uncorrelated and $N(\mu, \sigma^2)$ variables is $N(n \cdot \mu, n \cdot \sigma^2)$ distributed. This fundamental result will also be used in Sect. 5.3.2.2 to characterize strings of resistors. The situation is more complicated if h is not a linear map. Sophisticated solutions can be found for special cases. Let $y = h(\underline{x}) = \sum_{i=1}^{m} x_i^2$ and X_i independent standard normal distributed variables $X_i \sim N(0, 1)$. Then Y is χ^2-distributed with m degrees of freedom. Further rules can be established for products and other complicated relationships. The distribution of the ratio of random variables will be discussed in Sect. 5.3.2.

Determination of the Worst Case Point in the Case of Linear Models

A worst case point $\underline{x}^{\mathrm{wc}}$ is the most likely parameter set at which the performance of interest is exactly at the specification limit y^{wc} under worst-case operating conditions. Let h describe the relation between performance values and process parameters under worst case operating conditions and $f_{\underline{X}}$ the PDF of $\underline{X}$. The following relations have to be fulfilled

$$f_{\underline{X}}\left(\underline{x}^{\mathrm{wc}}\right) \rightarrow \max \tag{2.84}$$

$$h(\underline{x}^{\mathrm{wc}}) = y^{\mathrm{wc}}. \tag{2.85}$$

For the linear model using (2.81), we can also analytically determine the worst case points. Considering the PDF of the multivariate normal distribution (2.38), the worst case point $\underline{x}^{\mathrm{wc}}$ has to fulfill the following conditions

$$\left(\underline{x}^{\mathrm{wc}} - \underline{\mu}\right)^T \boldsymbol{\Sigma}^{-1} \left(\underline{x}^{\mathrm{wc}} - \underline{\mu}\right) \rightarrow \min \tag{2.86}$$

$$y_{\mathrm{nom}} + \underline{a}^T \cdot \left(\underline{x}^{\mathrm{wc}} - \underline{\mu}\right) = y^{\mathrm{wc}}. \tag{2.87}$$

Using the substitution $\underline{x}^{\mathrm{wc}} = \underline{\mu} + \boldsymbol{G} \cdot \underline{z}^{\mathrm{wc}}$, we can formulate the equivalent problem

$$(\underline{z}^{\mathrm{wc}})^T \cdot \underline{z}^{\mathrm{wc}} \rightarrow \min \tag{2.88}$$

$$\underline{a}^T \cdot \boldsymbol{G} \cdot \underline{z}^{\mathrm{wc}} = y^{\mathrm{wc}} - y_{\mathrm{nom}}. \tag{2.89}$$

The second equation (2.89) represents a line and the first one (2.88) measures the shortest distance between the origin of the coordinate system and the line. Thus, $\underline{z}^{\mathrm{wc}}$ must be a multiple of $(\underline{a}^T \cdot \boldsymbol{G})^T = \boldsymbol{G}^T \cdot \underline{a}$. Finally, we get

$$\underline{x}^{\mathrm{wc}} = \underline{\mu} + \frac{y^{\mathrm{wc}} - y_{\mathrm{nom}}}{\underline{a}^T \boldsymbol{\Sigma} \underline{a}} \cdot \boldsymbol{\Sigma} \cdot \underline{a} \tag{2.90}$$

for the worst case point. The norm of the vector $\underline{x}^{\mathrm{wc}} - \underline{\mu}$ is called worst case distance.

Figure 2.11 demonstrates the situation described by (2.86)-(2.90). We can give a geometric interpretation for this figure. The worst case point is that point where the line in the domain region of h that belongs to the specification limit y^{wc} touches an m-ellipsoid given by (2.39).

We have to distinguish between a worst case point and a corner case point. The corner case point is usually determined by deflecting all components of the random vector X by an arbitrary fraction of the respective standard deviation toward the specification limit. That means, a corner case point is given by

$$\underline{x}^{\mathrm{cc}} = \underline{\mu} + \nu \cdot \sqrt{\tilde{\boldsymbol{\Sigma}}} \cdot \mathrm{sign}(\underline{a}) \cdot \mathrm{sign}(y^{\mathrm{wc}} - y_{\mathrm{nom}}), \tag{2.91}$$

where $\tilde{\boldsymbol{\Sigma}}$ is the matrix that only contains the diagonal elements of the covariance matrix $\boldsymbol{\Sigma}$. All other elements are zero.

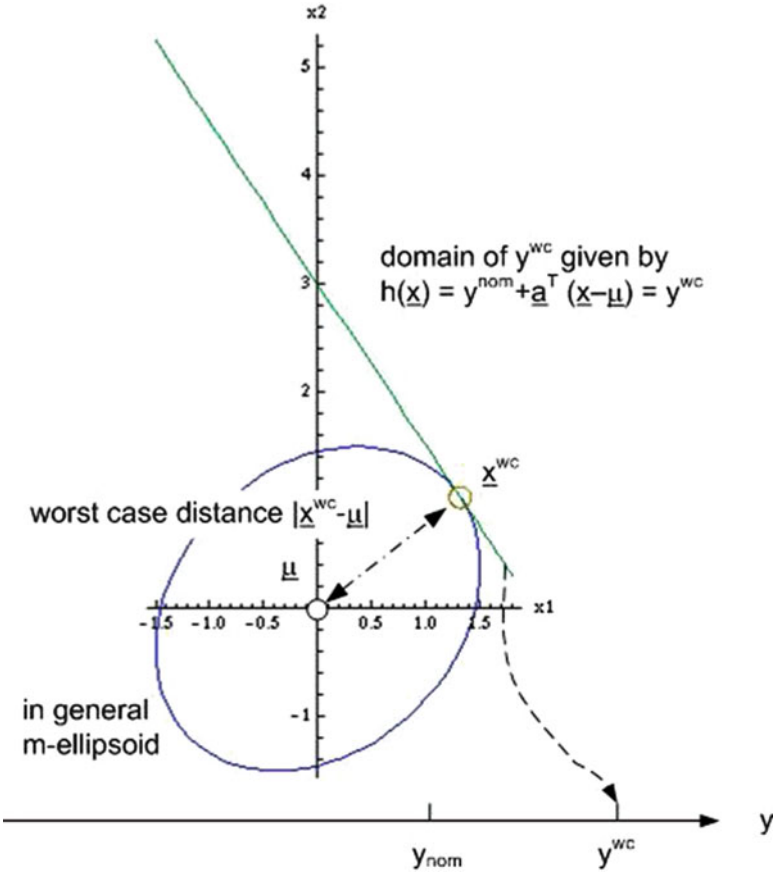


Fig. 2.11 Example of the position of a worst point $\underline{x}^{\mathrm{wc}}$ for a linear model

2.2.7.3 Second-Order Models

Another special case is the description of the response surface by a quadratic function. Such a function can be represented by

$$h : \mathscr{D} \subset \mathbb{R}^m \longrightarrow \mathbb{R}, \qquad \underline{x} \mapsto y = y_{\mathrm{nom}} + \underline{a}^T \cdot (\underline{x} - \underline{\mu}) + (\underline{x} - \underline{\mu})^T \cdot \boldsymbol{B} \cdot (\underline{x} - \underline{\mu}) \quad (2.92)$$

$\underline{a}$ and $\boldsymbol{B}$ are an m-dimensional real vector and an $m \times m$ real-valued symmetric matrix, respectively. This special case offers the opportunity to determine the moments of a performance variable $\underline{Y} = h(\underline{X})$ in an easy way if $\underline{X}$ is a $N(\underline{\mu}, \boldsymbol{\Sigma})$ multivariate normally distributed random vector. The idea behind is to transform h in such a manner that $\underline{Y}$ is the sum of independent random variables. We will briefly figure out how this transformation can be carried out.

If we use the representation $\underline{X} = \underline{\mu} + \boldsymbol{G} \cdot \underline{Z}$ where $\Sigma = \boldsymbol{G} \cdot \boldsymbol{G}^T$ and $\underline{Z}$ is $N(\underline{0}, \boldsymbol{I}_m)$ distributed (see (2.40))[3], we get

$$Y = y_{\text{nom}} + \underline{a}^T \cdot \boldsymbol{G} \cdot \underline{Z} + \underline{Z}^T \cdot \boldsymbol{G}^T \cdot \boldsymbol{B} \cdot \boldsymbol{G} \cdot \underline{Z}. \tag{2.93}$$

$\boldsymbol{G}^T \cdot \boldsymbol{B} \cdot \boldsymbol{G}$ is also a symmetric matrix with a rank $r \leq m$. A spectral decomposition delivers $\boldsymbol{G}^T \cdot \boldsymbol{B} \cdot \boldsymbol{G} = \boldsymbol{P} \cdot \boldsymbol{D} \cdot \boldsymbol{P}^T$, where $\boldsymbol{D} = \text{diag}(\lambda_1, \cdots, \lambda_r, 0, \cdots, 0) \in \mathbb{R}^{m \times m}$ is a diagonal matrix that contains the eigenvalues. $\boldsymbol{P} \cdot \boldsymbol{P}^T = \boldsymbol{I}_m$ with $\boldsymbol{P} \in \mathbb{R}^{m \times m}$. is built up by all normalized eigenvectors that are pairwise orthogonal. We can now introduce the substitutions $\tilde{\underline{Z}} = \boldsymbol{P}^T \cdot \underline{Z}$ and $\underline{Z} = \boldsymbol{P} \cdot \tilde{\underline{Z}}$ and get with $\tilde{\underline{a}} = \boldsymbol{P}^T \cdot \boldsymbol{G}^T \cdot \underline{a}$ and completing the square sum expressions in (2.95)

$$Y = y_{\text{nom}} + \tilde{\underline{a}}^T \cdot \tilde{\underline{Z}} + \tilde{\underline{Z}}^T \cdot \boldsymbol{D} \cdot \tilde{\underline{Z}} \tag{2.94}$$

$$Y = y_{\text{nom}} + \sum_{i=1}^{r} (\tilde{a}_i \cdot \tilde{\underline{Z}}_i + \lambda_i \cdot \tilde{\underline{Z}}_i{}^2) + \sum_{i=r+1}^{m} \tilde{a}_i \cdot \tilde{\underline{Z}}_i \tag{2.95}$$

$$Y = y_{\text{nom}} - \sum_{i=1}^{r} \frac{\tilde{a}_i^2}{4\lambda_i} + \sum_{i=1}^{r} \lambda_i \cdot \left(\tilde{\underline{Z}}_i + \frac{\tilde{a}_i}{2\lambda_i} \right)^2 + \sum_{i=r+1}^{m} \tilde{a}_i \cdot \tilde{\underline{Z}}_i. \tag{2.96}$$

The random vector $\tilde{\underline{Z}} = \boldsymbol{P}^T \cdot \underline{Z}$ is also normal distributed with mean value $E[\underline{X}] = \underline{0}$ and covariance matrix $E\left[\tilde{\underline{Z}} \cdot \tilde{\underline{Z}}^T\right] = E\left[\boldsymbol{P}^T \underline{Z} \cdot (\boldsymbol{P}^T \underline{Z})^T\right] = \boldsymbol{P}^T \cdot \boldsymbol{I}_m \cdot \boldsymbol{P} = \boldsymbol{I}_m$. That means $\tilde{\underline{Z}}$ is also $N(\underline{0}, \boldsymbol{I}_m)$ distributed. As a consequence, it follows that $\tilde{\underline{Z}}$ is built up by a constant and a sum of independent random variables. This makes it easy to determine not only $E[Y] = y_{\text{nom}} + \sum_{i=1}^{r} \lambda_i$ but also higher order moments of Y.

This characteristic can be used to determine marginal probabilities at the tail of a distribution using the saddle-point method [66].

2.2.7.4 Higher-Order Polynomial Models and Central Moment Calculation Method

Knowing a polynomial relationship between the performance parameter Y and the process parameters $X_1, X_2, \ldots, X_n$ in the circumference of a working point $\underline{x}_0$.

Model Assumptions

The probability density function of each process parameter is assumed to be symmetrically with respect to the $\underline{x}_0$. Therefore,

$$EX_i = \bar{x}_i = \underline{x}_{0i}. \tag{2.97}$$

[3] PCA can be applied in the same manner.

The symmetry causes that the central moments for each X_i of odd order are

$$\mu_k(X_i) = 0,\ k = 1,3,5,\ldots, \tag{2.98}$$

and the knowledge of the second, fourth, ..., central moments (the even ordered) is assumed, as well as the independence of the process parameters, e.g., there is no correlation between X_i and X_j, $i \neq j$. The central moments of odd order are especially zero for normal distributed random variables (see also Table 2.5).

Then the corresponding moments of Y given by

$$EY^k = E\left(a_0 + \sum_{i=1}^{m} a_i(x_i - \bar{x}_i) + \sum_{i=1}^{m}\sum_{j=i}^{m} b_{ij}(x_i - \bar{x}_i)(x_j - \bar{x}_j)\right)^k, \tag{2.99}$$

can be calculated explicitly applying the relations recapitulated in Sect. 2.2.2.

Example.

$$y = a_0 + \sum_{i=1}^{m} a_i(x_i - \bar{x}_i) + \sum_{i=1}^{m}\sum_{j=i}^{m} b_{ij}(x_i - \bar{x}_i)(x_j - \bar{x}_j) \tag{2.100}$$

leads to

$$EY = a_0 + \sum_{i=1}^{m} b_{ii}\,\mu_2(X_i), \tag{2.101}$$

$$EY^2 = a_0^2 + \sum_{i=1}^{m}\left(2a_0 b_{ii} + a_i^2\right)\mu_2(X_i) + \sum_{i=1}^{m-1}\sum_{j=i+1}^{m} b_{ij}^2\,\mu_2(X_i)\,\mu_2(X_j) + \sum_{i=1}^{m} b_{ii}^2\,\mu_4(X_i). \tag{2.102}$$

Analogously, the higher moments of Y can be calculated. This approach can be made for higher order polynomials equivalently. If a relationship $y = h(\underline{x})$ and the moments of the process parameters X_i are known, the central moment calculation method allows a simple evaluation of the moments of the performance parameter Y, see Zhang et al. [67].

2.2.7.5 Analyzing Models by Numerical Calculations

Knowing the joint probability density function of the process parameters $f_X(x_1, x_2, \ldots x_m)$ and the relationship describing the response surface $y = h(x_1, x_2, \ldots, x_m)$, we need to determine the distributional characteristics of the process parameter Y.

The CDF is given by the integral formula

$$F_Y(y_0) = P(Y < y_0) = \underbrace{\int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty}}_{h(\underline{x})<y_0,\, m\ integrals} f_X(x_1, x_2, \ldots, x_m)\, \mathrm{d}x_1 \mathrm{d}x_2 \ldots \mathrm{d}x_m, \quad (2.103)$$

or transforming the integral by use of the Jacobian

$$|J| = \left| \frac{\partial(x_1, \ldots, x_m)}{\partial(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_m)} \right| = \left| \frac{\partial x_i}{\partial y} \right|, \quad (2.104)$$

$$F_Y(y_0) = \int_{-\infty}^{y_0} \underbrace{\int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty}}_{m-1\ integrals} f_X(x_1, \ldots, x_{i-1}, x_i(y), x_{i+1}, \ldots, x_m)\, |J|\, \mathrm{d}x_1 \ldots \mathrm{d}x_{i-1} \mathrm{d}x_{i+1} \ldots \mathrm{d}x_m \mathrm{d}y, \quad (2.105)$$

and analogously the corresponding PDF

$$f_Y(y_0) = \underbrace{\int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty}}_{m-1\ integrals} f_X(x_1, \ldots, x_{i-1}, x_i(y_0), x_{i+1}, \ldots, x_m)\, |J|_{y=y_0} \times \mathrm{d}x_1 \ldots \mathrm{d}x_{i-1} \mathrm{d}x_{i+1} \ldots \mathrm{d}x_m, \quad (2.106)$$

which can be evaluated by numerical integration. For a fast and efficient numerical evaluation, the Gauss integration procedure is suggested.

If there is a large number of process parameters, the numerical calculation will be more and more time expensive and inaccurate.

A further possibility is an imitation of the Monte Carlo simulation. Knowing the relationship $y = h(\underline{x})$, we can calculate the values y_j corresponding to the configurations $x_{1,j}, x_{2,j}, \ldots, x_{m,j}$, $j = 1, 2, \ldots, n$ without a simulation run for a large number n. Knowing the residual variance σ^2_{Res} of the response surface, the value y_j can be adapted by addition of a normal distributed random variable $Z \sim N(0, \sigma_{\mathrm{Res}})$.

2.2.8 *Transformation of Random Variables by Numerical Methods*

2.2.8.1 Basic Concepts of Monte Carlo Simulation

The Monte Carlo simulation is a method for uncertainty propagation, where the goal is to determine how random variations, lack of knowledge, or errors affects

the sensitivity, performance, or reliability of a circuit, cell, chip or system that is modeled. Monte Carlo simulation is categorized as a sampling method because the inputs are randomly generated from probability distributions to simulate the process of sampling from an actual population. So we try to choose a distribution for the input data that best represents our current state of knowledge. Practical hints how to generate random number for several distributions can be found in [68] and the Appendices B.1 and B.2. The results generated from the simulation can be represented as probability distributions (or histograms) or converted to error bars, reliability predictions, tolerance zones, etc.

Monte Carlo simulation is a random experiment, applied if an analytical description of the system seems to be hardly or not possible. Simulations of integrated circuits (transistors, library cells, chips, etc.) are among this category.

To obtain sufficiently many values $y_i = h\left(\underline{x_i}\right)$ for the performance value of interest, a great number n of simulations has to be made via suitable software tools, where the process parameters $\underline{x_i}$ are random samples considering their probability distribution. The determined values y_i impact an impression on the probability distribution of the random variable Y. To carry out an appropriate number n of simulation runs, we check the confidence of the simulation results.

As a result of the Monte Carlo simulation runs, we can estimate the expected value of Y by (2.35).

The estimated $\hat{\mu}$ value is itself a sample of a random variable $\hat{M}$. This random variable is the nth part of the sum of n independent random variables Y_i with the same distribution as Y. Thus, the mean value of $\hat{M}$ is $E[Y]$. If we know the standard deviation σ of Y, then the standard deviation of $\hat{M}$ equals $\frac{\sqrt{n\cdot\sigma^2}}{n} = \frac{\sigma}{\sqrt{n}}$.

The precision of the sample means improves with the square root of the sample size. This is called "Square-Root Law."

Because of the Central Limit Theorem, the sum and also the nth part of the sum will converge against a normal distribution. Thus, based on (2.30), we get ($z > 0$)

$$\text{Prob}\left(\left|\frac{\hat{\mu} - E[Y]}{\frac{\sigma}{\sqrt{n}}}\right| \le z_{1-\alpha/2}\right) = \Phi\left(z_{1-\alpha/2}\right) - \Phi\left(-z_{1-\alpha/2}\right) = \text{erf}\left(\frac{z_{1-\alpha/2}}{\sqrt{2}}\right) = 1-\alpha \tag{2.107}$$

with the CDF of the $N(0,1)$ distribution is Φ. α is called significance level:

$$\hat{\mu} - z_{1-\alpha/2}\cdot\frac{\sigma}{\sqrt{n}} \le E[Y] \le \hat{\mu} + z_{1-\alpha/2}\cdot\frac{\sigma}{\sqrt{n}}. \tag{2.108}$$

$E[Y]$ belongs with the probability $1-\alpha$ to the confidence interval given by (2.108). The so-called $1-\alpha/2$ quantiles or percentiles $\Phi(1-\alpha/2) = z_{1-\alpha/2}(= c)$ of the standard normal distribution and $1-\alpha$ probability values that correspond can be found in Table 2.4 and also in appropriate references. We get for instance $z_{1-\alpha/2} = 1.959\ldots$ and $1-\alpha = 0.95$.

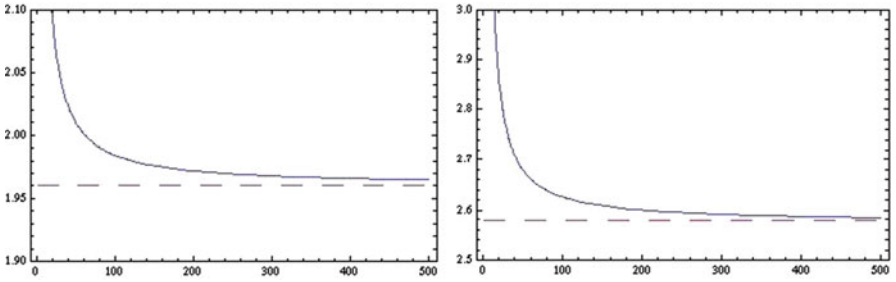


Fig. 2.12 $z_{1-\alpha/2}$ (*dashed line*) and $t_{1-\alpha/2;n-1}$ values depending on n for $1-\alpha = 0.95$ (*left*) and $1-\alpha = 0.99$

If the standard deviation σ is unknown, it can be estimated using (see (2.36))

$$\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \hat{\mu})^2. \tag{2.109}$$

Substituting Φ in (2.107) by the CDF of Student's t distribution with $n-1$ degrees of freedom it can be shown that

$$\text{Prob}\left(\left|\frac{\hat{\mu} - E[Y]}{\frac{\hat{\sigma}}{\sqrt{n}}}\right| \leq t_{1-\alpha/2;N-1}\right) = I\left(\frac{t^2_{1-\alpha/2;n-1}}{n-1+t^2_{1-\alpha/2;n-1}}; \frac{1}{2}, \frac{n-1}{2}\right) = 1-\alpha, \tag{2.110}$$

where $I(z;a,b)$ is the regularized Beta function. $t_{1-\alpha/2;n-1}$ is also known as the $1-\alpha/2$ (one-sided) quantile of Student's t distribution with $n-1$ degrees of freedom. Thus, $E[Y]$ belongs with the probability $1-\alpha$ to the interval given by

$$\hat{\mu} - t_{1-\alpha/2;n-1}\cdot\frac{\hat{\sigma}}{\sqrt{n}} \leq E[Y] \leq \hat{\mu} + t_{1-\alpha/2;n-1}\cdot\frac{\hat{\sigma}}{\sqrt{n}}. \tag{2.111}$$

For the probabilities $1-\alpha = 0.95$ and $1-\alpha = 0.99$, the Fig. 2.12 represents $z_{1-\alpha/2}$ and $t_{1-\alpha/2;n-1}$ values depending on the number n of simulation runs.

It follows from (2.107) and (2.110) that for a given significance level the number n of required simulation runs only depends on the standard deviation of the observed random performance variable Y and not of the number of random parameters X_i. For values greater than $30\ldots100$, there is only a small difference between the z and the t curves. That means for greater values (2.108) describes the confidence interval for known as well as for the estimated standard deviation of Y. To check the confidence interval using (2.108) the evaluated samples y_i must be generated independently. If this is not the case, other methods to determine the confidence interval must be applied. Bootstrap methods are, for instance, recommended in such cases. They are especially of interest, when the characteristic under investigation depends on the

probability distribution of Y. We assume that the results of s simulation runs can be used to determine one value of such a characteristic. The standard bootstrapping technique bases on a resampling of the results y_i of the simulation runs. The elements of b bootstrap samples $(y^*_{k1}, y^*_{k2}, \ldots, y^*_{ks})$ with $k = 1, \ldots, b$ are obtained by random sampling of the original y_i with replacement. Based on the b bootstrap samples, the expected value of the characteristic and the associated confidence interval are estimated [69].

Example. Let us apply (2.108) to a simple example. The domain of Y shall be the set of the values 0 and 1. The last value shall announce that a performance variable (for instance the delay) is behind its limit. We are interested in the probability p that the specification limit is violated. Assuming n simulation runs are carried out. We observe n_1 times the value 1. We estimate $\hat{p} = \frac{n_1}{n}$ and $\frac{\hat{\sigma}^2}{n} = \frac{n_1 \cdot \left(1 - \frac{n_1}{n}\right)^2 + (n - n_1) \cdot \left(0 - \frac{n_1}{n}\right)^2}{n \cdot (n-1)} = \frac{n_1 - \frac{n_1^2}{n}}{n \cdot (n-1)} \approx \frac{\hat{p} \cdot (1 - \hat{p})}{n}$. Thus, it follows that the number of Monte Carlo simulations runs to assure that the probability p belongs to the 95% confidence interval $[\hat{p} - \nu\hat{p}, \hat{p} + \nu\hat{p}]$ must meet the inequality $1.959\sqrt{\frac{\hat{p} \cdot (1-\hat{p})}{n}} \leq \nu\hat{p}$. Therefore, it must be required $n \geq \frac{1.959^2}{\nu^2} \cdot \frac{1-p}{p} \approx \frac{4}{\nu^2} \cdot \frac{1-p}{p}$. If we accept a value $\nu = 0.1 = 10\%$ and expect, for instance a marginal probability p of $10^{-4} = 0.1$ promille it follows n must be greater about $4 \cdot 10^6$. Thus, more than a million simulation runs are required. This shows the limitations of the method.

The Monte Carlo simulation is a simple and ubiquitous applicable utility in the investigation of complex systems, if there is any uncertainty of the behavior of the process parameters. However, the Monte Carlo simulation may reach its limits in some application case because of the computational effort and/or the accuracy of the results. That is the reason, why more efficient approaches must be investigated [70]. Special strategies have been developed to reduce the number of simulation runs such as stratified sampling, importance sampling or Latin hypercube sampling (LHS). For instance, Latin hypercube sampling generates a characteristic collection of parameter values from a multidimensional distribution. These methods are especially of interest in cases, where small probabilities have to be determined with a high accuracy. Yield analysis requires such methods. This will be discussed in more detail in Sect. 4.6.

2.2.8.2 The ANOVA Method

ANalysis Of VAriance (ANOVA) is a method to detect significant differences between samples. It can be thought as a generalization of the simple t-test. The ratio of the mean square between different samples and the mean square within a sample is calculated. The exceedance of a critical value given by the F-distribution indicates significant differences inside the data set. It allows to distinct between random and systematical differences. The assumptions for ANOVA are

– Independence of the cases
– Normality (distributions of the residuals are normal)
– Equality (variance of data in groups should be the same).

2.2.8.3 Variance Reduced Monte Carlo Approaches: Importance Sampling

As shown in the last example the standard Monte Carlo approach requires a huge number of simulation runs to estimate small probabilities with an appropriate accuracy. However, the accuracy can also be increased if an estimator with a lower variance can be applied. This is the basic idea behind variance reduction methods as stratified sampling and others [70].

One of the methods that is aimed at the same objective is importance sampling. Instead of the original probability density function, a modified function is used to generate random samples of parameters. Broadly speaking, it is tried to apply a modified function that delivers more performance values in the critical region of interest than the original one. That is, values are sampled with respect to their importance. We will try to figure out the basics.

We assume that the parameters can be described by a multivariate random vector $\underline{X}$ with the PDF $f_{\underline{X}}$. The relation between parameters and the (univariate) performance value under investigation is given by a function $h : \mathscr{D} \subset \mathbb{R}^m \longrightarrow \mathbb{R}$. Thus, the performance value $Y = h(\underline{X})$ is also a random variable with a PDF f_Y. We are now interested in the probability $I = \text{Prob}(Y > y^{\text{wc}})$ that the performance value is greater than $y^{\text{wc}} \in \mathbb{R}$. For instance, to determine small values of I is a typical task if yield shall be investigated. We will now figure out some basic steps using importance sampling. The wanted probability is given by

$$I = \int_{y^{\text{wc}}}^{\infty} f_Y(y)\text{d}y. \tag{2.112}$$

The idea is now to evaluate the equation

$$I = \int_{y^{\text{wc}}}^{\infty} \frac{f_Y(y)}{g_Y(y)} \cdot g_Y(y)\text{d}y, \tag{2.113}$$

where g_Y is used instead of f_Y as trial distribution. To estimate I, random values distributed with the probability density function g are generated. Thus after n simulation runs, I can be estimated by

$$\hat{I} = \frac{\sum_{i=1}^{n} \delta(y_i) \cdot \frac{f_Y(y_i)}{g_Y(y_i)}}{\sum_{i=1}^{n} \frac{f_Y(y_i)}{g_Y(y_i)}} \approx \frac{1}{n} \sum_{i=1}^{n} \delta(y_i) \cdot \frac{f_Y(y_i)}{g_Y(y_i)} \tag{2.114}$$

with

$$\delta(y_i) = \begin{cases} 0 & \text{for } y_i \leq y^{\text{wc}} \\ 1 & \text{for } y_i > y^{\text{wc}} \end{cases}. \tag{2.115}$$

The standard deviation $\sigma_{\hat{I}}$ of the estimator $\hat{I}$ can be determined by (see for instance [72])

$$\frac{\sigma_{\hat{I}}}{I} = \frac{1}{\sqrt{n}} \left(\frac{I_2}{I^2} - 1 \right)^{\frac{1}{2}} \tag{2.116}$$

with $I_2 = \int_{y^{\text{wc}}}^{\infty} \frac{f_Y{}^2(x)}{g_Y(x)} \mathrm{d}x$ and I given by (2.112).

The main and difficult task is to find a good distribution g_Y that can be applied in importance sampling. In theory, the best distribution is given by [73]

$${}^{\text{best}}g_Y(y) = \frac{\delta(y) \cdot f_Y(y)}{\int_{y^{\text{wc}}}^{\infty} f_Y(y) \mathrm{d}y}. \tag{2.117}$$

Looking at (2.116), this probability density distribution would be indeed the "best" choice and deliver an estimator with standard deviation zero. However, this is of little practical interest because the value I we want to estimate is needed as denominator in (2.117). But what we see is that the shape of g_Y should be near $\delta(y) \cdot f_Y(y)$.

Importance sampling by scaling and translation are widely used with the density functions

$$g_Y(y) = \frac{1}{a} \cdot f_Y \left(\frac{y}{a} \right) \tag{2.118}$$

and

$$g_Y(y) = f_Y(y - c), \tag{2.119}$$

respectively. Of practical importance is the usage of a mixed density function using the original distribution f_Y and r (at least one) other distribution h_{i_Y} [74]

$$g_Y(y) = \left(1 - \sum_{i=1}^{r} \lambda_i \right) \cdot f_Y(y) + \sum_{i=1}^{r} \lambda_i \cdot h_{i_Y}(y) \tag{2.120}$$

with $\sum_{i=1}^{r} \lambda_i \leq 1$ and $\forall_{1 \leq i \leq r} \quad \lambda_i \geq 0$.

Nevertheless, the choice of an adequate trial function g_Y for importance sampling remains a critical task. If the performance value Y depends on a random parameter value $\underline{X}$, practical experience show that importance sampling often only can be applied if a low number of parameters has to be considered.

Example: A typical case is the case where Y is normal distributed. We try to use a translated function (2.119) as trial function for the importance sampling analysis of (2.112). It can be shown that in this case the standard deviation of the estimator (given by (2.116)) is a minimum when $c \approx y^{\text{wc}}$. If Y depends on random parameters $\underline{X}$ with a multivariate normal distribution $f_{\underline{X}}$ with mean value $\underline{\mu}$, then the mean value of an appropriate trial function should be the associated worst case point $\underline{x}^{\text{wc}}$.

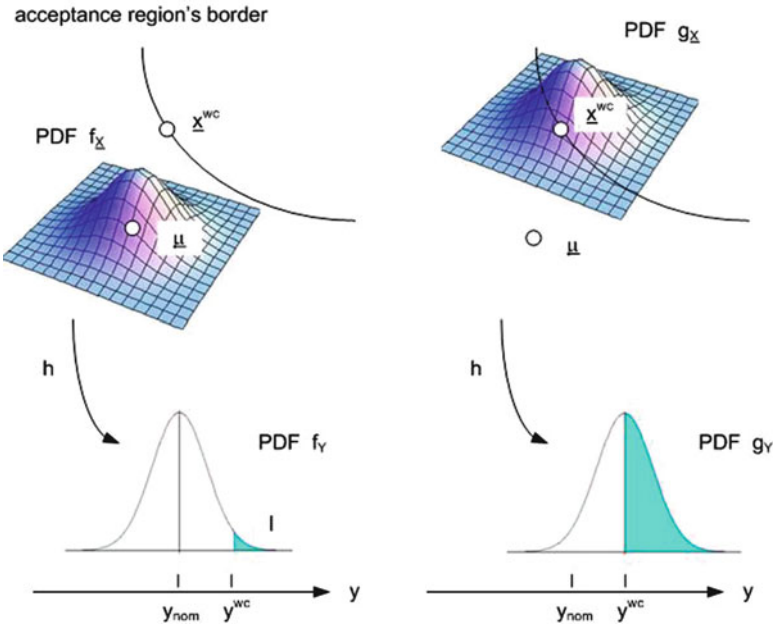


Fig. 2.13 Example for using standard Monte Carlo approach and importance sampling

This is the point of the acceptance region's border with the greatest probability. Thus, a possible choice for the mixed density function is

$$g_{\underline{X}}(\underline{x}) = (1-\lambda)\cdot f_{\underline{X}}(\underline{x}) + \lambda \cdot f_{\underline{X}}(\underline{x} - (\underline{x}^{wc} - \underline{\mu})) \tag{2.121}$$

with a λ between 0 and less than 1, for instance $\lambda = 0.5$. Then the I value can be estimated by

$$\hat{I} \approx \frac{1}{n}\sum_{i=1}^{n} \delta\left(h(\underline{x}_i)\right) \cdot \frac{f_{\underline{X}}(\underline{x}_i)}{g_{\underline{X}}(\underline{x}_i)}. \tag{2.122}$$

Figure 2.13 demonstrates the procedure.

2.2.9 Evaluation of Results

2.2.9.1 Statistical Tests

To check the distributional properties of some performance parameters the following statistical tests can be useful

- χ^2-test of goodness of fit
- Kolmogorov-Smirnov test.

They allow to compare the CDF F_Y of a random variable Y with that of a theoretical one F^*, or even two CDF's F_{Y_1} and F_{Y_1} of Y_1 and Y_2.

Furthermore, the independency of Y_1 and Y_2 can be checked by the χ^2-test of independency.

2.2.9.2 Discussion to the Extreme Value Behavior

The extreme value theory is a very promising application area for an evaluation of the quality of integrated circuits. There are manifold methods to investigate the so-called tail behavior, to evaluate probabilities $P(Y > y_{\text{thr}})$ for great differences of a performance parameter from the working point.

One of them is the peak over threshold (POT) method.

Its basic idea is the approximation of the sample by a generalized Pareto distribution (GPD) function

$$G_{\xi,\beta}(y) = \begin{cases} 1 - \left(1 + \frac{\xi y}{\beta}\right)^{-1/\xi}, & \xi \neq 0 \\ 1 - \exp\left(\frac{-y}{\beta}\right), & \xi = 0, \end{cases} \tag{2.123}$$

where the parameters ξ and β are estimated from the sample by a maximum likelihood method, being the solution of

$$\xi k = \sum_{i=1}^{k} \log\left(1 + \frac{\xi\, y^{(i)}}{\beta}\right), \tag{2.124}$$

$$\sum_{i=1}^{k} \frac{y^{(i)}}{\beta + \xi y^{(i)}} = \frac{k}{1+\xi}, \tag{2.125}$$

where k is the number of excesses $y^{(1)}, y^{(2)}, \ldots, y^{(k)}$ of a threshold value y_{thr} in the sample $y_1, y_2, \ldots, y_n$.

For basics in extreme value theory see [75–77]. Applications in the field of microelectronics do not seem to appear in the related literature so far.

2.2.9.3 Projection from Cell to Full Chip

Knowing performance characteristics of a single cell, extrapolation methods to more complicated structures are desired. An example is the extrapolation of the leakage from a single cell to a full chip.

In the traditional cell leakage analysis, the leakage of a cell is given by

$$\log(I_{\text{Cell}}) = a_0 + \sum_{i=1}^{m} a_i x_i, \tag{2.126}$$

where the x_i are $\Delta V_{\text{th}}, \Delta T_{\text{ox}}, \Delta L, \ldots$ and $a_0, a_1, a_2, a_3, \ldots$ real coefficients and the leakage of a full chip is given by the sum of the particular cells

$$I_{\text{Chip}} = \sum_{i=1}^{n} I_{\text{Cell},i}, \tag{2.127}$$

where n is the number of cells in the chip.

An extended leakage analysis (cf. [78]) is that basing on a quadratic response surface

$$\log(I_{\text{Chip}}) = \underline{x}^T \mathbf{A} \underline{x} + \underline{b}^T \underline{x} + c, \tag{2.128}$$

where $\mathbf{A}$ means a full rank $(n \cdot n)$-matrix, $n \approx 10^6$, a vector $\underline{b} \in \mathbb{R}^n$ and a real constant c.

To reduce the costs of the calculation of a full matrix, a low rank matrix $\tilde{\mathbf{A}}$ is determined, $\tilde{\mathbf{A}}$ is a sum of dominant eigenvalues and eigenvectors of the matrix $\mathbf{A}$, where the difference $\left\|\mathbf{A} - \tilde{\mathbf{A}}\right\|_F$ is minimized, $\|\cdot\|_F$ denotes the Frobenius norm. This step reduces the modeling costs by a factor of approximately $10^2 \ldots 10^4$.

A further extension , the so-called *incremental leakage analysis* facilitates a quick update on the leakage distributions after local changes to a circuit. The change of a few terms is much cheaper than a full new calculation. Simple replacements of $I_{\text{Cell}}^{\text{Old}}$ by $I_{\text{Cell}}^{\text{New}}$, allow to update the calculation of I_{Chip}.

References

1. Shichman, H., Hodges, D.A.: Modeling and simulation of insulated-gate field-effect transistor switching circuits. IEEE J. Solid-State Circuits **3**(5), 285–289 (1968)
2. Meyer, J.E.: MOS models and circuit simulation. RCA Review **32**, 42–63 (1971)
3. Ward, D.E., Dutton, R.W.: A charge-oriented model for MOS transistor capacitances. IEEE J. Solid-State Circuits **13**(5), 703–708 (1978)
4. Foty, D.P.: MOSFET Modeling with SPICE - Principles and Practice. Prentice Hall, Upper Saddle River, NJ (1997)
5. Sheu, B.J., Scharfetter, D.L., Ko, P.K., Jen, M.C.: BSIM Berkeley short-channel IGFET model for MOS transistors. IEEE J. Solid-State Circuits **22**(4), 558–566 (1987)
6. Synopsys: HSPICE MOSFET Models Manual, version z-2006.03 edn. (2007). Chapter 6
7. Liu, W.: MOSFET Models for SPICE Simulation, Including BSIM3v3 and BSIM4. John Wiley & Sons, New York (2001)
8. Enz, C.C., Krummenacher, F., Vittoz, E.A.: An analytical MOS transistor model valid in all regions of operation and dedicated to low-voltage and low current applications. J. Analog Integr. Circuits Signal Process **8**, 83–114 (1995)
9. Compact Model Council. http://www.geia.org/index.asp?bid=597
10. BSIM3, BSIM4 homepage. http://www-device.eecs.berkeley.edu/~bsim3/
11. Miura-Mattausch, M., Feldmann, U., Rahm, A., Bollu, M., Savignac, D.: Unified complete MOSFET model for analysis of digital and analog circuits. IEEE Trans. CAD/ICAS **15**(1), 1–7 (1996)
12. Gildenblat, G., Li, X., W.Wu, Wang, H., Jha, A., van Langevelde, R., Smit, G., Scholten, A., Klaassen, D.: PSP: An advanced surface-potential-based MOSFET model for circuit simulation. Electron Devices, IEEE Transactions on **53**(9), 1979–1993 (2006)

13. PSP homepage. http://pspmodel.asu.edu/
14. HISIM homepage. http://home.hiroshima-u.ac.jp/usdl/HiSIM.html
15. Tsividis, Y.: Operation and Modeling of the MOS Transistor, 2nd Edn. McGraw-Hill, New York (1999)
16. Taur, Y., Ning, T.: Fundamentals of modern VLSI devices. Cambridge University Press (1998)
17. Wang, A., Calhoun, B.H., Chandrakasan, A.P.: Sub-threshold Design for Ultra Low-Power Systems. Springer (2006)
18. Moore, G.E.: Cramming more components onto integrated circuits. Electronics **38**, 114 ff. (1965)
19. Dennard, R., Gaensslen, F., Rideout, V., Bassous, E., LeBlanc, A.: Design of ion-implanted MOSFET's with very small physical dimensions. IEEE Journal of Solid-State Circuits **9**(5), 256–268 (1974)
20. Rabaey, J.: Low Power Design Essentials. Springer, Boston, MA (2009). DOI 10.1007/978-0-387-71713-5
21. Kenyon, C., Kornfeld, A., Kuhn, K., Liu, M., Maheshwari, A., Shih, W., Sivakumar, S., Taylor, G., VanDerVoorn, P., Zawadzki, K.: Managing process variation in Intel's 45nm CMOS technology. Intel Technology Journal **12**(2) (2008). URL http://www.intel.com/technology/itj/2008/v12i2/3-managing/1-abstract.htm
22. Asenov, A.: Random dopant induced threshold voltage lowering and fluctuations in sub-0.1 um MOSFET's: A 3-D "atomistic" simulation study. IEEE Transactions on Electron Devices **45**(12), 2505–2513 (1998). DOI 10.1109/16.735728
23. Diaz, C.H., Tao, H.J., Ku, Y.C., Yen, A., Young, K.: An experimentally validated analytical model for gate line-edge roughness (LER) effects on technology scaling. IEEE Electron Device Letters **22**(6), 287–289 (2001). DOI 10.1109/55.924844
24. Asenov, A., Kaya, S., Davies, J.H.: Intrinsic threshold voltage fluctuations in decanano MOSFETs due to local oxide thickness variations. IEEE Transactions on Electron Devices **49**(1), 112–119 (2002). DOI 10.1109/16.974757
25. Kaushik, V.S., O'Sullivan, B.J., Pourtois, G., Van Hoornick, N., Delabie, A., Van Elshocht, S., Deweerd, W., Schram, T., Pantisano, L., Rohr, E., Ragnarsson, L.A., De Gendt, S., Heyns, M.: Estimation of fixed charge densities in hafnium-silicate gate dielectrics. IEEE Transactions on Electron Devices **53**(10), 2627–2633 (2006). DOI 10.1109/TED.2006.882412
26. Lucovsky, G.: Intrinsic limitations on the performance and reliability of high-k gate dielectrics for advanced silicon devices. In: Proc. IEEE Int. Integrated Reliability Workshop Final Report (2005). DOI 10.1109/IRWS.2005.1609592
27. Capodieci, L.: From optical proximity correction to lithography-driven physical design (1996-2006): 10 years of resolution enhancement technology and the roadmap enablers for the next decade. In: Proceedings of SPIE, vol. 6154 (3) (2006)
28. Nag, S., Chatterjee, A., Taylor, K., Ali, I., O'Brien, S., Aur, S., Luttmer, J.D., Chen, I.C.: Comparative evaluation of gap-fill dielectrics in shallow trench isolation for sub-0.25 /spl mu/m technologies. In: Proc. Int. Electron Devices Meeting IEDM '96, pp. 841–845 (1996). DOI 10.1109/IEDM.1996.554111
29. Tsang, Y.L., Chattopadhyay, S., Uppal, S., Escobedo-Cousin, E., Ramakrishnan, H.K., Olsen, S.H., O'Neill, A.G.: Modeling of the threshold voltage in strained si/si1-x gex/si1-ygey(x-y) cmos architectures. IEEE Transactions on Electron Devices **54**(11), 3040–3048 (2007). DOI 10.1109/TED.2007.907190
30. Al-Bayati, A., Graoui, H., Spear, J., Ito, H., Matsunaga, Y., Ohuchi, K., Adachi, K., Miyashita, K., Nakayama, T., Oowada, M., Toyoshima, Y.: Advanced CMOS device sensitivity to USJ processes and the required accuracy of doping and activation. In: Proc. 14th Int. Conf. Ion Implantation Technology 2002, pp. 185–188 (2002). DOI 10.1109/IIT.2002.1257969
31. Lorenz, J., Bär, E., Clees, T., Jancke, R., Salzig, C., S., S.: Hierarchical simulation of process variations and their impact on circuits and systems: Methodology. IEEE Trans. on Electron Devices, Special Issue Vol. **58**(8) (2011), pp. 2218–2226

32. Lorenz, J., Bär, E., Clees, T., Jancke, R., Salzig, C., S., S.: Hierarchical simulation of process variations and their impact on circuits and systems: Results. IEEE Trans. on Electron Devices, Special Issue Vol. **58**(8) (2011), pp. 2218–2226
33. Jancke, R., Kampen, C., Kilic, O., Lorenz, J.: Hierarchischer ansatz für die monte-carlo-simulation komplexer mixed-signal-schaltungen. In: 11. ITG/GMM-Fachtagung ANALOG. Erfurt (2010)
34. Yamaoka, M., Onodera, H.: A detailed vth-variation analysis for sub-100-nm embedded SRAM design. In: Proc. IEEE Int. SOC Conf, pp. 315–318 (2006). DOI 10.1109/SOCC.2006.283905
35. Pelgrom, M.J.M., Duinmaijer, A.C.J., Welbers, A.P.G.: Matching properties of mos transistors. IEEE Journal of Solid-State Circuits **24**(5), 1433–1439 (1989). DOI10.1109/JSSC.1989.572629
36. Petzold, L., Li, S., Cao, Y., Serban, R.: Sensitivity analysis of differential-algebraic equations and partial differential equations. Computers & Chemical Engineering **30**(10-12), 1553 – 1559 (2006). DOI 10.1016/j.compchemeng.2006.05.015
37. Özyurt, D.B., Barton, P.I.: Cheap second order directional derivatives of stiff ODE embedded functionals. SIAM J. Sci. Comput. **26**, 1725–1743 (2005). DOI 10.1137/030601582
38. Cao, Y., Li, S.T., Petzold, L., Serban, R.: Adjoint sensitivity analysis or differential-algebraic equations: The adjoint DAE system and its numerical solution. Siam Journal on Scientific Computing **24**(1), 1076–1089 (2003). DOI 10.1137/S1064827501380630
39. Sakurai, T., Newton, A.R.: Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas. IEEE Journal of Solid-State Circuits SC **25**(2), 584–594 (1990)
40. Bowman, K.A., Austin, B.L., Eble, J.C., Tang, X., Meindl, J.D.: A physical alpha-power law mosfet model. IEEE Journal of Solid-State Circuits **34**(10), 1410–1414 (1999). DOI 10.1109/4.792617
41. Rabaey, J.M., Chandrakasan, A., Nikolic, B.: Digital Integrated Circuits: A Design Perspective. Prentice Hall (2003)
42. Stolk, P.A., Widdershoven, F.P., Klaassen, D.B.M.: Modeling statistical dopant fluctuations in MOS transistors. IEEE Transactions on Electron Devices **45**(9), 1960–1971 (1998). DOI 10.1109/16.711362
43. Narendra, S.G.: Effect of MOSFET threshold voltage variation on high-performance circuits. Ph.D. thesis, Massachusetts Institute of Technology. Dept. of Electrical Engineering and Computer Science (2002)
44. Roy, K., Mukhopadhyay, S., Mahmoodi-Meimand, H.: Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits. In: Proceedings of the IEEE, pp. 305–327 (2003)
45. Veendrick, J.M.H.: Nanometer CMOS ICs: From basics to ASICs, 1st Edn. Springer, Heidelberg (2008)
46. Srivastava, A., Blaauw, D., Sylvester, D.: Statistical Analysis and Optimization for VLSI: Timing and Power. Springer Science+Business Media Inc, Boston, MA (2005). DOI 10.1007/b137645
47. Chan, T.Y., Chen, J., Ko, P.K., Hu, C.: The impact of gate-induced drain leakage current on mosfet scaling. In: Proc. Int. Electron Devices Meeting, vol. 33, pp. 718–721 (1987). DOI 10.1109/IEDM.1987.191531
48. Bouhdada, A., Bakkali, S., Touhami, A.: Modelling of gate-induced drain leakage in relation to technological parameters and temperature. Microelectronics and Reliability **37**(4), 649–652 (1997). DOI 10.1016/S0026-2714(96)00062-5
49. Mulaik, S.A.: Foundations of factor analysis, 2nd Edn. Chapman & Hall/CRC statistics in the social and behavioral sciences series. CRC Press, Boca Raton, FL (2010)
50. Johnson, R.A., Wichern, D.W.: Applied multivariate statistical analysis, 6th Edn. Pearson Prentice Hall, Upper Saddle River N.J. (2007)
51. Weisstein, E.W.: Regularized gamma function. http://mathworld.wolfram.com/RegularizedGammaFunction.html. From MathWorld – A Wolfram Web Resource
52. Rencher, A.C.: Methods of multivariate analysis (2002). DOI 10.1002/0471271357

53. Johnson, N., Kotz, S.: Distribution in Statistics I. Continuous univarite distributions. Wiley (1970)
54. Karian, Z.A., Dudewicz, E.J.: Fitting statistical distributions: The Generalized Lambda Distribution and Generalized Bootstrap methods. CRC Press, Boca Raton (2000)
55. Shlens, J.: Tutorial on Principal Component Analysis. Tech. Rep. Version 2, Systems Neurobiology Laboratory, Salk Insitute for Biological Studies and Institute for Nonlinear Science, University of California, San Diego (2005). CiteSeerX 10.1.1.115.3503
56. Jolliffe, I.T.: Principal Component Analysis, 2nd Edn. Springer (2002)
57. Jackson, J.E.: A User's Guide to Principal Components. Wiley Series in Probability and Statistics. Wiley (2003)
58. Skillicorn, D.B.: Understanding complex datasets: Data mining with matrix decompositions. Chapman & Hall/CRC data mining and knowledge discovery series. Chapman & Hall/CRC Press, Boca Raton (2007)
59. Kalman, D.: A singulary valuable decomposition : The SVD of a matrix. The College Mathematical Journal **27**(1), 1–23 (1996). URL http://www1.math.american.edu/People/kalman/pdffiles/svd.pdf
60. Shamsi, D., Boufounos, P., Koushanfar, F.: Noninvasive leakage power tomography of integrated circuits by compressive sensing. In: ISLPED '08: Proceedings of the 2003 international symposium on Low power electronics and design, pp. 341–346. ACM, NY, USA, Bangalore (2008)
61. Box, G.E.P., Draper, N.R.: Empirical model-building and response surfaces. Wiley series in probability and mathematical statistics. Wiley, New York (1987)
62. Poggio, T.: On optimal nonlinear associative recall. Biol. Cybernetics **19**, 201–209 (1975)
63. Lauridsen, S., Vitali, R., van Keulen, F., Haftka, R.T., Madsen, J.: Response surface approximation using gradient information. In: World Congress of Structural and Multidisciplinary Optimization WCSMO-4. Dalian, China (2001). CiteSeerX 10.1.1.16.2135
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.16.2135
64. Park, S., H.J., K., Cho, J.I.: Recent advances in linear models and related areas. In: Optimal Central Composite Designs for Fitting Second Order Response Surface Linear Regression Models, pp. 323–339. Physica-Verlag HD (2008). DOI 10.1007/978-3-7908-2064-5_17
65. Cheng, L., Xiong, J., He, L.: Non-Gaussian statistical timing analysis using second-order polynomial fitting. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on **28**(1), 130–140 (2009). DOI 10.1109/TCAD.2008.2009143
66. Sohrmann, C., Muche, L., Haase, J.: Accurate approximation to the probability of critical performance. In: 2. GMM/GI/ITG-Fachtagung Zuverlässigkeit und Entwurf, pp. 93–97 (2008)
67. Zhang, M., Olbrich, M., Seider, D., Frerichs, M., Kinzelbach, H., Barke, E.: CMCal: An accurate analytical approach for the analysis of process variations with non-Gaussian parameters and nonlinear functions. In: Design, Automation & Test in Europe Conference & Exhibition, 2007. DATE '07, pp. 1–6 (2007). DOI 10.1109/DATE.2007.364598
68. Saucier, R.: Computer generation of statistical distributions. Tech. rep., Army Research Laboratory (2000). URL http://ftp.arl.mil/random/
69. Cheng, R.C.: Boostrap methods in computer simulation experiments. In: Proceedings of the 1995 Winter Simulation Conference, pp. 171–177 (1995)
70. Liu, J.S.: Monte Carlo Strategies in Scientific Computing. Springer Publishing Company, Incorporated (2008)
71. Rao, R., Srivastava, A., Blaauw, D., Sylvester, D.: Statistical estimation of leakage current considering inter- and intra-die process variation. In: Proceedings of the 2003 International Symposium on Low Power Electronics and Design ISLPED '03, pp. 84–89 (2003), DOI 10.1109/LPE.2003.1231840
72. Denny, M.: Introduction to importance sampling in rare-event simulations. EUROPEAN JOURNAL OF PHYSICS **22**(4), 403–411 (2001)
73. Robert, C.P., Casella, G.: Monte Carlo statistical methods, 2nd Edn. Springer texts in statistics. Springer, New York, NY (2004). ISBN 978-0-387-21239-5

74. Hesterberg, T.: Advances in importance sampling. Statistics Department, Stanford University (1998)
75. Hein, A.: Parameter- und Quantilschätzung in der Extremwerttheorie. Uni Kaiserslautern (2001)
76. Reiss, R., M., T.: Statistical Analysis of Extreme Values. Birkhäuser (2007)
77. de Haan, L., Ferreira, A.: Extreme value theory. An Introduction. Springer series in operations research and financial engineering. Springer (2000)
78. Li, X., Le, J., Pileggi, L.T.: Projection-based statistical analysis of full-chip leakage power with non-log-normal distributions. In: Proc. DAC 2006, pp. 103–108 (2006)
79. GSA & IET International Semiconductor Forum, London UK, 18-19 May 2010 "Better Analog Modeling and Integration with iPDKs"
80. Hu, C.: Future CMOS scaling and reliability. Proceedings of the IEEE, **81**(5) (1993)
81. Wong, B.P., Mittal, A., Cao, Y., Starr, G.: NANO-CMOS Circuit and physical design, John Wiley & Sons, New York (2005)
82. Robertson, J.: High dielectric constant gate oxides for metal oxide Si transistors. Rep. Prog. Phys. **69**, 327–396 (2006) Institute physics publishing
83. Ge precursors for strained Si and compound semiconductors, semiconductor international, (2006)
84. Risch, L.: Pushing CMOS beyond the roadmap, Proceedings of ESSCIRC, p. 63 (2005)
85. Subramanian, V.: Multiple gate field-effect transistors for future CMOS technologies. IETE Technical review **27**, 446–454 (2010)

Chapter 3
Examination of Process Parameter Variations

Emrah Acar, Hendrik Mau, Andy Heinig, Bing Li, and Ulf Schlichtmann

Chapter 3 presents an overview on the sources of variations in the manufacturing process. Section 3.1 deals with the so-called Front-End Of Line (FEOL) variations that refer to the variations on the device level. Besides the extrinsic variability that is caused by the imperfections of the manufacturing process, the intrinsic variability due to atomic-level differences is gaining importance. At the nanoscale level, even an uncertainty of a few atoms may adversely affect the parameters and the behavior of microelectronic devices. Some details are going to be figured out in the first section of this chapter.

Besides transistors, the interconnections of devices play a decisive role in the determination of the time and energy behavior of a circuit. Aspects of the interconnect lines on a wafer are the subject of the Sect. 3.2 of this chapter. Back-End Of Line (BEOL) variations impact these interconnect lines. Sources of variations will be classified in the second section. Environmental and manufacturing factors will be compared as well as spatial and temporal variations. The sources of variability in

E. Acar (✉)
Research Staff Member, IBM Research, IBM T.J. Watson Research Center,
1101 Kitchawan Rd, Yorktown Heights, NY 10598, USA
e-mail: emrah@us.ibm.com

H. Mau
GlobalFoundries, Dresden Design Enablement, Wilschdorfer Landstraße 101,
01109 Dresden, Germany
e-mail: hendrik.mau@globalfoundries.com

A. Heinig
Fraunhofer Institute for Integrated Circuits IIS, Design Automation Division EAS,
Zeunerstraße 38, 01069 Dresden, Germany
e-mail: andy.heinig@eas.is.fraunhofer.de

B. Li • U. Schlichtmann
Technische Universität München, Institute for Electronic Design Automation,
Arcisstraße 21, 80333 Munich, Germany
e-mail: b.li@tum.de; ulf.schlichtmann@tum.de

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6_3,

the basic Cu-damascene process steps are going to be presented. Essential variations result from physical (e.g. interferences of light) and technological (asymmetry caused by the Chemical Mechanical Polishing Process CMP) reasons. Mainly, the interconnect lines are characterized by their resistances and capacitances to ground and between lines. These characteristics depend on the width and thickness of the wires, the distance between neighbouring wires, and the height of the dielectric layers. The dependency of the resulting variations of resistances and capacitances on process steps will be discussed.

Mathematical models to consider process variations will be presented in Sect. 3.3. Process variations have to be modeled in a way that allows for considering them in simulation and analysis methods used in the design process. Therefore, methodologies that reduce the complexity of problems, break occurring dependencies of parameters down and try to eliminate them have been developed. Because of the huge number of components and complex interconnections, it is important to distinguish between important variations and less important ones in order to achieve a result with reasonable computational effort. This requires a compromise between precision and clarity for the designer. Time behavior is one of the most important characteristics of digital circuits. For that reason, the last section specifically deals with the delays of basic logic components and examines how variations affect time behavior. In this context, different time constraints will be considered.

Emrah Acar wrote Sect. 3.1. Hendrik Mau and Andy Heinig are the authors of the Sect. 3.2. The last Sect. 3.3 of the third chapter was prepared by Bing Li and Ulf Schlichtmann.

3.1 Parameter Variations in the Manufacturing Process

During this decade, the experts were hotly debating about the end of device scaling, which is still the major driving force of the contemporary electronics industry. Since 1990s, we can see in many circles, the scaling had been claimed dead, as late as 2005 on a major design automation conference by one of the most famous technology company head expert. What was claimed was not about the end of the ever-shrinking device sizes, printability of smaller feature sizes or smaller feature size printability with the existing manufacturing capabilities, but more about an admission of the end of the performance gains obtained by device scaling. One of the major reasons for this misfortune is the existence of parameter variations within the manufacturing process that are inevitable when the devices are shrunk all the way to their, respectively, feasible limits with imperfect lithographic equipment operations and material processing systems.

In this section, we will discuss the contributors to variability in CMOS devices. In this section, we will categorize, outline, and discuss about the front-end of the line variations, also referred to as device variations. These variations are physics based and represent themselves as parametric variations in the device models, which represent the physical behavior of the device and its interaction with the rest of

Table 3.1 Categories for device variations ©IBM 2006

Proximity	Spatial	Temporal reversible	Temporal irreversible
Variation of chip mean	Parameter means, L_{gate}, V_{th}, t_{ox}	Environmental, operating temperature	Hot-electron effect, NBTI shift
Within-chip mean	Pattern density, layout-induced transconductance	On-die hotspots	Hot-spot-enhanced NBTI
Device-to-device variations	Atomistic dopant variation, line-edge roughness, parameter std. dev.	SOI body history, self-heating	NBTI-induced V_{th} variation

the systems. The parameter variations for the devices are major contributors to variability issues in power, delay, temperature, and reliability of current integrated circuits and must be assessed properly during the design stages.

Device variability, also known as Front-End Of Line (FEOL) variability mainly refers to the variations at the device level. This affects the response of the most active electrical components such as transistor devices fabricated in the silicon. As previously discussed, the device performance is heavily influenced by the effective channel length, poly gate length, spacer widths, gate-oxide thickness, and device edge variations. Furthermore, there are other types of variations that affect the device performance including atomistic dopant variations, self-heating, hot-electron effect, negative bias temperature instability (NBTI), and history body effect for SOI devices. When devices are fabricated and put in operation, these variations can be observed by the corresponding device metrics including channel current, threshold voltage, gate-leakage current, subthreshold current, etc. The variations in these performance metrics are mainly caused by the parametric variations in the device internals.

3.1.1 Categorizing Variability

An effective way of describing device variability is by utilizing a categoric approach as shown in Table 3.1 [1]. This table is particularly useful to differentiate different variation mechanisms and the domains in which they represent themselves. For example, each row indicates variations that display according to their spatial domains, that is, the variations of the chip mean statistics, and variations displayed within-chip (also referred to as within-die), and finally variations displayed device-to-device randomly. The temporal columns are variations displayed during the operation of the device, some of which are reversible variations, meaning they are transient in nature and have a reverse possibility by time. These are typically reliability and aging variations concerning the life-time of the device and chip.

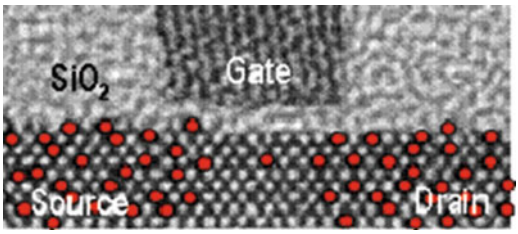


Fig. 3.1 Random dopants in a device ©A. Brown et al., IEEE Nanotechnology

3.1.1.1 Intrinsic Variability

Alternatively, we can categorize the device variations in terms of their physical domains. [1] talks about intrinsic and extrinsic device variabilities, along with placement-induced device variation, wear-out- and use-induced circuit variations. In this categorization, intrinsic variations are due to the atomic-level differences between devices even at the same layout geometries and operating/manufacturing environments. Such differences exist in device dopant profiles, film thicknesses, and line-edge roughness parameters due to the manufacturing process and equipment. The random dopant profiles that are intrinsic in nature are very random and can display major variations in threshold voltage especially beyond 65 nm technology nodes. The causes for these variations are the implant and annealing process that goes in a rather random positioning within the channel area. This has been a hot topic area for quite some time [2]. By device scaling and decrease in the dopant counts, the dopant variability within the channel area is highly emphasized. Quantum-mechanical effects within the channel also increase the threshold voltage variability along with doping in the gate [3]. Especially for small size devices, such as SRAMs, this variation is highly significant and must be assessed carefully for robustness and performance. Atomic-scale fluctuations in doping levels, and the dopants positions also cause variations in source/drain areas, affecting the overlap capacitance and source resistance. Figure 3.1 shows the randomly placed dopant atoms in a top view of a device indicating the fluctuations in doping level causing uncertainty in the source/drain edges.

Similarly, line-edge roughness (LER) effects further exacerbate the device variations. LER is, an intrinsic device variation, mainly a product of lithographic exposure process and uncertainties in photon counts and molecular composition of the photoresist. Due to the LER, the line edge for the gate shows a noisy pattern creating device length and edge variations.

Intrinsic device variations also include thickness variations for gate-oxide. Currently, the gate-oxide thicknesses are less than 1 nm, in the range of a few inter-atomic spacing. The variation in formation of the gate-oxide is quite significant for tunneling leakage current, which varies exponentially. These intrinsic variations are random for each device and are slated to increase for upcoming technology nodes (Figs. 3.2 and 3.3).

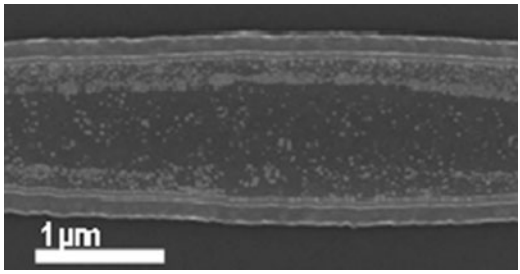


Fig. 3.2 Line-edge roughness example from a SEM camera

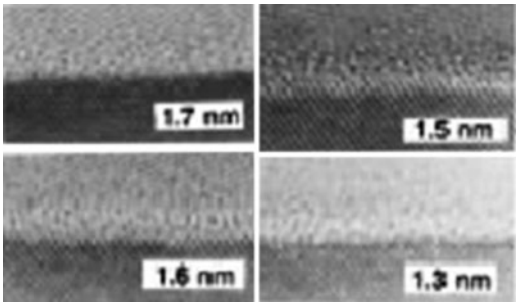


Fig. 3.3 Gate oxide variations reported in ©Momose et al., IEEE Trans. ED, 45(3) 1998

3.1.1.2 Extrinsic Variability

Extrinsic variation is caused by the imperfections in the contemporary manufacturing conditions. Unlike internal problems, it is not related to atomistic nature of the device.

Extrinsic variation can represent themselves as sets of wafers, positional errors, lens aberration, etc.

3.1.2 Variations from a Perspective for Model Parameters

[4] also discusses this topic in an introductory chapter and considered the device variations within, categories more related to device model parameter terms, such as short-channel effects, across–chip length variations, threshold voltage variations, hot carriers, negative bias temperature instability, and body effect. [5] also shares this perspective when the authors describe device variations in terms of model parameters, such as device length, device width, and threshold voltage.

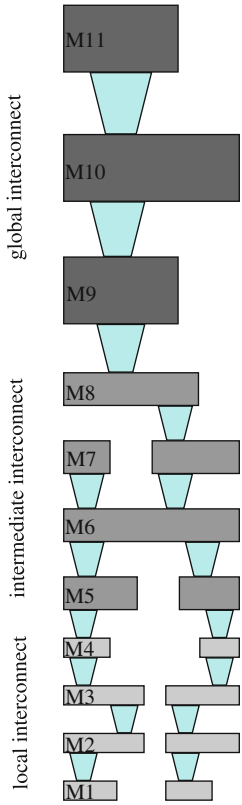


Fig. 3.4 Schematic cross-section of an interconnect stack

3.2 Variation of Interconnect Parameters

3.2.1 State-of-the Art Interconnect Stacks

Todays interconnect stacks are characterized by

- Up to 11 metal layers
- Use of low k material as dielectric
- Use of copper as interconnect
- Aspect ratio (h/w) of up to 2
- CD below $\lambda/2$ of litho wave length

Figure 3.4 shows a schematic cross-section of an interconnect stack as used in MPU designs [6].

Despite the replacement of Aluminium with Copper and the introduction of low-k material as dielectric to allow further decreasing size of interconnect wire dimensions, the resistance and capacitance have increased such that the interconnect delay dominates over the gate delay as shown in Fig. 3.5.

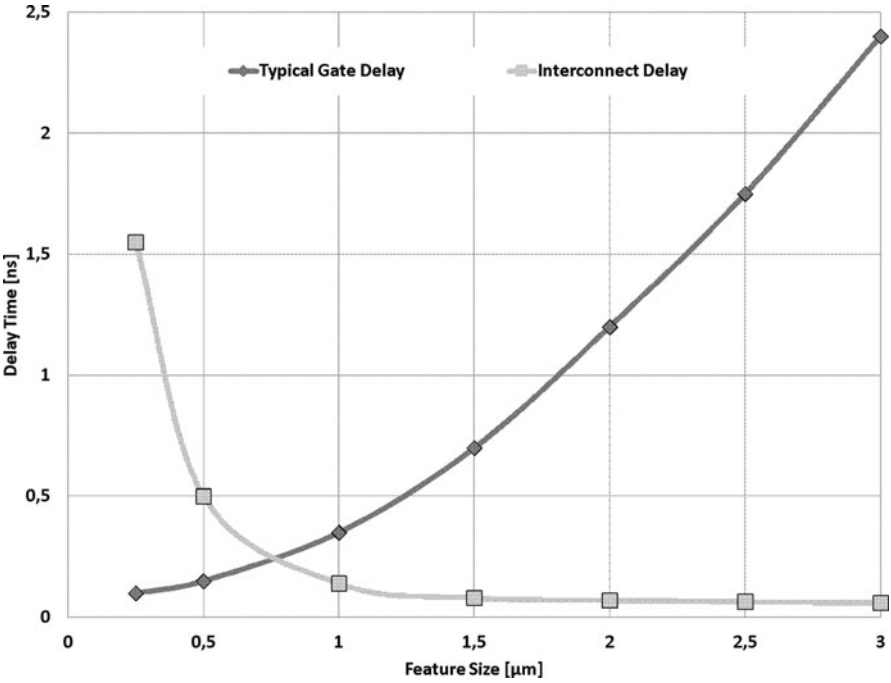


Fig. 3.5 Delay of interconnect and gate versus feature size

This domination of the interconnect delay made it necessary to model the nominal values of parasitic capacitance and resistance but also their variability with increasing accuracy to predict design performance and check functionality prior to tapeout.

Although variations of interconnect stack parameters were always occurring, with the increase of the interconnect delay this variability is now impacting the parametric performance of modern designs increasingly, regardless whether looked at Analog Mixed Signal or Digital routed designs.

3.2.2 Sources of Interconnect Variation

3.2.2.1 Classification Schemes

Before discussing the sources of variability for interconnects, it is helpful to recall some of the main classification schemes for variations and variability factors briefly. Since they are discussed elsewhere in much greater detail, a short overview will be given only.

Table 3.2 Comparison between systematic and random variations

Systematic variations	Random variations
Variations in data due to factors causing the parameter of interest to be shifted away from the expected value in a given direction	Variations in data due to factors affecting the value in a random manner. The variation can be described by distributions.
e.g., lithography, pattern proximity	e.g., line-edge roughness

Table 3.3 Comparison between environmental and intrinsic variability factors

Environmental factors	Intrinsic factors
Occurring during the operation of a circuit	Process variations and limitations causing variations in geometry and material parameters
e.g., temperature of the environment	e.g., variation in metal line thickness

Table 3.4 Comparison between spatial and temporal variations

Spatial variations	Temporal variations
Variation in space	Variation in time
e.g., variation of line resistivity across a wafer	e.g., degradation of conductivity due to electromigration

Variations can be divided into systematic and random as shown in Table 3.2. Depending on the level of understanding of underlying causes and the capability to model those variations accordingly, often systematic variations are considered as random due to their complex nature and difficulty to describe.

Another scheme is to separate variation based on environmental and intrinsic factors causing them as shown in Table 3.3.

A third and quite often used scheme is dividing up variations into spatial and temporal as Table 3.4 shows.

Spatial variations are occurring on very different scales. While intra-cell variations can be described on a scale of nanometer up to several microns, intrawafer variations are on a scale of many millimeters. Figure 3.6 depicts the different scales with their respective factors.

The large ratio of the spatial scales has important consequences when modeling variability. For instance, the variability of the line thickness caused by changes in the surrounding metal density can be modeled as a systematic way. However, while this is true at the wafer level and downward to the die level, it is not valid anymore at the standard cell level. At the time of designing those cells, the metal density of the neighborhood is not known and the metal thickness cannot be modeled as systematic but only as random. This affects the post-layout verification as well as the formulation of models for the electrical behavior.

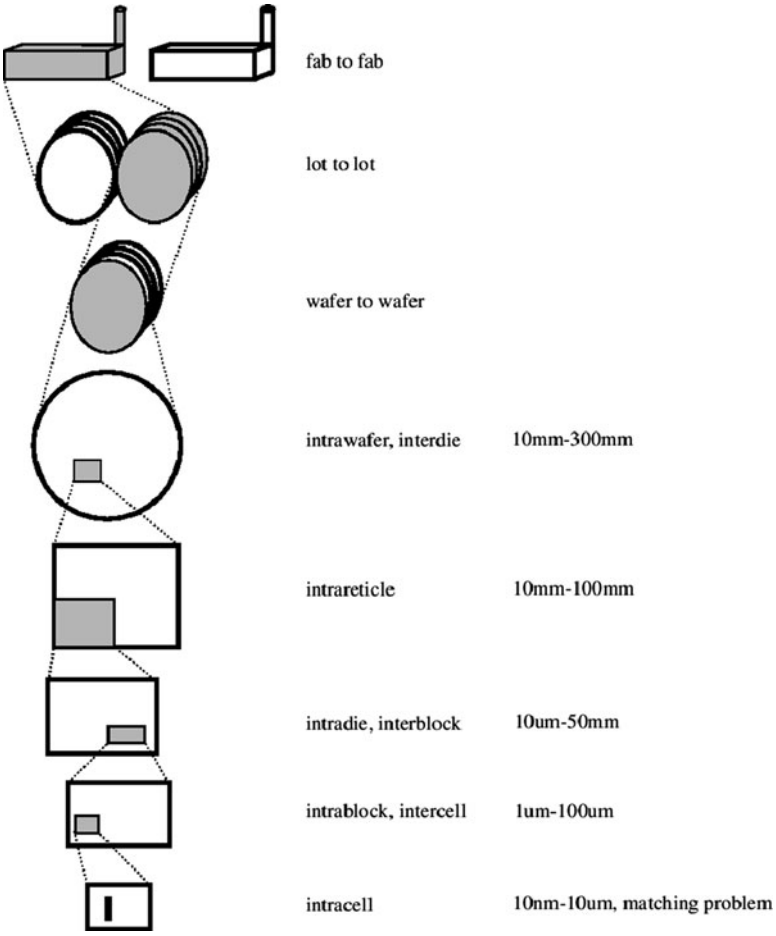


Fig. 3.6 Subdivision of spatial variations and their respective scaling factors

3.2.2.2 Source of Variability in the Cu Damascene Process

The basic Cu-damascene process steps are

- Lithography
- Etch
- Cu Deposition
- Chemical Mechanical Polishing Process (CMP)

and causing random and deterministic variations. Random variations in the resistance of the interconnect, for instance, can be caused by the grain structure of the deposited Cu film as well as by the line-edge roughness of the trench caused by the etching and deposition processes. Deterministic variations in the electrical

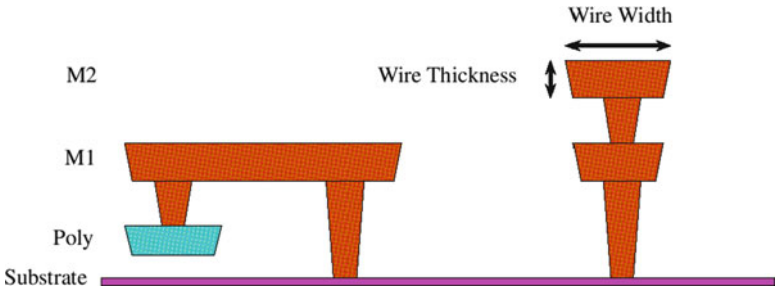


Fig. 3.7 caption

Table 3.5 Measures of the metal interconnect and their variability with causing process steps

Variation	Cause	Impact	Relevance
Trench CD or wire width	Lithography, etch	R,C	High
Trench height or wire height	Etch depth, CMP removal	R,C	High
Trench depth	ILD thickness variations, etch depth	C	High
Barrier thickness	Liner	R	Medium
Trench bottom shape	Liner etch	C, (R)	Medium
ILD thickness	Etch depth	C	High

parameters can be caused by dishing and erosion caused by the CMP process as it is dependent on the metal density in the neighborhood of the interconnect line. Figure 3.7 shows a schematic cross-section of a damascene Cu interconnect line. The geometrical parameters used there are explained in Table 3.5 together with their variability impact (Relevance).

3.2.3 Modeling of Variation of Electrical Parameters of Interconnects

Table 3.5 lists the most relevant geometrical parameters determining the interconnect variation. In a simplified model, those parameters can be used to model the electrical parameter variation. Figure 3.8 shows the width w and the thickness t of the trench, the height h of the dielectric layer, and the distance between the neighboring wires.

With those parameters and the model as described in [7], the total capacitance per unit length C' and resistance per unit length can be determined by

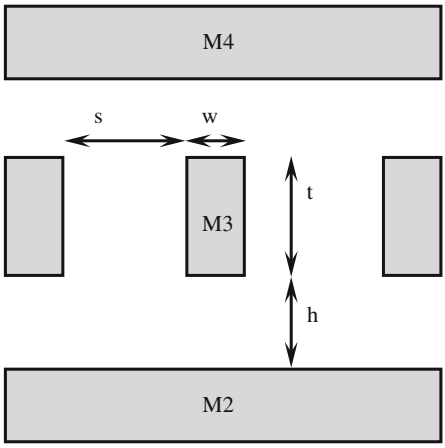


Fig. 3.8 Geometrical parameters for modeling of electrical parameters

$$C'^{a}_{\text{couple}} = \frac{t}{s}\left(1 - 1.5\mathrm{e}^{\frac{t}{2.5s}}\mathrm{e}^{-\frac{h}{0.31s}} + 1.5\mathrm{e}^{-\frac{h}{0.08s}} - 0.13\mathrm{e}^{-\frac{t}{1.3s}}\right) \tag{3.1}$$

$$C'^{fr}_{\text{couple}} = \left(\frac{h}{s}\right)^{0.2}\left(1.53 - 0.98\mathrm{e}^{-\frac{w}{0.35h}}\right)\cdot \mathrm{e}^{-\frac{s}{0.65h}} + 0.01 \tag{3.2}$$

$$C'^{a}_{\text{self}} = \frac{w}{h} \tag{3.3}$$

$$C'^{fr}_{\text{self}} = \left(1.05 + 0.63\mathrm{e}^{-\frac{t}{s}} - \mathrm{e}^{-\frac{s}{1.2h}}\right)\cdot\left(\frac{s}{s+2h}\right)^{0.05}\left(\frac{t}{h}\right)^{0.25} + 0.063 \tag{3.4}$$

$$C' = \varepsilon\left(2\cdot C'^{a}_{\text{couple}} + 2\cdot C'^{fr}_{\text{couple}} + 2\cdot C'^{a}_{\text{self}} + 2\cdot C'^{fr}_{\text{self}}\right) \tag{3.5}$$

$$R' = \rho\frac{1}{wt}. \tag{3.6}$$

Using a Monte Carlo approach, the RC-curve as shown in Fig. 3.9 can be obtained which is in good agreement with measured data.

The correlation plots in Fig. 3.10 show the impact of the variation of t, w, and h assuming constant pitch $s + w = constant$ on the electrical parameters. As can be seen from the plots, the resistance depends strongly on the thickness of the line and slightly on the width. The height of the dielectric does not have an impact as expected. For the capacitance variation, it can be seen that the width and thickness impact dominates in comparison with the height.

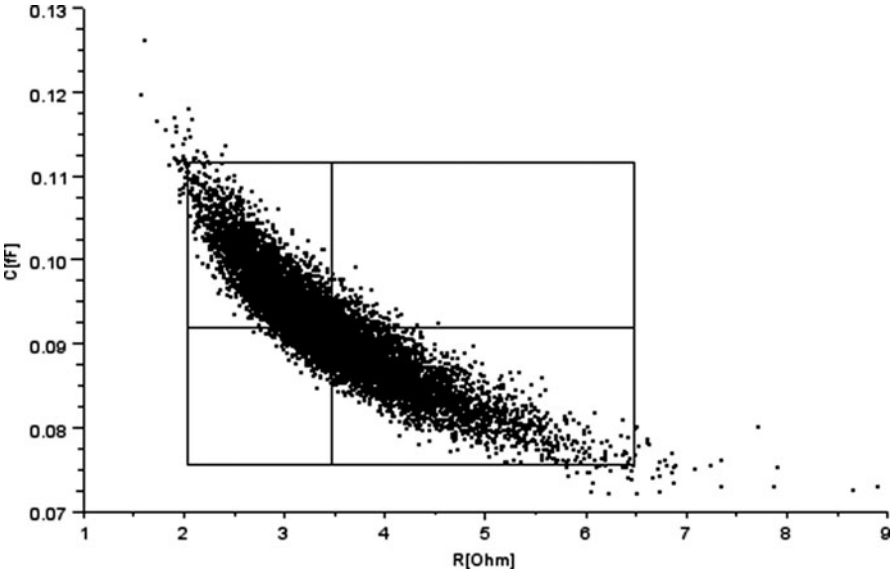


Fig. 3.9 Resistance vs. capacitance scatter plot

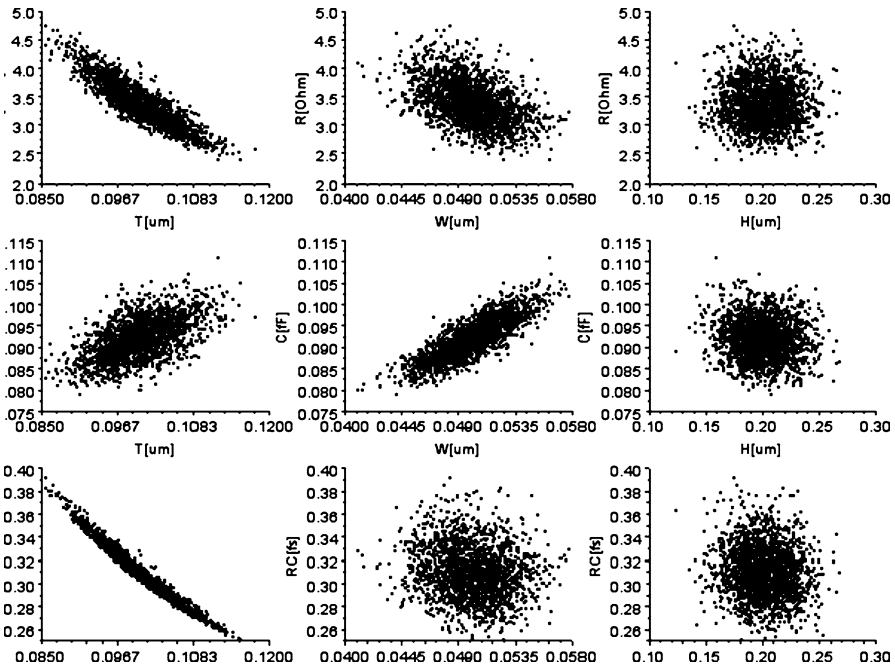


Fig. 3.10 Correlationsplots

3.2.4 Modeling of Variation in Post Layout Functional Verification

When performing post-layout functional verification of electrical circuits, it is necessary to check not only the correct function at nominal values, but also under variation of the electrical parameters of transistor and interconnect parasitics.

The most common way is to determine the variability of height, width, and thickness of the interconnect parameters used in Fig. 3.8 from electrical measurements and then to derive special backend corners based three sigma values of the parameters. By doing this corners with maximum and minimum coupling capacitance, maximum and minimal resistance as well as maximum and minimum RC product can be derived resulting in a least seven corners.

Since this approach requires multiple post-layout extractions and resulting files (SPEF, SPF, DSPF) contain data only of one particular corner, there have been efforts to combine all into one extraction run and result file providing so-called sensitivity extraction. This sensitivity extraction adds sensitivity data to the extracted parasitics about by how much a value will change based due to variation of an interconnect parameter such as width.

3.3 Mathematical Modeling of Process Variations

Parameter variations can be classified based on whether the variation occurs within one die (intra-die variations) or between different dies (inter-die variations). The latter variation can further be classified into die-to-die, wafer-to-wafer, and lot-to-lot, which however usually does not have a significant influence on analysis and optimization approaches for the design. Process parameter variations are modeled in parameters for transistor level simulation, which in turn are the foundation for modeling on gate level. Gate-level models are typically employed for statistical analysis and optimization during the design process. The timing and power behavior of gates can be modeled in linear or higher-order dependency on the process parameters. A linear model, commonly called the canonical model, has emerged as a de-facto standard in research. Models also differ in how they account for systematic vs. purely random variation, and how they incorporate correlation (both spatial and resulting from the circuit structure). In the following, we will use gate delay as an example to demonstrate the modeling of process variations. Other properties of a gate, e.g., power consumption, may be modeled similarly.

3.3.1 Decomposition of Process Variations

Process variations are usually classified into different categories [8], as shown in Fig. 3.11. Systematic variations can be determined before manufacturing. Once

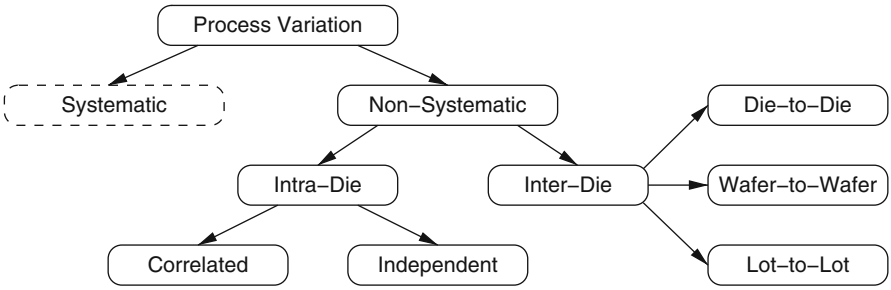


Fig. 3.11 Variation classification, adapted from [8]

physical synthesis is finished, these variations can be measured and modeled with fixed values. A typical example of systematic variations is the randomness of interconnect metal thickness. After layout and routing, the patterns of interconnects can be accurately analyzed. Therefore, the layout-related metal thickness variations in different areas can be predicted. With this information, the resistance and capacitance of interconnects can be modeled more accurately in sign-off analysis. Regarding active devices, gate length is affected by variations in lithography for mask optimization. These variations can be determined by computing the post-OPC gate lengths on the critical path to achieve more accurate timing analysis results [9]. In both cases, systematic variations are represented using fixed values instead of statistical variables. This is more accurate than simply analyzing circuit performance assuming random variations in metal thickness and from lithography. Both effects, however, can be incorporated only after physical synthesis. During the first iteration of logic synthesis, the circuit can only be optimized corresponding to the performance by modeling systematic variations as random variables.

Unlike systematic variations, nonsystematic variations cannot be determined before manufacturing. These variations result from the inaccuracy of process control and are independent of circuit design. Therefore, they can only be modeled with random variables in the complete design flow. Examples are variations in doping density and in layout-independent metal thickness of interconnects.

According to their spatial characteristics, nonsystematic variations are further partitioned into inter-die and intra-die variations. Inter-die variations affect all devices and interconnects on a die equally, i.e., all devices and interconnects have fully correlated random components. On a wafer, inter-die variations come from the nonuniformity of process control across the wafer surface. Therefore, chips at different positions have different performances. For example, the chips in the center of a wafer are normally faster than the chips near the periphery of the wafer. This type of variation is also called die-to-die variation. Similarly, wafer-to-wafer and lot-to-lot variations exist because of process control between wafers and lots.

Intra-die variations affect devices or interconnects inside a die differently. The physical parameters of two devices can shift in different directions, i.e. they are not fully correlated. Intra-die variations come from the inaccuracy of process control

across the surface of the die. For example, there is still a variation residue after modeling the systematic and inter-die variations of critical dimension (CD).

Furthermore, intra-die variations can be partitioned into a correlated part and an independent part. Although intra-die variations on devices or interconnects are not fully correlated, they still show a similar trend to some degree. This trend can be modeled by sharing the same variables as a part of intra-die variations, or by establishing correlation between these variations directly. Besides the correlated variation component, intra-die variations still exhibit a purely random effect. The purely random variations come from the random fluctuation during manufacturing processes, which thus imposes its effect on each device without correlation. Because of the inaccuracy of manufacturing equipments and process control, purely random variations exist in nearly every processing step. Examples are the random distortion of the lens used during the photolithography step and the purely random variation of the doping.

3.3.2 Correlation Modeling

Process variations are normally measured as a lumped distribution. Thereafter, the measured data are decomposed into different components [10]. The overall variations are then modeled as sums of these decomposed variables. Inter-die variations are shared by all devices or interconnects on the chip and cause correlation between their physical parameters, called global correlation or inter-die correlation. Because the uncertainties during manufacturing process vary continuously, intra-die variations exhibit proximity correlation. This correlation depends on the distance between two devices on the die [11]. The larger the distance is, the smaller the correlation becomes. For convenience, the correlation from intra-die variation is called local correlation.

Different methods are proposed to model correlation between process parameters. The quadtree model in [12, 13] uses different grid layers to model correlation between process parameters, as illustrated in Fig. 3.12. For a process parameter, a variable is assigned to each grid cell at each level. The process parameter of a device is modeled as the sum of all the variables of the grid cells directly above this device. The correlation between process parameters is established by sharing the same variables of the corresponding levels. Because the variable at level 0 is shared by all devices, it models the correlation from inter-die variation. The local correlation is modeled by sharing the variables with higher level numbers. If two devices are nearby on the die, they share more variables so that they have more correlation. If two devices are near enough to be located in the same grid cell at level 2, they become fully correlated. By increasing the number of grid, the accuracy of correlation modeling can be increased. However, this model can not represent the local correlation uniformly. For example, the distances from (2,4) to (2,1) and from (2,4) to (2,13) are equal. From this model, the parameters in (2,4) and (2,1) share the same variable at layer 1, but the same parameters in (2,4) and (2,13) do not

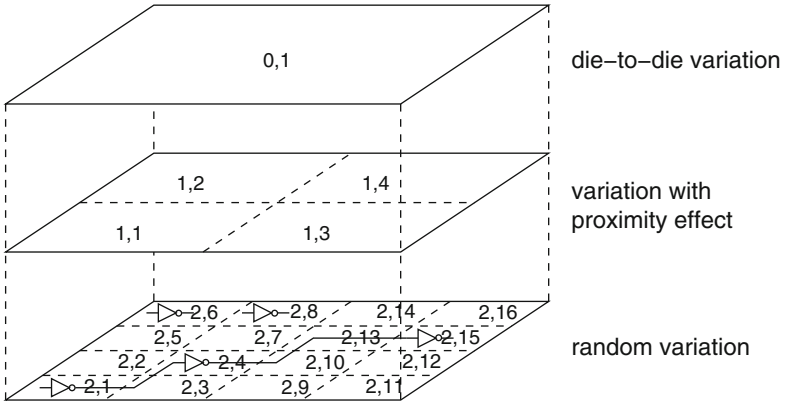


Fig. 3.12 Quadtree correlation model, adapted from [12, 13]

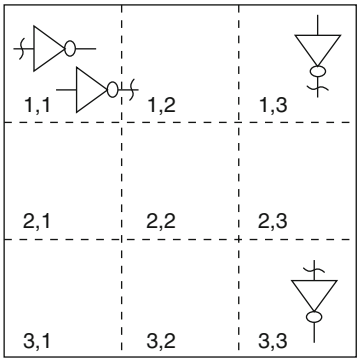


Fig. 3.13 Uniform grid correlation model [14]

share such variable. Consequently, correlations between parameters with the same distance may be modeled differently in this model. This contradicts the fact that intra-die correlation depends on distance between devices because of the proximity effect during manufacturing process.

Another correlation model is proposed in [14]. In this model, the die area is partitioned into a uniform grid, as shown in Fig. 3.13. For each grid cell, a random variable is assigned. The correlations between these random variables are computed or identified from the characterization of manufacturing technology, for example with the method in [15]. For n grid cells on the die, in total n variables are assigned. For the convenience of statistical timing analysis algorithms, the n correlated variables are decomposed into linear combinations of independent random variables, using an algorithm such as principal component analysis (PCA) [16]. After this decomposition, only the independent variables with large coefficients are kept in the linear combinations, so that the number of variables modeling correlation can be drastically reduced. This correlation model is very flexible because it can handle

any correlation between process parameters. The only reason to partition the die area to grid is to reduce the number of variables. For better modeling accuracy, a smaller cell size can be used, at the expense of a larger number of variables and larger correlation matrix. A similar correlation model is proposed in [17], where hexagonal grid cells are used to partition the die area. The advantage of such a model is that a grid cell in the partition has only one type of neighboring cell. Additionally, the distances from the neighbors of a cell to it are equal. This makes the hexagonal partition a better approximation in modeling proximity-related correlations.

Additionally, a correlation model is proposed in [18]. In this model, the die area is partitioned into grid with square cells. A process parameter in a grid cell is modeled as the sum of independent variables assigned to the corners of the grid cell. That is, each process parameter is decomposed into a linear combination of four independent random variables. This method can generate simple parameter decomposition, but no theoretical proof is provided for accuracy. Additionally, the method to map correlation data to the proposed model is not explained.

The correlations in the discussed models are all first-order. This is only enough to model the dependency between Gaussian random variables. To incorporate higher order dependency, methods such as independent component analysis [19] can be used, e.g., in [20, 21].

3.3.3 Process Parameter Modeling

The first step of statistical timing analysis is to model process variations in a form that can simplify modeling of gate delays and arrival time propagation. A process parameter is a sum of components modeling inter-die variations, intra-die variations, and purely random variations. The additive form of a process parameter p is written as

$$p = p_0 + p_g + p_l + p_r, \tag{3.7}$$

where p_0 is the nominal value of the parameter. p_g models the inter-die variation and is shared by all gates. p_l is the intra-die variation specific to each gate and is correlated with each other. p_r is an independent variable modeling the purely random effect in manufacturing processes.

The parameter p for a device may have Gaussian or non-Gaussian variations. In [14, 22, 23], all process variations are assumed as Gaussian in order to reduce the complexity of timing analysis. The Gaussian assumption, however, cannot provide enough accuracy because only the first two moments of process parameters are captured. To improve modeling accuracy, non-Gaussian variables are used in [24]. Additionally, the independent component analysis based non-Gaussian model is proposed in [20, 21]. In both methods, the random variables representing process variations can be in any form in addition to Gaussian.

3.3.4 Gate Delay Representation

Statistical timing analysis uses abstracted gate delays to evaluate circuit performance. A gate delay is defined as the time difference between points of measurement of the input and output waveforms. For a given input waveform, the output waveform of a gate depends on transistor parameters of the gate. For example, the effective gate length affects the gate delay dominantly. Assuming that all process parameters are denoted as a vector $\mathbf{p}$, a gate delay W is expressed as

$$W = f(\mathbf{p}), \tag{3.8}$$

where f denotes the mapping function from process parameters to the gate delay. The mapping function is theoretically very complex. Therefore, SPICE simulation is often used to obtain accurate samples of gate delays.

With process variations considered, a gate delay becomes a random variable. Because of the correlation between process variations, gate delays are correlated with each other. For example, the delays of two gates vary in a similar way when these gates are near on the die. When their distance is large, both gate delays exhibit more randomness. In order to incorporate the correlation from process variations, gate delays are described as simplified functions of process parameters, instead of identifying the numeric characteristics, e.g., means and standard deviations, of their distributions directly. In other words, the mapping function f in (3.8) is replaced with a simpler form at the expense of accuracy.

The canonical delay model in [14, 23] uses linear mapping functions. A gate delay in this method is expressed as

$$W = \mathbf{kp}, \tag{3.9}$$

where $\mathbf{k}$ is the coefficient vector and can be computed by sensitivity analysis [13], or identified by linear regression [25] from the results of SPICE-based Monte Carlo simulation.

According to (3.7), a parameter is partitioned into different parts. If each variable in (3.9) is replaced into the form of (3.7), the gate delay is transformed as

$$W = \mathbf{kp}_0 + \mathbf{kp}_g + \mathbf{kp}_l + \mathbf{kp}_r = W_0 + \mathbf{kp}_g + \mathbf{kp}_l + p_\tau. \tag{3.10}$$

In (3.10), $\mathbf{p}_0$ represents nominal values of parameters and all its elements are fixed, so that $\mathbf{kp}_0$ can be merged into a constant W_0. Because the first-order moments are merged into W_0, the means of $\mathbf{p}_g$, $\mathbf{p}_l$ and $\mathbf{p}_r$ are all zero. Representing inter-die variations, $\mathbf{p}_g$ is shared by all gate delays. $\mathbf{p}_r$ models purely random manufacturing effects, so that it is merged into one random variable p_τ. Unlike the other vectors in (3.10), $\mathbf{p}_l$ models correlated intra-die variations and needs further processing.

As discussed in Sect. 3.3.2, proximity correlation exists between within-die variables. Consider two gate delays W_a and W_b,

$$W_a = W_{0,a} + \mathbf{k}_a \mathbf{p}_g + \mathbf{k}_a \mathbf{p}_{l,a} + p_{\tau,a} \tag{3.11}$$

$$W_b = W_{0,b} + \mathbf{k}_b \mathbf{p}_g + \mathbf{k}_b \mathbf{p}_{l,b} + p_{\tau,b}, \tag{3.12}$$

where $\mathbf{p}_{l,a}$ and $\mathbf{p}_{l,b}$ are correlated random variables. During arrival time propagation, these random variables cannot be merged because of their correlation. Additionally, the correlation between $\mathbf{p}_{l,a}$ and $\mathbf{p}_{l,b}$ causes the computation of the correlation between W_a and W_b to be very slow, as will be explained later.

In order to reduce the runtime of timing analysis, PCA [16] is used to decompose correlated random variables. Assume that variable vector $\mathbf{p_l}$ with m elements is the vector containing all the correlated random variables modeling within-die process variations, so that $\mathbf{p}_{l,a}$ and $\mathbf{p}_{l,b}$ both are parts of $\mathbf{p_l}$. The correlation matrix of $\mathbf{p_l}$ is denoted as $\mathbf{C}$. Under Gaussian assumption, each element in $\mathbf{p_l}$ can be expressed as a linear combination of a set of independent components after applying PCA.

$$\mathbf{p_l} = \mathbf{A}\mathbf{x} \approx \mathbf{A}^r \mathbf{x}^r, \tag{3.13}$$

where $\mathbf{A}$ is the orthogonal transformation matrix formed by the eigenvectors of $\mathbf{C}$. $\mathbf{x} = [x_1, x_2, \ldots x_m]^T$ is a vector of independent Gaussian random variables with zero mean. The standard deviation vector of $\mathbf{x}$ is formed by the square root of eigenvalues of $\mathbf{C}$ corresponding to the eigenvectors in $\mathbf{A}$. If there are eigenvalues which are very small compared to other larger eigenvalues, the corresponding variables in $\mathbf{x}$ contribute relatively less than other variables in (3.13). Therefore, these variables can be discarded to reduce the number of the independent variables. Assume $\mathbf{x}$ is truncated to $\mathbf{x}^r$ with n_r variables. The original intra-die variations can be approximated by linear combinations of the n_r independent random variables $\mathbf{x}^r$. $\mathbf{A}^r$ is a column truncated matrix of $\mathbf{A}$.

Because any random variable from $\mathbf{p_l}$ can be approximated by a linear combination of $\mathbf{x}^r$ by selecting the row of $\mathbf{A}^r$ corresponding to the random variable, as shown in (3.13), the gate delay in (3.10) can be written as

$$W = W_0 + \mathbf{k}\mathbf{p}_g + \mathbf{k}\mathbf{A}_s^r \mathbf{x}^r + p_\tau \tag{3.14}$$

$$= c_0 + \sum_{i=1}^{n} c_i v_i + c_r v_r, \tag{3.15}$$

where $\mathbf{A}_s^r$ is formed by the rows of $\mathbf{A}^r$ corresponding to the variables of $\mathbf{p}_l$ in (3.10). The gate delay in (3.14) is generalized into the canonical linear delay form [23] as in (3.15), where v_i are independent random variables and shared by all gate delays. v_r is the purely random variably specific for each delay. c_0 is the nominal value of the delay. c_i and c_r are the coefficients of the random variables. The correlations between gate delays are represented by sharing the same set of random variables v_i.

In the canonical delay model (3.15), the mapping function f from parameters to delays is assumed as linear. With such linear delay form, arrival times can be propagated very fast with simple computations [23]. The expense of this simple

assumption is the loss of accuracy [26,27]. To improve the approximation accuracy, quadratic timing models are proposed in [18, 27, 28], where a gate delay is mapped as a second-order function of process parameters. If PCA is still used to decompose correlated random variables, a parameter in the quadratic form is replaced by a linear combination of uncorrelated random variables. For a second-order term, this replacement results in many cross terms, which make timing analysis complex and slow. To reduce the number of cross terms in a quadratic model, orthogonalization method is used in [27]. In addition to quadratic models, a gate delay is mapped as a linear function of independent Gaussian and non-Gaussian variables in [20,21]. A more general delay mapping method is proposed in [24]. In this model, a gate delay is mapped as a sum of linear and nonlinear functions of Gaussian and non-Gaussian random variables. Therefore, it can handle any delay functions without limitation. Using non-Gaussian random variables can improve the modeling accuracy of process variations; using nonlinear functions can improve the accuracy of approximating the mapping from process parameters to gate delays and interconnect delays. Both methods, however, increase complexity in the following steps of statistical timing analysis.

References

1. Bernstein, K., Frank, D., Gattiker, A., Haensch, W., Ji, B., Nassif, S., Nowak, E., Pearson, D., Rohrer, N.: High-performance cmos variability in the 65-nm regime and beyond. IBM Journal Research Development **50**(4), 433–449 (2006)
2. Cheng, B., Dideban, D., Moezi, N., Millar, C., Roy, G., Wang, X., Roy, S., Asenov, A.: Benchmarking statistical compact modeling strategies for capturing device intrinsic parameter fluctuations in BSIM4 and PSP. IEEE Design Test of Computers **PP**(99), 1–1 (2010). DOI 10.1109/MDT.2010.2
3. Asenov, A., Brown, A., Davies, J., Kaya, S., Slavcheva, G.: Simulation of intrinsic parameter fluctuations in decanaonmeter and nanometer-scale mosfets. Electron Devices, IEEE Transactions on **50** (2003)
4. Bernstein, K.: High Speed CMOS Design Styles. Springer Publishing Company, Incorporated (1998)
5. Orshansky, M., Nassif, S., Boning, D.: Design for Manufacturability and Statistical Design. Springer (2008)
6. International technology roadmap for semiconductors (2009)
7. Sim, S.P., Krishnan, S., Petranovic, D., Arora, N., Lee, K., Yang, C.: A unified RLC model for high-speed on-chip interconnects. IEEE Transactions on Electron Devices **50**(6), 1501–1510 (2003)
8. Blaauw, D., Chopra, K., Srivastava, A., Scheffer, L.: Statistical timing analysis: From basic principles to state of the art. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **27**(4), 589–607 (2008). DOI 10.1109/TCAD.2007.907047
9. Yang, J., Capodieci, L., Sylvester, D.: Advanced timing analysis based on post-OPC extraction of critical dimensions. In: ACM/IEEE Design Automation Conference (DAC), pp. 359–364 (2005)
10. Stine, B.E., Boning, D.S., Chung, J.E.: Analysis and decomposition of spatial variation in integrated circuit processes and devices. IEEE Transactions on Semiconductor Manufacturing **10**(1), 24–41 (1997)

11. Cline, B., Chopra, K., Blaauw, D., Cao, Y.: Analysis and modeling of CD variation for statistical static timing. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 60–66 (2006)
12. Agarwal, A., Blaauw, D., Zolotov, V.: Statistical timing analysis for intra-die process variations with spatial correlations. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 900–907 (2003)
13. Agarwal, A., Blaauw, D., Zolotov, V., Sundareswaran, S., Zhao, M., Gala, K., Panda, R.: Statistical delay computation considering spatial correlation. In: IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 271–276 (2003)
14. Chang, H., Sapatnekar, S.S.: Statistical timing analysis considering spatial correlations using a single PERT-like traversal. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 621–625 (2003)
15. Xiong, J., Zolotov, V., He, L.: Robust extraction of spatial correlation. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **26**(4), 619–631 (2007)
16. Jolliffe, I.T.: Principal Component Analysis, 2. edn. Springer (2002)
17. Chen, R., Zhang, L., Visweswariah, C., Xiong, J., Zolotov, V.: Static timing: back to our roots. In: IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 310–315 (2008)
18. Khandelwal, V., Srivastava, A.: A general framework for accurate statistical timing analysis considering correlations. In: ACM/IEEE Design Automation Conference (DAC), pp. 89–94 (2005)
19. Hyvärinen, A., Karhunen, J., Oja, E.: Independent component analysis. Wiley & Sons (2001)
20. Singh, J., Sapatnekar, S.: Statistical timing analysis with correlated non-Gaussian parameters using independent component analysis. In: ACM/IEEE Design Automation Conference (DAC), pp. 155–160 (2006)
21. Singh, J., Sapatnekar, S.S.: A scalable statistical static timing analyzer incorporating correlated non-Gaussian and Gaussian parameter variations. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **27**(1), 160–173 (2008)
22. Kang, K., Paul, B.C., Roy, K.: Statistical timing analysis using levelized covariance propagation. In: Design, Automation and Test in Europe (DATE), pp. 764–769 (2005)
23. Visweswariah, C., Ravindran, K., Kalafala, K., Walker, S., Narayan, S.: First-order incremental block-based statistical timing analysis. In: ACM/IEEE Design Automation Conference (DAC), pp. 331–336 (2004)
24. Chang, H., Zolotov, V., Narayan, S., Visweswariah, C.: Parameterized block-based statistical timing analysis with non-Gaussian parameters, nonlinear delay functions. In: ACM/IEEE Design Automation Conference (DAC), pp. 71–76 (2005)
25. Seber, G.: Linear Regression Analysis. John Wiley & Sons (1977)
26. Li, X., Le, J., Gopalakrishnan, P., Pileggi, L.T.: Asymptotic probability extraction for non-normal distributions of circuit performance. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD) (2004)
27. Zhan, Y., Strojwas, A.J., Li, X., Pileggi, L.T., Newmark, D., Sharma, M.: Correlation-aware statistical timing analysis with non-Gaussian delay distributions. In: ACM/IEEE Design Automation Conference (DAC), pp. 77–82 (2005)
28. Zhang, L., Chen, W., Hu, Y., Gubner, J.A., Chen, C.C.P.: Correlation-preserved non-Gaussian statistical timing analysis with quadratic timing model. In: ACM/IEEE Design Automation Conference (DAC), pp. 83–88 (2005)

Chapter 4
Methods of Parameter Variations

Christoph Knoth, Ulf Schlichtmann, Bing Li, Min Zhang, Markus Olbrich, Emrah Acar, Uwe Eichler, Joachim Haase, André Lange, and Michael Pronath

Chapter 4 presents various dedicated methods that support variability handling in the design process. Using these methods, the designer can analyze the effect of variations on his design and identify possible improvements.

An important requirement for modeling digital circuits is a precise characterization of the used library cells. The first two sections are devoted to this task. In Sect. 4.1, models will be described which characterize the timing behavior of digital logic cells. In addition, different approaches to model the timing will be explained. These approaches will be evaluated with respect to their efficiency. The main part of the section is dedicated to current source models (CSMs) which represent the frontier of academic research. Their structure will be explained and an efficient concept

C. Knoth (✉) • U. Schlichtmann • B. Li
Technische Universität München, Institute for Electronic Design Automation, Arcisstraße 21, 80333 Munich, Germany
e-mail: christoph.knoth@tum.de; ulf.schlichtmann@tum.de; b.li@tum.de

M. Zhang • M. Olbrich
Institute of Microelectronic Systems, Leibniz University of Hannover, Appelstraße 4, 30167 Hannover, Germany
e-mail: min.zhang@ims.uni-hannover.de; markus.olbrich@ims.uni-hannover.de

E. Acar
Research Staff Member, IBM Research, IBM T. J. Watson Research Center, 1101 Kitchawan Rd, Yorktown Heights, NY 10598, USA
e-mail: emrah@us.ibm.com

U. Eichler • J. Haase • A. Lange
Fraunhofer Institute for Integrated Circuits IIS, Design Automation Division EAS, Zeunerstraße 38, 01069 Dresden, Germany
e-mail: uwe.eichler@eas.iis.fraunhofer.de; joachim.haase@eas.iis.fraunhofer.de; andre.lange@eas.iis.fraunhofer.de

M. Pronath
MunEDA GmbH, Stefan-George-Ring 29, 81929 Munich, Germany
e-mail: michael.pronath@muneda.com

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6_4,

for characterizing them will be shown. The first section draws the conclusion that CSMs are well-suited to deal with the requirements of advanced nanometer process technologies.

In Sect. 4.2 of this chapter, methods to generate individual cell models adapted to the accuracy requirements will be presented. The approach is based on rapid higher-order polynomial modeling. The developed methods enable an automation of the characterization of huge libraries. The effectiveness of the approach will be shown by experiments on industrial standard cell libraries which will demonstrate gate level leakage characterization. However, this approach is not restricted to a special circuit performance.

Statistical Static Timing Analysis (SSTA) is one of the advanced approaches that handles variations of gate and interconnect delays in the design of digital circuits. Therefore, Sect. 4.3 describes how SSTA works and which results can be expected.

Besides time performance, energy consumption is the second most important constraint for the design of digital circuits. The static as well as the dynamic energy consumption are going to be demonstrated in Sects. 4.4 and 4.5. Recently, leakage power has become a more and more dominant performance limiter in integrated circuit technologies resulting in limitations of downscaling. Thus, planning, estimation, and optimization of leakage power have become a major design objective. Different types of leakage current are described in Sect. 4.4. In addition, leakage models for logic cells and circuits will be introduced. Methods for leakage estimation for a device will be described and generalized to larger circuit blocks. Finally, parametric variability of leakage is also going to be discussed.

In Sect. 4.5, two aspects of dynamic power consumption are going to be covered: first, a probabilistic method for the analysis of glitches and the additional power caused by them, and second, a new digital simulation method for precise statistical power estimation. The second approach uses cell libraries extended by additional process parameter data which are accessed directly by the cell model during digital gate-level simulation. This enables a process-dependent analysis of timing and power as well as an increased simulation accuracy by direct consideration of signal slope times, which is also suitable for advanced glitch detection.

Section 4.6 introduces the basics of a commercial tool that considers the operating range and variations of the manufacturing process in order to increase the yield especially for basic building blocks given on the transistor level. To this end, the concepts of worst-case point and worst-case distance will be introduced and applied on nominal optimization and a design centering procedure. The main optimization steps will be explained with the help of an example.

Section 4.7 deals with concepts for robustness analysis. Robustness analysis aims at determining the contributors to circuit performance variability. A measure for cell robustness will be defined that takes into account the variability of different performance parameters. Such a measure allows for evaluating and comparing different realizations of a cell or a cell library with respect to their robustness. By using different kinds of examples, the section provides several hints for the usage of the robustness analysis.

In summary, this chapter gives an overview on different aspects and current possibilities that allow for investigating circuits with regard to manufacturing variations and for reducing the influence of these variations on the final design.

Christoph Knoth and Bing Li are the authors of Sects. 4.1 and 4.3 respectively. Ulf Schlichtmann delivered contributions to both sections. Min Zhang and Markus Olbrich are the authors of Sect. 4.3. Emrah Acar wrote Sect. 4.4. The Sect. 4.5 was written by Markus Olbrich, Uwe Eichler and Joachim Haase. Authors of the Sects. 4.6 and 4.7 are Michael Pronath and André Lange respectively.

4.1 Characterization of Standard Cells

In industrial design flows, library standard cells today are represented in Nonlinear Delay Models (NLDMs). This model and its limitations are described here. It has been recognized since a number of years that this model appears to be increasingly incapable of dealing with the requirements of advanced process technologies. Almost a decade ago, the Scalable Polynomial Delay Model (SPDM) was proposed as an alternative, but failed to catch on in industry. Today, the focus is on current source models (CSMs). EDA companies have proposed two types of CSM, and there is also much academic research on CSMs. Unfortunately, the term CSM refers to significantly different approaches in the commercial realm (used by EDA companies) and in academic research. The concepts and shortcomings of commercial CSMs will be explained as well in the following as an overview of current academic research on CSMs will be given. We will also describe how the required CSM parameters can be determined by efficient characterization methods.

4.1.1 Introduction

Digital integrated circuits are typically implemented using at least partly a semi-custom design methodology based on standard cells. These standard cells have been designed, sized, and layouted individually. They are provided as libraries to the design teams, which then use these cells to create logic functions. The important information to describe a standard cell are logic function, area, power consumption, driver strength, and timing. In this section, the focus is on timing information of standard cells. Different delay models at gate level are reviewed. The approaches are sorted by increasing accuracy, which also reflects the chronological order in which they were introduced.

In the following, a logic stage is defined to span from the input of a logic cell to the input of subsequent logic cells while including polysilicon and metallic interconnect. With this definition, the total signal delay of a stage is given as

$$d_{\text{stage}} = d_{\text{intrinsic}} + d_{\text{load}} + d_{\text{ic}}. \tag{4.1}$$

Hereby $d_{\text{intrinsic}}$ denotes the intra-cell delay contribution for switching transistor capacitances and charging internal nets. d_{load} accounts for the time needed to charge the attached interconnect and input capacitances of connected logic cells. d_{ic} finally models the propagation delay along the passive interconnect. Despite the formal separation in (4.1), stage delays are definitely nonlinear functions and these three influences cannot be distinguished so clearly. On the other hand, at the time of generating the timing library for the standard cells, nothing is known about the loads and interconnects which have to be driven. The only basis for all timing models is the SPICE subcircuit file. It may include parasitic resistors and capacitors from the layout extraction. By performing transient simulations with the individual subcircuits, timing information is obtained and can be used to create a timing model.

4.1.2 Fixed and Bounded Delay Model

In the early days of IC design, power consumption was of minor interest and also the potential for delay optimization on gate level was neglected. In fact, gates for driving different loads have been sized to match a fixed, globally common, delay value [1]. Since feature sizes were large, interconnect resistivity was small and thus interconnect delay negligible. Hence, the total delay of a combinatorial block was known before placement and routing as the number of stages in the longest path.

To meet the demand for more area and power efficiency while raising operating frequencies, standard cells have been designed to have different delay values. Hence, instead of a globally common delay value, each cell was described by an individual fixed cell delay $d_{\text{intrinsic}}$. A variant of the fixed delay model is the bounded delay model, which states a minimum and a maximum delay value for each cell. Despite their limited accuracy, such models are still valid in early design phases for quickly providing delay estimates.

4.1.3 Load-Slew-Models

With exponentially decreasing VLSI feature sizes and increasing instance count, capacitive interconnect load and interconnect delay became significant. Since the actual workload of a cell is unknown before place and route, delay models must be parametric in terms of the capacitive output load. The continued shrinking of feature sizes also involved lowering the supply voltage to reduce field strength and leakage currents in MOS transistors. The reduced voltage swing required more accurate signal models than the step function [2]. The new waveform model used the transition time tr_{in}, also named slew or slope, to describe the finite time needed for switching logic values. Avoiding ambiguous tails at the beginning and end of transitions, slew refers to a voltage span from typically 10–90% or 30–70% of the total voltage swing.

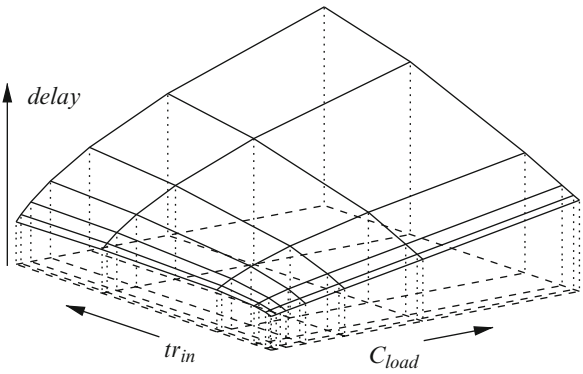


Fig. 4.1 NLDM cell delay lookup table

4.1.3.1 Nonlinear Delay Model (NLDM)

The nonlinear delay model (NLDM) is widely used in industry. It provides values for output slew and cell delay as a function of capacitive output load C_{load} and input transition time tr_{in}. These values are stored in lookup tables and are provided for every timing arc, i.e., form every input pin to every output pin and for all combinations of other input pins that make the cell sensitive to the switching input pin. Figure 4.1 depicts the tabulated delay values for a buffer cell. The values of C_{load} and tr_{in} are usually stored together with the lookup table.

Generating NLDMs is done by attaching capacitors of different values to the output of the cell and applying input signals with different transition times. Usually, the waveforms are smoothed ramp waveforms which represent typical waveforms. For each of these combinations the delay values and output slew are measured. The values are stored using one of different industry standard timing formats: Synopsys' Liberty format (.lib), its extended version Advanced Library Format (ALF), or Cadence's Timing Library Format (TLF).

4.1.3.2 Handling Parameter Variation

The cell delay obviously is also a function of process parameters such as oxide thickness as well as the environmental parameters supply voltage and temperature. To account for their impact, cell performances are analyzed at different points of the parameter space, denoted as Process-Voltage-Temperature (PVT) corners (see also Sect. 2.2 and [3]). The PVT corners are obtained by enumerating all permutations of parameters being set to their bounds or three sigma values. Since the delay is usually monotonic in a parameter, the PVT corners are the parameter vectors for which the cells have extreme delay values.[1] PVT corners are derived for the whole library and

[1]Finding corner cases is nontrivial not only because of the large number of parameters. For some parameters, such as temperature, nonlinear dependencies have been observed [4].

not for individual cells or designs. Since PMOS and NMOS transistors, and hence rising and falling delay, are affected differently by parameters, four different corners are distinguished: fast–fast, fast–slow, slow–fast, and slow–slow. The whole cell library is characterized at each of these four PVT corners. Static timing analysis is then performed for the different PVT corners using the corresponding timing library.

In addition to global variation which affects the entire chip and is modeled by PVT corners, there is also local variation of process parameters, supply voltage and temperature. Each cell instance will therefore have an individual delay, which slightly differs from the value stored in the library. This complicates timing validation since it is possible that although the chip is generally slower than expected (slow–slow PVT corner), signal propagation along individual paths might be not as bad. This could result in violating the hold time constraint if such a path is the clock path to the registers at the end of a combinatorial block. To account for such scenarios, the individual variations of supply voltage and temperature are estimated by power and IR drop analysis. Their impact on cell delay around the defined extreme PVT corners is then modeled by applying linear scaling factors to the cell delay, known as "k-factors". These k-factors are also used in when a chip is to be designed with a slightly different supply voltage and/or operating temperature profile than the cell library was characterized for, to avoid the costly effort of recharacterizing an entire cell library.

Similarly, the derating factors for on-chip variation (OCV) account for local variation of process parameters. Hence, for the above-mentioned scenario different OCV derating factors might be set to decrease signal delay along the clock path to the capture registers while increasing delay values of cells in data paths [5]. Note however that derating factors have no physical meaning. They are used to artificially worsen the timing behavior but cannot be related to particular parameters.

Newer approaches refine this concept to reduce pessimism which is introduced by assigning one derating factor for all data paths and one to all clock paths. Since OCV models statistical variation of cell delays, both the number of cells in a path and its geometrical footprint after placement should influence the derating factor.

This methodology is hence denoted as level-based and location-based OCV [6]. Foundries provide lookup tables with derating factors in terms of total number of cells in the path and length of the diagonal of the enclosing rectangle. An example is given in Fig. 4.2. The more cells in a path, the smaller the derating factor to account for statistical averaging of faster and slower cells. If the placement is already known, also the distance can be used to further adjust derating factors. Note also that the clock paths begin at the branching point. This is done to reduce the so-called common path pessimism; cells like the first buffer might be slow or fast, but they contribute the same delay to both paths.

4.1.3.3 Scalable Polynomial Delay Model (SPDM)

NLDM standard cell libraries might require a significant amount of memory (see also Sect. 4.1.6). For every timing arc separate lookup tables of delay values and

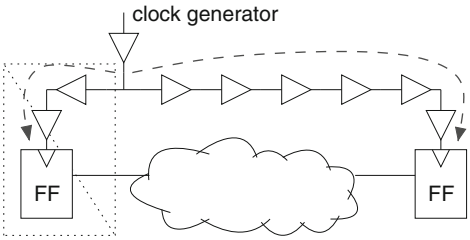


Fig. 4.2 Advanced on-chip variation based on levels and location

Distance	Level				
	1	2	4	8	16
0	1.14	1.13	1.13	1.12	1.09
1000	1.15	1.14	1.14	1.12	1.10
2000	1.17	1.16	1.14	1.13	1.12
4000	1.19	1.17	1.16	1.15	1.14

output transition times are generated for rising and falling transitions. Since the whole timing library must be kept in memory during timing analysis, it is desirable to express cell delays more compactly.

It can be seen in Fig. 4.1 that for small transition times tr_{in} the cell delay of this buffer is almost linearly dependent on the output load C_{L}. For large values of tr_{in} a square root function fits better. Hence, the superposition of different functions of output load C_{L} and transition time tr_{in} can be used to provide characteristic delay equations such as

$$d = k_1 + k_2 \cdot C_{\text{L}} + k_3 \cdot C_{\text{L}}^3 + (k_4 \cdot k_5 \cdot C_{\text{L}}) \cdot tr_{\text{in}}. \tag{4.2}$$

These templates of delay equation are fitted to the delay values measured during simulation by adjusting the coefficients k_i. Still the selection of adequate template functions is nontrivial. The most general approach for delay approximation is the scalable polynomial delay model (SPDM). The model consists of a sum of base functions ϕ, weighted by model coefficients k_i.

$$d = \sum_{i}^{n_k} k_i \phi_i(\mathbf{x}) = \sum_{i}^{n_k} k_i \prod_{j=0}^{n_x} x_j^{j_i} \quad \text{with} \quad j_i \leq m_x. \tag{4.3}$$

These base functions are polynomials in terms of the n_x model parameters x_j with a maximum order of m_x. This framework of (4.3) also allows to model the delay dependence on process parameters by including them into the vector of model parameters. This unified delay equation therefore has two major advantages compared to NLDM. The first is a much more compact library representation. Typical NLDMs use 7×7 lookup tables indexed by output load and transition time. To provide delay values for three different supply voltage, three temperatures, and three process parameters, already $3 \cdot 3 \cdot 3 \cdot 49 = 1{,}323$ entries have to stored. On the

other hand, the SPDM uses a much smaller number of coefficients. Furthermore, this SPDM yet provides more information. While NLDM accounts for small parameter changes by applying linear derating factors to all values in the lookup table, SPDM also captures higher order effects and cross term dependencies of parameters, output load, and input transition time. Nonetheless, SPDM is not widely used in industry. One reason might be the higher complexity in library generation. The polynomial order must be chosen with care to trade between required accuracy and number of coefficients. On the other hand, while polynomials of high order are needed for accuracy at the measured points, this might lead to severe errors elsewhere due to oscillation. Library generation therefore not only requires additional time and manpower to fit the models but also to verify library consistency and quality.

Finally, since the introduction of SPDM in 2000 the amount of available memory and hard disks space has continuously increased and eased at least NLDM's size drawbacks.

4.1.3.4 Receiver Modeling

The presented NLDM and SPDM timing models provide cell delays as functions of the transition time of the input signal and the capacitive load. This output load represents the interconnect and the receiver models of all attached logic cells. Hence, besides the timing information output slew and delay, every cell provides at least one value describing its input capacitance. This capacitance represents the amount of charges that flows into the cell's input pin and will be stored on parasitic capacitances and gate capacitances of the transistors during a signal transition (see Fig. 4.6 on page 103). It is therefore characterized by attaching a ramp voltage source at the input pin and integrating current through this voltage source.

$$C_{\text{in}} = \frac{\Delta Q}{\Delta V} = \frac{1}{\Delta V} \cdot \int i_{V_{\text{IN}}}(t)\mathrm{d}t. \tag{4.4}$$

Note, however, that a cell's input capacitance is not independent of its output load. The transition at the input might result in an output transition. Due to capacitive coupling inside a cell and the nonlinearity of the transistor capacitances, the input capacitance further depends on the input slew. Nonetheless, output load and input slew have significantly less impact on input capacitance than on the cell delay. Therefore, only minimum and maximum values are provided in the libraries.

4.1.3.5 Resistive Interconnects

Load-slew models are based on the assumptions that signal waveforms are sufficiently described by linear ramps and that the output load is purely capacitive. Due to high integration and reduced feature sizes, interconnect resistance became notable, being the source of three major challenges. First, modeling interconnect delay became imperative since it accounts for significant fractions of the total path delay (see also Sect. 3.2). Second, voltage waveforms at the output of the driver

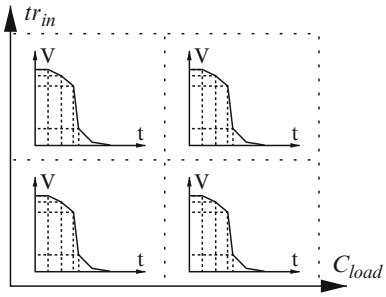


Fig. 4.3 ECSM stores output voltage waveforms for different combinations of input slew and output load

and the input of the receivers are different due to the low pass characteristics. Furthermore, the impact of the transition tails on cell delay became significant. Third, logic cells have been characterized using purely capacitive loads. However, a driver's load is not just the sum of attached receiver input capacitances plus interconnect loading. Resistive shielding reduces the load "seen" by a driver and therefore decreases the cell delay. The concept of effective capacitances account for the last two effects [7]. It matches the average current injection into the output net such that the crossing points of the delay voltage threshold coincide. However, effective capacitance and driver waveform are mutually dependent resulting in an iterative process to determine stage delays and output waveforms. More detailed descriptions are given in [7, 8].

4.1.3.6 Effective Current Source Model (ECSM) and Composite Current Source Model (CCS)

EDA vendors introduced new delay models named ECSM [9] and CCS [10] to account for the growing influence of resistive interconnect loads. Both are compatible with the existing library formats for NLDM. Values for cell delay and output slew are still tabulated in terms of input slew and capacitive output load. The real improvements are more accurate driver and receiver models. For every combination of output load and input slew, additionally the output voltage waveform (ECSM) or the current waveform (CCS) is stored. Figure 4.3 illustrates the ECSM-lookup table. Note that there is no difference between the two models since one can be converted to the other through

$$i_{\text{out}} \approx C_{\text{load}} \cdot \frac{V(t_{n+1}) - V(t_n)}{t_{n+1} - t_n}. \tag{4.5}$$

This transient information is used for driver modeling. The current flowing into a particular load is obtained by interpolating between different waveforms that are stored in the lookup table. It is finally integrated over time to obtain output waveform and stage delay.

The receiver model for CCS are two time-wise defined capacitances. This is conceptionally similar to (4.4). Only the voltage difference is not the complete

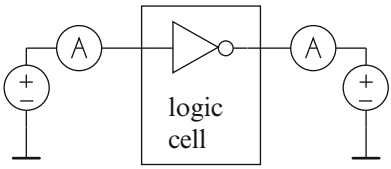


Fig. 4.4 Current source model characterization setup to measure port currents instead of cell delays

swing. Instead, a transition is split at the timepoint when the delay threshold voltage is crossed. The first capacitance models the average current during the first part of the transition, the second the latter. CCS receiver capacitances can be modeled as lookup tables in terms of transition time, or in timing arc notation in terms of transition time and output load.

4.1.3.7 Analytical Cell Delay Models

Some approaches have been published for cell delay prediction based on analytical expressions. The principle is to derive a formula (4.6) and solve it for the cell delay d.

$$|V_{\text{out}}(t_{arrival}) - V_{\text{out}}(t_{\text{arrival}} + d)| = \frac{1}{2}V_{\text{DD}} = \int_{t_{\text{arrival}}}^{t_{\text{arrival}}+d} f\,(C_{\text{load}}, tr_{\text{in}}, \mathbf{p})\mathrm{d}t. \tag{4.6}$$

The model parameters **p** are obtained through SPICE simulations. The principle of most approaches is to model the voltage trajectory across the cell's output current. Usually, the coefficients of simple MOS transistor current equations based on a power law are fitted to the cell under consideration [11]. Provided there is a linear ramp input signal and a purely capacitive load, explicit delay equations can be derived. Nonetheless, these approaches usually only work for simple cells such as inverters without parasitics. More complex cells have to be mapped to an effective inverter [12].

Analytical cell delay models never gained much popularity in industry. This is due to their complexity, limited applicability to industrial cells, and additional inaccuracy compared to NLDM.

4.1.4 Waveform Independent Model (Current Source Model)

Current source models (CSMs) are fundamentally different from NLDM, ECSM, and CCS. They do not provide output waveforms or delay values as functions of parameters such as input slew or output load. Instead, they provide a cell's output current as a function of port voltages. Figure 4.4 depicts the general characterization setup in which the port currents are measured for a set of SPICE simulations.

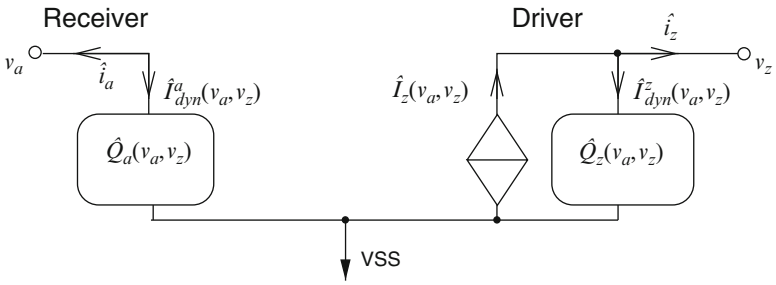


Fig. 4.5 Typical CSM with voltage-controlled current source and voltage-controlled charges

The cell delay is obtained during analysis through a SPICE-like simulation, in which the CSM provides the voltage-dependent port current that is flowing into the load model. Since this method does not impose restrictions on signal shapes or load models, arbitrary input waveforms and an arbitrary loads are supported [13]. However, this generality is usually traded for simulation performance. If the load is restricted or converted to single capacitors or CRC-Π-structures, closed-form expressions can be used to avoid iterative output waveform calculation [14–16]. Another technique for faster simulation of circuits with current source drivers with arbitrary resistive-capacitive loads is described in [17]. It applies matrix diagonalization to efficiently solve the equation systems.

Figure 4.5 depicts the typical structure of a CSM comprising a receiver and a driver model. In almost every CSM approach, the transient output current is modeled as the composite of a static current $I(\mathbf{v})$ and an additional dynamic contribution $I_{\mathrm{dyn}}(\mathbf{v}, \mathbf{v}')$. The voltage vector $\mathbf{v}$ usually contains the port voltages and $\mathbf{v}'$ their time derivatives. For complex cells, also important internal nodes might be included [18].

The voltage-controlled current source models the DC output current of a logic cell for every combination of port voltages. Its values are obtained either from lookup tables using interpolation or from approximation functions. To correctly predict cell delay and output waveforms, additional components are required. They account for cell internal dynamic effects such as the finite time for transistors to charge or discharge the channels or the reduced output current due to charging internal nodes. Different implementations use either voltage-controlled charges or (non)linear capacitors. In one of the first CSMs, only a linear output capacitor is used to reshape the output waveform [14]. An additional delay value is added afterward to match the total cell delay. In later approaches, low-pass filters were used to partially account for the cell delay [13, 19]. Voltage-controlled nonlinear capacitors or charges have been introduced to improve accuracy and to provide a receiver model [19, 20, 22]. Additional nonlinear capacitors are used to explicitly model capacitive coupling between driver and receiver (Miller Effect) [22, 23]. Some approaches follow a very general methodology and use the driver model consisting of static current source and dynamic element for every port of the cell [18, 24], although the static input currents are significantly smaller than the dynamic ones (see Sect. 4.1.5). Nonetheless, these approaches provide CSM for simultaneous switching of all inputs, whereas other CSMs are provided for every timing arc.

The model characterization usually is a two-step process. First, the function for the static port current is obtained. As shown in Fig. 4.4, DC voltage sources are attached to the ports. Their values are swept from v_{SS} to v_{DD} to cover the whole range of port voltages. The measured currents are then stored as lookup tables or are approximated by fitting functions. The latter requires less data to be stored and hence increases simulation performance.

Thereafter dynamic elements are characterized which requires more effort. These elements are either obtained by error minimization to a set of typical input waveforms [14, 24] or by special simulations with step or ramp functions [19, 20]. Here, the principle is to identify additional current contributions in case of voltage changes. The difference of observed transient current when applying step functions to the already known static current is related to charges or capacitors.

In the approaches of [18, 25, 26], the logic cell is treated as multi-port system. Its admittance matrix $\mathbf{Y} = \mathbf{G} + j\omega\mathbf{C}$ is linearized at all combinations of port voltages to account for nonlinearity. Based upon the susceptance matrix $\mathbf{C}$, nonlinear voltage-dependent capacitors are derived to connect all ports [18].

4.1.5 Physically Motivated CSM

Despite their obvious benefits, at the time of writing this book CSMs are more research topics than industrial reality. The higher accuracy provided by them comes with significantly lower simulation performance compared to STA using NLDM, ECSM, or CCS. Further, commercial EDA tools currently do not support CSMs and there exists no standardized format. Finally, library generation is much more complex compared to the current standards. As discussed in the previous subsection, most CSM approaches only try to match the observable port currents but treat the cell as a black box model. CSM generation therefore requires a large number of simulations. An alternative approach has been proposed in [13]. The CSM components are not artificial circuit elements for error minimization but are related to the original netlist elements of a logic cell. This approach for physically motivated CSMs is described in more detail since it visualizes the principles of CSM modeling.

The inverter subcircuit in Fig. 4.6 is used throughout this description. The aim of every CSM is to produce output voltage waveforms that are identical to those of the cell's transistor netlist description for any given input signal and any attached load. This will be realized when the CSM output current $\hat{i}(t)$ always equals the original current $i(t)$.

$$i(\mathbf{v}(t), \mathbf{v}'(t)) = \hat{i}(\hat{\mathbf{v}}(t), \hat{\mathbf{v}}'(t)). \tag{4.7}$$

While accuracy is one requirement, simulation performance is another. Evaluating the right-hand side of (4.7) must be significantly faster than computing the transistor currents in the left-hand side. Only if this is provided, CSMs are applicable to complex digital circuits. The first step of complexity reduction is in reducing the number of model parameters. The voltage vector $\hat{\mathbf{v}}(t)$ only contains port voltages

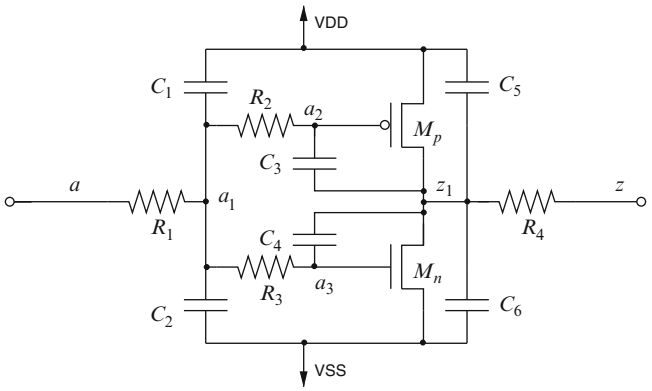


Fig. 4.6 CMOS inverter with layout parasitics

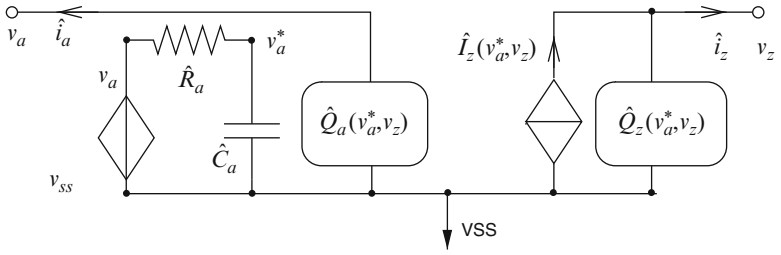


Fig. 4.7 Proposed CSM for logic cells with parasitic elements

but no cell internal nodes. The next step is to select a model topology to fulfill (4.7). Obviously, a static current source $\hat{I}$ is required to handle the special case

$$i(\mathbf{v},0) = \hat{i}(\hat{\mathbf{v}},0) = \hat{I}(\hat{\mathbf{v}}). \tag{4.8}$$

In cases of voltage changes of port nodes a and z, additional dynamic currents result from charging the parasitic capacitors and transistor capacitances. This dynamic current is modeled by an associated port charge $\hat{Q}$.

$$\hat{i}(\hat{\mathbf{v}}(t),\hat{\mathbf{v}}'(t)) = \hat{I}(\hat{\mathbf{v}}(t)) + \frac{\mathrm{d}}{\mathrm{d}t}\hat{Q}(\hat{\mathbf{v}}(t)). \tag{4.9}$$

Finally in very large cells the passive parasitic network causes a notable signal delay at the input. A low-pass filter is therefore inserted into the CSM, which produces a delayed input voltage in $\hat{\mathbf{v}}^* = [v_a^*, v_z]^T$. The final topology of the CSM is shown in Fig. 4.7.

$$i(\mathbf{v},\mathbf{v}'(t)) = \hat{I}(\hat{\mathbf{v}}^*(t)) + \frac{\mathrm{d}}{\mathrm{d}t}\hat{Q}(\hat{\mathbf{v}}^*(t)). \tag{4.10}$$

Once the CSM structure has been derived, the next step is to define the functions of $\hat{I}$ and $\hat{Q}$. This is done by a topology analysis of the given subcircuit in which the contributions of every netlist element to the port components are identified. Starting at the port node, all resistively connected nodes are visited. At each node, symbolic expressions for node charge and static current contributions are derived. The sum of these charges is denoted as the associated port charge. Similarly, the sum of all current contributions defines the total port current. For the example of Fig. 4.6, the resulting expressions are listed below.

$$\begin{aligned}\hat{Q}_a = {} & Q_{\mathrm{g}}^{M_p}(v_{z_1}, v_{a_2}, v_{\mathrm{DD}}) + Q_{\mathrm{g}}^{M_n}(v_{z_1}, v_{a_3}, v_{\mathrm{SS}}) + C_1 \cdot (v_{a_1} - v_{\mathrm{DD}}) \\ & + C_2 \cdot (v_{a_1} - v_{\mathrm{SS}}) + C_3 \cdot (v_{a_2} - v_{z_1}) + C_4 (v_{a_3} - v_{z_1}) \end{aligned} \tag{4.11}$$

$$\begin{aligned}\hat{Q}_z = {} & Q_{\mathrm{d}}^{M_p}(v_{z_1}, v_{a_2}, v_{\mathrm{DD}}) + Q_{\mathrm{d}}^{M_n}(v_{z_1}, v_{a_3}, v_{\mathrm{SS}}) + C_5 \cdot (v_{z_1} - v_{\mathrm{DD}}) \\ & + C_6 \cdot (v_{z_1} - v_{\mathrm{SS}}) + C_3 \cdot (v_{z_1} - v_{a_2}) + C_4 \cdot (v_{z_1} - v_{a_3}) \end{aligned} \tag{4.12}$$

$$\hat{I}_a = I_{\mathrm{g}}^{M_p}(v_{z_1}, v_{a_2}, v_{\mathrm{DD}}) + I_{\mathrm{g}}^{M_n}(v_{z_1}, v_{a_3}, v_{\mathrm{SS}}) \tag{4.13}$$

$$\hat{I}_z = I_{\mathrm{d}}^{M_p}(v_{z_1}, v_{a_2}, v_{\mathrm{DD}}) + I_{\mathrm{d}}^{M_n}(v_{z_1}, v_{a_3}, v_{\mathrm{SS}}). \tag{4.14}$$

Q_{g} denotes the charges located at the gate pin of a transistor. In (4.13), only the static gate currents contribute to the static port current. Since these are magnitudes smaller than dynamic or output currents, they can be neglected without loss of accuracy.

The symbolic expressions for the model components are expressed as SPICE measurement statements. Their numerical values are obtained by DC simulations. This is possible because all node potentials in (4.11)–(4.14) have fairly small time constants. Consequently, these node potentials are almost in algebraic relationship to the port voltages. Hence, the circuit elements are mostly controlled by the absolute values of port voltages and not by their time derivatives or internal states of node potentials. Hence, to characterize port charges and output current, DC voltages sources are attached to input and output port and swept from v_{SS} to v_{DD}. For each combination of port voltages, the measurements (4.11), 4.12), (4.14) are executed, and the values are stored in lookup tables.

The above-mentioned algebraic dependency does not hold for cells consisting of more than one stage such as buffers or ANDS. These cells are split into channel connected blocks which are then modeled individually. It is further problematic for large cells with a long parasitics network. Here, the low-pass filter is used to account for the additional delay. It is sized to match the average cutoff frequency of the transistor gate pins.

The functions for the port elements of the inverter cell are shown in Fig. 4.8. While the output current is strongly nonlinear, the charges are fairly linear and might be approximated by a first-order polynomial. However, for more complex cells also the charges show significant nonlinearities. Therefore, all elements might be implemented as voltage-controlled elements which perform bilinear interpolation on lookup tables. For efficiency, a separate CSM is provided for each timing arc, which limits the dimension of lookup tables to two.

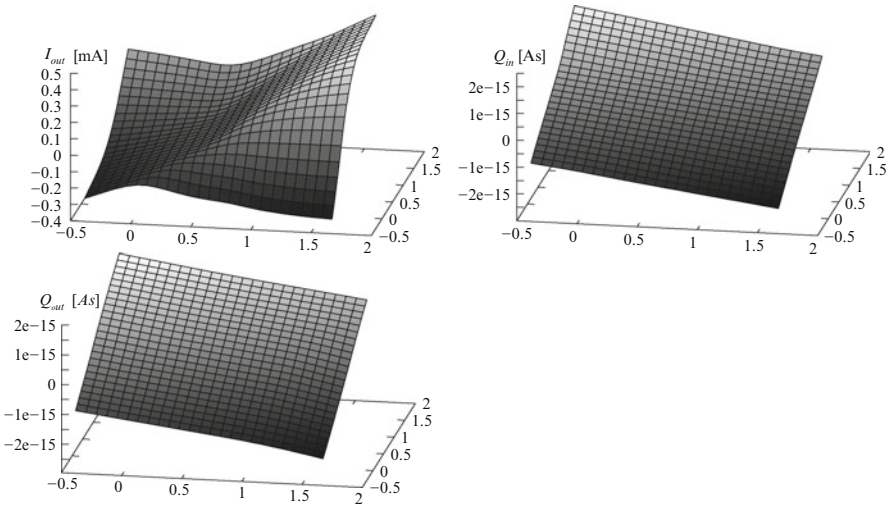


Fig. 4.8 Functions for static output current (*top left*), input charge (*top right*), and output charge (*bottom left*)

4.1.5.1 Parameter Variation Analysis with CSMs

Process variation affects not only cell delays but also the resulting output waveforms. These, in turn, affect the delay of the next logic stage. First academic approaches have been proposed to use CSMs in statistical static timing analysis considering waveform variations [22, 27, 28]. The timepoints when voltage thresholds are crossed are modeled as random variables instead of fixed values.

4.1.6 Memory Issues

The required accuracy improvements in timing analysis and timing modeling result in increasing library sizes. Using the bounded delay model, only two values must be stored per cell. A refinement of this model stores two delay values per pin or per timing arc.

Using a slew-load-model such as NLDM or SPDM, additionally input capacitances and output slews must be provided. Most data must be available for rising and for falling transitions. Typical NLDM lookup tables provide timing information for 7×7 combinations of input transition and output load. These are $2 \cdot 2 \cdot 49 = 196$ values for one timing arc of a cell at one PVT corner. CCS and ECSM additional require to store the current or voltage waveforms for each combination of transition time and output load. When modeling a signal at 10 timepoints, these are additional

$2 \cdot 20 \cdot 49 = 1{,}960$ entries per timing arc. Finally, these models provide the DC output current for at least 10×10 voltage combinations per cell. CCS libraries tend to be larger than ECSM libraries due to differences in the format.

The large differences of current source models complicate the size estimation. For most approaches, one CSM is required for one timing arc but captures rising and falling transitions. To provide some degree of accuracy at least 10×10 combinations of input and output voltage are required. Since there are three to four nonlinear elements in a CSM, these are 300 data points. However, since CSMs are mostly defined to channel connected blocks, this number must be multiplied by the number of stages in a cell.

4.2 Library Characterization

In statistical analysis, modeling circuit performance for nonlinear problems demands high computational effort. In semicustom design, statistical leakage library characterization is a highly complex yet fundamental task. The linear model provides an unacceptable accuracy in modeling a large number of standard cells. Instead of assuming one model for the entire library beforehand, we developed an approach generating models individually. In our approach, the statistical learning theory is utilized. The key contribution is the use of a cross term matrix and an active sampling scheme, which significantly reduces model size and model generation time. The effectiveness of our approach is clearly shown by experiments on industrial standard cell libraries. For quasilinear problems, a small amount of additional effort is required to verify the linear dependency. For strong nonlinear problems, our approach reaches high accuracy with affordable computational effort. As we regard the circuit block as a black box, our approach is suitable for modeling various circuit performances.

4.2.1 Motivation

In today's ICs, leakage current is an essential part of power consumption. At the 65 nm technology node, leakage current can exceed 50% of total power [29, 30]. Consequently, leakage variability arising from variations in Process-Voltage-Temperature (PVT) conditions has gained importance in the last decade. For instance, it has been reported in [79] that a 10% variation in the effective channel length results in a 3X difference in subthreshold leakage. A 10% oxide thickness variation even contributes to a 15X difference in gate leakage current. Our experiments on industrial standard cell libraries show that a 10% temperature variation leads to a 4X difference in cell leakage.

To address the leakage variation issue in semicustom design, building a statistical library at gate level is a highly complex yet fundamental task for statistical analysis.

A typical BSIM-model based industrial library has more than 500 standard cells. Every state of each logic cell must be separately characterized. In our experiments, 12 varying process parameters were considered in the analysis. Variation in supply voltage and temperature also has a significant impact.

Considering this complexity the traditional corner case analysis and Monte Carlo simulation are too expensive. Diverse analytical and numerical approaches have recently been proposed to model the cell leakage as a function of the underlying process parameters. In analytical approaches, empirical equations are applied to model the leakage components separately [32, 33, 79].

Numeric approaches regard the logic cell as a black box responding (in leakage value) to stimuli (i.e., the process parameter values). The approximation of the black box is often described as a *response surface model*. Most existing approaches assume that the leakage current has a log-normal distribution. Consequently, the logarithm of leakage, $\log(I)$, is expressed as a linear function of the process parameters. However, our experiments show that the linear model provides an unacceptable poor accuracy in modeling a large number of standard cells. For nonlinear problems, high-order models are needed to improve modeling accuracy. Certainly, this leads to a rapidly increasing computation effort for high-dimensional problems, as the number of coefficients to be determined rises significantly. To overcome this problem, reduced rank regression techniques have been employed on a quadratic model [34–37]. It is yet not clear how these methods can be efficiently extended to higher order models.

In terms of library characterization, a pre-assumed model is either inaccurate (e.g., the linear model) or too computing intensive (e.g., the high-order models). Our experiments on several standard cell libraries show that the $\log(I)$s of many cells can be accurately modeled by the linear model. At the same time, there are a large number of cells whose $\log(I)$s are strongly nonlinear, and even not compatible with the quadratic model.

Another important aspect of statistical modeling is the selection of samples. In most existing approaches, the simulation samples used for coefficient calculation are randomly generated. The samples' representativeness for the entire parameter space is rarely discussed.

In this section, we present an approach based on statistical learning theory. The algorithms developed generate the response surface model for each cell individually. Its main contributions are fourfold: (1) the order of each process parameter is estimated based on the real dependency of the leakage current on this parameter; (2) the structure of each model (i.e., which terms should be included in the model) is decided by a sensitivity analysis-based method; (3) to avoid reliance on random simulation samples, a sampling scheme is proposed, taking into account both the sample location and its impact on leakage current, and (4) the accuracy of models is validated by cross validation.

The remainder of this section is organized as follows: in Sect. 4.2.2 the basis of statistical learning theory is introduced. Section 4.2.3 describes our algorithms in detail. The results of the experiments on typical industrial library cells are presented and discussed in Sect. 4.2.4. Section 4.2.5 provides a short conclusion.

4.2.2 Background

4.2.2.1 Problem Formulation

In the context of statistical modeling, a logic cell is regarded as a black box. For each of its logic states, the observations are formulated as input–output pairs generated by an unknown deterministic mapping function:

$$F : \underline{X} \mapsto Y, \tag{4.15}$$

where $\underline{X} = [X_1, X_2, \cdots, X_n]^T$ represents a set of process parameter values and Y the leakage value.

We assume that a true underlying polynomial function $g(\underline{x})$ exists, so (4.15) can be formulated as:

$$F(\underline{x}) = g(\underline{x}) + \Phi, \tag{4.16}$$

where the noise Φ is negligible.

The target, then, is to construct an estimator of $g(\underline{x})$, based on the dataset D with a minimum error:

$$f_D(\underline{x}) = \sum_{p_1=0}^{P_1} \cdots \sum_{p_n=0}^{P_n} \beta_{p_1,\cdots,p_n} \prod_{i=1}^{n} x_i^{p_i}. \tag{4.17}$$

In (4.17), $\underline{x}$ represents the process parameters, P_i represents the highest order of the parameter x_i, $\prod_{i=1}^{n} x_i^{p_i}$ denotes one term of the polynomial, and $\beta_{p_1,\ldots,p_n}$ is the coefficient of this term. A term with at least two parameters is defined as a *cross term*. A term with one parameter is defined as a *single term*. In statistical learning, the error of $f_D(\underline{x})$ is defined by the loss function (see Sect. 4.2.2.2).

4.2.2.2 Statistical Learning

Statistical learning methods are broadly employed in diverse fields of science and industry. The objective of statistical learning is to explore the underlying knowledge from a given set of data (*learning from data*). The most important categories are regression, classification, and data clustering [38]. The statistical modeling of circuit performance is a typical regression task, which can be roughly divided into three phases: model selection, model fitting, and model validation.

The objective of model selection is to decide which terms (i.e., $\prod_{i=1}^{n} x_i^{p_i}$ in (4.17)) should be included in the model. In building high-order models for high-dimensional problems, the large amount of additional terms leads to a run time explosion. First, significantly more simulations are needed to determine the coefficients. Moreover, the run time of the model fitting process itself also increases

exponentially with the sample size as well as with the number of terms. At the same time, the sample density in high-dimensional problems is normally very low. This phenomenon is commonly referred to as the *curse of dimensionality* [39].

Modeling a logic cell with 12 varying process parameters, for example, the quadratic model requires 91 terms. Using the cubic model, i.e., $P_i = 3$ and $\sum p_i \leq 3$ for each term in (4.17), the number of terms increases to 455. Obviously, most of these terms (418 of 455) are cross terms having at least two parameters.

The key idea of our approach uses a sensitivity analysis-based technique to decide the selection of cross terms. This technique is described in detail in Sect. 4.2.3.

Model fitting is the procedure determining the coefficients of (4.17). This procedure is also denoted as the *training process*. As mentioned, the function characterizing the error of $f_D(\underline{x})$ is often denoted as the *loss function*. The most frequently used loss function is the squared error, measuring the square of the deviation between $f_D(\underline{X})$ and Y for each sample. Using the ordinary least square method (OLSM), the coefficients are determined ensuring minimization of the expectation value of the squared error (also *MSE*: Mean Squared Error):

$$\underline{\beta} = \arg\min_{\underline{\beta}_i}\{E[(Y - f_D(\underline{x}))^2]\}; \tag{4.18}$$

$$MSE = E[(Y - f_D(\underline{x}))^2] = \int_{\Omega} (Y - f_D(\underline{x}))^2 PDF(\underline{x}) \mathrm{d}\underline{x}. \tag{4.19}$$

$PDF(\underline{x})$ is the joint probability density function of $\underline{x}$ and Ω denotes the entire parameter space.

Substituting Y with (4.16), the *MSE* can be deconstructed into three parts [43]:

$$MSE = Bias^2 + Variance + Noise; \tag{4.20}$$

$$Bias = E[g(\underline{x})] - E_D[f_D(\underline{x})]; \tag{4.21}$$

$$Variance = E_D[(f_D(\underline{x}) - E_D[f_D(\underline{x})])^2]; \tag{4.22}$$

$$Noise = E[(g(\underline{x}) - F(\underline{x}))^2]. \tag{4.23}$$

The notation $E_D[.]$ represents the expectation value with respect to dataset D. (4.23) indicates that *Noise* depends neither on the dataset D nor on the estimator $f_D(\underline{x})$. This is thus an uncontrollable factor in improving the effectiveness of estimator $f_D(\underline{x})$.

Bias measures the average accuracy of the estimate and indicates the discrepancy between the true polynomial $g(\underline{x})$ and $f_D(\underline{x})$. If the model is properly chosen, *Bias* will be minimized by OLSM. Where the model is misspecified, OLSM does not minimize *Bias* even with an infinite number of samples [41]. *Variance* shows the degree of variability in $f_D(\underline{x})$ between datasets [42]. Due to the *Bias–Variance Dilemma* [38, 41–44], a compromise between *Bias* and *Variance* must be reached during model selection.

In OLSM, the simulation samples used to determine the coefficients (the *training samples*) are normally randomly chosen. As a rule of thumb, the number of samples needed is twice the number of the model terms. The quality of the parameter space representation has been rarely discussed previously. We have developed a scheme for more active selection of training samples.

Model validation is utilized to verify the accuracy of the model with independent samples not used in the training process. The average error of these samples is defined as the prediction error:

$$Err_{\text{pred}} = \frac{1}{N}\sum_{j=1}^{N}\left(Y_j - f\left(\underline{X}_j\right)\right)^2, \tag{4.24}$$

where $(\underline{X}_j, Y_j)$ are the prediction samples.

In a data-rich situation, the training sample set and the prediction sample set should be completely disjoint. As the simulation cost for a large number of samples is prohibitively high in process variation analysis, cross validation is applied [38].

4.2.3 Dynamic Model Generation

4.2.3.1 Model Selection Using Cross Term Matrix

In our approach, the existence of terms, $\prod_{i=1}^{n} x_i^{p_i}$ in (4.17), is represented in the *cross term matrix* M_{CT}. The cross term matrix is a square matrix of order n, where n is the number of process parameters. The key concept of generating M_{CT} is described in Algorithm 1. Given $PDF(\underline{x})$ and a circuit block (e.g., one logic cell), we first execute a sensitivity analysis at the nominal point P_0, where the value of each parameter equals the mean value of its distribution. The *most significant parameters* (MS parameters) in the center area of the parameter space are identified from the first order sensitivity vector $\underline{S}$. They are then stored in the list L^0_{MS}:

$$L^0_{\text{MS}} = \left\{ x_i \middle| ABS\left(\frac{\partial F}{\partial x_i}\Big|_{P_0}\right) \geq \lambda_1 \varepsilon \right\}, \tag{4.25}$$

where $ABS(.)$ denotes the absolute value, $\lambda_1 \in (0,1)$ is a tuning parameter, and ε can be either a pre-defined value or the greatest value in $\underline{S}$.

Following the identification of the MS parameters in the center area of the parameter space, the black box function can be formulated as:

$$F(\underline{x}) = h(\underline{x}_I) + \underline{S}_{II}\underline{x}_{II}^T + \boldsymbol{\Phi}. \tag{4.26}$$

Here, $\underline{x}_I$ denotes the MS parameters in the center area. The remaining parameters are included in $\underline{x}_{II}$, and their first-order sensitivities are represented in $\underline{S}_{II}$. As in (4.16), $\boldsymbol{\Phi}$ is the noise that can be ignored.

Input : **PDF**($\underline{\mathbf{x}}$): Joint Probability Density Function of Process Parameters;
CB: Circuit Block
Output: $\mathbf{M_{CT}}$: Cross Term Matrix

`GenerateCrossTermMatrix` (**PDF**($\underline{\mathbf{x}}$), **CB**) **begin**
  $\underline{\mathbf{S}} \leftarrow$ `SensitivityAnalysis` ($[\mu_1, \cdots, \mu_n]^T$, **CB**);
  $\mathbf{L_{MS}^0} \leftarrow$ `InitialiseMostSignificantParameter` ($\underline{\mathbf{S}}$);
  $\mathbf{L_{MS}} = \mathbf{L_{MS}^0}$;
  foreach ($\mathbf{x_i} \in \mathbf{L_{MS}}$) **do**
    $\underline{\mathbf{S_i}} \leftarrow$ `SensitivityAnalysis` ($[\mu_1, \cdots, \mu_i \pm \mathbf{3}\sigma_i, \cdots, \mu_n]^T$, **CB**);
    $\mathbf{L_{CT}} \leftarrow$ `IdentifyCrossTermParameter` ($\mathbf{x_i}, \underline{\mathbf{S}}, \underline{\mathbf{S_i}}$);
    $\mathbf{M_{CT}} \leftarrow$ `FillMatrix` ($\mathbf{x_i}, \mathbf{L_{CT}}$);
    $\mathbf{L_{MS}} \leftarrow$ `UpdateMostSignificantParameter` ($\mathbf{L_{MS}}, \mathbf{L_{CT}}$);
  end
  return $\mathbf{M_{CT}}$;
end

Algorithm 1: Generation of cross term matrix

For each parameter x_i in $\underline{x}_I$ (i.e., each parameter stored in L_{MS}), a sensitivity analysis is executed at its $\mu_i \pm 3\sigma_i$ points (P_i^+ and P_i^-), while the values of the other parameters are kept on nominal values. The changes in first-order sensitivities of all parameters are explored to select the cross term parameter for x_i. The parameter x_j is defined as a *cross term parameter* of x_i, if the change in its first order sensitivity on the point P_i^+ or P_i^- exceeds the threshold $\lambda_2 \varepsilon$, where $\lambda_2 \in (0,1)$ is another tuning parameter.

$$L_{\mathrm{CT}}^i = \left\{ x_{j,j\neq i} \middle| \; ABS\left(\frac{\partial F}{\partial x_j}\Big|_{P_0} - \frac{\partial F}{\partial x_j}\Big|_{P_i^+} \right) \geq \lambda_2 \varepsilon \quad \text{or} \right.$$
$$\left. ABS\left(\frac{\partial F}{\partial x_j}\Big|_{P_0} - \frac{\partial F}{\partial x_j}\Big|_{P_i^-} \right) \geq \lambda_2 \varepsilon \right\}. \tag{4.27}$$

A numerical example shall demonstrate the algorithm. In the following, we suppose that the unknown $h(\underline{x}_I)$ to be approximated in (4.26) is

$$h(\underline{x}_I) = x_1 + x_3 + x_5 + x_2^2 + x_4^3 + x_1^2 x_2 + e^{x_3 + x_4} + \sin(x_4 x_6) + x_2 x_5^4 x_6^2. \tag{4.28}$$

As $\underline{x}_I$ and $\underline{x}_{II}$ are disjoint, for each MS parameter (i.e., x_i in $\underline{x}_I$) it is true that $\frac{\partial F(\underline{x})}{\partial x_i} = \frac{\partial h(\underline{x}_I)}{\partial x_i}$. Applying this to (4.26) and (4.28), the first-order sensitivity of x_4 obviously varies with the value of x_3:

$$\frac{\partial F(\underline{x})}{\partial x_4} = \frac{\partial h(\underline{x}_I)}{\partial x_4} = 3x_4^2 + \mathrm{e}^{x_3 + x_4} + x_6 \cos(x_4 x_6). \tag{4.29}$$

Fig. 4.9 A cross term matrix

$$A = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{array} \begin{array}{c} \begin{array}{cccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Fig. 4.10 Illustration of the cross term matrix: the cross terms are suggested by the maximum possible submatrix populated exclusively by 1 s

a

$$\begin{array}{c} \\ x_1 \\ x_4 \\ x_3 \\ x_2 \\ x_5 \\ x_6 \end{array} \begin{array}{c} \begin{array}{cccccc} x_1 & x_4 & x_3 & x_2 & x_5 & x_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

b

$$\begin{array}{c} \\ x_1 \\ x_4 \\ x_6 \\ x_2 \\ x_5 \\ x_3 \end{array} \begin{array}{c} \begin{array}{cccccc} x_1 & x_4 & x_6 & x_2 & x_5 & x_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Setting λ_2 to zero, x_4 is identified as a cross term parameter of x_3. The polynomial model (4.17) must, therefore, include terms containing x_3 and x_4 to model the dependency of $\frac{\partial F(\underline{x})}{\partial x_4}$ on x_3.

By contrast, x_j $(j = 1,2,5,6)$ is not a cross term parameter of x_3 as its first-order sensitivity is not dependent on x_3. Terms containing x_j and x_3 should, therefore, not be included. A considerable number of unnecessary cross terms are thereby excluded from the polynomial model.

Figure 4.9 illustrates the cross term matrix A for (4.28). Both the rows and the columns represent the process parameters $x_1 - x_6$. The entry a_{kl} is set to 1, when x_l is a cross term parameter of x_k, otherwise it is set to 0. In most applications, the cross term matrix is sparsely occupied. The cross terms are identified based on the maximum possible square submatrix populated exclusively by 1 s. In Fig. 4.9, two 2×2 submatrices are highlighted, which indicate the existence of cross terms $\{x_1^{p_1} x_2^{p_2} | 0 < p_{1,2} \leq P_{1,2}\}$ and $\{x_3^{p_3} x_4^{p_4} | 0 < p_{3,4} \leq P_{3,4}\}$. The highest power of each parameter, P_i, remains to be estimated. In Fig. 4.9, the 2×2 submatrix $A[5,6;5,6] = \begin{pmatrix} a_{55} & a_{56} \\ a_{65} & a_{66} \end{pmatrix}$. cannot suggest cross terms as it is part of the 3×3 matrix $A[2,5,6;2,5,6]$. In Fig. 4.10a, this is illustrated by exchanging the rows and columns of x_2 and x_4. Similarly, another submatrix $A[4,6;4,6]$ has been highlighted in Fig. 4.10b.

In summary, the cross term matrix (Fig. 4.9) suggests the following cross terms:

$$\left\{ x_1^{p_1} x_2^{p_2}, x_3^{p_3} x_4^{p_4}, x_2^{p_2} x_5^{p_5} x_6^{p_6}, x_4^{p_4} x_6^{p_6} \middle| 0 < p_i \leq P_i \right\}, \tag{4.30}$$

where i indicates the process parameter index and P_i is the highest power of the parameter. The number of terms is enormously reduced in comparison with comprehensive high order polynomial models.

P_i is determined by sweeping the ith parameter along one line in the multidimensional parameter space, where x_i varies and the other parameters are constant. On this sweep line, (4.26) becomes:

Input : **PDF**($\underline{\mathbf{x}}$): Joint Probability Density Function of Process Parameters;
$\mathbf{N_{tr}}$: Training Sample Size
Output: $\mathbf{L_{tr}}$: Training Sample List

```
GenerateTrainingSamples(PDF(x), N_tr) begin
    L_tr ← ImportSamplesFromModelSelection();
    while (Size(L_tr) ≤ N_tr) do
        L_m ← GenerateRandomSamples(PDF(x), m);
        X ← SelectTrainingSample(L_m);
        L_tr ← AddNewTrainingSample(L_tr, X);
    end
    return L_tr;
end
```

Algorithm 2: Generation of training samples

$$\tilde{F}_{x_i}(\underline{x}) = \sum_{p_i=1}^{N_i} C_{p_i} x_i^{p_i} + \Phi. \tag{4.31}$$

The sweep points selected are equidistant and the OLSM is used to determine the coefficients C_{p_i}. The highest power P_i is defined as:

$$P_i = \max\{p_i | C_{p_i} > \Theta\}, \tag{4.32}$$

where Θ is a threshold. As, on the sweep line, the single terms of x_i in (4.26) are mapped into (4.31), P_i also indicates the highest power of the single terms. Combining (4.32) with (4.30), the terms to be included in the model are determined.

4.2.3.2 Model Fitting Using Active Sampling

Conventionally, the training samples used to determine the term coefficients are randomly generated. In contrast, our approach chooses the training samples in a more active manner. Algorithm 2 provides an overview of the sampling scheme.

The first training sample set is acquired from the model selection. Reusing these samples ensures that the edge area of the parameter space, which is critical in nonlinear problems, is always accounted for.

The second training sample set is pseudorandomly selected. To choose one new training sample, m samples ($m > 1$) are randomly generated. Among these, m samples the one with the greatest distance from the set of existing samples is selected:

$$\underline{X} = \arg\max_{\underline{X}_j} [D(\underline{X}_j, L_{tr})], \qquad \underline{X}_j \in L_m. \tag{4.33}$$

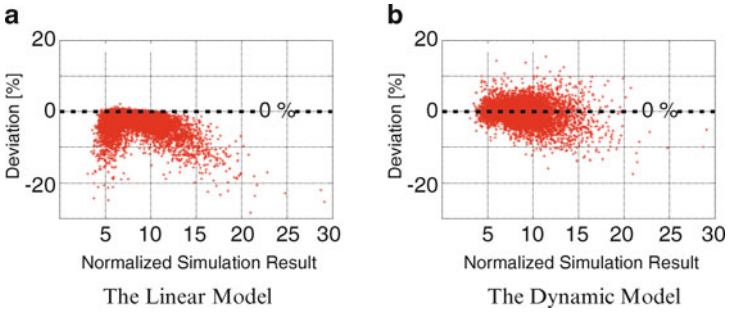


Fig. 4.11 3-Input NAND gate at 135°C

The distance is defined as:

$$D(\underline{X}_j, L_{tr}) = \sum_{P_{tr,n} \in L_{tr}} ||\underline{X}_j, P_{tr,n}||^2, \tag{4.34}$$

where $||.||$ denotes the euclidean distance between two points.

4.2.4 Experimental Results

Our approach was applied to typical CMOS standard cells from three industrial 90 nm leakage libraries. The results of several typical cases are presented here. In the experiments, the logarithm of leakage was modeled. The accuracy of various models and the computation time required to generate such models are compared. To calculate the leakage, an industrial in-house analog simulator was used, in which the adjoint network method is employed to determine first-order sensitivities.

For each test case, 10,000 randomly generated test samples were used to verify the accuracy of various models. The deviation of $Model_i$ at sample point $\underline{X}$ is

$$Deviation_i(\underline{X}) = \left| \frac{Model_i(\underline{X}) - Simulation(\underline{X})}{Simulation(\underline{X})} \right|. \tag{4.35}$$

As tail region of leakage distribution is particularly important in statistical leakage library characterization, each model's error in 99%-quantile has been compared.

The first example is a 3-Input NAND gate with 12 varying process parameters at 135°C. Figure 4.11 illustrates the model accuracy comparison between the linear model and the model generated by our approach. Applying the linear model, the majority of the testing samples have negative deviations (Fig. 4.11a). This indicates a negative bias in the model, which means that systematic errors have occurred in the modeling process. Our model, however, shows relatively balanced sample deviations along the 0% line, as shown in Fig. 4.11. Table 4.1 describes

Table 4.1 Results comparison for a 3-input NAND at 135 and 60°C

Temperature	135°C		60°C	
Fit model	LM	DGM	LM	DGM
Number of terms	13	26	13	36
Run time	∼ 10 s	∼40 s	∼10 s	∼90 s
Deviation				
Group A: 0–4%	78.0%	90.8%	10.4%	90.6%
Group B: 4–8%	12.2%	8.1%	27.0%	7.9%
Group C: Above 8%	9.8%	1.1%	62.6%	1.5%
Error in quantile				
99%-Quantile	28.0%	5.8%	12.2%	2.5%

LM linear model, *DGM* dynamically generated model

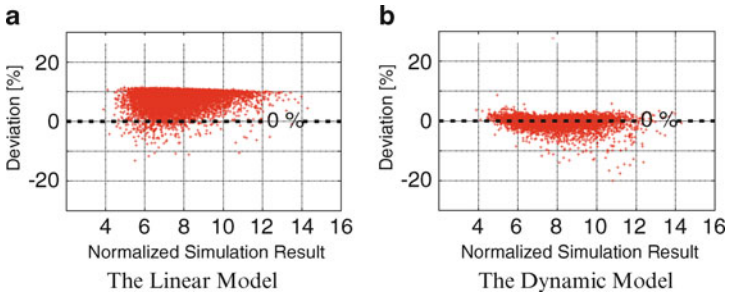


Fig. 4.12 3-Input NAND gate at 60∘C

the comparison more precisely. There are 13 coefficients to be determined in the linear model. The run time is 10 s. The testing samples are divided into three groups, according to the definition in (4.35). Group A contains the samples with a deviation rate of less than 4%; group B, those with 4–8% and group C, those above 8%. For the linear model, for example, 78.0% of all 10,000 samples fall into group A.

Our approach includes 26 terms in the model. Forty seconds were spent generating this model. The computational effort primarily consists of the *simulation time* spent computing leakage values and the *training time* necessary to determine the coefficients by OLSM. The results show visible improvements in accuracy: 90.8% of the testing samples now fall into group A. The error in 99%-quantile is reduced from 28% to 5.8%.

In the first example, the linear model shows an acceptable deviation level in center region of leakage distribution. Our experiments on the library characterization, however, show the linear model to be entirely unsuitable in a large number of cases. In the next example, the same experiment is applied to the same NAND gate, with the temperature adjusted to 60 °**C**. Figure 4.12a clearly indicates that the *Bias* of the linear model is substantially more than 0%.

Table 4.1 shows the results at 60°C. For the linear model, 62.6% of the testing samples have a deviation greater than 8%. Our approach includes 36 terms in the

Table 4.2 Results comparison for a full adder

Fit model	LM	QM2	CM	QM4	DGM
Number of terms	13	91	455	1,820	42
Number of simulations	1 sens.	182	910	3,640	8 sens. +76
Run time	∼6 s	∼11 min	∼15 h	–	∼2 min
Deviation					
Group A: 0–4%	0.6%	51.4%	83.2%	Aborted	74.6%
Group B: 4–8%	0.8%	35.7%	12.7%	Aborted	19.2%
Group C: Above 8%	98.6%	12.9%	4.1%	Aborted	6.2%
Error in quantile					
99%-Quantile	32.5%	8.2%	3.1%	Aborted	4.5%

LM linear model, *QM2* quadratic model, *CM* cubic model, *QM4* quartic model, *DGM* dynamically generated model

model, which is more complex than that in Example 1. The run time now is 90 s. The accuracy is, however, enormously improved: 90.6% of testing samples have a deviation smaller than 4% and the 99%-quantile error is 2.5%.

A more detailed comparison can be seen in the third example. The experiment is applied to a full adder with 12 varying process parameters. Again, 10,000 testing samples were divided into three groups for each model. Table 4.2 shows that the number of terms rapidly increases with the model order. The linear model has only 13 terms. This rises to 91 for the quadratic model, 455 for the cubic model and 1,820 for the quartic model (i.e., with a model order of 4). Linear modeling uses the first order sensitivities of each parameter at the nominal point, calculated directly by the analog simulator. For higher order models, the OLSM is employed to determine the coefficients. The number of training samples needed is twice as much as the number of terms included in the model. As mentioned, the two major run time contributions are simulation time and training time. Our experiments show that the training time of OLSM grows exponentially with the number of terms, and with the number of training samples. For this example, the quadratic model needs 11 min and the cubic model 15 h. Modeling with the quartic model was aborted due to the prohibitive computation effort. Creating and analyzing the cross term matrix resulted in the inclusion of only 42 terms in our model. The run time is 2 min, which is considerably less than that required for the quadratic model.

The benefit of our approach, regarding accuracy, is obvious. Using the linear model, only 0.6% of the testing samples fall into group A. Accompanied by highly increased computation cost, the quadratic model has 51.4% group A testing samples and the cubic model 83.2%. Modeling with the quartic model was aborted. Applying our approach results in 74.6% group A testing samples. This represents an over 20% improvement compared to the quadratic model. It is also worth noting that for the high deviation group (Group C) the dynamic model shows similar results as the cubic model (6.2 and 4.1%), despite a radically reduced run time. The improvement in 99%-quantile error has been shown clearly: 32.5% for the linear model, 8.2% for the quadratic model, 3.1% for the cubic model, and 4.5% for the dynamic model.

4.2.5 Conclusion

In this section, we have presented an approach for rapid high-order polynomial-based statistical modeling. Instead of assuming the model type beforehand, our approach dynamically generates a model for each individual case. The complexity of the model is decided by the dependency of the circuit performance on the process parameters. We developed a sensitivity-guided method to generate a cross term matrix. Exploring the cross term matrix allows unnecessary terms to be excluded from the model for nonlinear problems. For linear problems, the cross term matrix becomes almost a zero matrix.

To determine the coefficients of the model, our approach selects training samples in a more active way. First, the samples in the model selection phase are reused, so that the edge area of the sampling space can always be accounted for. Second, we integrate the sample distance into the sampling scheme, resulting in relatively broadly populated training samples.

The benefits of the approach were clearly shown by experiments in gate level leakage library characterization, in which both quasilinear cases and strongly nonlinear cases exist. For quasilinear cases, a small amount of extracomputational effort was required to verify the linear dependency. For strong nonlinear cases, our approach addresses the modeling challenge with a high degree of accuracy and with affordable computational effort. Finally, it should be mentioned that, because our approach regards the circuit block as a black box, it is suitable for modeling various circuit performances.

4.3 Statistical Static Timing Analysis

In recent years, academic and industrial research has produced a multitude of approaches to address SSTA (statistical static timing analysis). They differ primarily in whether they take a block-based or path-based approach, whether they use a linear or higher order dependency of gate delay on process variations, whether they assume Gaussian variations or allow arbitrary variations, and to which degree they consider spatial correlations. An overview of these approaches is presented in the following.

4.3.1 Background of Timing Analysis

Most of the circuits used in industry are sequential circuits. The typical structure of a digital circuit is shown in Fig. 4.13. The input combinational logic generates the data for the registers and the output logic for the primary outputs. The outputs of registers are connected back to the input logic, forming combinational paths between registers. The registers store the data at their inputs when the triggering signal, called clock, is valid.

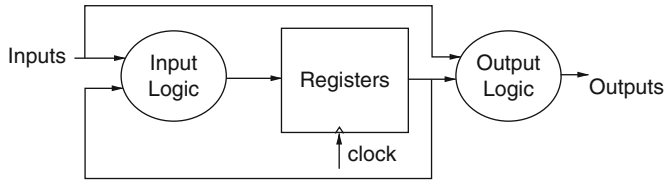
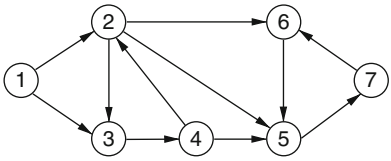


Fig. 4.13 Sequential circuit structure

Fig. 4.14 Example of reduced timing graph



Because of the simplicity of design and verification, flip-flop-based circuits are the most popular circuit type, where all registers are implemented by flip-flops. A flip-flop transfers the data at its input to its output only when the predefined clock edge appears. Without losing generality, all flip-flops are assumed to be triggered at the rising clock edge in the following.

In order to work correctly, a flip-flop has special requirements to the data at its input. Assuming the valid clock edge is at time t_c, the data at the input of the flip-flop should be stable between $(t_c - s_i)$ and $(t_c + h_i)$, where s_i is called setup time and h_i hold time. During this time period, any data change at the input of the flip-flop may cause it to enter metastability state with a certain probability [45]. In this state, the output of the register stays between 0 and 1, and is considered as a circuit failure. A hold time constraint violation happens when the signal from a register propagates to the next stage too fast. It can be corrected easily, e.g., by delay insertion and padding [46]. Setup time constraints determine the maximum clock frequency and should be checked when verifying a circuit against different clock frequencies. To correct violations of setup time constraints further circuit optimization is required. This optimization usually enlarges the die size and increases design time. In the following, only setup time constraints will be discussed.

For convenience to explain timing specifications of sequential circuits, *reduced timing graphs* [47] are used to represent the structural connections between flip-flops. An example of the reduced timing graph is illustrated in Fig. 4.14. In a reduced timing graph, a node represents a register, a primary input of the circuit, or a primary output. An edge represents the maximum delay between a pair of nodes, denoted Δ_{ij}.

To compute these delays, combinational circuits between flip-flops are traversed. For this type of circuits, the *timing graph* is used to represent its structural timing properties. Figure 4.15 shows an example of the timing graph of the circuit c17 from ISCAS85 benchmarks. A node in a timing graph corresponds to a pin of a gate if interconnects are considered. Otherwise, a node corresponds to a net in the circuit,

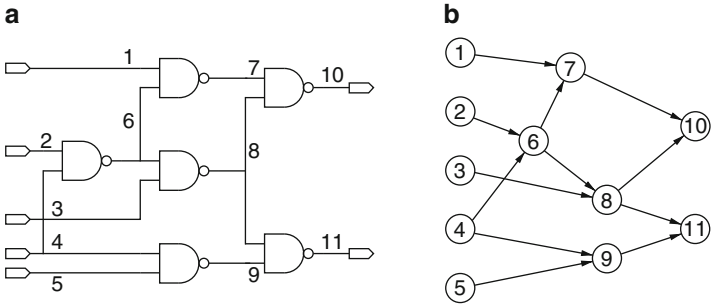


Fig. 4.15 c17 Benchmark circuit and timing graph

e.g., in Fig. 4.15. Additionally, primary inputs and outputs are also represented by nodes. An edge represents the delay W_{ij} between two nodes in the timing graph.

At the time of the nth active clock edge $t_{c,n}$, a signal starts to propagate to the output of a flip-flop i and further to the input of flip-flop j at the next stage. The latest time that this signal reaches j is $t_{c,n} + q_i + \Delta_{ij}$, where q_i denotes the propagation delay of the flip-flop. This time is the arrival time of the signal at the input of j through i, denoted as A_{ij}. The data change at the input of j must meet its setup time constraint, so that

$$A_{ij} = t_{c,n} + q_i + \Delta_{ij} \leq t_{c,n+1} - s_j. \tag{4.36}$$

Normally, flip-flop j has more than one fanin node in the reduced timing graph. After a valid clock edge, data signals propagate from all these fanin nodes to j. Each arrival time must meet the setup constraint described in (4.36). Consequently, the maximum of these arrival times should meet the setup time constraint, i.e.,

$$\max_{i \in \psi_j} \{A_{ij}\} = \max_{i \in \psi_j} \{t_{c,n} + q_i + \Delta_{ij}\} \leq t_{c,n+1} - s_j \Longleftrightarrow \tag{4.37}$$

$$t_{c,n} + \max_{i \in \psi_j} \{q_i + \Delta_{ij}\} + s_j \leq t_{c,n+1} \Longleftrightarrow \tag{4.38}$$

$$\max_{i \in \psi_j} \{q_i + \Delta_{ij}\} + s_j \leq t_{c,n+1} - t_{c,n} = T, \tag{4.39}$$

where ψ_j is the set of all fanin nodes of j in the reduced timing graph. Clock skew is not considered in (4.39) for simplicity. The constraint (4.39) should be met at all flip-flops in the circuit. With φ defined as the set of all flip-flops, the setup time constraint for the circuit is

$$\max_{j \in \varphi} \{\max_{i \in \psi_j} \{q_i + \Delta_{ij}\} + s_j\} \leq T. \tag{4.40}$$

The constraint (4.40) defines that the arrival time from any flip-flop node in the reduced timing graph to each of its sink nodes should meet the setup time constraint. Therefore, the constraint (4.40) can be written as

$$\max_{(i,j)\in\phi} \{q_i + \Delta_{ij} + s_j\} \leq T, \tag{4.41}$$

where ϕ is defined as the set of flip-flop pairs between each of which there is at least one combinational path in the original circuit.

The timing performance of a sequential circuit is represented by the maximum clock frequency. This clock frequency is determined by the minimum clock period, which can meet the timing constraint described in (4.41). To verify these constraints, the maximum delays Δ_{ij} used in (4.41) should be computed first from the combinational circuit between flip-flops.

In contrast to sequential circuits, a combinational circuit consists of no storage components but only combinational gates. If a signal reaches an input of such a gate, it continues to propagate instantly and reaches the output of this gate after the time equal to the delay of the gate. To compute Δ_{ij} for i and j in the reduced timing graph, the timing graph of the combinational circuit between them should be traversed. Two types of traversal methods exist to compute the maximum delay of a combinational circuit: path-based and block-based. In a path-based method, the paths from inputs to outputs of the timing graph are enumerated [48–50]. The delay of a path is computed by summing up the edge delays on the path. Although this path enumeration method is feasible to evaluate small designs, it cannot handle all paths in large ones, since the number of paths increases exponentially with circuit size.

The second method is the block-based method, or block-oriented method [51–53]. This method visits each node in the timing graph no more than once to compute the maximum delay from inputs to outputs. For each node, the arrival time represents the maximum delay from all the inputs to it. The arrival time of a node is updated just after all its fanin nodes are updated. As the first step of this computation, the arrival times from fanin nodes and the edge delays are added. Thereafter, the arrival time of the current node is computed by the maximum of the results from the previous step. This iterative computation stops after all outputs are visited.

Instead of computing the maximum delay Δ_{ij} between flip-flop i and j individually for all pairs of flip-flops, the inner maximum in (4.40) is computed by one arrival time traversal, resulting in the desired maximum circuit delay. This approach significantly reduces the computational effort. For this purpose, a virtual combinational circuit is formed. All outputs of flip-flops are considered as primary inputs of the virtual circuit, and all inputs of flip-flops as primary outputs. All the combinational components between flip-flops together form the combinational logic in between. The arrival times at primary inputs of the virtual circuit are set to the propagation delays of the corresponding registers. The resulting arrival times at the primary outputs of the virtual circuit are maximum arrival times from all primary inputs. In other words, it is the maximum delay from all fanin flip-flops to the input of a flip-flop, equal to the result of the inner maximum in (4.40). Thereafter, the left

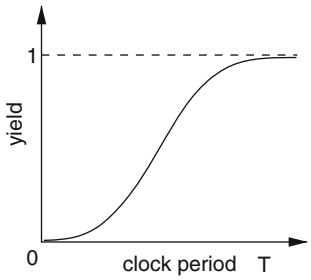


Fig. 4.16 Graphic representation of yield computation

side of (4.40) is computed by the maximum of the sums of the arrival time and the setup time at all flip-flops. This maximum specifies the minimum clock period for the flip-flop-based circuit without timing violation.

4.3.2 Statistical Timing Analysis

With process random variations modeled as variables, all gate delays become random variables. The static timing analysis algorithms described in the previous section can be adapted to compute the minimum clock period of a circuit similarly. The resulting clock period, however, is a random variable, denoted as $T_{\min}$. For a given clock period T, the timing yield of a circuit is evaluated by computing the probability that $T_{\min}$ is smaller than T, i.e.,

$$yield = Prob\{T_{\min} \leq T\}, \quad 0 < T < \infty, \tag{4.42}$$

where Prob$\{\cdot\}$ denotes the probability.

Because all gate delays are positive, the computed minimum clock period $T_{\min}$ is also positive. According to probability theory [56], yield computation in (4.42) is equivalent to the definition of *cumulative distribution function* (CDF) of the random variable $T_{\min}$. The graphic representation of (4.42) is illustrated in Fig. 4.16, where circuit yield approximates 0 when T approximates 0, and 1 when T is large enough. The latter case indicates that a sequential circuit can work properly at a reasonably low clock frequency, if no hold time constraint is violated.

In statistical timing analysis, the timing graph traversal is completely the same as in static timing analysis. The two computations, maximum and sum, however, must be adapted to handle random gate delays. In order to use the same sum and maximum computations at all nodes, arrival times in statistical timing analysis are usually represented in the same form as gate delays. When an arrival time and a gate delay are added, corresponding coefficients of different variables are summed directly, whether linear or quadratic gate delays are used. Because of the complexity in computing the maximum of two random variables and the requirement that the result of the maximum should have the same form as a gate delay, such computation

is always approximated in statistical timing analysis. In the following, only the sum and maximum computations of two random variables are discussed because other cases involving more than two random variables can be processed by applying these two computations iteratively.

Using the canonical delay model (3.15) described in Sect. 3.3.4, [55] introduces an arrival time propagation method, which can process the maximum computation efficiently, meanwhile keeping the correlations between arrival times accurately. Consider two random variables A and B

$$A = a_0 + \sum_{i=1}^{n} a_i v_i + a_r v_{r_a} \tag{4.43}$$

$$B = b_0 + \sum_{i=1}^{n} b_i v_i + b_r v_{r_b}. \tag{4.44}$$

The sum of A and B is computed as

$$A + B = (a_0 + b_0) + \sum_{i=1}^{n} (a_i + b_i) v_i + (a_r v_{r_a} + b_r v_{r_b}) \tag{4.45}$$

$$= s_0 + \sum_{i=1}^{n} s_i v_i + s_r v_{r_s}, \tag{4.46}$$

where s_r is identified by matching the variances of $s_r v_{r_s}$ and $a_r v_{r_a} + b_r v_{r_b}$.

To compute the maximum of A and B, denoted as $\max\{A,B\}$, the tightness probability (T_P) [55] is first computed. In [55], T_P is defined as the probability that A is larger than B. If A and B are both Gaussian, T_P is computed by

$$T_\mathrm{P} = Prob\{A \geq B\} = \Phi\left(\frac{a_0 - b_0}{\theta}\right), \tag{4.47}$$

where Φ is the cumulative distribution function of the standard Gaussian distribution. $\theta = \sqrt{\sigma_A^2 + \sigma_B^2 - 2\mathrm{Cov}(A,B)}$, where σ_A^2 and σ_B^2 are the variances of A and B, respectively. $\mathrm{Cov}(A,B)$ is the covariance between A and B, and is computed according to [56] as

$$\begin{aligned}\mathrm{Cov}(A,B) = &\sum_{i=1}^{n}\sum_{j=1}^{n} a_i b_j \mathrm{Cov}(v_i, v_j) + \sum_{i=1}^{n} a_i b_r \mathrm{Cov}(v_i, v_{r_b}) \\ &+ \sum_{i=1}^{n} b_i a_r \mathrm{Cov}(v_i, v_{r_a}) + a_r b_r \mathrm{Cov}(v_{r_a}, v_{r_b}).\end{aligned} \tag{4.48}$$

Because the random variables v_{r_a}, v_{r_b} and v_i in (4.43) and (4.44) are independent of each other, (4.48) is simplified to

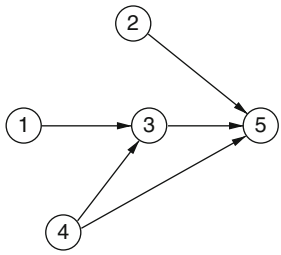


Fig. 4.17 Correlation example in statistical arrival time propagation

$$\mathrm{Cov}(A,B) = \sum_{i=1}^{n} a_i b_i \mathrm{Cov}(v_i, v_i) = \sum_{i=1}^{n} a_i b_i \sigma_{v_i}^2. \tag{4.49}$$

Comparing (4.48) and (4.49), the computation is drastically simplified because the random variables are uncorrelated. This is the motivation that the correlated random variables in Sect. 3.3.4 are decomposed.

According to [65], the mean (μ) and variance (σ^2) of $\max\{A,B\}$ are computed by

$$\mu = T_\mathrm{P} a_0 + (1 - T_\mathrm{P}) b_0 + \theta \phi \left(\frac{a_0 - b_0}{\theta} \right) \tag{4.50}$$

$$\begin{aligned} \sigma^2 =& T_\mathrm{P}(\sigma_A^2 + a_0^2) + (1 - T_\mathrm{P})(\sigma_B^2 + b_0^2) \\ &+ (a_0 + b_0)\theta \phi \left(\frac{a_0 - b_0}{\theta} \right) - \mu^2, \end{aligned} \tag{4.51}$$

where ϕ is the probability density function of the standard Gaussian distribution. In order to apply the sum and maximum computations iteratively to propagate arrival times, $\max\{A,B\}$ is approximated in the same form of (3.15) as

$$\max\{A,B\} \approx M_{A,B} = m_0 + \sum_{i=1}^{n} m_i v_i + m_r v_{r_m}, \tag{4.52}$$

where m_0 is equal to μ. m_i is computed by $m_i = T_\mathrm{P} a_i + (1 - T_\mathrm{P}) b_i$. m_r is computed by matching the variance of the linear form (4.52) and σ^2 in (4.51).

The sum and maximum computations discussed till now process correlation between arrival times implicitly. This will be discussed in the following in more detail. An example of such correlation is illustrated in Fig. 4.17, where all ede delays are correlated, e.g., due to manufacturing variations. The arrival times from nodes 2 and 3 to 5 respectively, denoted as A_{25} and A_{35}, can be expressed as

$$A_{25} = A_2 + W_{25} \tag{4.53}$$

$$A_{35} = \max\{A_1 + W_{13}, A_4 + W_{43}\} + W_{35}, \tag{4.54}$$

where A_1, A_2, and A_4 are arrival times at node 1, 2, and 4, respectively. In the method from [55], the computation of the maximum of A_{25} and A_{35} requires the covariance between them. This covariance is computed as

$$\begin{aligned}\mathrm{Cov}(A_{25}, A_{35}) &= \mathrm{Cov}(A_2 + W_{25}, \max\{A_1 + W_{13}, A_4 + W_{43}\} + W_{35}) \qquad (4.55)\\ &= \mathrm{Cov}(A_2, \max\{A_1 + W_{13}, A_4 + W_{43}\})\\ &\quad + \mathrm{Cov}(W_{25}, \max\{A_1 + W_{13}, A_4 + W_{43}\})\\ &\quad + \mathrm{Cov}(A_2, W_{35}) + \mathrm{Cov}(W_{25}, W_{35}) \qquad (4.56)\end{aligned}$$

In [55], the maximum in the first two terms in (4.56) is approximated by a linear form. In order to compute the covariance correctly, the covariance computed by this linear form approximation should be equal to the covariance computed with the original maximum. This requirement is met in [55] by guaranteeing that the linear approximation has the same covariance to any other random variable. That is, for a third random variable C in linear form, written as

$$C = c_0 + \sum_{i=1}^{n} c_i v_i + c_r v_{r_c} \qquad (4.57)$$

the maximum and its linear approximation $M_{A,B}$ in (4.52) of two random variables A and B defined in (4.43) and (4.44) should meet

$$\mathrm{Cov}(\max\{A, B\}, C) = \mathrm{Cov}(M_{A,B}, C). \qquad (4.58)$$

According to [65], the left side of (4.58) can be computed by

$$\begin{aligned}\mathrm{Cov}(\max\{A, B\}, C) &= T_\mathrm{p}\mathrm{Cov}(A, C) + (1 - T_\mathrm{p})\mathrm{Cov}(B, C) \qquad (4.59)\\ &= T_\mathrm{p} \sum_{i=1}^{n} a_i c_i \sigma_{v_i}^2 + (1 - T_\mathrm{p}) \sum_{i=1}^{n} b_i c_i \sigma_{v_i}^2. \qquad (4.60)\end{aligned}$$

Similar to (4.48) and (4.49), the right side of (4.58) can be computed by

$$\mathrm{Cov}(M_{A,B}, C) = \sum_{i=1}^{n} m_i c_i \sigma_{v_i}^2 = T_\mathrm{p} \sum_{i=1}^{n} a_i c_i \sigma_{v_i}^2 + (1 - T_\mathrm{p}) \sum_{i=1}^{n} b_i c_i \sigma_{v_i}^2. \qquad (4.61)$$

From (4.59) to (4.61), (4.58) is proved, so that the arrival time computation of the method in [55] can handle correlation correctly.

The property (4.58) guarantees that the linear approximation in the maximum computation of [55] can preserve the correlation of the maximum to any random variable. Therefore, the correlation of the maximum to any independent variable v_i

is also preserved. This is the basis of the method proposed in [58]. The advantage of the method in [55] is that the correlation is handled implicitly and the computation of (4.47) and (4.49)–(4.51) needs only to be done once in a maximum computation. Therefore, this method is more efficient than [58].

In addition to the correlation between gate delays, reconvergent structures in the circuit cause further correlation. In Fig. 4.17, the arrival time A_4 at node 4 has a purely random variable v_{r_4}. The two arrival times from 3 to 5 and from 4 to 5 are partially correlated because v_{r_4} becomes a part of the arrival time of node 3 after the maximum computation at node 3. This correlation, however, is discarded in [55], because the purely random variables are merged into one variable in the maximum computation. At node 5, all the random parts of the incoming arrival times are assumed as independent. This assumption is not true because a purely random part may converge from different paths at following nodes, thus causing structural correlation [59, 60]. To solve this reconvergence problem, the canonical delay model (3.15) in [55] is extended in [61]. Instead of merging the initial purely random variables of gate delays, these variables are kept separately in arrival times during propagation. Therefore, the correlation from these random variables can be incorporated.

The linear timing analysis methods require that gate delays are approximated by linear combinations of Gaussian random variables. As in modeling gate delays, statistical timing analysis methods using nonlinear or non-Gaussian gate delays or both are proposed to improve timing accuracy. In [62], gate delays and arrival times are represented as quadratic functions of independent Gaussian random variables. The maximum computation is performed in a way similar to [58], where the covariances between the maximum and each term in the quadratic form are matched. As in [58], the first-order correlation between the maximum and other variables are preserved. The disadvantage of this method is that numerical integration is needed for each coefficient identification, which makes the proposed method slow. In order to reduce the runtime of [62], a parameter dimension reduction technique is proposed in [63]. Another method with a quadratic model is proposed in [64]. This method still uses the tightness probability from [55], but only when the maximum of two quadratic variables is Gaussian. This Gaussian property is evaluated by computing the skewness of the maximum using the formula in [65]. If the skewness is smaller than a threshold, the maximum is assumed to be Gaussian and is approximated by a linear combination of the two quadratic inputs. If the skewness is larger than the threshold, the maximum is not computed and the corresponding arrival times are directly propagated as a collection of quadratic forms. At each maximum computation, the skewness is evaluated so that the collections of quadratic forms can be compressed as soon as possible.

Representing gate delays as linear combinations of non-Gaussian variables, the method in [66, 67] approximates the maximum of two variables also using tightness probability. The difference from [55] is that the tightness probability is computed from two non-Gaussian random variables, with the formulas proposed in [68]. This method has high efficiency, but the correlation between random variables is compromised during the maximum approximation. In the nonlinear non-Gaussian

case, the method in [69] samples the nonlinear non-Gaussian parts of the variables, so that the rest part of the arrival times are linear combinations of Gaussian variables, which can therefore be processed with the method in [55]. The accuracy of this sampling-based method depends heavily on the number of samples. If the distributions of non-Gaussian variables are very complex and the number of them is large, this method faces runtime problem for moderate accuracy.

From the discussion above, correlation handling is always the source of complexity for statistical timing analysis. To avoid this complexity, correlation is simply discarded in [70], where it is proved that the result without considering correlation is an upper bound of the result with correlation after the maximum computation. Without considering correlation, the statistical bounds in [70] are very loose. Therefore, selective enumeration is deployed in [59, 71] to improve the bounding accuracy.

The algorithms discussed above are all block-based. Similar to static timing analysis, path-based methods are also explored to process statistical gate delays, e.g., in [72, 73]. To apply these methods, critical paths should be first identified. However, without a statistical timing method, the critical paths identified from static timing analysis can not be guaranteed to be critical [74]. Additionally, any path in the circuit contributes to the circuit delay distribution with certain probability. Consequently, it is not very clear how many paths should be selected for path-based methods to cover the paths which are statistically critical. Furthermore, it is very hard to implement incremental timing analysis with path-based methods, because any revision in the circuit can change the critical paths. Given these disadvantages, path-based methods are currently limited to specific areas of application.

In summary, timing analysis of flip-flop-based circuits is similar to the method for static timing analysis. The delays between flip-flops are computed with a statistical timing engine described above. The minimum clock period is computed using (4.40) with the maximum and sum replaced by the statistical computations. The result $T_{\min}$ is a random variable, whose properties define the performance distribution of the circuit. The clock feeding to all flip-flops must have a period larger than $T_{\min}$ to guarantee the proper behavior of the circuit. Therefore, timing yield of a flip-flop-based circuit at clock period T, defined as the probability that the circuit works correctly with clock period T, can be computed by (4.42).

4.4 Leakage Analysis

Leakage power is an important challenge to downscaling of the CMOS semiconductor technology. The transistor device leakage currents grow exponentially with scaling device sizes and threshold voltages. Starting with 90 nm technology and beyond, leakage current became a significant performance variable limiting scaling, and it is expected to grow as much as 3x until 2012, and further reach to 5x by 2016 [105]. As seen in current techologies, leakage has further gained emphasis since it

displays significantly higher variability, as much as orders of magnitude with respect to frequency/delay variability [106].

In this section, we will discuss the increasing importance of leakage power in integrated circuit design, describe major leakage current mechanisms, and ways to model leakage for CMOS circuits. We will also touch on statistical modeling of the leakage power and describe techniques to analyze leakage variations.

4.4.1 Background

Technology scaling enables supply voltages go down for better performance at similar power densities. However, the desired transistor threshold voltage V_t scaling factor cannot be maintained at the same rate due to increase in leakage power and related reliability issues such as various short channel effects. Hence, the current state-of-art integrated circuits push their limits with V_t to achieve fast switching performance at the cost of increased the leakage and dynamic power, especially for technology nodes after 90 nm. To solve the short channel effects, and to compensate the relatively slow scaling threshold voltages, engineers develop thinner gate oxide devices to enhance the gate drive currents, but this also increases the tunneling current across the gate and results in higher gate leakage currents. Furthermore, the physical limits of the channel engineering, the existence of fewer dopant atoms in the device channel, sophisticated halo designs and higher manufacturing and operating environment variabilities all contribute to the significance of leakage power for high performance integrated circuits.

Current design methodologies often include planning, estimation, and optimization for leakage power as a major design objective none less important than dynamic power. As we approach the fierce frequency wall and limits of device scaling, where the circuit and system performances are more and more limited to power consumption of various products such as ubiquitous handheld mobile products and multicore microprocessor chips, the leakage power has rightfully gained its own importance for semiconductor manufacturers.

4.4.2 Types of Leakage Current

Leakage phenomenon in CMOS and SOI transistor devices is studied extensively. The term leakage stems from the current that is undesirable and not functionally useful, since the CMOS device is intended to be at the off state, and is expected not to leak, therefore hold state indefinitely. The non-existence, or negligible quantities of leakage current of CMOS devices before 130 nm technology node was a big force for moving integrated circuit designs from BJT analog transistors to CMOS technology. But for current deep sub-micron technologies, leakage current, that is

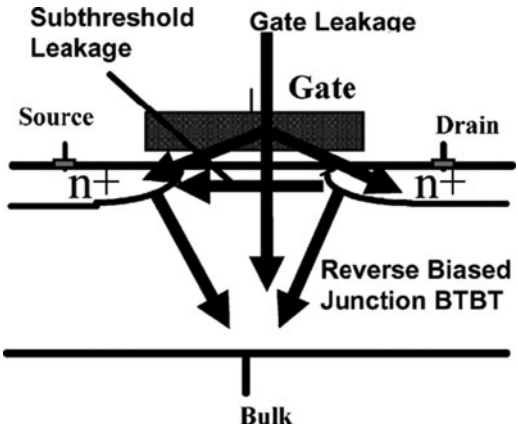


Fig. 4.18 Various types of leakages for an Nfet transistor

the undesired current flowing through the device channel at their non-operational behavior is no longer negligible, and causes excess power dissipation.

The major contributors of the transistor device leakage are subthreshold leakage, gate oxide leakage and band-to-band tunneling leakage.

The transistor device leakage can be modeled with the governing device topology. The subthreshold current flows from drain to source nodes, the gate leakage flows from the gate node to source, drain and substrate nodes, and the band-to-band tunneling current is divided into its drain and source components, both flowing to the substrate node. For SOI devices, we don't have tunneling current due to the insulator layer and hence it can be ignored. The leakage currents are all voltage controlled current sources and functions of gate, source and drain voltages. The figure below depicts these types of leakage currents in a nfet CMOS transistor (Fig. 4.18).

Next section, we will model these types of leakage currents in more detail.

4.4.2.1 Subthreshold Leakage

Subthreshold current, I_{off}, is the drain current of a transistor in the subthreshold region and can be expressed as

$$I_{\text{sub}} = I_0 10^{(V_{\text{gs}} - V_t)/S(T)}, \tag{4.62}$$

where I_0 is the drain current with $V_{\text{gs}} = V_t$, and $S(T)$ is the subthreshold slope at the requested temperature. Subthreshold leakage is caused by the minority carriers drifting across the channel from drain to source when the transistor device operates when $V_{\text{gs}} < V_t$. To calculate I_0, device models can be used for subthreshold operating regime of the device that is often implemented in BSIM device models. For simplicity, we could model I_0 as

$$I_0 = \mu_0 C_{\text{ox}} (W_{\text{eff}}/L_{\text{eff}}) (kT/q)^2 (1 - e(V_{\text{ds}}/V_t)), \tag{4.63}$$

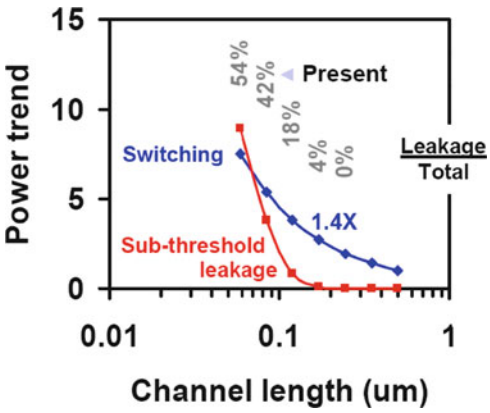


Fig. 4.19 Subthreshold leakage trend for VLSI technology nodes [30]

where W_{eff} and L_{eff} are effective device width and lengths, μ_0 is the mobility, C_{ox} is the gate capacitance, k is the Boltzmann constant. More advance models introduce short channel effects, body effect, and narrow channel effects. Further studies also accounted for quantum mechanical confinement of electron/holes in the depletion/inversion regions of the device. Such extensions impact the threshold voltage of the device that significantly impacts the subthreshold leakage of the device [75].

Previous projections show that in the 90 nm process node, the subthreshold leakage power can contribute as much as 40% of the total power [76]. Hence, it is imperative to model the subthreshold current as accurate as possible. Subthreshold current is exponentially dependent on the threshold voltage, and hence accurate models should account for threshold voltage variation effects (Fig. 4.19).

4.4.2.2 Gate Oxide Leakage

Gate oxide leakage is due to the tunneling of electrons from the bulk silicon and drain/source overlap region through the gate oxide barrier into the gate area. Gate oxide leakage increases exponentially with the decrease in the oxide thickness and the increase in the potential drop across the oxide. It has been considered as the sum of gate oxide leakage components, including the source/drain overlap region current, gate to channel current, and gate to substrate leakage currents (Fig. 4.20).

$$I_{\text{gate}} = I_{\text{gc}} + I_{\text{god}} + I_{\text{gos}}. \tag{4.64}$$

Equations for I_{gate}, I_{gos}, and I_{god} have same functional form, with the difference of being dependent on V_{gs} and V_{gd}, respectively. These variables are functions of effective gate length L_{eff}, terminal voltages, oxide thickness, and temperature. A simplified gate oxide leakage model can be provided as:

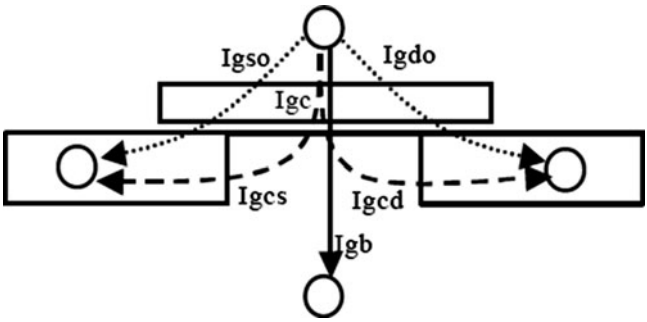


Fig. 4.20 Components for gate oxide leakage phenomenon

$$I_{\text{gate}} = (AC)W_{\text{eff}}L_{\text{eff}} \exp\left(-BT_{\text{ox}}\alpha/V_{\text{gs}}\right). \tag{4.65}$$

As seen, I_{gate} is strongly influenced by the gate voltage and the oxide thickness.

4.4.2.3 Tunneling Leakage

In an nfet device, when the drain or source is biased higher than the device substrate, a significant tunneling current flows through the drain-substrate and source-substrate junctions. For SOI, this current is negligible. The classical diode models can be applied to estimate the tunneling leakage for large scale circuits. For such band-to-band tunneling leakage (BTBT), [75] explains a highly accurate numerical model that integrates the sum of the currents flowing through the drain-substrate and source-substrate junctions.

4.4.3 Leakage Model for Logic Cells and Circuits

Using the models for leakage currents, one can use a detailed device model to be used with a transistor simulation environment. For this, one can use controlled current sources across the device terminals to build the required numerical model [75]. The overall leakage is then analyzed using this controlled current source model as shown in Fig. 4.21.

For circuits composed of multiple devices, the current source models can all be integrated and analyzed via a circuit simulation engine that honors universal current and voltage preservation laws. Note that the characterization of the leakage of a cell requires calculation of the leakage currents for each input vector of the gate. Since leakage is calculated for a steady-state condition, a DC analysis can be performed via the circuit simulator for the requested circuit inputs.

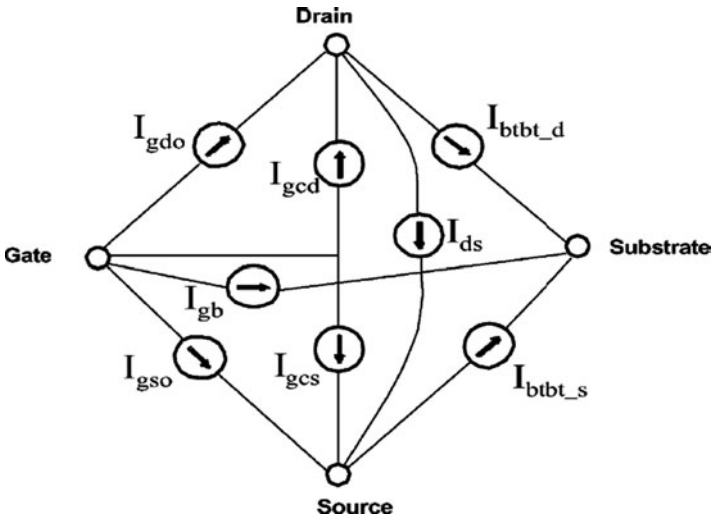


Fig. 4.21 Controlled current source model for leakage analysis

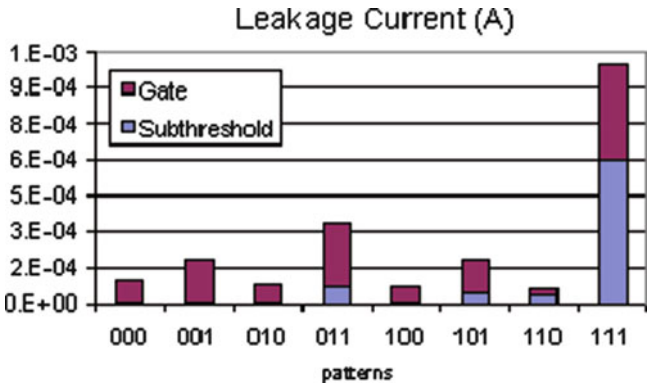


Fig. 4.22 Leakage currents for each input for 3-input NAND gate

Since leakage is a strong function of the terminal voltages, we see significant variations within different inputs. Figure 4.22 depicts this for a 3-input NAND gate, and table displays the leakage for each input vector. Here, we performed device models from a state-of-art process technology node.

Most generally, dynamic evaluation of the leakage for all possible input combinations is quite costly and inefficient. In a large logic circuit, not all the cells are generally at their high leakage states. The balancing of high and low leakage states for various cells implies an averaging effect for each cell. Hence for more efficient leakage estimation, static (input-independent) analysis techniques are preferred within the design methodologies. Over a long period of circuit operation, the cells in the logic circuit may be at numerous states, and assuming all input states

are equi-probable, we can characterize the leakage by its mean statistic. If the probabilities of each cell input are known either by designer intuition, known primary input characteristics or via logic simulation results, we could model the mean cell leakage as:

$$I_{\text{cell}} = \sum_{\text{state}_k} \text{Prob}(\text{state}_k) I_{\text{state}_k}. \tag{4.66}$$

For the logic circuit consisting of many cells, we could simply add the individual cell leakages due to the near-perfect isolation between the logic boundaries, and therefore the mean circuit leakage can be derived as:

$$I_{\text{circuit}} = \sum_{\text{cell}_i} I_{\text{cell}_i} \tag{4.67}$$

4.4.4 Static Leakage Estimation

When the operating environment and the technology are kept constant, it is desirable to develop a static (input-independent) method for predicting the average leakage under possible input conditions. As technology requires smaller and faster logic stages and more radical design styles, the leakage becomes more dependent on the input states. This is also true with the existence of control inputs as for the case of header/footer designs, or special inputs which can turn off some sections of the circuit. Therefore in this section, we will introduce a probabilistic static leakage estimation method. We will focus on combinational circuit as the core building blocks of conventional digital integrated circuits.

The logic circuits often hold full logic values at the cell boundaries and the total leakage of the logic circuit is mainly the sum of leakages coming from all the cells combined. Let us assume, the leakage for each cell type is pre-characterized for all its input states, i.e., I_{state_i} . This can be done once with an accurate circuit simulator during library generation step.

Using a concept of occurrence probability that describes the likelihood of a circuit boundary node (input or output) holding a full logic value, we define the node occurrence probability of node n, as the likelihood of observing the node n at a logic value 1: $\pi_n = \text{Prob}(n = 1)$. Hence, the probability of observing n at value of 0 would be $1 - \pi_n$. We can further define the state occurrence probability $\text{Prob}(\text{state}_i(x))$, as the probability of observing the cell i at state x. If the cell inputs are independent, computation of $\text{Prob}(\text{state}_i(x))$ is simply the multiplication of the associated node occurrence probabilities. Furthermore, one can simply propagate node occurrence probabilities from cell inputs to cell outputs via following the logic functionality of the cell as described in [77].

Using the node and state occurrence probabilities, we can evaluate the average leakage of the logic circuit as the weighted sum of the leakage for all cells in each state. The weights are simply the state occurrence probabilities. Moreover, same weights can be applied for leakage components, i.e., gate and subthreshold leakages. These values can also be pre-characterized in the library creation step.

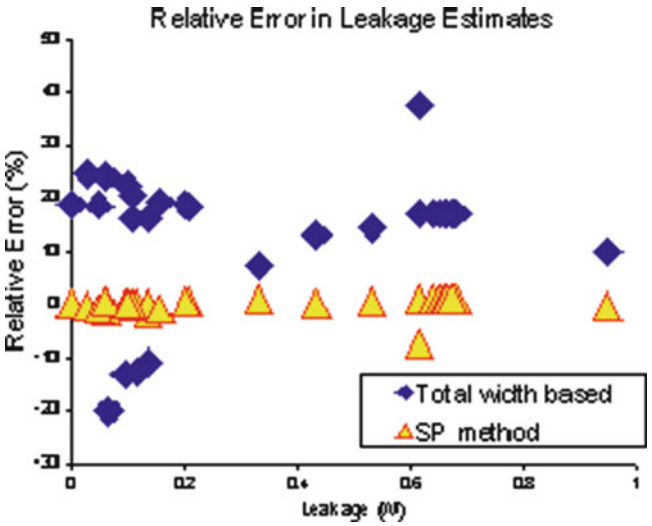


Fig. 4.23 Accuracy for probabilistic simulation for leakage analysis

This approach was experimented for ISCAS combinational circuits that are synthesized using a library of basic gates with delay constraints. The total leakage of each circuit for a given input vector is estimated via circuit simulator at nominal conditions. Each primary input is assumed to have binary node occurrence probability of 0.5. The results shown in Fig. 4.23 validates accuracy of the average relative error for the probabilistic technique. Compared to total-width-based leakage estimate, which totally ignores the probabilities of the logic states and averages out all the devices, the probabilistic simulation-based techniques have accuracy within 2% versus 20–30%. This is achieved by accurately capturing the high and low leakage states of the circuit cells from the state occurrence probabilities and accounting their leakage estimates accurately in the total circuit leakage power estimate.

4.4.5 Stack Factor-Based Leakage Estimation

The models mentioned above are useful when the logic cells are fully characterized. When such libraries do not exist and the logic circuits are more custom designed with arbitrary circuit topologies, empirical models for leakage are often used. These models simply accumulate leakages of each transistor in the circuit. Since leakage is dominated by the subthreshold leakage component, overall model quality depends on the accuracy of the subthreshold leakage.

The most simplest approach for subthreshold leakage estimation is area or total width based. However to properly account for leakage of channel-connected devices, we must consider the stacking effect for leakage.

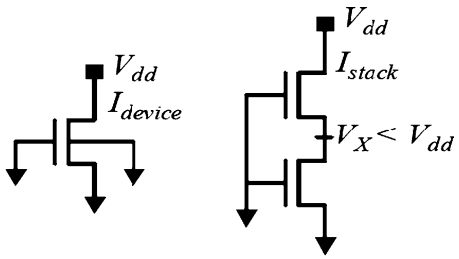


Fig. 4.24 Stacked transistor device

Figure 4.24 demonstrates a stack of transistors. The upper device has an internal source node causing a lower V_{gs} and therefore a higher threshold voltage. Hence, it would generate a lower subthreshold current than its nonstacked version. We can also explain this by the bottom device with a lower V_{ds}, compared to its nonstacked version that sees the full supply voltage level V_{DD}. Since subthreshold current for the stack of transistors will be limited by these factors, the total leakage of the stack of devices is considerably lower than the sum of their nonstacked versions. The same applies to higher stack sizes, and can be generalized [78].

In today's microprocessors and high-performance circuits, the stack depth is often less than 4 for performance constraints. Hence from the characterization point of view, we can build models for various stack sizes and build models for leakage estimation [78].

Like stacking, body biasing also reduces subthreshold leakage exponentially. Also, the leakage is also modulated by DIBL effect, as V_{ds} increases, the channel energy barrier between the drain and source is lowered. Hence, this effect also exponentially increases the subthreshold leakage. The derivation of the stacking factors could include all these physical effects.

One could model the stacking effect using the original device model equation (4.62) but this can be too time consuming for chip-level analysis. To overcome this, the following empirical model can be used:

$$I_{\text{sub,stacked}} = I_{\text{sub}} W_{\text{tot}} / X_{\text{S}}, \tag{4.68}$$

where I_{sub} is the subthreshold leakage for a device with no stacking, W_{tot} is the total transistor width, and X_{S} is the empirical stacking factor. X_{S} is the de-rating factor for the reduction in the subthreshold leakage due to the stacking effects. [78] studies the derivation of X_{S} for various stack topologies and concludes that typical values of 2 and 3 are used for cache and core circuits for high-performance microprocessor designs.

For more accuracy, certain blocks of circuits can be pre-characterized to determine the best stacking factor. Such primitive cells can use basic, domino, complex gates with standardized stacks. Once the proper stacking factors are found, it can be used in full-chip level leakage analysis.

4.4.6 Leakage Current Estimation Under Variability

Leakage current components, subthreshold and gate in particular, are super-linearly dependent on effective device length and threshold voltages. For gate oxide leakage, we also have a similar dependence on the oxide thickness. In technologies for 90 nm and beyond, we do see significant variability of these device parameters due to imperfect manufacturing conditions and tool, fab limitations. Moreover, the operating environment for the integrated circuit may present significant variations in temperature and supply voltage. Under such variability, leakage performance is amenable to show large amount of variability. This motivates us to model leakage current within the existence of parametric variabilities.

Lets assume the threshold voltage of a device, V_t as a Gaussian distribution with mean V_{t0} and standard deviation σ_{V_t}. If we convolve this distribution with the subthreshold leakage model with respect to V_t, the result is that leakage is not distributed as Gaussian and the mean leakage is skewed due to the nonlinear relationship between the subthreshold leakage and V_t. Similar study could be done on channel length variability, δ_{L}. Therefore, the average leakage estimate under parametric variations need a more careful study.

Our new leakage model that considers parametric variability will be based on the average leakage estimate and will contain a multiplier that reflects the uplift resulted in the variability of the underlying parameter. This could be formalized as:

$$I_{\text{leakage}} = I_{\text{nominal}} * f(\Delta P), \tag{4.69}$$

where P is the process parameter that impacts the leakage current. Like in BSIM models, the typical functions for $f()$ is quite nonlinear and complex. Hence, more efficient analysis can be performed by accurate empirical equations valid within the operation and manufacturability regions.

The formalism in parameter P also allows the decomposition of global and local variability, i.e.,

$$\Delta P = \Delta P_{\text{global}} + \Delta P_{\text{local}}, \tag{4.70}$$

where ΔP_{global} and ΔP_{local} models the die-to-die variability (a.k.a. global) and within-die variability (a.k.a. local), respectively.

We illustrate the method using a single parameter on gate leakage. The oxide thickness is reduced to increase device mobility and speed, but this increases gate leakage current significantly leading to nonlinear relationship between variations in ΔT_{ox} and I_{gate}. Other parameters do not affect gate leakage at the same magnitude and can be ignored.

Hence, we can write, with significant variability in oxide thickness, ΔT_{ox}, the gate leakage term can be written as:

$$I_{\text{gate}} = I_{\text{gate,nom}} \exp(f(\Delta T_{\text{ox}})). \tag{4.71}$$

For simplicity, let us express the $f(\Delta T_{\text{ox}})$ as a linear function as $-T_{\text{ox}}/\beta_1$. T_{ox} can also be decomposed into global and local components as:

$$T_{\text{ox}} = T_{\text{ox},g} + T_{\text{ox},l}. \tag{4.72}$$

Hence, the gate leakage with variability due to T_{ox} can be written as:

$$I_{\text{gate}} = I_{\text{gate,nom}} \exp(-\Delta T_{\text{ox},g}/\beta_1) \exp(-\Delta T_{\text{ox},l}/\beta_1). \tag{4.73}$$

We denote the gate leakage current of a single device with unit width as I_{gate} and its nominal term $I_{\text{gate,nom}}$ is modeled when no variations in T_{ox}. We further assume that $\Delta T_{\text{ox},g}$ and $\Delta T_{\text{ox},l}$ are both zero-mean normal random variabilities. The model in (4.73) leads to the estimate for the average I_{gate} as:

$$E[I_{\text{gate}}] = S_{\Delta T_{\text{ox},l}} I_{\text{gate},\Delta T_{\text{ox},g}}, \tag{4.74}$$

where the uplift factor due to the within-die variations in T_{ox} is

$$S_{\Delta T_{\text{ox},l}} = \exp(\sigma^2_{\Delta T_{\text{ox},l}}/2\beta_1^2) \tag{4.75}$$

and, the baseline gate leakage estimate for the global chip-mean $\delta T_{\text{ox},g}$ is:

$$I_{\text{gate},\Delta T_{\text{ox},g}} = I_{\text{gate,nom}} \exp(-\Delta T_{\text{ox},g}/\beta_1). \tag{4.76}$$

The uplift factor is based on the standard deviation of the local component in T_{ox} variability, and the baseline leakage estimate can be found by the known global variation in $\Delta T_{\text{ox},g}$ for all the devices. By using (4.74), one can assess the average gate leakage under various global and local combinations in T_{ox} variability. The uplift factor $S_{\Delta T_{\text{ox},l}}$ is responsible for the within-die variations and exponentially increase with its standard deviation.

The same model can be extended to the subthreshold leakage I_{sub}. As previously noted, the subthreshold leakage has an exponential relation with its major contributor, V_{th} which leads to:

$$I_{\text{sub}} = I_{\text{sub,nom}} \exp(\Delta V_{\text{th}}). \tag{4.77}$$

This exponential relationship is the cause of significant uplift of the average leakage under parametric variations in V_{th}, as much as a few orders of magnitude. Similar important parameters that impact device leakage and performance are channel length L_{eff}, oxide thickness T_{ox}, dopant concentration N_{sub}. Most of these parameters may have interrelated and can be represented via their principal components. In similar manner, we can model threshold voltage variations as a sum of variability due to channel length, and dopant concentrations, as:

$$f(\Delta V_{\text{th}}) = f(\Delta L_{\text{eff}}) + f(\Delta N_{\text{sub}}). \tag{4.78}$$

This is simply the generalization of the model derived for gate leakage term with ΔT_{ox} variations. As derived in [79], it leads to a similar model of subthreshold

leakage as a function of global and local variations in channel length and dopant concentrations, as:

$$E]I_{\text{sub}}] = S_{\Delta_{\text{L}_{\text{eff},l}}} S_{\Delta_{\text{NSUB},l}} I_{\text{sub},\Delta_{\text{L}_{\text{eff},g}},\Delta_{\text{NSUB},g}}, \tag{4.79}$$

where $S_{\Delta_{\text{L}_{\text{eff},l}}}$ and $S_{\Delta_{\text{NSUB},l}}$ are uplift factors for variability in channel length and dopant concentration affecting threshold voltage variations. For more details on this derivation, the reader may refer to [79].

For total leakage of a chip, we simply sum the subthreshold and gate leakage currents for all the devices using the uplift factors derived from the within-die variation statistics. The result would give average leakage estimate under various combinations of global and local parametric variability and gives very useful yield assessments when coupled with frequency estimates.

4.4.7 Conclusion

Leakage analysis has become a key topic for recent integrated circuit technologies, as power especially leakage power became a more dominant performance limiter. In this section, we outlined the leakage current phenomenon, and discussed details on subthreshold and gate leakage types in general. We discussed modeling leakage for a device and circuit, and its generalization to larger circuit blocks, all the way to chip-level. We presented the issues on static and probabilistic estimation of leakage power and the model extensions that cover parametric variability.

Leakage analysis is a key component in circuit design, optimization, and manufacturing. It is an essential topic in design verification for years to come.

4.5 Dynamic Power Analysis

Since leakage analysis is described in Sect. 4.4 in detail, this section focuses on statistical analysis methods for dynamic power.

The dynamic power of a gate cell is caused by two different effects: (1) the charging and discharging of gate-internal and external parasitic capacitances and (2) short-circuit currents through the gate during switching. The short-circuit power depends on the amount of time where the input voltage is in a range between the thresholds of both, the nMOS and pMOS transistors, so that both are open during that time resulting in a current flow from V_{dd} to ground. This effect mainly depends on the input slope time and usually counts for a smaller part of the total dynamic power. The dominant part is caused by charging and discharging of parasitic capacitances, which can be divided into gate-internal capacitances and capacitive loads of wires and driven gates connected to the gate output. Usually, these are combined to an effective load capacitance C_{load}, which is charged at each

switching event of the gate output. Because both effects directly depend on signal transition rates, the calculation of switching activities for each net in the circuit is an important task.

In this section, two aspects of dynamic power are covered: (1) Glitch power and how it can be determined by probabilistic methods and (2) the influence of process variations on timing and power and how to consider them in digital gate-level simulation.

4.5.1 Probabilistic Glitch Power Analysis

The dynamic power caused by net n can be calculated by

$$P_n = \frac{1}{2T_C} V_{\text{dd}}^2 C_{\text{load}} \alpha_n, \tag{4.80}$$

where T_C is the clock frequency, V_{dd} equals the supply voltage, and α_n is the switching probability of net n, which is also called transition density. C_{load} is the effective load capacitance that is switched by the driving gate cell of net n.

Several approaches have been presented to determine signal transition densities. Since simulation-based methods are very time consuming, faster approaches have been developed based on probabilistic methods. In this section, an overview of these methods will be given. It is crucial to consider delays precisely when estimating transition rates. Methods will be shown that use pattern- and slope-dependent delay models, taking into account process variations.

We define glitches as the functionally unnecessary portion of the total signal transitions. They are caused by unbalanced paths from throwing latches to the inputs of a logic gate. This results in different signal arrival times that unnecessarily forces a gate to switch and switch back during one clock cycle. These glitches are also propagated through the circuit. However, if the difference Δ_t between the arrival times is below a gate-specific delay, then no glitch occurs at the output. This phenomenon is called hazard filtering and it is used to leverage the constraints for a glitch-free design. In order to determine glitches at gate level, the delay model must be sufficiently accurate (see [80]). A simple method to distinguish between glitches and functional transitions is to determine transition rates in two different ways. Functional transition rates $\alpha_{i,\text{func}}$ are obtained by using a zero delay model. Using a more precise delay model considering hazard filtering leads to $\alpha_{i,\text{all}}$ which includes both functional transitions and glitches. Glitch rate then equals the difference $\alpha_{i,\text{glitch}} = \alpha_{i,\text{all}} - \alpha_{i,\text{func}}$.

Monte Carlo simulation results are very accurate if an appropriate number of samples is used. Since this is very time consuming, other methods have been developed to overcome this problem. Najm et al. proposed a probabilistic simulation method, which was implemented in the early power estimation tool CREST [81].

Table 4.3 Extended probabilistic waveform table

$P_{(00,00)}$	$P_{(10,00)}$
$P_{(00,01)}$	$P_{(10,01)}$
$P_{(11,10)}$	$P_{(01,10)}$
$P_{(00,11)}$	$P_{(01,11)}$

4.5.1.1 Probabilistic Simulation

In contrast to ordinary digital simulations, probabilistic simulation methods use signal and transition probabilities as signal values. Each gate maps the input signal combination probabilities to output probabilities. These probabilities are propagated starting at the primary inputs through the whole combinational part of the circuit only once.

Using probabilistic waveforms, a signal is defined by a tupel of four probabilities at any time: the probability that the signal remains low (P_{00}), transitions from low to high (P_{01}), transitions from high to low (P_{10}), and that the signal remains high (P_{11}). The sum of these probabilities always equals one.

Output signals of a gate can be derived from the input probabilities by using lookup tables. Efficient probabilistic waveform simulation can be performed using an event-driven mechanism, which allows to consider transport delays easily.

4.5.1.2 Extensions for Hazard Filtering

Using the formulation from above, it is not possible to take hazard filtering into account. The reason is that no information on temporal correlation between different events exists. In order to determine glitch power more precisely, probability waveforms were reformulated as presented in [82].

The idea is to separate the probabilities for different cases. Probability tupels allow to describe different signal histories given by the previous event. Doing so an event is characterized by 8 probabilities instead of 4 probabilities. Table 4.3 shows the events that can occur. $P(e_\mathrm{p}, e_\mathrm{c})$ denotes the probability for the combination of the current event e_c with the previous event e_p. An event can be 00, 11 for keeping low and high or 01, 10 for transitions from low to high or from high to low, respectively.

This concept can also be extended to represent more previous events. Although the number of separate probabilities strongly increases with the number of previous events that are considered. Therefore, a trade-off must be chosen. In the following, we use the formulation from Table 4.3. By observing an inertial delay window, the extensions allow to identify glitches, which are not propagated due to inertial gate delay (see [82]).

Table 4.4 Results of transition density determination

	Monte Carlo simulation	Extended probabilistic simulation [82]
Runtime	0.364 µs	0.011 µs
Speedup		33
Max. error		24.30%
Avg. error		5.97%

4.5.1.3 Experimental Results

A 1-bit full adder was chosen as an example to investigate both speedup and error compared to Monte Carlo simulation. Table 4.4 shows the averaged transition density results for all nets.

The speedup of 33 is payed by an average error of nearly 6%. The example circuit has several reconverged paths where the signals are correlated. The error of the probabilistic simulation is caused by neglecting these correlations between the input signals of the reconverging gates and by a limited consideration of temporal correlation.

4.5.2 Monte Carlo Digital Power Simulation

In simulation-based power estimation approaches, transition densities are determined in a digital gate-level simulation and then fed to the actual power estimation tool. This is done for certain interesting input patterns resulting in quite accurate, testcase-specific power measures. One methodology to estimate the influence of process variations on dynamic power and also on timing behavior of a digital circuit is Monte Carlo simulation, which can be very accurate regarding the effects of variations but time consuming if the single simulation run takes longer time. Transistor-level models such as SPICE netlists provide the advantage that they support to map many of the interesting process parameters directly to parameters of the used transistor models such as BSIM, but they have the disadvantage that they need too much computational effort for larger circuits. This makes them less suitable for the Monte Carlo approach. A huge acceleration can be achieved when moving from transistor level to behavioral models as used in digital gate-level cell libraries. Here, the process variations have to be mapped to parameters of the cell models which are usually the input slope times and output load capacitances. This section describes such an approach for a statistical gate-level simulation flow based on parameter sensitivities and generated VHDL cell models. The solution provides a good speed/accuracy tradeoff by using the event-driven digital simulation domain together with an extended consideration of signal slope times directly in the cell model.

4.5.2.1 Modeling Approach

The characterization of digital components using the nonlinear delay and power model (NLDM, NLPM) is widely used as an industry standard. Input-to-output delays, slope times of the output signals, and the consumed energy of a switching event are characterized with respect to the slope time of the input signals and the output load capacitance considering the logic state and the direction of signal edges (compare Sect. 4.1).

$$\begin{aligned} \text{delay} &= f(\text{slope}_{\text{in}}, C_{\text{load}}) \\ \text{slope}_{\text{out}} &= g(\text{slope}_{\text{in}}, C_{\text{load}}) \\ \text{energy}_{\text{sw}} &= h(\text{slope}_{\text{in}}, C_{\text{load}}). \end{aligned} \tag{4.81}$$

For N nominal parameters $p_{\text{nom}} \in \mathbb{R}^N$, the functions f, g, and h are represented by two-dimensional lookup tables determined by SPICE simulations for each combination of typical input slope and output load values during cell library characterization. In the nominal case, the standard cell library will not contain any parameter data and is valid only for the PVT corner it was characterized for.

In the case of statistical analysis, the parameters can vary and are characterized by random variables. The dependency of delay, slope, and switching energy on the parameters can be expressed in the simplest way by first-order (linear) sensitivities $\frac{\partial f}{\partial p_i}$, $\frac{\partial g}{\partial p_i}$, and $\frac{\partial h}{\partial p_i}$ for each parameter p_i. The functions f, g, and h are then extended by a variable part:

$$\begin{aligned} \text{delay} &= f(\text{slope}_{\text{in}}, C_{\text{load}}) + \sum_{i=1}^{N} \frac{\partial f}{\partial p_i}(\text{slope}_{\text{in}}, C_{\text{load}}) \cdot \Delta p_i \\ \text{slope}_{\text{out}} &= g(\text{slope}_{\text{in}}, C_{\text{load}}) + \sum_{i=1}^{N} \frac{\partial g}{\partial p_i}(\text{slope}_{\text{in}}, C_{\text{load}}) \cdot \Delta p_i \\ \text{energy}_{\text{sw}} &= h(\text{slope}_{\text{in}}, C_{\text{load}}) + \sum_{i=1}^{N} \frac{\partial h}{\partial p_i}(\text{slope}_{\text{in}}, C_{\text{load}}) \cdot \Delta p_i. \end{aligned} \tag{4.82}$$

The parameters p_i shall describe statistically independent variations and can be derived from the usually correlated, technology-specific device parameters by statistical methods such as principal component analysis (PCA) for Gaussian-distributed variables. The linear sensitivities with respect to these independent parameters have to be determined once by the cell characterization step, too. For that purpose, some SPICE simulators provide special analyses for single-run sensitivity calculation to reduce the characterization effort [83]. Special consideration needs the distinction between intra-die (local) and inter-die (global) variations. For the application on gate-level netlists, it turned out to be more practical to introduce separate variables for the local components of parameters because separate, instance-specific random values have to be calculated for them during simulation. This additionally provides

the possibility of further abstraction by combining these local components to a single new parameter describing all local variations of a gate cell instance.

The linear approach provides valid results only as long as the dependencies of the cell quantities f, g, and h on the technology parameters are approximately linear in the interesting range. In many practical cases, this has been proven for the timing and dynamic power quantities. However, as shown in [83], the method may also be applied to nonlinear problems when a transformation to a new, linear-dependent quantity can be found as it is possible for leakage current variations. Furthermore, general nonlinear functions f, g, and h are possible as they are, for instance, used in [69] for statistical static timing analysis (SSTA).

In the rest of this section, process parameters are assumed to be Gaussian distributed. In general, this has the advantage that all quantities that are linear-dependent on these parameters are also Gaussian, which eases the prediction of their probability distribution by mean and variance. At cell level, this is always the case for the quantities f, g, and h because of the linear sensitivity approach. At chip level, quantities such as path delays or mean dynamic power may have a more nonlinear dependency because of the complexity of possible paths through the netlist and the occurrence of glitches. But in many cases, some of these chip-level quantities show Gaussian distribution too.

For analysis and modeling of parameter variations and correlations, also compare Sects. 2.2 and 3.3.

4.5.2.2 VHDL Modeling and Design Flow Aspects

Because the active power of a larger circuit strongly depends on the applied stimuli, the toggle activity at each node of the digital circuit has to be determined before any realistic power estimation. Conventional tools typically can use either a given mean activity at the circuit's input pins or the result of a prior functional digital simulation, e.g., in VCD format. These two steps can be joined into a common simulation of timing behavior and power consumption when the cell model directly accesses the timing and energy tables of the cell library and considers the signal slope times dynamically as described below. This results in a slightly increased simulation effort but has two advantages. First, there is no static timing analysis (STA) step needed like in standard digital gate-level simulation with back-annotated static delays (e.g., in SDF format). Second, by evaluating the slopes dynamically during simulation, delays and the shape of signal edges can be modeled more exactly which enables further analyses, e.g., of glitches and the portion of dynamic power caused by them.

As a prerequisite for the statistical simulation, the parameter sensitivities have to be characterized and provided as additional tables in the cell library. This can be realized either by a user-defined extension of the commonly used Liberty format which allows such extensions by its syntax or by a full proprietary format, e.g., based on a scripting language such as Perl to ease further processing. (Sect. A.3 contains an example listing from a Liberty file with such extensions.) From this extended library, the delay, slope, energy, and sensitivity tables can then be extracted

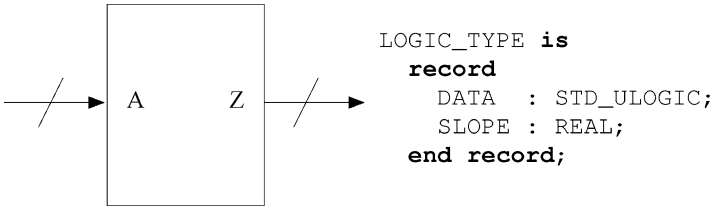


Fig. 4.25 Buffer cell with ports A and Z of type LOGIC_TYPE

```
library VHDL_UTILITY;
use VHDL_UTILITY.STATISTICS.all;

package INTER_CELL_PKG is
  ...
  -- calc. gauss. random value with mean 0.0 and std.dev. 1.0
  constant D_TOX_THIN : REAL := NORMAL(0.0, -1.0, 1.0, FALSE,
      1.0);
  ...
end package INTER_CELL_PKG;
```

Listing 4.1 Package to determine random parameter variations Δp_i

and written to corresponding VHDL array variables. As far as the cell library also contains sufficient information about the logical function of each cell type, the complete VHDL cell models can also be generated automatically.

The cell model has the task to calculate cell delay, output slope time, and switching energy for each transition at one of its input signals based on (4.82). The required arguments slope_{in} and C_{load} are provided in different ways: the effective load capacitance can be assumed to be constant over time and is therefore passed as a generic parameter to the cell model instance; the input slope time, however, depends on the output slope time of the driving cell and is therefore carried together with the logic value from cell to cell using a two-valued signal data type (see Fig. 4.25) replacing the standard logic data type STD_ULOGIC.

The independent random parameter values Δp_i for the global variations can be determined once during the elaboration phase of each simulation run using functions of the VHDL-AMS standard package SAE J2748 for different probability density functions [84, 85]. To avoid an additional scaling of the calculated random numbers to the individual range of each parameter, all parameter and sensitivity values are normalized to a mean of 0.0 and a standard deviation of 1.0. Listing 4.1 demonstrates how a global variation of the normalized parameter oxide thickness D_TOX_THIN is determined. Using a local file to remember the last random number and taking it as seed for the next call, the function NORMAL delivers a new random value for each simulator start.

This is done for each parameter, and then (4.82) are applied by multiplications of the Δp_i values with the tabulated sensitivities and a summation of all these variation portions with each of the three nominal tables resulting in a set of three varied tables

```
{entity R_SBUF is
  generic ( Z_LOAD : REAL := 0.0 );
  port    ( A : in   LOGIC_TYPE;
            Z : out  LOGIC_TYPE );
end entity R_SBUF;

architecture behav of R_SBUF is
  -- first, look up values for the given eff.
  -- load capacitance Z_LOAD for arc A->Z
  constant Z_A_DELAY_VEC      := LOOKUP_2D_CUT(Z_LOAD,
                                               LOAD_INDICES,
                                               SLOPE_INDICES,
                                               DELAY_TABLE);
  constant Z_A_SLOPE_OUT_VEC  := LOOKUP_2D_CUT(Z_LOAD,
                                               LOAD_INDICES,
                                               SLOPE_INDICES,
                                               SLOPE_TABLE);
  ...
begin
  ...
  process (A.DATA)
  begin
    Z_NEW   := A.DATA; -- logic function
    Z_DELAY := LOOKUP_1D(A.SLOPE,
                         SLOPE_INDICES,
                         Z_A_DELAY_VEC) * (1 ns);
    Z_SLOPE := LOOKUP_1D(A.SLOPE,
                         SLOPE_INDICES,
                         Z_A_SLOPE_OUT_VEC);
    Z.DATA  <= A.DATA  after Z_DELAY;
    Z.SLOPE <= Z_SLOPE after Z_DELAY;
    ...
  end process;
end architecture;}
```

Listing 4.2 Essential parts of the VHDL cell model for the buffer example

for delay, slope, and energy. During simulation, the cell model then looks up values only from these modified tables. Listing 4.2 shows essential parts of the cell model's VHDL code of the buffer example from Fig. 4.25.

User-Defined Functions for Importance Sampling

In the case of standard Monte Carlo simulation, a huge number of simulation runs are needed if marginal probabilities are estimated. One method to overcome this problem is importance sampling (see also Sect. 2.2.8). The samples are generated in a special way that reduces the variance of the estimator of a probability compared to standard Monte Carlo simulation for the same number of runs. The presented simulation approach also allows to declare special distribution functions for importance sampling in addition to the functions provided by SAE J2748 [84].

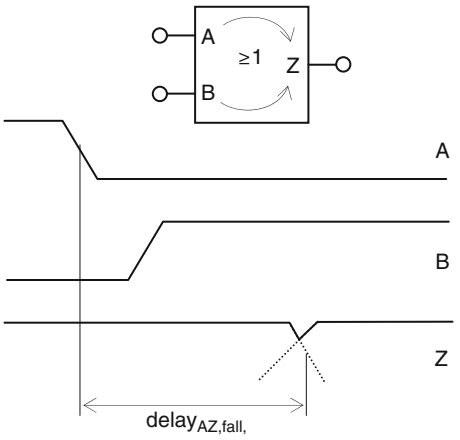


Fig. 4.26 Possible simplified glitch shape for an XOR gate (not to scale)

Power Analysis

Like in standard tools, the active power calculation is based on counting of switching events at cell outputs. For each such switching event, a corresponding energy value is taken from the cell library and can then be added up to a total average power or traced to a discrete power signal over time. Beyond that, the direct consideration of slope times by the model as described above allows a much more precise estimation of the real, analog waveforms of the output signals than in standard digital simulation.

This is very helpful when analyzing the occurrence and scale of single glitches. Figure 4.26 shows an example where the peak voltage of the glitch is less than half of the full swing, caused by different delays of the two timing arcs through the gate. This small glitch should not cause the subsequent gates to toggle and should not be visible with standard digital simulators because of the inertial delay model of the cell model, which ignores pulses that are shorter than the cell delay. But it will consume energy in the driving gate cell itself, which would also be ignored in a conventional power estimation flow. However, using the slope information, the glitch can be recorded into a file and considered for power calculations in a postprocessing step. Longer glitches are visible also during simulation and can thus be considered also for online calculations. Nevertheless, a calculation in the postprocessing step provides better accuracy because information on the glitch peak voltage is not available until the second edge of the pulse has occurred so that an appropriate downscaling of the corresponding energy values of the two signal transitions cannot really be done in real time. In a conventional power estimation flow, both transitions would count as complete transitions, which lead to inaccuracies.

Besides the mean power results for certain application scenarios, a time-based power analysis can help to estimate power peaks due to high switching activity when they occur. This is needed, e.g., for dimensioning of the supply network, where larger currents may lead to an unwanted drop of the supply voltage. Thus, a fast estimation of a circuit's total current consumption would be desirable in addition to

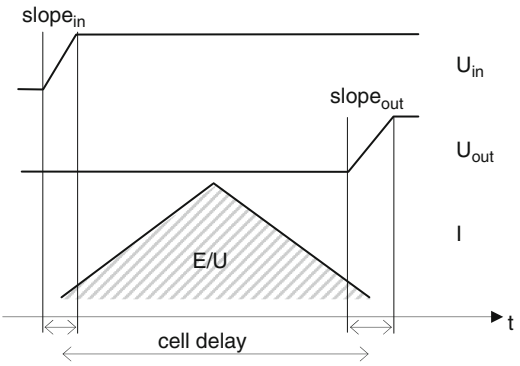


Fig. 4.27 Switching current modeling using a triangular current waveform shape

the time-based power analysis. One possibility to do so is to assume a fixed current waveform during the toggling time of the gate cell. This waveform can then be scaled for each switching event such that its integral matches the quotient of the looked-up energy value and the nominal supply voltage (see Fig. 4.27, (4.83)).

$$\frac{E}{U} = \int_t I \, \mathrm{d}t \qquad U = \text{const.} \tag{4.83}$$

By an additive superposition of the currents of all cell instances in the netlist, an overall current waveform can be calculated that is an estimate for the case of an ideal supply net because the supply voltage was assumed to be constant (Fig. 4.29). The accuracy of the current estimation can be improved by choosing appropriate waveform shapes, by adjusting the points in time when the waveform should begin and end (begin, middle, or end of the slope time) and by a proper consideration of special cases such as glitches and events where only input signals but no output signals change.

Interconnect Wires

Although power is drawn in the gate cells per definition, their behavior is significantly influenced by the wiring between them. On the one hand, the parasitic capacitances of wires increase the effective load capacitance and thus usually increase cell delay and switching energy. On the other hand, wires introduce additional delays and change slope times, which influences the appearance of glitches and the behavior of subsequent gates. Because the presented simulation approach directly calculates delays and power using the cell library data, no back-annotation of delays from a static timing analysis is needed for the cell instances. For the wires, however, it is still required because they are design specific and thus cannot easily be pre-characterized like the elements of a standard cell library. At least the effective load capacitances for each cell output and the nominal static delays of the wire instances have to be calculated, e.g., by a timing analysis tool such as PrimeTime [86]. The results can be written to VHDL packages using cell instance names as identifiers to make them available to the cell model.

```
introduce a new characterization port to the entity of the netlist;
foreach input slope characterization value do
    apply slope value to characterization port;
    foreach wire in the netlist do
        connect an input signal of the driving cell to the characterization port;
        run delay calculation for the wire;
        write resulting input slope of the wire, wire delay and wire output slope to a VHDL
        package which can be used by the cell model to look up values;
    end
end
```

Algorithm 3: High-level wire delay characterization

This is the level of accuracy also used in standard flows. However, using the slope information, it is additionally possible to include the dependency of the wire's delay and output slope time on its input slope time. To determine these dependencies for a whole design within an acceptable time, a high-level characterization can be applied using the delay calculation algorithms of PrimeTime using Algorithm 3.

Further effects such as the nonlinear dependency of the gate input pin capacitance, which contributes to the effective load capacitance of the driving cell, on the slope time cannot really be considered in an event-driven simulation because this would lead to a mutual dependency of slope and load and would require network analysis methods to be solved; here, constant, mean pin capacitances from the cell library have to be used.

Simulation Flow

The general flow for postlayout gate-level simulation is sketched in Fig. 4.28. It requires three main parts of input data: the cell library with additional sensitivities for the process parameters, a wire model with extracted parasitic elements, e.g., in the common SPEF format, which is needed by the STA tool to calculate the effective load capacitances and wire delays, and the corresponding postlayout netlist, which has to be adapted to the new signal data type and the additional generic parameters of the cell models. From these inputs, the VHDL model can then be generated in single pre-processing steps – once for the cell-specific files and once for the circuit-specific data. This leads to a compound model that can be used for a simultaneous simulation of timing and power behavior and also for both, nominal cases without process dependency and for full statistical analysis using the sensitivity data.

4.5.2.3 Application of the Approach

The performance of the described approach depends on several conditions such as the number of considered parameters, the portion of local parameters, the implementation of table access and interpolation, the consideration of glitches, or

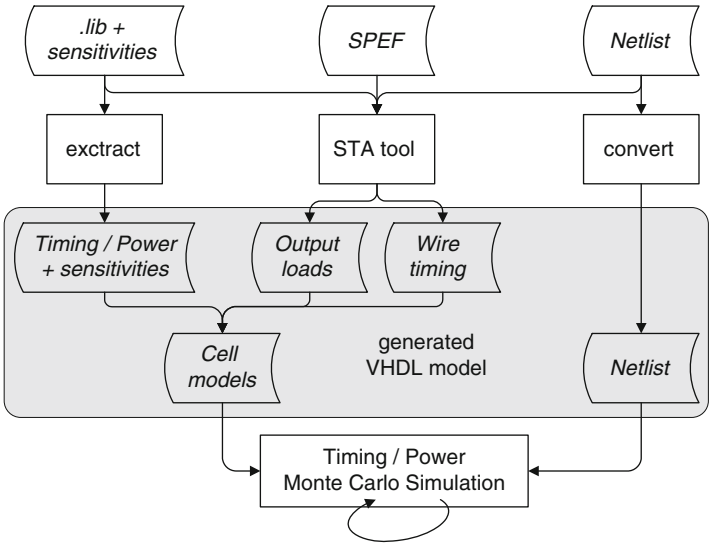


Fig. 4.28 Simulation flow for gate-level digital Monte Carlo analysis

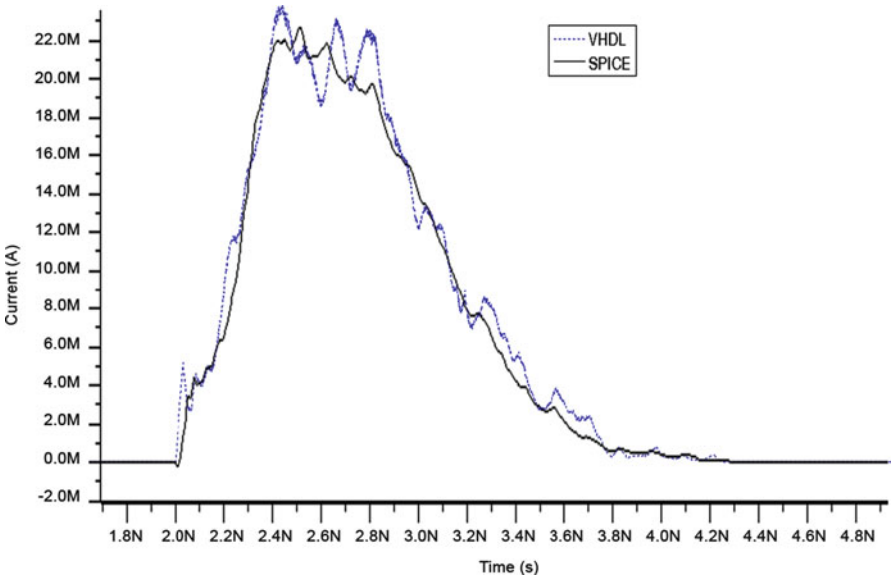


Fig. 4.29 Comparison of VHDL current model with SPICE simulation

the number of oversampling points for the calculation of the current waveforms. In general, however, the performance should stay in the same order of magnitude like with conventional digital gate-level simulation. For an example design with about 1,000 cells and 15 parameters a simulation time increase by a factor of about 5 was

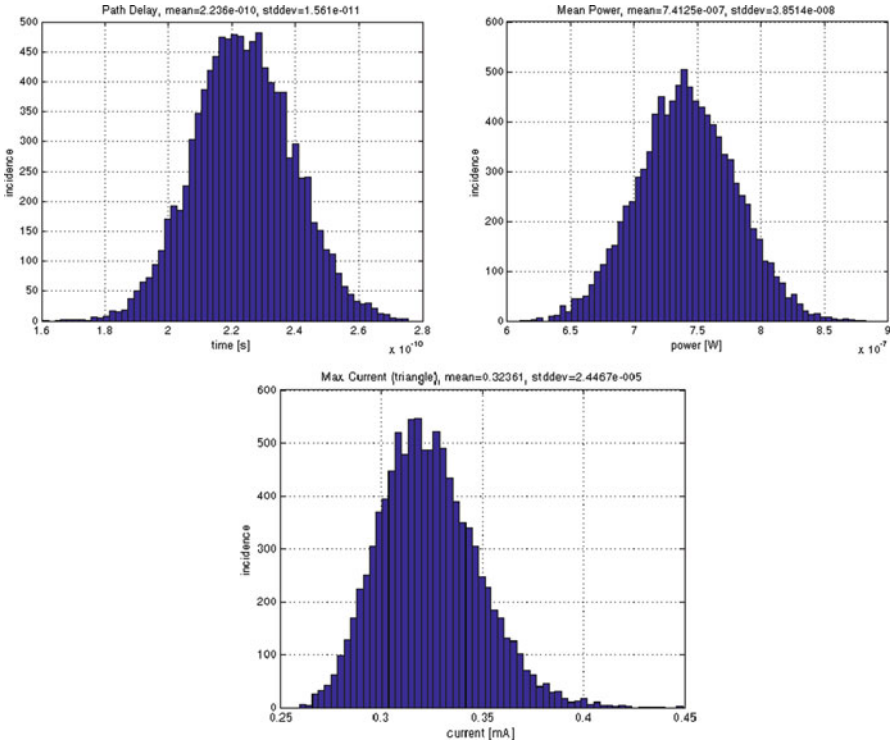


Fig. 4.30 Probability distribution function for path delay, mean power, and maximum current for inverter chain example

found compared to the corresponding standard Verilog model with back-annotated SDF data, but compared with the SPICE reference model it is still faster by a factor of about 12,000 [87]. This should make the described gate-level approach suitable for Monte Carlo analyses of medium-sized full designs or subcircuits of larger ones.

For good accuracy, a seamless integration of the sensitivity analysis regarding the process parameters into the library characterization flow is needed. Especially the used SPICE models have to be parameterized in a way that in case of nominal values for all considered process parameters the resulting cell library contains identical values like in the standard characterization flow for that PVT corner without sensitivity analysis. In the mentioned example, a maximum deviation of 4% for the total number of toggles and of 7% for the mean power could be reached.

Besides the analysis of process variations, the introduced model can be used for a more accurate calculation of delays and slope times, which may increase the accuracy of timing and power analysis and enables a reasonable investigation of glitches.

The current estimation – as an additional feature – should be done during postprocessing for accuracy reasons like it was mentioned above for glitch power estimation too. This needs further time – but has to be done only once for all simulation runs. Figure 4.29 shows a comparison of the calculated total currents for

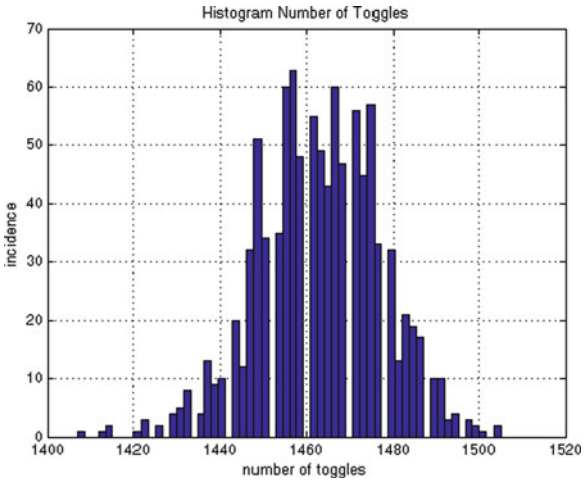


Fig. 4.31 Probability distribution function for the number of toggles for multiplication circuit example

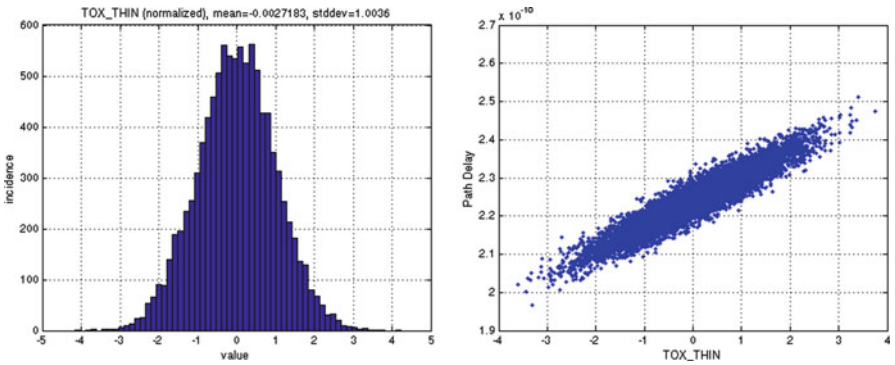


Fig. 4.32 Probability distribution function for global parameter TOX_THIN and dependency of path delay on TOX_THIN for inverter chain example

the VHDL and the SPICE reference model. The VHDL result is based on a nominal simulation with a triangle shape for single transitions as shown in Fig. 4.27. In the simulation scenario, both input vectors of a multiplication unit were inverted at the same time to cause high switching activity.

Several different statistical analyses may be applied to the results of a Monte Carlo simulation depending on the quantities of interest. As a common way for the visualization of results, histogram plots for selected quantities are shown in Fig. 4.30. These are based on 10,000 simulation runs using an inverter chain test design. Figure 4.31 shows the number of toggles during 1,000 runs of the multiplication circuit example and illustrates the influence of the process variations on the number of glitches. Finally, Fig. 4.32 demonstrates the analysis for a single process parameter. In the right plot, the dependency of the circuit-level quantity path

delay on the global parameter TOX_THIN is shown. As expected, it shows a linear dependency because it is always the sum of the same single inverter delays. The remaining noise is caused by the local variations.

4.6 Methods for Analysis and Optimization of Parametric Yield

Based on statistical transistor models, statistical SPICE circuit simulation is explained. Higher-level analysis (Monte Carlo analysis, worst-case analysis, Mismatch analysis and sensitivities) is explained. Optimization of performance and robustness is then explained.

4.6.1 Statistical Analysis

In order to simulate statistical variation and device degradation, model parameters of the transistor model are varied, e.g., the long-channel zero body bias threshold voltage V_{th0}, or mobility μ_0.[2] All such variable model parameters are collected in the parameter vector $\mathbf{s}$.

After production, model parameters are assumed to be Gaussian distributed with a mean vector $\mathbf{s}_0$ and a covariance matrix $\mathbf{C}$.

A specification is a lower bound on a performance, for example slew rate $SR \geq 3\,\mathrm{V\,\mu s^{-1}}$. One manufactured instance of a circuit is considered OK if it fulfills all specifications at all required operating conditions (e.g., temperature range). Including the worst-case operating conditions into the statistical analysis is important for a realistic yield estimate. In the following, we do not include them explicitly in the formalism, but keep in mind that fulfillment of a specification always means that it has to be fulfilled at its respective worst-case operating condition.

If we denote each individual specification with $f_i(\mathbf{s}) \geq b_i$, then the set of process parameters that fulfills a specifications i is

$$A_i = \{\mathbf{s} | f_i(\mathbf{s}) \geq b_i\}, \tag{4.84}$$

with a similar definition for upper bounds. The partial parametric yield Y_i of a circuit regarding one specification i is then simply the probability that its process parameters are inside A_i:

$$Y_i = P\{\mathbf{s} \in A_i\}, \tag{4.85}$$

[2]Note that electrical characteristics such as Isat or transconductance gm are simulation results, but not parameters of a transistor model like BSIM3.

while the total yield is the probability that all specifications are fulfilled:

$$Y = P\{\mathbf{s} \in \bigcap_i A_i\}. \tag{4.86}$$

4.6.1.1 Monte Carlo Analysis

Monte Carlo simulation is an integration method based on random sampling. A sample of size N is taken from the distribution of process parameters, so that there are N vectors $\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \ldots, \mathbf{s}^{(N)}$. For all of them, the performance values are simulated, which results in one vector of simulation results $f(\mathbf{s}^{(1)}), \ldots, f(\mathbf{s}^{(N)})$.

Monte Carlo simulation is used to estimate the parametric yields of a given circuit:

$$\hat{Y}_i = \frac{1}{N}\sum_j \delta_j \qquad \text{with} \qquad \delta_j = \begin{cases} 1 & \text{if } f(\mathbf{s}^{(j)}) \geq b_i \\ 0 & \text{else} \end{cases}. \tag{4.87}$$

Monte Carlo simulation is also used to estimate the mean values and standard deviations of a performance:

$$\bar{f} = \hat{\mu}_f = \frac{1}{N}\sum_j f\left(\mathbf{s}^{(j)}\right) \tag{4.88}$$

$$s_f^2 = \hat{\sigma}_f^2 = \frac{1}{N-1}\sum_j \left(f\left(\mathbf{s}^{(j)}\right) - \bar{f}\right)^2. \tag{4.89}$$

The accuracy and effort (required sample size N) of Monte Carlo simulation is a popular discussion topic and deserves attention. It depends on the observed statistics:

- *Mean value*: Accuracy grows with the square root of the sample size N. Accuracy can be improved by latin hypercube sampling (LHS). But for most specifications, the mean value can be estimated with one simulation at the typical (tt) corner anyway. Hence, the mean is usually among the least interesting results of Monte Carlo.
- *Standard deviation σ or variance σ^2*: Accuracy improves with the square root of sample number N, too. LHS does not significantly improve the accuracy.
- *Partial parametric yield Y*: The variance of yield estimation is $Y \cdot (1-Y)/N$. In the range of up to a yield of 90%, a small sample number such as $N = 50$ is quite sufficient to see that there is a low yield issue. If the yield approaches 100%, usually the failure rate $1-Y$ is considered in ppm, and the (unsymmetrical) 95% confidence interval is calculated. To statistically "prove" to a 95% confidence that the failure rate is less than 1,350 ppm (equivalent to a Gaussian distance of 3 sigma from mean to spec limit), the sample number has to exceed $3/(1-Y) = 3/1{,}350\,\text{ppm} = 2{,}200$. For a 4σ Gaussian distance, it is $3/32\,\text{ppm} = 95{,}000$, and for a 6σ Gaussian distance ($1 - Y = 10^{-9}$), the sample size has to exceed $3 \cdot 10^9$ (Fig. 4.33).

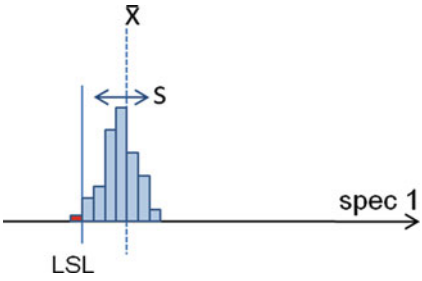


Fig. 4.33 Monte Carlo result histogram. Arithmetic mean and empiric standard deviation are shown

It is these prohibitively high simulation counts for estimating high yields that gave Monte Carlo a bad reputation for being too expensive. But let us consider that this method of estimating the failure rate by counting failed samples is so expensive because it makes no assumptions on the distribution shape. It is a robust method for estimating the failure rate in the presence of extreme outliers, long-tail distributions and multimodal distributions. We will see below that methods that are based on robustness distances (like sigma-to-spec) require a lot fewer simulation runs.

Unfortunately, there is no sampling trick that reduces the required sample size significantly and is still equally robust as standard Monte Carlo for estimating the parametric yield. There are adaptive sampling strategies that can theoretically reduce the required sample number dramatically for certain problems, but unfortunately not far enough for widespread practical applications in parametric yield estimation. Typical sample sizes for adaptive importance sampling lie in the range of several ten thousands, while the methods do not scale down well and tend to become unstable for smaller sample sizes. For many applications of Monte Carlo circuit simulation, a feasible sample size is a factor 10–100 smaller than what adaptive importance sampling needs; hence, it has not found widespread application in circuit simulation, despite its popularity in other fields such as finance or theoretical physics. Particularly for large circuits with long simulation times, even $N = 100$ can be challenging. In this situation, reducing the sample size from 10^8 to $10^{4.5}$ has little practical meaning.

Given these problems, for many practical applications, the estimated distance from mean value to specification bound in multiples of standard deviations is a better robustness measure than the estimated failure rate. In Fig. 4.34, we would consider spec 2 to be more robust than spec 1, although the shapes of the distributions may not be exactly Gaussian and the sample size is only $N = 150$.

If the performance distribution shape were known to be exactly Gaussian, then we could verify high robustness with surprisingly little effort: A simulated distance of 4 s in a sample size of only $N = 45$ is already sufficient to accept a $> 3\sigma$ robustness with 95% confidence.

We can collect the required minimum distance for 95% confidence in Table 4.5.

As an example, Table 4.5 is read as follows: If you want to verify that the distance of the mean from spec of a performance that has a normal distribution is more than 4σ, then run a Monte Carlo simulation with a sample size of $N = 50$. If the distance

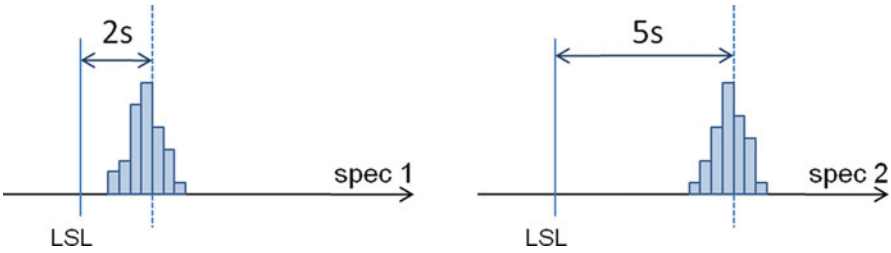


Fig. 4.34 Robustness estimates from Monte Carlo

Table 4.5 Required sigma distance by sample number

	3σ	4σ	5σ
$N = 50$	3.66	4.85	6.04
$N = 100$	3.44	4.57	5.70
$N = 300$	3.24	4.31	5.38
$N = 1{,}000$	3.13	4.16	5.20

of the mean from the spec in the sample is larger than 4.85σ, then it is OK because you may safely assume that it will not drop below 4.0 even if you increased the sample size to infinity. If the distance in your $N = 50$ sample is only a little less than 4.85σ, then increase the sample size for more accuracy. Else, accept that the design is not robust enough and fix it.

For non-Gaussian distributed performances with long tails, the rule is not so simple anymore. One extreme example is the log-normal distribution. A one-sided yield of 99.87% on the long-tail side does not require a 3σ distance like a perfect Gaussian distribution would, but 8.6σ. In a sample of size $N = 300$, the spec-to-mean distance has to be larger than 11.9σ to verify this yield with 95% confidence. These numbers look high, but they are not since reaching 11.9σ is much more likely in a log-normal distribution than in a Gaussian distribution.

For performances that are known to be close to a log-normal distribution, like leakage current or certain timing measurements in logic, it is more robust to check the distribution of their logarithm. Performances such as CMRR of an amplifier should be analyzed not in dB but in the linear signed domain, where they show a much more linear behavior.

4.6.1.2 Worst-Case Analysis

Figure 4.35 shows the mean value, covariance ellipsis and one specification bound in process parameter space. Of all process parameter sets which violate a specification, the point that is closest to the mean value is called the *worst-case point* $\mathbf{s}_{\mathrm{wc}}$. It marks the position in the process parameter space where the probability density of parametric faults has its maximum. The Mahalanobis distance between $\mathbf{s}_{\mathrm{wc}}$ and $\mathbf{s}_0$ is the *worst-case distance* β_{wc}.

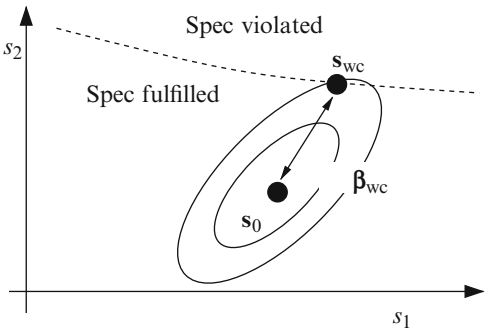


Fig. 4.35 Definition of worst-case point

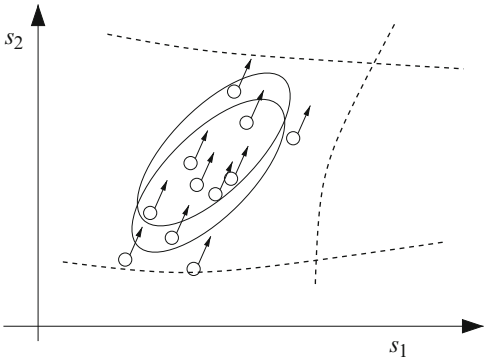


Fig. 4.36 Process drift or device degradation

$$d_{\mathbf{C}}(\mathbf{x},\mathbf{y}) = \sqrt{(\mathbf{x}-\mathbf{y})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x}-\mathbf{y})} \tag{4.90}$$

$$\beta_{\mathrm{wc}} = \pm d_{\mathbf{C}}(\mathbf{s}_{\mathrm{wc}},\mathbf{s}_0). \tag{4.91}$$

There is a sign convention for β_{wc}: if the specification is fulfilled at the nominal point, then β_{wc} is positive, else it is negative. In this way, the yield estimate from β_{wc} is equal to the Gaussian cumulative density function (see (4.94)).

Due to device degradation during operation, the mean value and the covariance matrix of the circuits change with time t: $\mathbf{s}_0(t)$, $\mathbf{C}(t)$. The initial values after production at $t = t_0$ are $\mathbf{s}_0(t_0)$ and $\mathbf{C}(t_0)$. Therefore, the percentage of circuits that still fulfill their specification at time t is

$$Y(\mathbf{p},t) = \int\limits_{A(\mathbf{p})} |2\pi\mathbf{C}(t)|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\mathrm{d}_{\mathbf{C}(t)}(\mathbf{s},\mathbf{s}_0(t))^2\right) \mathrm{d}\mathbf{s}. \tag{4.92}$$

If we consider the influence of process variation on the sensitivity toward stress-induced degradation as a second-order effect, then we may assume $\mathbf{C}(t)$ to be constant. The effect of degradation during operation on the yield is then formally similar to a process drift during manufacturing (see Fig. 4.36).

For small changes in the position of the mean value, the change of worst-case distance over time is

$$\beta_{\mathrm{wc}}(t) = \beta_{\mathrm{wc}}(t_0) - \frac{\left(\mathbf{s}_0(t) - \mathbf{s}_0(t_0)\right)^{\mathrm{T}} \mathbf{C}^{-1} \left(\mathbf{s}_{\mathrm{wc}}(t_0) - \mathbf{s}_0(t_0)\right)}{\beta_{\mathrm{wc}}(t_0)}. \tag{4.93}$$

This change can be positive or negative, i.e., a performance can become better or worse by device degradation.

The worst-case distance can be used to estimate the partial yield for the performance:

$$\hat{Y}(\beta_{\mathrm{wc}}) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\beta_{\mathrm{wc}}} \mathrm{e}^{-\xi^2/2} \mathrm{d}\xi. \tag{4.94}$$

4.6.2 Optimization for Robustness

In order to resist process drift and device degradation, it is not sufficient to optimize only the yield figure $Y(\mathbf{p},0)$, because this value goes into saturation at 100%. Standard methods for the estimation of Y, which means counting Monte Carlo samples, are not accurate enough to estimate the worst-case distance. A robust and a nonrobust design may show the same yield value $Y(\mathbf{p},0)$, but different worst-case distances, which means different sensitivities toward process drift or device degradation. Optimization for yield and robustness, therefore, has to focus on the worst-case distances as the primary targets for optimization of robustness and yield [88].

As a result, this advantage of worst-case distance optimization in contrast to optimization of Y becomes even more important for the design of robust and reliable analog circuits. The combination of worst-case distance optimization and SOAs is the basis of our approach. The SOAs can be formalized as functions of the design parameters, which impose further constraints on the optimization problem:

$$\mathbf{c}(\mathbf{p}) \geq 0. \tag{4.95}$$

During the worst-case distance optimization, design points are accepted as valid, only if they fulfill all such constraints. The solution has to show high worst-case distances for each performance f_i, while satisfying all constraints $\mathbf{c} \geq 0$.

The optimization consists of three main steps (cf. Fig. 4.37). First, the operating points of basic structures, e.g., differential pairs, are optimized using a constraint matrix concept. Then the circuit performance is improved regarding operating range, such as supply voltage and temperature. Finally, design centering is carried out to maximize the yield.

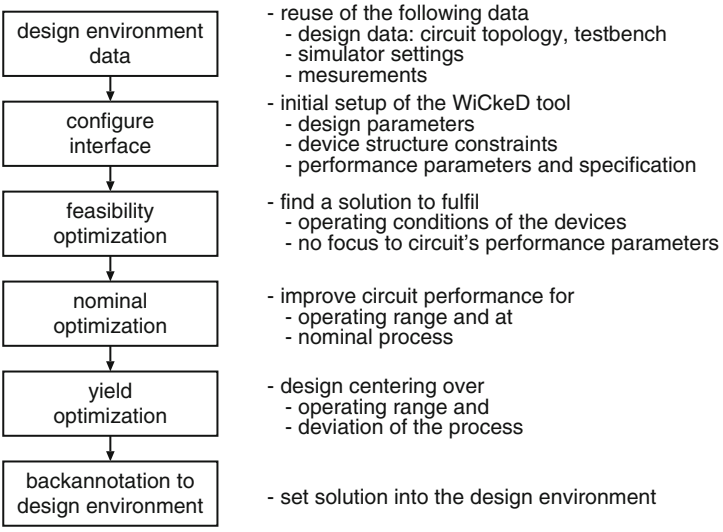


Fig. 4.37 WiCkeD's main optimizations steps

4.6.2.1 Structural Constraints

Analog circuits are composed of basic building blocks such as current mirrors or differential pairs. Unlike digital gates, the analog ones depend on their geometries and operating point to operate correctly. Usually, being in saturation is an important constraint on many analog transistors, as well as current symmetries or certain nodes being at the same potential. Since these constraints are related neither to the specification, nor to the layout level like design rules are, but come from the structure of the circuit, they are called "structural constraints." We distinguish four types of constraints:

1. Geometric equality, like "equal lengths $l_1 = l_2$ in a current mirror"
2. Geometric inequality, like "$w_1 l_1 > 6L_{\text{min}}^2$"
3. Electrical equality, like "$I_1 = I_2$" in a current mirror
4. Electrical inequality, like "$V_{\text{gs}} - V_{\text{th}} > 50\,\text{mV}$" (strong inversion).

The geometric constraints can be guaranteed by construction. To check the electrical inequality constraints, a simulation has to be done that shows by which amount c_k each constraint k is over-fulfilled ($c_k > 0$) or is violated ($c_k < 0$).

Like design rules on layout level, structural constraints on schematic level do not at all guarantee that the circuit fulfills the specification, but a violation indicates a structural problem that may result in a low yield but remains undetected when simulating a rather high-level circuit specification.

For a typical analog circuit consisting of 100 transistors, more than 400 constraints may be created. Since many structural constraints can be derived from

requirements on basic structures such as current mirrors, generation of many of these constraints can be performed automatically [89].

4.6.2.2 Feasibility Optimization (FO)

Structural constraints are useful to automatically find a good initial sizing and to ensure that tools for automatic nominal optimization and design centering provide technically feasible results [90]. For that purpose, FO modifies the vector $\mathbf{d} = (d_1, \ldots, d_{n_d})$ of design parameters (such as transistor geometries and resistor values) so that all constraints are fulfilled, i.e., $\mathbf{c}(\mathbf{d}) \geq \mathbf{0}$. Usually, a reasonable initial sizing $\mathbf{d}_{\text{init}}$ is available and a solution close to it is preferred:

$$\begin{aligned} &\min_{\mathbf{d}} \|\mathbf{d} - \mathbf{d}_{\text{init}}\| \\ &\mathbf{c}(\mathbf{d}) \geq \mathbf{0}\,. \end{aligned} \tag{4.96}$$

The number of independent design parameters n_d grows with the number of elements to be sized and is reduced by geometric equality constraints. For typical analog, circuits n_d can be expected to be between 15 and 30, while complex designs, e.g., the OTA presented in [91], can have up to 100 degrees of freedom.

4.6.2.3 Nominal Optimization (NO)

Analog circuits are characterized by performance measures, for example, gain A_0, slew rate SR, and noise figure NF. The specification requires the values of these measures not to exceed certain upper and/or lower bounds, for example $A_0 \geq 80\,\text{dB}$.

We denote the performance measures by the vector $\mathbf{f} = (f_1, \ldots, f_{n_f})$, with the vectors of lower bounds $\mathbf{f}^L$ and upper bounds $\mathbf{f}^U$. The performance measures depend on design parameters: $\mathbf{f}(\mathbf{d})$, and the specification is

$$\mathbf{c}(\mathbf{d}) \geq \mathbf{0} \quad \wedge \quad \mathbf{f}^L \leq \mathbf{f}(\mathbf{d}) \leq \mathbf{f}^U\,. \tag{4.97}$$

The goal of nominal optimization is finding values for $\mathbf{d}$ that satisfy (4.97).

Moreover, this must be achieved for a defined range of operating parameters such as temperature or Vdd. We denote the operating parameters by the vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{n_\theta})$ with lower and upper bounds $\boldsymbol{\theta}^L$ and $\boldsymbol{\theta}^U$. Then $\mathbf{f}$ depends also on the operating conditions: $\mathbf{f}(\mathbf{d}, \boldsymbol{\theta})$, and the specification is

$$\mathbf{c}(\mathbf{d}) \geq \mathbf{0} \quad \wedge \quad \bigvee_{\boldsymbol{\theta}^L \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}^U} \mathbf{f}^L \leq \mathbf{f}(\mathbf{d}, \boldsymbol{\theta}) \leq \mathbf{f}^U. \tag{4.98}$$

The goal of nominal optimization with operating conditions is finding values for $\mathbf{d}$ that satisfy (4.98).

Two types of algorithms are available for nominal optimization in WiCkeD: gradient-based optimization with parameter distances [92] and stochastic (global) optimization. The nature of most analog sizing problems is optimization of performance functions that show strong trade-offs, are expensive to evaluate in terms of simulation time, but are monotonous or convex – not in the full design space but in the small feasible design space that is restricted by the large number of structural inequality constraints. Gradient-based methods can be adapted to this type of problem very efficiently. Therefore, it is reasonable to run these methods first, and to resort to stochastic optimizers only when simulation results and design knowledge indicate that multiple local optima actually exist.

4.6.2.4 Design Centering

Process variation and mismatch have a large influence on the performance measures of analog circuits. For simulation, this effect is modeled by varying randomly a few standard Gaussian distributed model parameters, for example tox or Vth. The vector of random model parameters is denoted by $\mathbf{s} = (s_1, \ldots, s_{n_s})$ with the null vector $\mathbf{0}$ as mean and unity covariance matrix. Process variation and mismatch are both contained in $\mathbf{s}$, so for a typical analog circuit consisting of 100 transistors, n_s can be expected to be between 200 and 250.

One standard method for estimating the distributions of performance measures is Monte Carlo simulation. A sample of size N of $\mathbf{s}$ is generated and simulated, yielding N result vectors $\mathbf{f}^{(i)} = \mathbf{f}(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s}^{(i)})$, $i = 1 \ldots N$. The parametric yield Y is estimated as the percentage of samples that lie within the specification bounds $(\mathbf{f}^L, \mathbf{f}^U)$. Monte Carlo is only an analysis method, but does not vary $\mathbf{d}$ and hence shows little information on how to improve the yield by changing $\mathbf{d}$.

Yield improvement can be accomplished by worst-case distance methods. A design that satisfies (4.98), i.e., that fulfills the specification for the typical process and no mismatch ($\mathbf{s} = \mathbf{0}$) and for all required operating conditions, could still violate the specification for some $\mathbf{s} \neq \mathbf{0}$. If process conditions $\mathbf{s}$ causing violations are close to the mean value (i.e., $f_i(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s}) < f_i^L$ for some $\boldsymbol{\theta}$ and small $\|\mathbf{s}\|$), then there will be severe parametric yield loss. Therefore, an important measure for a performance f_i is the worst-case distance $\beta_{\mathrm{wc}}^{(i)}$, which is the shortest distance between the mean value and a process condition that causes $f_i(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s})$ to fail its specification. For a lower bound f_i^L of a spec that satisfies (4.98),

$$\begin{aligned} \beta_{\mathrm{wc}}^{(i)} &= \min_{\boldsymbol{\theta}, \mathbf{s}} |\beta| \\ f_i\left(\mathbf{d}, \boldsymbol{\theta}, \beta \frac{\mathbf{s}}{\|\mathbf{s}\|}\right) &= f_i^L \\ \boldsymbol{\theta}^L \leq \boldsymbol{\theta} &\leq \boldsymbol{\theta}^U. \end{aligned} \tag{4.99}$$

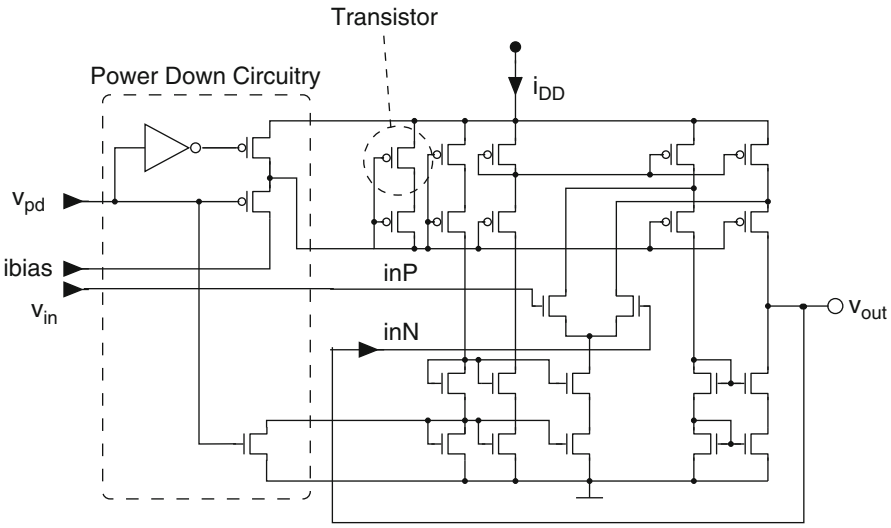


Fig. 4.38 Schematic of an OpAmp

The worst-case distance for an upper bound is similarly defined.

A worst-case distance is a function of the design parameters. They are useful goals to maximize over **d** and thereby achieve a design that is centered in the process space regarding the specification bounds [88, 93].

4.6.2.5 Example

In this section, some of the concepts discussed above are shown on an example circuit, see Fig. 4.38. This kind of operational amplifiers are basic building blocks of many analog and mixed-signal circuits. For certain applications, the performance of the operational amplifier is critical for the performance of the whole system.

Table 4.6 shows the Monte Carlo simulation results for the circuit specifications after initial nominal circuit design. As can be seen from the row "Yield estimate $\hat{Y}_i$," many specs already have a large yield, but transit frequency F_t and phase margin ϕ_M still need improvement.

Table 4.7 compares yield estimation results from Monte Carlo simulation (see Sect. 4.6.1.1) and worst-case analysis (see Sect. 4.6.1.2). The worst-case distance for specification ϕ_M is negative, which corresponds to a yield estimate $<50\%$.

Table 4.8 shows one of the worst-case points in terms of the threshold voltages of the devices. The most dominant parameter is the threshold voltage of transistor MMN1, but MMP3 and MMP1 influence the transit frequency significantly, too.

Table 4.6 Example circuit results

Performance	Voff	A_0	F_t	ϕ_M	CMRR	Slew	Power
Spec	± 0.002	>70 dB	$>40 \cdot 10^6\,\mathrm{s}^{-1}$	>60°C	>90 dB	$>35\,\mathrm{MV\,s}^{-1}$	<0.003 W
Mean	0.00002	79.032	$41.061 \cdot 10^6$	59.434	109.10	39.646	0.001891
Standard deviation	0.0012	0.464	$0.6483 \cdot 10^6$	0.701	9.925	1.032	0.00008
Yield estimate $\hat{Y}_i$	0.91575	1	0.934066	0.2051282	1	1	1
Confidence interval for Q_i	[0.87627, 0.94584]	[0.98909, 1]	[0.89780, 0.96046]	[0.15883, 0.25795]	[0.98909, 1]	[0.98909, 1]	[0.98909, 1]

Table 4.7 Comparison of estimates of the partial yield from Monte Carlo analysis and worst-case analysis

Spec	$\hat{Y}_i$ from Monte Carlo (%)	95% confidence interval from Monte Carlo	Worst-case distance	$\hat{Y}_i$ from worst-case analysis (%)
A0	100	[0.9891,1]	>7.556	100
Ft	93.41	[0.8978,0.9605]	1.463	92.83
phm	20.51	[0.1588,0.2580]	−0.677	24.912
CMRR	100	[0.9891,1]	4.315	100
Slew	100	[0.9891,1]	4.179	100
Power	100	[0.9891,1]	>9.380	100

Table 4.8 The worst-case point of the performance Ft

Device	w_i	Device	w_i
MMN1	−1.227	MMP9	−0.008
MMP3	−0.496	MMN2	−0.005
MMP1	0.481	MMN12	0.002
MMN5	0.266	MMN10	−0.002
MMN3	0.182	MMN6	0.001
MMN11	0.140	MMP8	−0.001
MMN9	−0.138	MMP6	0.001
MMP5	0.100	MMN4	0.001
MMP2	0.048	MMN8	0.001
MMP4	−0.048	MMP10	−0.001
MMP7	−0.016	MMN7	0.001

4.7 Robustness Analysis for Semiconductor Blocks

Fluctuations in the manufacturing process introduce steady variations in the electrical properties of semiconductor devices and interconnects. As a result, the performance parameters of integrated circuits are subject to variation as well and hence they can be treated as random variables following certain probability distributions. By quantifying them robustness analysis figures out critical blocks or circuits in terms of variability.

It refers to two problems. On the one hand, the performance parameter distributions need to be quantified to measure variability and to compare performance characteristics in different physical domains. Summarizing these partial results leads to an overall performance robustness of the circuit under investigation. On the other hand, the major contributors to variability need to be known. This enables identifying critical process parameters and estimating their impact which is of particular importance in case of parameter drifts.

This section proposes solutions to both issues by presenting methods and figures of merit to rate performance parameter distributions and to account for performance parameter shifts.

4.7.1 Motivation and Goals

Fluctuations in manufacturing cause variations in device and interconnect parameters and may lead to circuit performance degradation or malfunctions. To account for this issue, two important aspects for circuit analysis are discussed in the following. While we focus on transistor-level circuits, the principles may also be applied to higher levels of abstraction assuming proper statistical modeling.

To counteract degradation, it is essential to know the major contributors to circuit performance variability. Therefore, two methods to determine significantly influencing variables are presented, whereas we focus on transistor circuits and outline the pros and cons.

Furthermore, we need to rate circuit performance variability. Techniques to estimate the parametric yield have already been proposed in Sect. 4.6. But since this figure of merit also has some disadvantages, we discuss some alternatives whereas we keep track of single performance characteristics and overall circuit performance.

In summary, robustness analysis is intended to contribute to answering the following questions.

- How can we determine the contributors to circuit performance variability?
- What are the feasible figures of merit to rate performance parameter fluctuations? Is there a way to combine the characteristics to rate overall circuit robustness?

Targeting these problems some methodologies are presented considering the example performance parameter

$$y_1 = x_1 + x_2^2. \tag{4.100}$$

The random parameters X_1 and X_2 are assumed to be statistically independent standard Gaussian variables, $X_i \sim N(0,1)$ so that the nominal point is $\mathbf{x}_0 = \mathbf{0}$.

4.7.2 Contributors to Variability and Their Impact

There are several methods to determine influential parameters x_i with respect to an arbitrary performance characteristic y_j. They assume that the variables x_i are uncorrelated. Although this assumption may not necessarily be true, it can be achieved by applying principal component analysis, see Sect. 2.2.

4.7.2.1 Sensitivity Analysis

Linear sensitivities

$$S_{i,j}^{(1)} = \left.\frac{\partial y_j}{\partial x_i}\right|_{\mathbf{x}_0}, \tag{4.101}$$

Table 4.9 Example for full-factorial experiments. The nominal value is shown for completeness

x_1	x_2	y_1		i	$\bar{y_1}(x_i = 1)$	$\bar{y_1}(x_i = -1)$	Effect (4.102)
0	0	0	$\Longrightarrow$	1	2	0	1
-1	-1	0		2	1	1	0
-1	1	0					
1	-1	2					
1	1	2					

the first derivatives of a performance parameter y_j with respect to the variables x_i at the nominal point $\mathbf{x}_0$, measure the first-order impact of the variables. Large absolute values outline significant contributors to variability, whereas small quantities indicate less influential variables.

Linear sensitivity data may be directly computed by the circuit simulator. Applying the one-factor-at-a-time approach [94] and the approximation using finite differences,

$$S_{i,j}^{(1)} \approx \left.\frac{\Delta y_j}{\Delta x_i}\right|_{\mathbf{x}_0} = \frac{y_j(\mathbf{x}_0, x_i = x_{0,i} + h) - y_j(\mathbf{x}_0, x_i = x_{0,i} - h)}{2h} \tag{4.102}$$

with the deviation parameter h, is an alternative. For the example performance parameter in (4.100), we determine $S_1^{(1)} = 1$ and $S_2^{(1)} = 0$ using an arbitrary parameter $h > 0$.

Design of Experiments (DoE) considering n variables with k levels each is a more effective alternative to the one-factor-at-a-time approach [94]. Multiple variables are deflected concurrently to decrease the simulation effort and to additionally enable analyzing cross-correlations. In comparison with the one-factor-at-a-time analyses, the efficiency increases with a rising number of variables. Full factorial experiments and fractional factorial experiments are distinguished. While the former consider all cross-correlations, the latter ignore higher-order interactions but decrease simulation costs. The full-factorial analysis of the example performance parameter in (4.100) is summarized in Table 4.9. Both variables take the values ± 1.

Beneath statistical analyses and response surface modeling, effect analysis is a possibility for evaluating the experiment results. The main effect of a variable x_i is the change in a performance parameter y_j when changing the variable between its levels a_k. In analogy to (4.102), it is defined by the difference in the corresponding mean performance parameter values $\bar{y_j}(x_i = a_k)$ and the variable deflection Δa_k. Two variables x_i and x_j interact if the effect of one variable depends on the level of the other one. In our example, the DoE results and the one-factor-at-a-time method are identical. Additionally, the values in Table 4.9 indicate that the variables x_1 and x_2 do not interact.

If a sample of data is created, for instance during a Monte Carlo analysis, it can be used to fit response surface models that we have introduced in Sect. 2.2.

In the vicinity of the nominal point $\mathbf{x}_0$, the first-order Taylor approximation of the performance parameter y_j contains the linear sensitivities defined in (4.101).

In our example, the quadratic influence of the variable x_2 cannot be detected due to the linear approach. This drawback may be overcome by increasing the model complexity, that is by increasing the order of the sensitivity by generalizing (4.101),

$$S_{i,j}^{(n)} = \left. \frac{\partial^n y_j}{\partial x_i^n} \right|_{\mathbf{x}_0} . \tag{4.103}$$

While the simulation effort rises with the model order, we gain accuracy. Finding a trade-off between reliable results and runtime is a major task when performing sensitivity analyses, which is directly influenced by the circuit topology.

An application of higher-order sensitivity data is proposed in [95] to calculate the probability P of critical performance parameter values. Simulation data is fitted to a second-order polynomial response surface model that is the input to an analytical evaluation. Principal component analysis of the coefficient matrix of the second-order variable impacts and a subsequent saddle point approximation enables estimating the cumulative distribution functions of the performance characteristics y_j. Numerical methods additionally estimate the sensitivity of the probability P to the variables x_i.

Assuming a properly chosen model complexity, sensitivity analysis provides reasonable results. The contributors to variability are outlined qualitatively and quantitatively, which is often applied in circuit optimization [96].

4.7.2.2 The Chi-Square Test as a Statistical Approach

Statistical tests may be an alternative for sensitivity analysis to determine significant contributors to circuit performance variability. Their input is a sample of the size N, which may be created by Monte Carlo simulations that are alternative sampling approaches.

In particular, the chi-square test [97] may be of interest. It is a test for statistical independence that checks the null hypothesis, the random variables X and Y are statistically independent, against its alternative, X and Y are statistically dependent. To find contributors to circuit performance fluctuations, the random variable X represents an arbitrary circuit variable x_i, whereas the performance parameter under investigation y_j is described by Y.

Usually, device parameters and performance characteristics are continuous random variables. Using partitions of equal width or probability, both random variables have to be classified into n_x and n_y categories to build up an n_x x n_y contingency table.

The test statistic χ^2 measures the deviation of the data in the contingency table from its expected distribution in case X and Y are statistically independent. Under this assumption, the test statistic follows a chi-square distribution with $(n_x - 1)(n_y - 1)$ degrees of freedom. If it exceeds a threshold value that is given

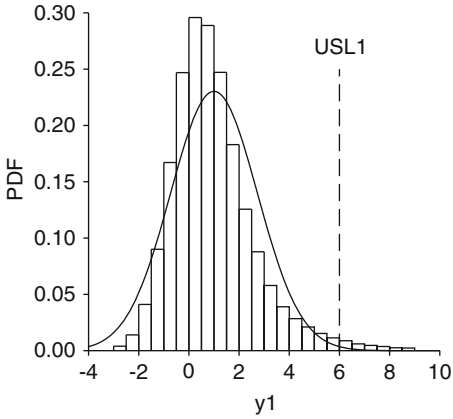


Fig. 4.39 Example performance parameter distribution, histogram plot, and Gaussian approximation

by the desired significance level, the null hypothesis will have to be rejected and a statistical dependence has to be assumed.

The chi-square test has to be performed sequentially for all combinations of variables x_i and performance parameters y_j to qualitatively separate significant and less influential variables. To avoid misinterpreting, the variables x_i have to be uncorrelated. In contrast to sensitivity analysis, a quantitative ranking is not possible.

4.7.3 Performance Parameter and Circuit Robustness

Let us consider our performance parameter y_1 in (4.100) for which the upper specification limit $USL_1 = 6$ defines the tolerance region. It may be represented by the random variable Y_1 with its mean value $E[Y_1] = 1$ and variance $\mathrm{Var}Y_1 = \sigma_1^2 = 1$. The histogram plot in Fig. 4.39 indicates that a Gaussian approximation does not fit the data. In the following sections, figures of merit to rate the performance parameter distributions will be presented.

4.7.3.1 Parametric Yield and Worst-Case Distances

In Sect. 4.6, the parametric yield Y has been defined as the percentage of samples within the specification limits. While being very intuitive, it is very costly to estimate high values of this figure of merit with sufficient accuracy. Furthermore, the tails of performance parameter distributions are of particular importance. But they may be inaccurate due to the lack of measurement data when modeling extreme device behavior. As an alternative, the worst-case distance β_{wc} has been introduced.

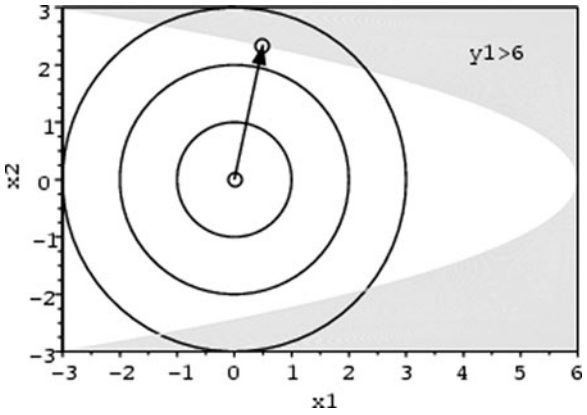


Fig. 4.40 Worst-case distance example

In our example, we have $Y = 0.983$. That is, 98.3% of all circuits will meet the performance requirements for y_1. The worst-case point $(0.5, \sqrt{5.5})$ results in the worst-case distance $\beta_{\text{wc}} = 5.75$ as it is illustrated in Fig. 4.40.

4.7.3.2 Process Capability Indices

As we have seen in Sect. 4.6, it is hard to estimate high parametric yield values with sufficient accuracy.

Beneath worst-case distances β_{wc}, process capability indices may be an alternative [98, 99]. Since they do not exhibit this disadvantage of parametric yield estimation, they have been widely used in process monitoring and statistical quality control. Additionally, they have been successfully applied in circuit optimization [100].

The two most important indices are C_p and C_{pk}. They are defined by lower and upper specification limits (LSL_j and USL_j), mean values $E[Y_j]$, and standard deviations σ_j,

$$C_p(y_j) = \frac{USL_j - LSL_j}{6 \cdot \sigma_j} \tag{4.104}$$

$$C_{pk}(y_j) = \frac{\min\left\{USB_j - E[Y_j], E[Y_j] - LSB_j\right\}}{3 \cdot \sigma_j}, \tag{4.105}$$

in case of Gaussian-distributed performance parameters. While index C_p solely relates the width of the tolerance region to the distribution spread, the measure C_{pk} additionally considers the distribution location with respect to the nearer specification limit. The definitions show that high values indicate robust performance characteristics. Due to the assumed Gaussian distribution, process capability indices

Table 4.10 Transformation of process capability indices and yield for Gaussian performance characteristics

C_p	Maximum yield	C_pk	Minimum yield
2/3	0.9544997	2/3	0.9544997
1	0.9973002	1	0.9973002
4/3	0.9999367	4/3	0.9999367
5/3	0.9999994	5/3	0.9999994

provide an estimate for the parametric yield,

$$2\phi(3\cdot C_{pk})-1 \leq Y \leq 2\phi(3\cdot C_p)-1, \tag{4.106}$$

whereas $\phi(.)$ is the cumulative distribution function (CDF) of the standard normal distribution. Providing some characteristic values for process capability indices and corresponding parametric yield, Table 4.10 shows an increased discriminatory power of the process capability indices compared to yield in the high-yield regime. Furthermore, since only mean values and standard deviations need to be determined, the indices are easy to use.

Since our performance characteristic y_1 is not bounded below, we have to use an infinite lower specification limit, $LSL_1 = -\infty$, in the calculations. Applying (4.104)–(4.106), we obtain

$$\begin{aligned} C_p(y_1) &= \infty\,, \\ C_{pk}(y_1) &= \frac{5}{9}\sqrt{3} \approx 0.962 \qquad \text{and} \\ 0.996 &\leq Y \leq 1. \end{aligned}$$

Using the process capability indices, we overestimate the parametric yield in comparison with Sect. 4.7.3.1. The reason is that the Gaussian approximation is not accurate as it neglects the upper distribution tail that we can see in Fig. 4.39.

To counteract, a more general definition for the process capability indices has been standardized [101],

$$C_p(y_j) = \frac{USB_j - LSB_j}{Q_j(0.99865) - Q_j(0.00135)} \tag{4.107}$$

$$C_{pk}(y_j) = \min\left\{\frac{USB_j - Q_j(0.5)}{Q_j(0.99865) - Q_j(0.5)}, \frac{Q_j(0.5) - LSB_j}{Q_j(0.5) - Q_j(0.00135)}\right\}, \tag{4.108}$$

where $Q_j(.)$ is the quantile function. Since these definitions can be applied to arbitrary performance parameter distributions, a transformation into parametric yield estimates is difficult or even impossible. Additionally, determining the 0.00135– and 0.99685–quantiles may be very costly in terms of the required sample size N and hence in terms of analysis runtime. The values

$$Q_1(0.5) = 0.738\,, \quad Q_1(0.99865) = 10.520 \qquad \Longrightarrow \qquad C_{pk}(y_1) = 0.538$$

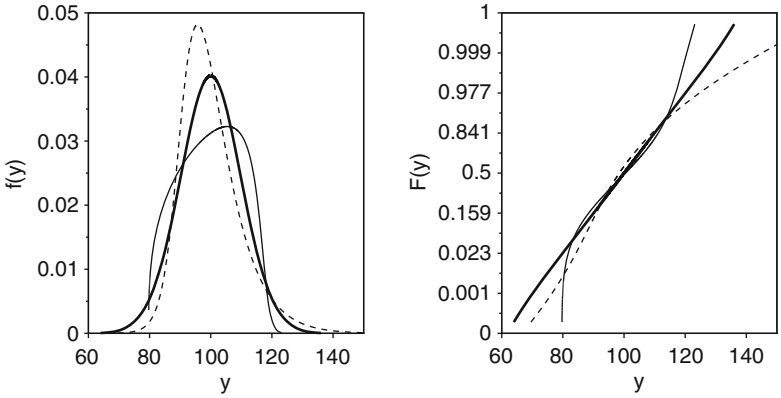


Fig. 4.41 Although their mean values $E[Y_j] = 100$, standard deviations $\sigma_j = 10$ and hence the coefficients of variation $CV_j = 0.1$ are identical, performance parameter distributions may differ due to their shape

show a 44% lower process capability index $C_{pk}(y_1)$ than the one determined by the approximation in (4.105). This underlines that using the Gaussian approximation is inaccurate in our example and overestimates the parametric yield.

In circuit analysis, mean values and standard deviations or quantiles have to be estimated applying an arbitrary sampling technique, see (2.35) and (2.36) for Monte Carlo analyses. Since the characteristics can only be estimated, the process capability indexes are estimators as well. Methods to define confidence intervals account for this limitation.

Since multiple performance parameters usually contribute to circuit performance, process capability indices for single performance characteristics need to be combined. Multivariate extensions have been developed for this purpose [98].

4.7.3.3 Coefficient of Variation

The coefficient of variation,

$$CV_j = \frac{\sigma_j}{E[Y_j]}, \tag{4.109}$$

also known as the relative variation of the performance parameter y_j, is a unit-free measure. High values imply a large spread in the distribution, which may be interpreted as a low robustness. The coefficient of variation can only be applied to distributions with positive mean values.

The value $CV_1 = \sqrt{3} = 1.732$ expresses that the standard deviation σ_1 is about 73% larger than the mean $E[Y_1]$, which appears quite high. But since the coefficient of variation does not consider specification boundaries and a reference performance parameter for comparison is not available, an interpretation is difficult. Nevertheless, constructing confidence intervals will increase the significance of the analysis results.

As the process capability indexes for Gaussian distributions in (4.104) and (4.105), the definition in (4.109) does not pay attention to the shape of the distribution so that distributions may differ, although their coefficients of variation are identical, see Fig. 4.41. To rate multiple performance parameters y_j, multivariate coefficients of variation have been proposed [102].

4.7.3.4 Taguchi

In manufacturing, Taguchi methods aim at improving quality [94]. The loss function,

$$L = k \cdot \left(y_j^{(i)} - T \right)^2, \tag{4.110}$$

with the constant k rates a manufactured item $y_j^{(i)}$ according to its deviation from its target value T. In general, minimizing the loss function in (4.110) achieves a quality improvement. It does not consider specification limits because it focuses on-target operation.

In circuit analysis, the item may be an arbitrary performance characteristic y_j and the target may be the nominal value or mean $E[Y_j]$. Estimating the sample variance σ_j for Monte Carlo simulations using (2.36) corresponds to determining the mean loss function with respect to the sample mean.

To rate performance parameter distributions, signal-to-noise ratios have been proposed. The used definition has to be chosen according to the scope of optimization [94], and it has to be aimed at high values.

$$S/N_T(y_j) = 10 \cdot \log \left(\frac{E[Y_j]^2}{\sigma_j^2} \right) \qquad \text{minimize variation} \tag{4.111}$$

$$S/N_L(y_j) = -10 \cdot \log \left(\frac{1}{N} \sum \frac{1}{(y_j^{(i)})^2} \right) \qquad \text{maximize } y_j \tag{4.112}$$

$$S/N_S(y_j) = -10 \cdot \log \left(\frac{1}{N} \sum (y_j^{(i)})^2 \right) \qquad \text{minimize } y_j. \tag{4.113}$$

Robustness analysis requires (4.111) because it considers the distribution spread and location. But as the coefficient of variation in Sect. 4.7.3.3, the signal-to-noise ratio $S/N_T(y_j)$ does not pay attention to the shape of the performance parameter distribution.

We can determine the value

$$S/N_T(y_1) = -4.77$$

for our example performance parameter y_1 in (4.100). The measure is negative because the sample standard deviation is larger than the mean value. As the coefficient of variation, the signal-to-noise ratio indicates a large spread in the distribution of performance parameter y_1.

Table 4.11 Summary of measures for robustness analysis

Name	Symbol	Definition (equation)	Specification required	Shape of distribution
Parametric yield	Y		Yes	Considered
Worst-case distance	β_{wc}		Yes	Considered
Process capability	C_p / C_{pk}	(4.104) and (4.105)	Yes	Not considered
Indices		(4.107) and (4.108)	Yes	Considered
Coefficient of variation	CV	(4.109)	No	Not considered
Signal-to noise ratio	S/N_T	(4.111)	No	Not considered

4.7.3.5 Summary

We have discussed several figures of merit for robustness analysis which are summarized in Table 4.11. When choosing one of the measures, the availability of specification boundaries, the importance of the distribution shapes and the interpretability have to be considered. We have to take into account that the presented figures of merit can only be estimated during the practical application. Determining confidence intervals will be necessary to increase the significance of the analysis results.

The parametric yield appears most intuitive since it is directly linked with the manufacturing process outcome. Due to the potential lack of reference values, the alternative figures of merit mainly fit circuit comparison purposes. While high process capability indices and signal-to-noise ratios according to Taguchi indicate robust circuits or blocks, the coefficient of variation should be minimized.

Currently, process capability indices are quite uncommon in integrated circuit design and analysis. By increasing the discriminatory power in the high-yield regime, they are an alternative for the parametric yield. A transformation is possible when the type of the performance parameter distribution is known.

Specification boundaries do not necessarily have to be defined. Then, the coefficient of variation and Taguchi's signal-to-noise ratio may be used to abstract performance parameter variability. As a disadvantage, both figures of merit do not take into account the distribution shape. We will present an alternative approach in the subsequent section and apply it to a sample set of standard cells.

4.7.4 Standard Cell Robustness Analysis

In Sect. 4.7.3.3, we have presented the coefficient of variation as a potential figure of merit for robustness analysis. Since it does not take into account the distribution shape, further information needs to be considered.

We propose to additionally use the interquartile range,

$$IQR_j = Q_j(0.75) - Q_j(0.25), \tag{4.114}$$

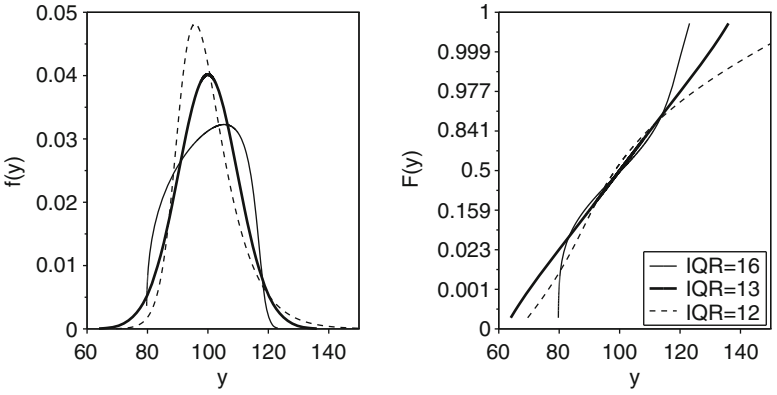


Fig. 4.42 The quotient IQR_j/σ_j indicates heavy-tailed distributions

that is known from the Cumulative distribution function (CDF). If the quotient IQR_j/σ_j (=1.349 for Gaussian distributions) is small, the distribution of the performance parameter y_j is heavy-tailed [97]. Figure 4.42, referring to Fig. 4.41, illustrates this fact.

Since the increased probability of extreme values, heavy-tailed distributions are considered less robust. In combination with the coefficient of variation, we define the performance parameter robustness [103],

$$R_j = \frac{IQR_j/\sigma_j}{CV_j} = \frac{IQR_j \cdot E[Y_j]}{\sigma_j^2}. \tag{4.115}$$

To account for multiple performance parameters y_j, we additionally have to combine their robustness measures. All performance parameters contribute to cell variability so that a cell cannot be more robust than any performance characteristics. Hence, we define the cell robustness

$$\frac{1}{R_{\text{cell}}} = \sum_j \frac{1}{R_j}. \tag{4.116}$$

Equation (4.116) may be adapted to different processes targets, for instance high performance or low power, by inserting weights according to the importance of performance parameters.

In Sect. 2.2, we have introduced that the sample mean and the sample standard deviation only approximate the true distribution parameters. This also holds for the inter-quartile range so that the performance parameter robustness R_j in (4.115) and the cell robustness R_{cell} in (4.116) can only be estimated as well. Since it is hard to analytically derive the distributions and confidence intervals of our robustness, we apply a bootstrapping approach [104].

Table 4.12 Standard cells for robustness analysis

Cell	Name
2x buffer	bufx2
1x inverter	invx1
2x inverter	invx2

Table 4.13 Performance parameters for standard cell robustness analysis

j	Performance parameter	Identifier	j	Performance parameter	Identifier
1	Cell delay	Delay	4	Leakage power	Leakage
2	Output transition	Transition	5	Noise immunity	Noise
3	Active power	Power	6	Setup/hold time	Setup/hold

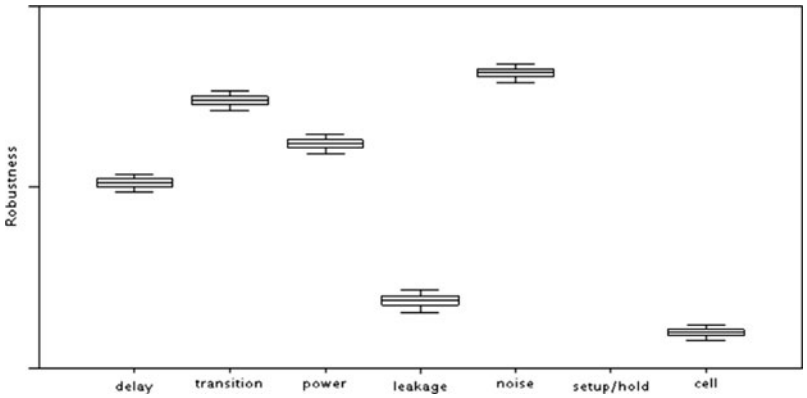


Fig. 4.43 Estimated performance parameter robustness for 2x buffer. Note the logarithmic scale

For every performance characteristic y_j, we determine the distributions of the estimators for mean, standard deviation, and inter-quartile range applying a Monte Carlo analysis, see Sect. 2.2 and [97]. Assuming that these distributions are independent, we draw sets of estimated means, standard deviations and inter-quartile ranges and calculate a sample of performance parameter robustness measures R_j using (4.115). After repeating the procedure for all performance characteristics, (4.116) combines the performance parameter robustness values to cell robustness estimators. As a result, we obtain the distributions of the performance parameter and cell robustness estimators. While we transform them into boxplots for visualization here, we are also able to read off confidence intervals to numerically evaluate our analysis results.

While we apply the approach in (4.115) and (4.116) to the standard cells in Table 4.12, we focus on the performance parameters in Table 4.13.

With a sample size $N = 1{,}000$, we obtain the results in Figs. 4.43 and 4.44. Since setup and hold constraints are not of concern in combinational logic, this performance parameter has an infinite robustness, $R_{\text{setup/hold}} = \infty$.

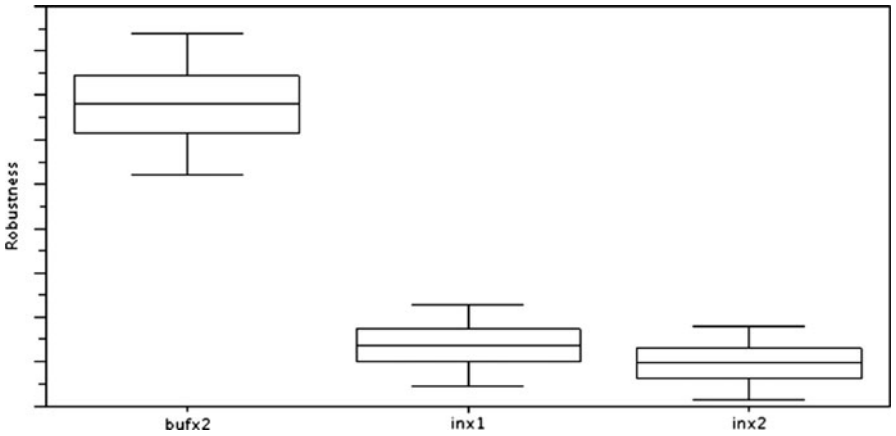


Fig. 4.44 Cell robustness estimation

Figure 4.43 shows the boxplots for the estimated robustness values of the 2x buffer, which are representative for all examined cells. Leakage power is the most critical performance parameter, followed by cell delay. Fluctuations of the output transition time and noise immunity can be neglected because they contribute only marginally to the cell variability.

The comparison of cell robustness in Fig. 4.44 shows that the 2x buffer is most robust. The two inverters cannot be classified because the confidence intervals of the robustness estimators overlap. To reduce variability, the inverters should primarily be optimized.

4.7.5 Conclusion

Robustness analysis aims at determining the contributors to circuit performance variability and to abstract performance fluctuations by feasible figures of merit.

In the first part, we have outlined methods to determine significant variables, which effect variability. Sensitivity analysis has been discussed as a well-known and widely used approach. By choosing model orders and a simulation methodologies, users may tune the analysis efficiency and accuracy. As a result, influential variables are outlined and their impact can directly be read off. Although they are not frequently used in circuit analysis, statistical tests may be an alternative approach. The chi-square test has been briefly introduced. While significant contributors to variability are outlined, the impact of variables cannot be quantitatively determined.

In addition to the parametric yield, which has been presented in Sect. 4.6, figures of merit to abstract performance parameter and cell variability have been proposed. They have to be chosen depending on the availability of specification boundaries and the importance of the distribution shapes. An enhancement to the coefficient of variation and its application to standard cell comparison have been shown.

Nevertheless, some enhancements to robustness analyses will be required. While reducing the analysis effort will be the major task, ensuring sufficient accuracy and providing robustness characteristics suitable to analyses on higher levels of abstraction are also topics for further investigations.

References

1. Scheffer, L., Lavagno, L., Martin, G. (eds.): EDA for IC implementation, circuit design, and process technology. CRC Press (2006)
2. Kayssi, A.I., Sakallah, K.A., Mudge, T.N.: The impact of signal transition time on path delay computation. IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing **40**(5), 302–309 (1993)
3. Nardi, A., Neviani, A., Zanoni, E.: Impact of unrealistic worst case modeling on the performance of VLSI circuits in deep submicron CMOS technologies. IEEE Transactions on Semiconductor Manufacturing (SM) **12**(4), 396–402 (1999)
4. Dasdan, A., Hom, I.: Handling inverted temperature dependence in static timing analysis. ACM Transactions on Design Automation of Electronic Systems **11**(2), 306–324 (2006)
5. Weber, M.: My head hurts, my timing stinks, and i don't love on-chip variation. In: SNUG (2002)
6. Incentia: Advanced on-chip-variation timing analysis. Tech. rep., Incentia Design Systems Inc. (2007)
7. Qian, J., Pullela, S., Pillage, L.: Modeling the "effective capacitance" for the RC interconnect of CMOS gates. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **13**(12), 1526–1535 (1994)
8. Celik, M., Pileggi, L., Odabasioglu, A.: IC Interconnect Analysis. Kluwer Academic Publishers (2004)
9. ECSM - effective current source model. http://www.cadence.com/Alliances/languages/Pages/ecsm.aspx (2007)
10. Composite current source. http://www.synopsys.com/products/solutions/galaxy/ccs/cc_source.html (2006)
11. Sakurai, T., Newton, A.R.: Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas. IEEE Journal of Solid-State Circuits SC **25**(2), 584–594 (1990)
12. Hirata, A., Onodera, H., Tamaru, K.: Proposal of a timing model for CMOS logic gates driving a CRC *pi* load. In: ICCAD'98: Proceedings of the IEEE/ACM International Conference on Computer-aided Design, pp. 537–544. ACM, New York, NY, USA (1998)
13. Knoth, C., Kleeberger, V.B., Nordholz, P., Schlichtmann, U.: Fast and waveform independent characterization of current source models. In: IEEE/VIUF International Workshop on Behavioral Modeling and Simulation (BMAS), pp. 90–95 (2009)
14. Croix, J., Wong, M.: Blade and razor: cell and interconnect delay analysis using current-based models. In: ACM/IEEE Design Automation Conference (DAC), pp. 386–389 (2003)
15. Feldmann, P., Abbaspour, S., Sinha, D., Schaeffer, G., Banerji, R., Gupta, H.: Driver waveform computation for timing analysis with multiple voltage threshold driver models. In: ACM/IEEE Design Automation Conference (DAC), pp. 425–428 (2008)
16. Liu, B., Kahng, A.B.: Statistical gate level simulation via voltage controlled current source models. In: IEEE International Behavioral Modeling and Simulation Workshop (2006)
17. Wang, X., Kasnavi, A., Levy, H.: An efficient method for fast delay and SI calculation using current source models. In: IEEE International Symposium on Quality Electronic Design, pp. 57–61. IEEE Computer Society, Washington, DC, USA (2008)

18. Kashyap, C., Amin, C., Menezes, N., Chiprout, E.: A nonlinear cell macromodel for digital applications. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 678–685 (2007)
19. Li, P., Feng, Z., Acar, E.: Characterizing multistage nonlinear drivers and variability for accurate timing and noise analysis. IEEE Transactions on VLSI Systems **15**(11), 1205–1214 (2007)
20. Goel, A., Vrudhula, S.: Current source based standard cell model for accurate signal integrity and timing analysis. In: Design, Automation and Test in Europe (DATE), pp. 574–579 (2008)
21. Fatemi, H., Nazarian, S., Pedram, M.: Statistical logic cell delay analysis using a current-based model. In: ACM/IEEE Design Automation Conference (DAC), pp. 253–256 (2006)
22. Fatemi, H., Nazarian, S., Pedram, M.: Statistical logic cell delay analysis using a current-based model. In: ACM/IEEE Design Automation Conference (DAC), pp. 253–256 (2006)
23. Mitev, A., Ganesan, D., Shanmugasundaram, D., Cao, Y., Wang, J.M.: A robust finite-point based gate model considering process variations. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 692–697 (2007)
24. Amin, C., Kashyap, C., Menezes, N., Killpack, K., Chiprout, E.: A multi-port current source model for multiple-input switching effects in CMOS library cells. In: ACM/IEEE Design Automation Conference (DAC), pp. 247–252 (2006)
25. Dabas, S., Dong, N., Roychowdhury, J.: Automated extraction of accurate delay/timing macromodels of digital gates and latches using trajectory piecewise methods. In: Asia and South Pacific Design Automation Conference, pp. 361–366 (2007)
26. Knoth, C., Kleeberger, V.B., Schmidt, M., Li, B., Schlichtmann, U.: Transfer system models of logic gates for waveform-based timing analysis. In: International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM^2ACD), pp. 247–252 (2008)
27. Goel, A., Vrudhula, S.: Statistical waveform and current source based standard cell models for accurate timing analysis. In: ACM/IEEE Design Automation Conference (DAC), pp. 227–230 (2008)
28. Zolotov, V., Xiong, J., Abbaspour, S., Hathaway, D.J., Visweswariah, C.: Compact modeling of variational waveforms. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 705–712. IEEE Press, Piscataway, NJ, USA (2007)
29. Yeo, K.S., Roy, K.: Low Voltage, Low Power VLSI Subsystems. McGraw-Hill Professional (2004)
30. Narendra, S., Blaauw, D., Devgan, A., Najm, F.: Leakage issues in IC design: Trends, estimation and avoidance. In: Proc. ICCAD 2003, Tutorial (2003)
31. Rao, R.R., Devgan, A., Blaauw, D., Sylvester, D.: Parametric yield estimation considering leakage variability. In: Proc. DAC 2004, pp. 442–447 (2004)
32. Agarwal, A., Kang, K., Roy, K.: Accurate estimation and modeling of total chip leakage considering inter- & intra-die process variations. In: ICCAD, pp. 736–741 (2005)
33. Li, T., Zhang, W., Yu, Z.: Full-chip leakage analysis in nano-scale technologies: Mechanisms, variation sources, and verification. In: Proc. DAC 2008, pp. 594–599 (2008)
34. Li, X., Le, J., Pileggi, L.T., Strojwas, A.: Projection-based performance modeling for inter/intra-die variations. In: Proc. ICCAD 2005, pp. 721–727 (2005)
35. Li, X., Le, J., Pileggi, L.T.: Projection-based statistical analysis of full-chip leakage power with non-log-normal distributions. In: Proc. DAC 2006, pp. 103–108 (2006)
36. Zhuo, F., Li, P.: Performance-oriented statistical parameter reduction of parameterized systems via reduced rank regression. In: Proc. ICCAD 2006, pp. 868–875 (2006)
37. Mitev, A., Marefat, M., Ma, D., Wang, J.M.: Principle hessian direction based parameter reduction with process variation. In: Proc. ICCAD 2007, pp. 632–637 (2007)
38. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer (2001)
39. Bellman, R.E.: Adaptive Control Processes. Princeton University Press (1961)
40. Geman, S., Bienenstock, E., Doursat, R.: Neural networks and the bias/variance dilemma. Neural Computation **4**(1), 1–58 (1992)

41. Sugiyama, M., Rubens, N.: Active learning with model selection in linear regression. In: The SIAM International Conference on Data Mining, pp. 518–529 (2008)
42. Maimon, O., Rokach, L.: Data Mining and Knowledge Discovery Handbook. Springer (2005)
43. Geman, S., Bienenstock, E., Doursat, R.: Neural networks and the bias/variance dilemma. Neural Computation **4**(1), 1–58 (1992)
44. Berk, R.A.: Statistical Learning from a Regression Perspective. Springer (2008)
45. Cadence: Clock Domain Crossing (White Paper) (2004)
46. Shenoy, N., Brayton, R., Sangiovanni-Vincentelli, A.: Minimum padding to satisfy short path constraints. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 156–161 (1993)
47. Zhang, L., Tsai, J., Chen, W., Hu, Y., Chen, C.C.P.: Convergence-provable statistical timing analysis with level-sensitive latches and feedback loops. In: IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 941–946 (2006)
48. Kamikawai, R., Yamada, M., Chiba, T., Furumaya, K., Tsuchiya, Y.: A critical path delay check system. In: ACM/IEEE Design Automation Conference (DAC), pp. 118–123 (1981)
49. Pilling, D.J., Sun, H.B.: Computer-aided prediction of delays in LSI logic systems. In: ACM/IEEE Design Automation Conference (DAC), pp. 182–186 (1973)
50. Sasaki, T., Yamada, A., Aoyama, T., Hasegawa, K., Kato, S., Sato, S.: Hierarchical design verification for large digital systems. In: ACM/IEEE Design Automation Conference (DAC), pp. 105–112 (1981)
51. Hitchcock, R.B.: Timing verification and the timing analysis program. In: ACM/IEEE Design Automation Conference (DAC), pp. 594–604 (1982)
52. Hitchcock, R.B., Smith, G.L., Cheng, D.D.: Timing analysis of computer hardware. IBM Journal Research Development **26**(1), 100–105 (1982)
53. Maheshwari, N., Sapatnekar, S.S.: Timing Analysis and Optimization of Sequential Circuits. Kluwer Academic Publishers (1999)
54. Feldman, D., Fox, M.: Probability, The Mathematics of Uncertainty. Marcel Dekker, Inc (1991)
55. Visweswariah, C., Ravindran, K., Kalafala, K., Walker, S., Narayan, S.: First-order incremental block-based statistical timing analysis. In: ACM/IEEE Design Automation Conference (DAC), pp. 331–336 (2004)
56. Feldman, D., Fox, M.: Probability, The Mathematics of Uncertainty. Marcel Dekker, Inc (1991)
57. Clark, C.E.: The greatest of a finite set of random variables. Operations Research **9**(2), 145–162 (1961)
58. Chang, H., Sapatnekar, S.S.: Statistical timing analysis considering spatial correlations using a single PERT-like traversal. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 621–625 (2003)
59. Agarwal, A., Zolotov, V., Blaauw, D.T.: Statistical timing analysis using bounds and selective enumeration. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **22**(9), 1243–1260 (2003)
60. Devgan, A., Kashyap, C.: Block-based static timing analysis with uncertainty. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 607–614 (2003)
61. Zhang, L., Hu, Y., Chen, C.P.: Block based statistical timing analysis with extended canonical timing model. In: IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 250–253 (2005)
62. Zhan, Y., Strojwas, A.J., Li, X., Pileggi, L.T., Newmark, D., Sharma, M.: Correlation-aware statistical timing analysis with non-Gaussian delay distributions. In: ACM/IEEE Design Automation Conference (DAC), pp. 77–82 (2005)
63. Feng, Z., Li, P., Zhan, Y.: Fast second-order statistical static timing analysis using parameter dimension reduction. In: ACM/IEEE Design Automation Conference (DAC), pp. 244–249 (2007)
64. Zhang, L., Chen, W., Hu, Y., Gubner, J.A., Chen, C.C.P.: Correlation-preserved non-Gaussian statistical timing analysis with quadratic timing model. In: ACM/IEEE Design Automation Conference (DAC), pp. 83–88 (2005)

65. Clark, C.E.: The greatest of a finite set of random variables. Operations Research **9**(2), 145–162 (1961)
66. Singh, J., Sapatnekar, S.: Statistical timing analysis with correlated non-Gaussian parameters using independent component analysis. In: ACM/IEEE Design Automation Conference (DAC), pp. 155–160 (2006)
67. Singh, J., Sapatnekar, S.S.: A scalable statistical static timing analyzer incorporating correlated non-Gaussian and Gaussian parameter variations. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **27**(1), 160–173 (2008)
68. Li, X., Le, J., Gopalakrishnan, P., Pileggi, L.T.: Asymptotic probability extraction for non-normal distributions of circuit performance. In: IEEE/ACM International Conference on Computer-Aided Design (ICCAD) (2004)
69. Chang, H., Zolotov, V., Narayan, S., Visweswariah, C.: Parameterized block-based statistical timing analysis with non-Gaussian parameters, nonlinear delay functions. In: ACM/IEEE Design Automation Conference (DAC), pp. 71–76 (2005)
70. Agarwal, A., Blaauw, D., Zolotov, V., Vrudhula, S.: Statistical timing analysis using bounds. In: Design, Automation and Test in Europe (DATE), pp. 62–67 (2003)
71. Agarwal, A., Blaauw, D., Zolotov, V., Vrudhula, S.: Computation and refinement of statistical bounds on circuit delay. In: ACM/IEEE Design Automation Conference (DAC), pp. 348–353 (2003)
72. Agarwal, A., Blaauw, D., Zolotov, V., Sundareswaran, S., Zhao, M., Gala, K., Panda, R.: Path-based statistical timing analysis considering inter- and intra-die correlations. In: ACM/IEEE International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems (TAU), pp. 16–21 (2002)
73. Orshansky, M., Bandyopadhyay, A.: Fast statistical timing analysis handling arbitrary delay correlations. In: ACM/IEEE Design Automation Conference (DAC), pp. 337–342 (2004)
74. Li, X., Le, J., Celik, M., Pileggi, L.T.: Defining statistical timing sensitivity for logic circuits with large-scale process and environmental variations. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **27**(6), 1041–1054 (2008)
75. Mukhopadhyay, S., Raychowdhury, A., Roy, K.: Accurate estimation of total leakage current in scaled CMOS logic circuits based on compact current modeling. In: DAC '03: Proceedings of the 40th annual Design Automation Conference, pp. 169–174. ACM, New York, NY, USA (2003). DOI 10.1145/775832.775877
76. Kao, J., Narendra, S., Chandrakasan, A.: Subthreshold leakage modeling and reduction techniques. In: Proc. ICCAD, 2002, pp. 141–148 (2002)
77. Acar, E., Devgan, A., Nassif, S.R.: Leakage and leakage sensitivity computation for combinational circuits. Journal of Low Power Europe **1**(2), 1–10 (2005)
78. Jiang, W., Tiwari, V., de la Iglesia, E., Sinha, A.: Topological analysis for leakage prediction of digital circuits. In: Design Automation Conference, 2002. Proceedings of ASP-DAC 2002. 7th Asia and South Pacific and the 15th International Conference on VLSI Design. Proceedings., pp. 39–44 (2002)
79. Rao, R.R., Devgan, A., Blaauw, D., Sylvester, D.: Parametric yield estimation considering leakage variability. In: Proc. DAC 2004, pp. 442–447 (2004)
80. Favalli, M., Benini, L.: Analysis of glitch power dissipation in CMOS ICs. In: ISLPED '95: Proceedings of the 1995 international symposium on Low power design, pp. 123–128. ACM, New York, NY, USA (1995). DOI 10.1145/224081.224103
81. Najm, F., Burch, R., Yang, P., Hajj, I.: CREST - a current estimator for CMOS circuits. In: IEEE International Conference on Computer-Aided Design, pp. 204–207 (1988)
82. Wang, L., Olbrich, M., Barke, E., Büchner, T., Bühler, M.: Fast dynamic power estimation considering glitch filtering. In: SOCC, pp. 361–364 (2009)
83. Häußler, R., Kinzelbach, H.: Sensitivity-based stochastic analysis method for power variations. In: ITG Fachbericht 196, 9. ITG/GMM-Fachtagung, ANALOG'06, Entwicklung von Analogschaltungen mit CAE-Methoden, pp. 125–130. Dresden (2006)

84. SAE International, Electronic Design Automation Standards Committee: J2748 VHDL-AMS Statistical Packages (2006)
85. Christen, E., Bedrosian, D., Haase, J.: Statistical modeling with VHDL-AMS. In: Forum on Specification and Design Languages, FDL'07, pp. 44–49. ECSI, Barcelona (2007)
86. Synopsys, Inc.: PrimeTime User Guide, B-2008.12 edn. (2008)
87. Dietrich, M., Eichler, U., Haase, J.: Digital statistical analysis using VHDL. In: Design, Automation & Test in EuropeDATE'10, pp. 1007–1010. Dresden, Germany (2010). URL http://www.date-conference.com/proceedings/PAPERS/2010/DATE10/PDFFILES/08.1_3.PDF
88. Schenkel, F., Pronath, M., Zizala, S., Schwencker, R., Graeb, H., Antreich, K.: Mismatch analysis and direct yield optimization by spec-wise linearization and feasibility-guided search. In: DAC, pp. 858–863 (2001)
89. Graeb, H., Zizala, S., Eckmueller, J., Antreich, K.: The sizing rules method for analog integrated circuit design. In: ICCAD, pp. 343–349 (2001)
90. Stehr, G., Pronath, M., Schenkel, F., Graeb, H., Antreich, K.: Initial sizing of analog integrated circuits by centering within topology-given implicit specifications. In: ICCAD (2003)
91. Rooch, K.H., Sobe, U., Pronath, M.: Circuit design-for-yield (DFY) for a 110dB Op-Amp for automotive and sensor applications. In: ITG Fachbericht 196, 9. ITG/GMM-Fachtagung, ANALOG'06, Entwicklung von Analogschaltungen mit CAE-Methoden, pp. 119–123. Dresden (2006)
92. Schwencker, R., Schenkel, F., Graeb, H., Antreich, K.: The generalized boundary curve – A common method for automatic nominal design and design centering of analog circuits. In: DATE, pp. 42–47 (2000)
93. Antreich, K.J., Graeb, H.E.: Circuit optimization driven by worst-case distances. In: The Best of ICCAD – 20 Years of Excellence in Computer-Aided Design, pp. 585–595. Kluwer Academic Publishers (2003)
94. Montgomery, D.: Design and Analysis of Experiments, 3rd edn. John Wiley & Sons, New York (1991)
95. Sohrmann, C., Muche, L., Haase, J.: Accurate approximation to the probability of critical performance. In: 2. GMM/GI/ITG-Fachtagung Zuverlässigkeit und Entwurf, pp. 93–97 (2008)
96. Pehl, M., Graeb, H.: RaGAzi: A random and gradient-based approach to analog sizing for mixed discrete and continuous parameters. In: Integrated Circuits, ISIC '09. Proceedings of the 2009 12th International Symposium on, pp. 113–116 (2009)
97. Hartung, J.: Statistik: Hand- und Lehrbuch der angewandten Statistik, 12 edn. R. Oldenbourg (1999)
98. Kotz, S., Johnson, N.: Process capability indices. Chapman & Hall, London (1993)
99. Rinne, H., Mittag, H.J.: Prozessfähigkeitsmessung für die industrielle Praxis. Hanser (1999)
100. Aftab, S., Styblinski, M.: IC variability minimization using a new C_p and C_{pk} based variability/performance measure. In: ISCAS '94., 1994 IEEE International Symposium on Circuits and Systems, vol. 1, pp. 149–152 (1994)
101. ISO 21747: Statistical methods – Process performance and capability statistics for measured quality characteristics (2006)
102. Bennett, B.: On multivariate coefficients of variation. Statistical Papers **18**(2), 123–128 (1977). Springer, Berlin
103. Lange, A., Muche, L., Haase, J., Mau, H.: Robustness characterization of standard cell libraries. In: DASS 2010 – Dresdner Arbeitstagung Schaltungs- und Systementwurf, pp. 13–18 (2010)
104. Marques de Sá, J.: Applied Statistics Using SPSS, STATISTICA, MATLAB and R, 2nd edn. Springer (2007)
105. International Technology Roadmap for Semiconductors: System drivers (2007)
106. Borel, J. (ed.): European Design Automation Roadmap. Parais (2009)

Chapter 5
Consequences for Circuit Design and Case Studies

Alyssa C. Bonnoit and Reimund Wittmann

The previous chapters introduced various methods to analyze the influence of variations of the manufacturing process on the performance of devices and circuits. These methods can be applied to evaluate designs for manufacturability. Variations imply negative effects in most cases that shall be reduced. However, there exist also applications where the variations bring an advantage into the design process. The consequences of both aspects regarding special design requirements will be figured out in this chapter.

Section 5.1 presents circuit techniques for the reduction of parameter variations. The section provides an overview on techniques to tolerate variability and to reduce variablity. One of these techniques is body biasing. Body biasing does not only influence leakage and timing by modifying the threshold voltage of a transistor. There is also a potential of this technique to control variability in foregoing technologies. This first section discusses properties of various body biasing schemes.

While body biasing is explored in the first section of this chapter by simple chip designs, Sect. 5.2 demonstrates the implications of this technique in high-performance microprocessors. The processors use dynamic voltage/frequency scaling. The effectiveness of five body biasing schemes at ages of zero and five years will be compared. The simulation infrastructure and workload examples for these tests will be described.

Section 5.3 demonstrates that variations of the manufacturing process can also improve the accuracy of a design. The basic idea will be demonstrated realizing a required value of a resistance by structures of small resistors. In this case, random

A.C. Bonnoit (✉)
IBM Corporation, 2070 Route 52, Hopewell Junction, NY 12533, USA
e-mail: abonnoit@us.ibm.com

R. Wittmann
IPGEN Microelectronics GmbH, Kortumstr. 36, 44787 Bochum, Germany
e-mail: reimund.wittmann@ipgenme.de

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6_5,

values of the small resistors are uncorrelated. Thereby, the variance of the random resistance of the structure of small resistors is much smaller than the variance of the resistance of one appropriate complex resistor. This approach can be applied to design high-precise digital-to-analog converters (DACs). The corresponding theory will be explained as well as some details of the structures involved. The presented approach offers a wide spectrum of applications for the design of high-precise integrated resistances and capacitances.

The first two sections of this chapter were prepared by Alyssa C. Bonnoit, Reimund Wittmann wrote Sect. 5.3.

5.1 Circuit Techniques for Reduction of Parameter Variations

As technology scales, circuit designers are confronted with increasing body biasing. We present an overview of a number of techniques for both dealing with and reducing this increasing variability. In particular, adaptive body biasing has been demonstrated to be effective at addressing variability. We examine the sensitivity to body biases in highly scaled technologies, and look at implementations of body biasing in a wide range of chip applications.

5.1.1 Background

Process variability increases significantly as transistor geometry is scaled because precise control of electrical parameters becomes increasingly difficult. Process variations can be further categorized into inter-die (between dies) or intra-die (within a die). Inter-die variability includes die-to-die, wafer-to-wafer, and lot-to-lot variations. For example, a source of die-to-die variations is fluctuation in resist thickness from wafer to wafer [1]. As a result of inter-die variability, some dies do not fall into any acceptable bin due to either high power or low frequency [2]. Intra-die variability affects transistors on the same die differently, and is further divided into those that are random and those that are systematic. Random variations are caused predominantly by random dopant fluctuations (RDFs) and line-edge roughness (LER). Random variations increase static power (since the leakage is dominated by the leakiest transistors), but have little impact on the frequency due to averaging over transistors along a path. As random variability increases, circuits that rely on matching between devices, such as SRAMs and analog differential pairs, require larger margins to ensure proper operation. Systematic variations include layout effects, for example, transistors in a region with a higher average pitch may print different from transistors in another region of the die, as well as process effects, such as a gradient across the exposure area of a reticle field. Systematic variability increases the amount of frequency margin required to ensure that the chip can operate, since the critical path may fall in a slow region. This, in turn, increases static power because the majority of the transistors on the chip are faster and leakier than required to meet the operating frequency.

Process variability is further compounded at run-time by thermal variability, because the temperature of the die fluctuates with the workload being run. Frequency of a path decreases as temperature increases because the mobility of the transistors decreases [3]. Static power increases exponentially with temperature, but to first order, dynamic power is independent of temperature. The speed binning and power limits must therefore be assessed at the highest (i.e., worst-case) temperature.

Finally, long-range temporal variability occurs because transistor performance degrades over time due to mechanisms such as negative-bias temperature instability (NBTI) and hot-carrier injection (HCI). Similarly, long-range temporal variability increases the frequency margin required. In turn, this increases the amount of static power wasted at low ages (since the frequency can still be met when the threshold voltage increases with age).

Process engineers are pursuing a significant number of approaches to reduce variability. Lithography uses resolution-enhancement techniques to ensure that the printed shapes look like the design shapes even when the wavelength of light is equal to (or smaller than) the wavelength of light used [4]. Amorphous Silicon can be used for gates in order to reduce line-edge roughness [5]. While these process techniques are necessary and reduce variability, they will not eliminate variability. As a result, designers are confronted with increasing variability. The designers can either tolerate variability by designing circuits with increased margin or flexibility, or reduce variability post-Silicon manufacturing by modulating the transistor characteristics.

5.1.2 Overview of Techniques to Tolerate Variability

Several design techniques have been proposed to reduce the sensitivity to variability. Liang and Brooks proposed variable-latency register file in which the slowest 20% of read accesses are done in two clock cycles instead of one, which allows the frequency margin to be reduced [6]. This results in a mean frequency improvement of 12%. They argue that the variations affecting logic and SRAM are different, and thus extend this work to include a time-borrowing floating point unit that translates slack in the SRAM read accesses to margin in the timing of the floating point unit.

Tiwari et al. proposed *ReCycle*, a flexible pipeline clocking scheme that uses time-borrowing between stages to increase clock frequency by reducing the margin required [7]. They also suggested inserting dummy "donor" pipeline stages to provide additional timing slack.

5.1.3 Overview of Techniques to Reduce Variability

Circuit designers can reduce variability using technology-aware layout with regular fabrics and logic bricks [8]. Standard library elements (inverter, NAND, NOR) are

grouped into a set of logic primitives, such as a buffer chain or a flip-flop. The set of logic primitives is significantly smaller than a standard cell library. Each of the logic primitives is translated to a logic brick layout, which consists only of lithography-friendly patterns. Because there are fewer logic bricks than standard cell library elements, extensive lithography simulations of each brick can be performed. With a larger number of bricks, the circuit designer has more flexibility, but the setup overhead increases. The regular fabric methodology reduces variability due to lithography because there are a small and finite number of possible patterns, all of which are designed to be lithography-friendly.

5.1.4 Body Biasing

Body biasing has been proposed as a means of addressing variations [9]. After fabrication, the threshold voltage (V_{TH}) of transistors can be modulated by changing the body-to-source voltage. In bulk MOSFETs, the V_{TH} is given by:

$$V_{TH} = V_{TH_0} + \gamma\left(\sqrt{|2\Phi_F - V_{BS}|} - \sqrt{|2\Phi_F|}\right), \tag{5.1}$$

where V_{TH0} is the device threshold voltage with no body bias applied, $2\Phi_F$ is the surface potential at strong inversion, and γ is the body effect coefficient [10]. For simplicity, we examine this equation for the case of an NFET with the source tied to ground. If a negative voltage is applied to the body, then the depletion width increases, which means that a higher gate voltage is required to form an inversion layer and thus the V_{TH} increases; this is known as a reverse body bias (RBB). Similarly, if a positive voltage is applied to the body while the source is grounded, then the depletion width decreases, and thus the V_{TH} decreases; this is known as a forward body bias (FBB). Throughout this work, V_{BSn} and V_{BSp} will represent the body to source voltage of NFETs and PFETs, respectively. Negative values of these parameters will indicate RBB and a positive one FBB, regardless of which direction the body-to-source voltage must actually be shifted.

There are several technology issues with body biasing in bulk MOSFETs. RBB increases short-channel effects, which increases variability within devices sharing a bias. This is especially problematic in circuits that are sensitive to device matching, such as SRAMs. FBB not only improves short-channel effects, but also increases junction leakage, potentially to the point where the source-to-bulk junction is forward biased. Additionally, an analog signal, the body bias, must be distributed a significant distance – in the extreme, across the entire die. This becomes increasingly problematic with scaling because cross-talk between wires worsens. Finally, the sensitivity of V_{TH} to the body bias decreases with scaling, because the channel doping increases.

Body biasing is limited in the magnitude of the V_{TH} shift that can be induced. The maximum forward-bias is limited by current flows across the P–N junction formed

between the n-well and p-well. A thyristor-like device is formed in the substrate by the two bipolar transistors. Oowaki et al. found that there was no latch-up effect with up to 0.5 V forward bias [11] (assumed by Miyazaki et al. [12], Tachibana et al. [13], and Narendra et al. [14]). The maximum reverse-bias is limited by high leakage and possible break-down across the reverse-biased drain-body junction, particularly during burn-in [14].

The sensitivity of threshold voltage to the body bias for NFETs and PFETs is shown in Fig. 5.1 for the 90, 45, and 22 nm predictive technologies [15]. While the sensitivity of V_{TH} to the body biases does decrease as technology scales, the decrease from 90 to 22 nm (4 technology generations) is only 12% for the NFET and 10% for the PFET. This indicates that body biasing will continue to be a viable solution to control variability in technologies going forward.

Figure 5.2 shows the impact of body biasing on these three metrics for a 13-stage ring oscillator implemented in a 22 nm high performance high-K metal gate predictive technology [15]. The NFET body bias was varied between 500 mV RBB and 500 mV FBB, while the PFET body bias was adjusted to keep the output of an I/O-connected inverter at $\frac{V_{dd}}{2}$, maintaining the inverter beta ratio at its nominal value. Full RBB is seen to reduce frequency by 32%, energy per switching event by 4.4%, and leakage power by 90%. Full FBB, on the other hand, increases frequency by 29%, energy per switching event by 5.6%, and leakage power by 1,300%.

A variety of body biasing schemes have been proposed for a wide range of chip applications. Intel's Xscale processor used reverse body biases (RBBs) during standby mode to reduce leakage [16]. Transmeta's Efficeon applied reverse body biasing on a per-chip basis to reduce power by a factor of 2.5 [17]. Narendra et al. implemented a communication router chip in 150 nm technology with a constant 450 mV FBB applied during active operation to achieve equivalent speed to no body biasing at lower supply voltage [18]. Borkar et al. suggest selecting body biases at test for high-performance microprocessors [19]. A simple general purpose microprocessor was implemented in 0.2 μm technology with adaptive body biasing, and reduced variation in the speed of a single workload by 18% [12, 20].

Three temporal granularities for computing the body biases have been considered on simple chips. First, the NFET and PFET body biases can be determined at test-time to address process variability. Tschanz et al. found a 7x reduction in the variation in die frequency by selecting body biases at test, improving yield from 50 to 100% in a 150 nm test chip [2]. Second, the body biases can be computed when the chip powers on to address long-term temporal variability (for example, NBTI) as well as process variability. Finally, the body biases can be computed continuously to address temperature variations as well as temporal and process variability. Teodorescu et al. found that continuously computing the body biases offered a 35% reduction in power and an 8% increase in frequency over choosing the body biases at test [21]. There has been considerable work on a circuit to adaptively determine the body biases [2, 12, 20, 22].

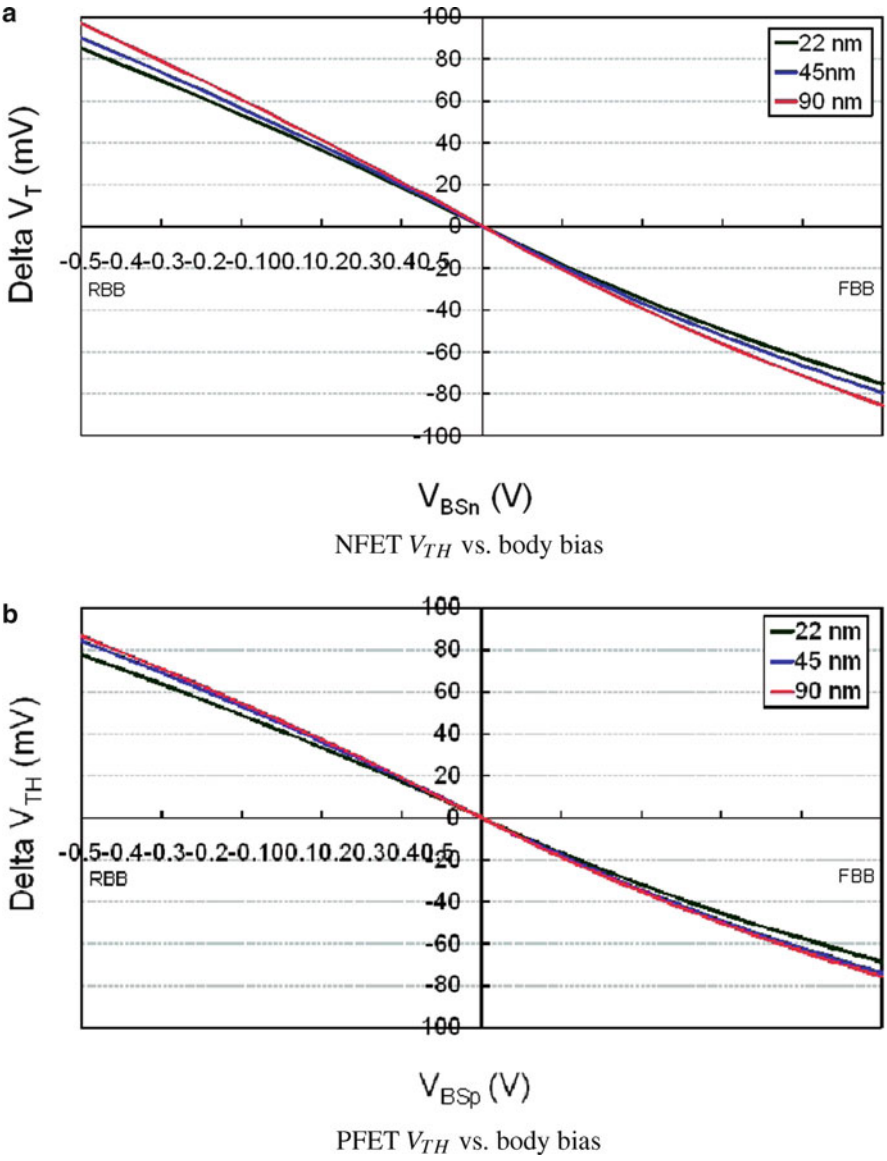


Fig. 5.1 Sensitivity of V_{TH} to body biasing for 90, 45, and 22 nm predictive technologies [15]

5.2 Microprocessor Design Example

In the previous section, we explored the applications of body biasing in a variety of simple chip designs. However, high-performance microprocessors use dynamic voltage/frequency scaling, which has significant implications for the use of body

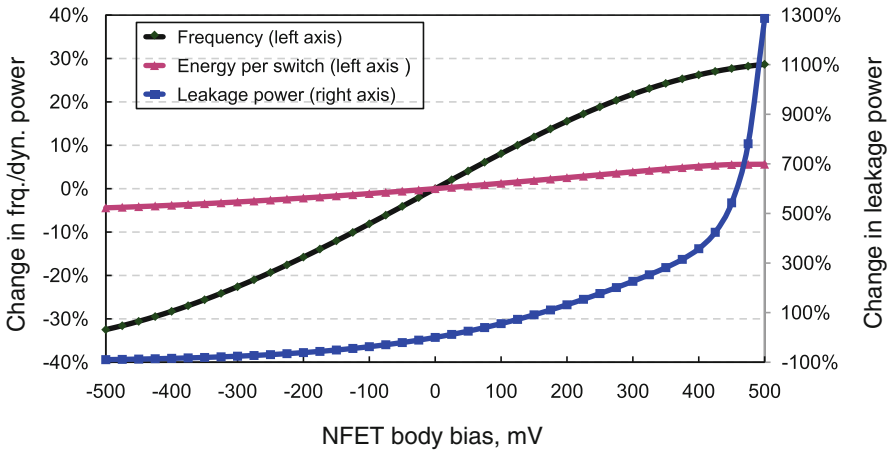


Fig. 5.2 Impact of body biasing on ring oscillator frequency, energy per switch, and leakage power

biasing. Every time the voltage and frequency of the microprocessor change, the body biases must be adjusted in order to ensure that the new frequency is met with the current body biases. In this section, we look at the implementation of body biasing in a 16-core chip-multiprocessor implemented in a high-performance 22 nm technology.

5.2.1 Dynamic Voltage/Frequency Scaling

Dynamic voltage/frequency scaling (DVFS) is implemented on virtually all microprocessors. With DVFS, the voltage and frequency are lowered during frequency-insensitive application phases, such as times when the system performance is limited by memory accesses. Thus, DVFS is able to achieve significant reductions in power for a modest reduction in performance.

Changing the V/F level leads to an abrupt change in the body biases required to meet frequency and balance the strength of the NFETs and PFETs, which changes the implementation costs. One possible implementation is to compute the body biases continuously. Alternatively, a single set of body biases can be found to meet the required frequency at all V/F levels. Four implementations arise from computing the body biases at test-time or at power-on, and computing a single set of body biases for all V/F levels or unique body biases for each V/F level. Instead of the three implementations of body biasing reviewed in the previous section, there are five possible implementations on modern high-performance microprocessors.

Table 5.1 Processor parameters

Parameter	Value
Technology	22 nm high-performance [15]
# of cores	16
Available V_{DD}s	0.5, 0.575, 0.65, 0.725, 0.8 V
DVFS interval	50 µs
L1-I/D caches	64 KB, 64 B blocks, 2-way, 2-cycle load-to-use, LRU, 4R/W
L2 cache	16 × 1 MB, 64 B blocks, 16-way, 20-cycle hit, LRU, 1R+1W per bank
Main memory	60 ns random access, 64 GB s^{-1} peak bandwidth
Pipeline	Eight stages deep, four instructions wide
ROB LSQ size	160
Store buffer size	64

5.2.2 System Architecture

A chip-multiprocessor with the parameters in Table 5.1 is divided into voltage/frequency islands (VFIs). The voltage/frequency levels for the core VFIs are chosen by the DVFS controller. There is one VFI for the L2 cache, network, and memory controller, which always runs at the nominal V_{DD} and the chip's speed bin frequency. On-chip voltage regulators [23] enable fast voltage transitions taking 5 ns. An interval of 50 μs is assumed for DVFS decisions. DVFS uses a threshold-based control algorithm, which keeps the retire slot utilization within given bounds [24]. The utilization is compared to the up threshold (T_{up}) and the down threshold (T_{down}) at every interval, and the V/F level is raised if $U > T_{up}$, lowered if $U < T_{down}$, and held constant while $U \in \left[T_{down}, T_{up}\right]$. An algorithm that directly considers power was not used because it would result in different performances for dies in the same speed bin, which presents marketing difficulties.

In order to determine speed bins, two million dies were generated via simulation. Extrapolating from recent trends in commercial microprocessor prices, body biasing is used to improve the performance of the speed bins (instead of lowering power at the same speed bins). Therefore, separate speed bins were created for no body biasing and body biasing (300 mV FBB was applied to the dies for the body biasing speed bins), each with a yield of 97%. The maximum frequency of the dies was computed at both an age of zero and five years, at the nominal V_{DD} of 0.8 V and worst-case temperature of 100°C. The speed bins were determined from the frequency at 0 years of age plus a fixed margin, equal to the maximum delta between the 0- and 5-year frequencies. The ZBB speed bins are located at 2.53, 2.8, 3.07, 3.33, and 3.60 GHz, and contain 26, 33, 20, 9, and 9% of dies. The BB speed bins are located at 2.93, 3.33, 3.73, 4.13, and 4.4 GHz, and contain 22, 31, 22, 12, and 10% of dies. This translates to an average frequency increase of 22% with body biasing.

Table 5.2 Workloads evaluated

Workload	Notes
Online transaction processing (TPC-C)	
tpcc-db2	DB2, 100 warehouses, 64 clients, 450 MB buffer pool
tpcc-oracle	Oracle, 100 warehouses, 16 clients, 1.4 GB SGA
Decision support systems (TPC-H on DB2)	
tpch-qry1	450 MB buffer pool, scan-dominated
tpch-qry2	450 MB buffer pool, join-dominated
Web server (SPECweb99)	
apache	16K connections, FastCGI, worker threading model
zeus	16K connections, FastCGI

5.2.3 Workloads Evaluated

The multi-threaded workloads in Table 5.2 were evaluated. The online transaction processing workloads consist of TPC-C v3.0 on both *IBM DB2 v8 ESE* and *Oracle 10g Enterprise Database Server*. The Decision Support Systems (DSS) workloads are two queries from TPC-H, both on DB2. Apache HTTP Server v2.0 and Zeus Web Server v4.3 are evaluated on SPECweb99 under saturation by requests.

5.2.4 Simulation Infrastructure

Extraction of the workload parameters was performed using Flexus CMPFlex.OoO, a microarchitecture-level simulator for chip-multiprocessors [25]. Flexus runs real workloads on real operating systems and model processors, main memory, disks, and all other components of a computer system. The power and thermal modeling extensions released by Herbert and Marculescu [24] were used to gather power statistics while accounting for the impact of temperature.

It is simple to determine throughput in the high-level model because all dies within a speed bin have identical performance. The first workload-specific parameter is $T_{i,j}$, the throughput of workload i running on a die from speed bin j. $T_{i,j}$ is obtained by recording the number of non-spin user-mode instructions retired during a Flexus run, representing the amount of progress made by the application.

The power drawn when running a given workload in a given speed bin is more complex. Workload-specific factors affect the utilization metric used in DVFS control (e.g., cache miss rates), so different workloads have different distributions of runtime over the V/F levels. The benefit of a body biasing scheme depends on the V/F level, so this must be accounted for when computing the average power drawn by a particular sample die running a particular workload. This is handled in

the high-level model by computing the power draw at each distinct V/F level and then weighting each power value by $L_{i,j,k}$, the portion of time workload i running on a die from speed bin j spends at V/F level k.

The power is split into dynamic and static components; both must be computed per V/F level. Detailed simulation is used to extract $P^{\text{dynamic}}_{i,j,k}$ and $P^{\text{static}}_{i,j,k}$, the dynamic and static power drawn when running workload i on a baseline die from speed bin j at V/F level k. The choice of baseline die is arbitrary, and a die matching the mean leakage of all dies in the speed bin was used. To obtain the dynamic and static power for a sample die, $P^{\text{dynamic}}_{i,j,k}$ and $P^{\text{static}}_{i,j,k}$ are scaled using the circuit-level dynamic energy per switching operation and static power models. This is done by multiplying the baseline value by the average model output across all variation map grid points on the sample die and dividing by the average model output across the baseline die.

For each workload, 50,000 dies were generated from the variation model. The frequency, static power, and energy per switch models were combined with the Flexus results to determine the throughput and power of each core. The HotSpot thermal simulator [26] was used to account for the dependence of power on temperature; simulations were iterated until the power and temperature values converged.

5.2.5 Schemes Evaluated

The baseline is a traditional DVFS scheme with no body biasing, referred to as *ZBB*. Five implementations of body biasing are considered, designated by a name starting with *BB*. Body biases are applied at the per-core level. In *BB-Cont.*, the body biases were continuously computed. The remaining body bias schemes are denoted *BB-x-y*, where *x* specifies the time when the body biases were computed ("T" for at test or "P" for at power-on), and *y* specifies whether a single set of body biases is used for all V/F levels ("S") or unique body biases are calculated for each V/F level ("A"). For example, *BB-T-A* indicates that the body biases were chosen at test, with unique body biases for each V/F level.

5.2.6 Results

Figure 5.3 compares the effectiveness of the five body biasing schemes at ages of zero and five years. Their effectiveness is measured by power over throughput squared (P/T^2), which is equal to the product of energy per instruction and time per instruction. The results are averaged across the workloads and the results of each bin are weighted by the percentage of dies in the bin. Each bar is normalized to the power of *ZBB* on the same hardware at the same age.

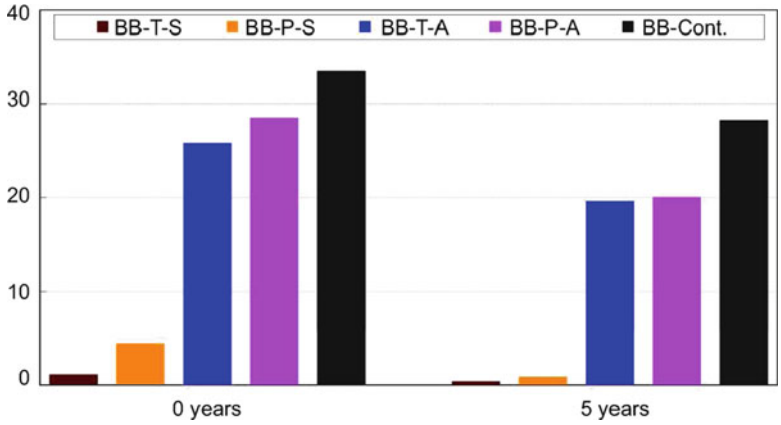


Fig. 5.3 P/T^2 by age, averaged across workloads and speed bins

The implementation of body biasing has a large effect on the improvement in P/T^2. The schemes that assign a single set of body biases for all V/F levels show little benefit over *ZBB* at both ages, while the three schemes that assign unique body biases per V/F level show significant improvements. The improvement decreases for all of the schemes between zero and five years. This is because static power decreases with aging and static power is much more sensitive to body biasing than dynamic power (static power is exponentially dependent on V_{th}; body biasing only modulates dynamic power through capacitance). The power-on schemes show 3.8% improvement over the test-time schemes at zero years. The power-on schemes still show a slight benefit over the test-time schemes at five years because the test-time scheme margins against worst-case (instead of typical) aging. *BB-Cont.* achieves 7% better P/T^2 than *BB-P-A* at 0 years. The delta between *BB-Cont.* and *BB-P-A* is larger at five years than at zero years because the power is lower at five years and thus the delta between the maximum temperature and the operating temperature is larger.

5.3 A Yield Prediction and Layout Optimization Technique for Digital Potentiometer IP cores

Fabrication nonidealities introduced by increasing the degree of integration have made analog IP design a challenging task. The design process for engineering reliable analog IP in selected nanoscale target processes has become very complex and time consuming. Design automation in this area is required urgently. In the following chapter, a method for yield prediction calculation of a configurable Digital Controlled Potentiometer (DCP) IP core has been worked out. This method comes along with a novel, mainly automated IP integration and characterization

process with excellent design migration capabilities. The presented DCP IP core is optimized to be used as a DAC and a SAR ADC in a wide specification range. The architecture is made robust against resistor parameter variations. It even takes advantage of them by applying a statistical averaging approach. By arranging raw unit devices in the layout in a compound way, the overall accuracy of these compound devices can be increased significantly depending on the number of used unit devices.

The behavioral model of the DCP topology includes all relevant realistic performance degradation parameters. The optimization potential in terms of yield is analyzed taking systematic and statistical properties of the DCP topology into account. An analytic yield-over-tolerance function is determined by mean value, standard deviation, and correlation matrix of the used unit resistors. Behavioral modeling and yield optimization of resistor string based architecture (potentiometer) Digital-to-Analog Converters (DACs) is presented to improve its reliability and area efficiency with focus on nanoscale CMOS processes, which suffer from large device tolerances. The optimization potential in terms of yield is analyzed taking systematic and statistical properties into account. Background mathematics is illustrated, and a comparison with corresponding Monte Carlo simulation is given.

The presented optimization technique is able to support resolutions up to 14 bit with guaranteed monotony and high linearity. The design optimization process can be mapped to individual process profiles and takes various parasitic effects into account. An accurate yield estimation algorithm detects the critical fabrication influence to IP performance in advance. Based on given IP specification and process selection, all design views (schematic, layout, behavioral model and verification testbenches) are generated automatically by using executable design descriptions. Measurement results of rapid integrations in different process technologies will be presented and discussed.

5.3.1 Limiting Matching Effects for Resistors in Analog Circuits

IC technology process scaling introduces increasing nonidealities. Manufacturing parameter variation is one of these. This work has already shown that predictable parameter variations help to design robust circuits, but in addition they can be utilized even to improve quality and reliability of analog circuit design beyond known limits. Recently, it has been found that regularity, which enables statistical averaging, is one of the keys for accurate and reliable analog circuit design in nanoscale process technologies. The mathematical background for applying statistical averaging is introduced in this section.

Exemplary the matching behavior of resistors is discussed here in detail. In principle, an analog discussion is possible also for capacitor arrays and transistor arrays (current mirrors, reference generation).

The matching behavior of resistors in analog circuits is influenced by several important parameters. These parameters are the spread in sheet resistance, lithography and etching errors, bend effects, contact resistance, and alignment errors. Lithography and etching errors lead to statistical and deterministic deviations of resistor ratios from the desired value. The precision of analog circuits based on the string principle (e.g., digital-to-analog converter, analog-to-digital converter) is usually independent of the design's center resistance value, which is defined as the averaged resistance of all matched resistors in one converter. The absolute value can fluctuate in a range of about 30% from one wafer to another. Statistical deviations from this value are large for small structures and become smaller if structure size increases. Increasing the resistor size, however, must be carried out with care because of the increasing influence of deterministic deviations for larger structures. Thus, gradient errors with respect to the resistor core depend on height, orientation, and slope of the spread in sheet resistance. An optimal geometry for resistor layout has to be found, which reduces the statistical errors to minimum, assuming the statistical errors still dominate when compared with gradient errors.

In hand-crafted analog design, it is usual to design well-matched resistors without any bends. In a flexible layout generator for different resolutions (2 in. resistors), the resistors must contain bends or the area requirements would become excessive. Effects that decrease matching due to deterministic errors caused by bends, lithography, and etching are minimized by using an identical basic resistor cell for all matched resistors.

Another critical problem stems from contacts interconnecting the array resistors. Their resistance varies widely and can degrade matching behavior. Especially in data converters featuring higher resolution in which low-valued resistors ($<10\,\Omega$) are used, contact resistance can easily dominate over the string resistance. Also, mask tolerances can lead to contact alignment errors. To avoid these problems, the resistors can be arranged in groups in which no contacts are used in the current path.

The resistors are interconnected without leaving the resistor layer. At the end of each resistor chain, where contacts are unavoidable, arrays of contacts are used for interconnection.

5.3.2 Modeling of Statistical Averaging Principle

Statistical averaging can be utilized to increase the overall accuracy of an arrangement of inaccurate unit devices, resulting in an improved yield [27]. The yield function, which depends on the tolerance of the single unit elements, is found typically by extensive use of Monte Carlo simulation. An alternative analytic yield calculation approach is presented here, which is in line with Monte Carlo simulation results, providing useful insight into yield properties and showing accuracy improvement potential of compound devices, exploiting the statistical average principle.

5.3.2.1 Mathematical Fundamentals

A normally distributed random variable X has an expectation μ, a variance σ^2 and can be written as $X = N(\mu, \sigma^2)$. The density function of X is:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \mathrm{e}^{\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)}.$$

For the multivariate case, an m-dimensional random variable has an m-dimensional normal distribution if it has the following joint density function:

$$f_{X_1 X_2 \cdots X_m}(x_1, x_2, \cdots, x_m) = \frac{1}{(2\pi)^{m/2}\sqrt{\det(\Sigma)}} \mathrm{e}^{\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)}. \tag{5.2}$$

Σ is the symmetric, positive definite (m,m)-covariance matrix, $x^T = (x_1, x_2, \cdots, x_m)$ and $\mu^T = (\mu_1, \mu_2, \cdots, \mu_m)$. Σ can be expressed taking the diagonal matrix $\boldsymbol{D}$ containing the standard deviations of the m components of the random vector $\underline{X} = (X_1, X_2, \cdots, X_m)^T$ and its correlation matrix $\boldsymbol{P}$ into account (compare (2.71)):

$$\Sigma = \boldsymbol{D} \cdot \boldsymbol{P} \cdot \boldsymbol{D}.$$

Distribution of the ratio of two normally distributed random variables.

In a note on page 48 in Sect. 2.2.7, we have shown how to describe the sum of a number of uncorrelated normally distributed random variables. For instance, this is important to characterize a series connection of resistors where its probability distributions are uncorrelated.

If we are interested in the behavior of a voltage divider for instance, this kind of description will be useful but is not sufficient. We need to describe the ratio of correlated normally distributed random variables X and Y. How to describe the PDF of this ratio will be derived in the following.

At first, we shortly recapitulate the result already presented by equation (2.42) in Sect. 2.2.3. Let X and Y be normal random variables with $X = N(\mu_X, {\sigma_X}^2)$ and $Y = N(\mu_Y, {\sigma_Y}^2)$. X and Y shall be correlated with each other, represented by the correlation coefficient ρ. The covariance matrix is (see also (2.41) on page 36)

$$\Sigma = \begin{pmatrix} {\sigma_X}^2 & \rho\sigma_X\sigma_Y \\ \rho\sigma_X\sigma_Y & {\sigma_Y}^2 \end{pmatrix} = \begin{pmatrix} \sigma_X & 0 \\ 0 & \sigma_Y \end{pmatrix} \cdot \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \cdot \begin{pmatrix} \sigma_X & 0 \\ 0 & \sigma_Y \end{pmatrix}. \tag{5.3}$$

Using (5.2) and (5.3), the bivariate density function $f_{X,Y}(x,y)$ of X and Y can be calculated as follows [28]:

$$f_{X,Y}(x,y) = \frac{\exp\left[-\frac{\left(\frac{x-\mu_X}{\sigma_X}\right)^2 - 2\rho\left(\frac{x-\mu_X}{\sigma_X}\right)\left(\frac{y-\mu_Y}{\sigma_Y}\right) + \left(\frac{y-\mu_Y}{\sigma_Y}\right)^2}{2\left(1-\rho^2\right)}\right]}{2\pi\sigma_X\sigma_Y\sqrt{1-\rho^2}}. \tag{5.4}$$

By using the following substitution:

$$w = \frac{x}{y} \Leftrightarrow x = w \cdot y, \tag{5.5}$$

the bivariate probability density function (PDF) of (5.4) can be utilized to calculate the probability density function of the ratio of X and Y:

$$f_{\frac{X}{Y}}(w) = \int_{-\infty}^{\infty} |y| \cdot f_{X,Y}(w \cdot y, y)\, \mathrm{d}y. \tag{5.6}$$

The argument of the exponential function in $f_{X,Y}(w \cdot y, y)$ of (5.4) is a polynomial of second order in y. Solving the integral of (5.6) leads to the following equation [29]:

$$f_{\frac{X}{Y}}(w) = \frac{b(w) \cdot d(w)}{\sqrt{2\pi}\sigma_X \sigma_Y a(w)^3} \left[2 \cdot \Phi\left(\frac{b(w)}{\sqrt{1-\rho^2}a(w)} \right) - 1 \right] + \frac{\sqrt{1-\rho^2}}{\pi \sigma_X \sigma_Y a(w)^2} \mathrm{e}^{-\frac{c}{2\left(1-\rho^2\right)}}, \tag{5.7}$$

where:

$$a(w) = \sqrt{\frac{w^2}{{\sigma_X}^2} - \frac{2\rho \cdot w}{\sigma_X \sigma_Y} + \frac{1}{{\sigma_Y}^2}}$$

$$b(w) = \frac{\mu_X w}{{\sigma_X}^2} - \frac{\rho(\mu_X + \mu_Y \cdot w)}{\sigma_X \sigma_Y} + \frac{\mu_Y}{{\sigma_Y}^2}$$

$$c = \frac{{\mu_X}^2}{{\sigma_X}^2} - \frac{2\rho \cdot \mu_X \mu_Y}{\sigma_X \sigma_Y} + \frac{{\mu_Y}^2}{{\sigma_Y}^2}$$

$$d(w) = \exp\left(\frac{b(w)^2 - c \cdot a(w)^2}{2(1-\rho^2) \cdot a(w)^2} \right).$$

φ and Φ are the PDF and CDF, respectively, of the standard normal distribution (see also (2.32) and (2.33)). The ratio of two normal distributed random variables, shown in equation (5.7) is not normal distributed anymore, having a median value instead of an expectation value and no standard deviation. Nevertheless, the PDF of (5.7) is similar to a normally distributed PDF but having longer tails.

5.3.2.2 Yield Calculation

Resistive Voltage Divider

Figure 5.4 shows the resistor string topology for a resistive voltage divider, consisting of 2 (case A), 4 (case B), and m (case C) normally distributed unit resistors.

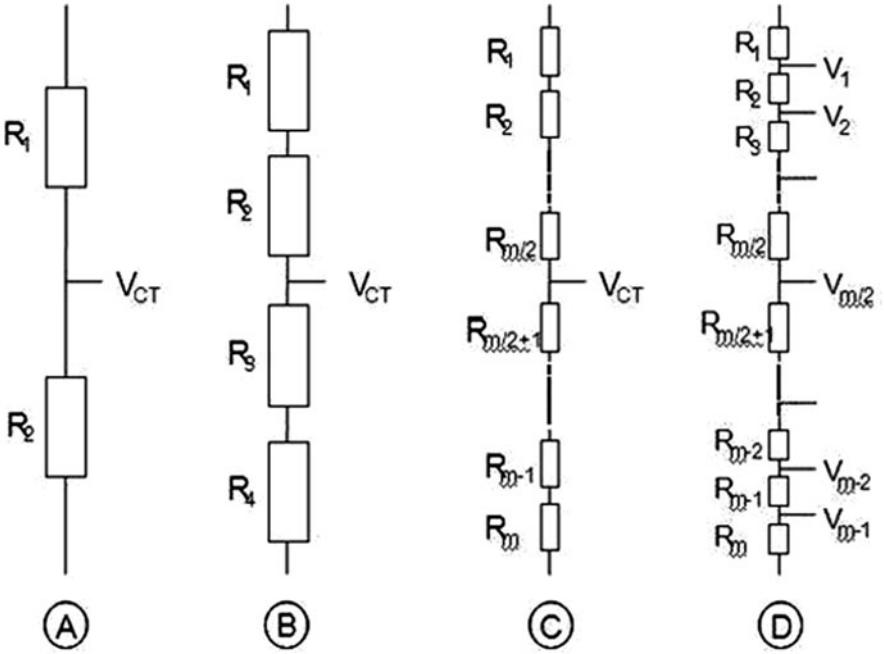


Fig. 5.4 Resistor string topologies

Let R_k be the kth resistor of a resistor string. All resistors are $N(\mu_R, {\sigma_R}^2)$ distributed and uncorrelated. The voltage at the center tap V_{CT} of the resistive voltage divider is the ratio of two correlated normal random variables X and Y depending on the resistors that have to be considered:

$$V_{\mathrm{CT}} = \frac{\sum\limits_{k=1}^{m/2} R_k}{\sum\limits_{k=1}^{m} R_k} = \frac{\frac{2}{m} \cdot \sum\limits_{k=1}^{m/2} R_k}{\frac{2}{m} \cdot \sum\limits_{k=1}^{m} R_k} = \frac{X}{Y}. \tag{5.8}$$

As X and Y are the sums of normally distributed random variables, they are themselves $N(\mu_X, {\sigma_X}^2)$ and $N(\mu_Y, {\sigma_Y}^2)$, respectively, normally distributed. Considering the relations from Sect. 2.2.2 and the fact that they are built up by a sum of uncorrelated random variables (see note on page 29), we can determine their expected values and variances. We get $E[X] = \mu_X = \frac{2}{m} \cdot \frac{m}{2} \cdot \mu_{\mathrm{R}} = \mu_{\mathrm{R}}$ and $\mathrm{var}X = {\sigma_X}^2 = \left(\frac{2}{m}\right)^2 \cdot \frac{m}{2} \cdot {\sigma_{\mathrm{R}}}^2 = \frac{2}{m} \cdot {\sigma_{\mathrm{R}}}^2$. Y is described by $E[Y] = \mu_Y = \frac{2}{m} \cdot m \cdot \mu_{\mathrm{R}} = 2 \cdot \mu_{\mathrm{R}} = 2 \cdot \mu_X$ and $\mathrm{Var}Y = {\sigma_Y}^2 = \left(\frac{2}{m}\right)^2 \cdot m \cdot {\sigma_{\mathrm{R}}}^2 = \frac{4}{m} \cdot {\sigma_{\mathrm{R}}}^2 = 2 \cdot {\sigma_X}^2$.

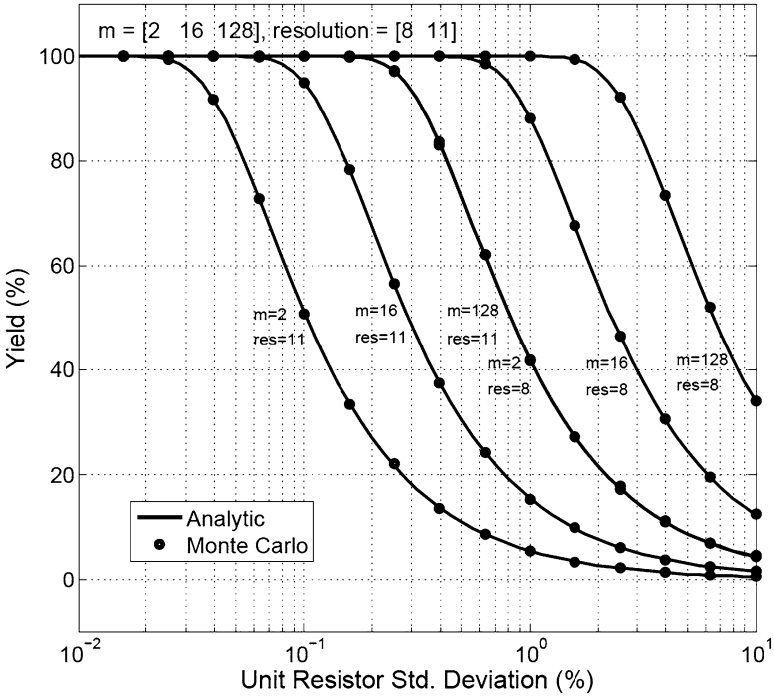


Fig. 5.5 Yield function for center tap: Monte Carlo vs. analytic results

$\frac{m}{2}$ resistors belong to X as well as to Y. Thus, both random variables are correlated. The correlation coefficient can be determined by

$$\rho_{X,Y} = \frac{\operatorname{cov}\left(\frac{2}{m}\cdot\sum_{k=1}^{m/2} R_k, \frac{2}{m}\cdot\sum_{k=1}^{m} R_k\right)}{\sigma_X\cdot\sigma_Y} = \frac{{\sigma_X}^2}{\sigma_X\cdot\sigma_Y} = \frac{\sigma_X}{\sigma_Y}. \tag{5.9}$$

Thus, the correlation coefficient for the center tap voltage divider case is $\rho = \frac{1}{\sqrt{2}}$ as the resistive sum below the center tap is always half the overall resistive sum, leading to $\sigma_Y = \sqrt{2}\cdot\sigma_X$. With the knowledge of the correlation coefficient, (5.7) can be applied to describe the PDF $f_{V_{\mathrm{CT}}}$ of the center tap voltage.

The yield-over-tolerance function for the center tap voltage divider can be calculated by integrating the PDF of (5.7) in the following way:

$$\text{Yield} = \frac{\int_{V_{\mathrm{CT,ideal}}-LSB/2}^{V_{\mathrm{CT,ideal}}+LSB/2} f_{V_{\mathrm{CT}}}(w)\mathrm{d}w}{\int_{-\infty}^{\infty} f_{\frac{X}{Y}}(w)\mathrm{d}w} = \int_{\frac{1}{2}-LSB/2}^{\frac{1}{2}+LSB/2} f_{V_{\mathrm{CT}}}(w)\mathrm{d}w \tag{5.10}$$

Figure 5.5 shows the yield function for the center tap of several resistive voltage dividers as function of the tolerance of the single unit resistor elements. The number of resistors m in the voltage divider is parameterizable and set to 2, 16, and 128. The resolution criteria *res* for the yield function is set to 8 bit (Least significant bit = LSB = $1/2^8$) and 11 bit (LSB = $1/2^{11}$), respectively. The calculated yield results based on integration of the probability density function (5.7) according to (5.10) and the statistical simulated results using Monte Carlo Analysis (10,000 simulations per yield point) show excellent matching.

By quadrupling the number of unit resistor elements, the accuracy can be increased by 1 bit without changing the yield function. This principle is shown in Fig. 5.5, where the yield functions for [$m = 2$ and $res = 8$] and for [$m = 2 \cdot 4^3 = 128$ and $res = 8 + 3 = 11$] are identical. This result is in line with the square-root law of information theory (the accuracy of information is equal to the square-root of the volume of information [30]). The complex probability density function of (5.7) can be rearranged for the center tap voltage divider case using $\mu_Y = 2 \cdot \mu_X$, $\sigma_Y = \sqrt{2} \cdot \sigma_X$ and $\rho = \frac{1}{\sqrt{2}}$ to:

$$f_{V_{\mathrm{CT}}}(w) = \frac{\mu_X \cdot \mathrm{Erf}\frac{\mu_X}{\sigma_X\sqrt{2K}} \cdot \exp\left(\frac{-{\mu_X}^2(2K-1)}{2{\sigma_X}^2 K}\right)}{\sqrt{2\pi} \cdot \sigma_X \cdot K^{3/2}} + \frac{\exp\left(-\frac{{\mu_X}^2}{{\sigma_X}^2}\right)}{\pi \cdot K}, \tag{5.11}$$

where $K = (1 - 2w + 2w^2)$. The probability density of V_{CT} is asymmetric and depends on w, defined in (5.5), expectation value μ_X and standard deviation σ_X. By increasing the number m of unit resistors $\sigma_X = \sqrt{\frac{2}{m}} \cdot \sigma_{\mathrm{R}}$ can be reduced.

The slope of the V_{CT} yield function (5.10) is independent of the number of used unit resistor elements, because of the fixed relationship between σ_X and σ_Y. By choosing the relevant correlation coefficient p the yield function of any single tap of a resistor string can be calculated analytically, being in line with the square-root law[1] of information theory and leading to the same results as the corresponding Monte Carlo simulations.

Simple Potentiometer DAC

Figure 5.4 (case D) shows the resistor string used as voltage reference for a potentiometer DAC. For the case of a DAC, every tap of the resistor string has to fulfill the accuracy requirements, leading to a worse yield function with steeper slopes compared to a resistive voltage divider and leading to the fact that the square-root law of information is not valid anymore, meaning quadrupling the number of unit resistors by four increases the DAC accuracy by less than 1 bit. Both matters of facts are shown in Fig. 5.6.

[1]For more information, see note concerning "Square-Root Law" on page 56.

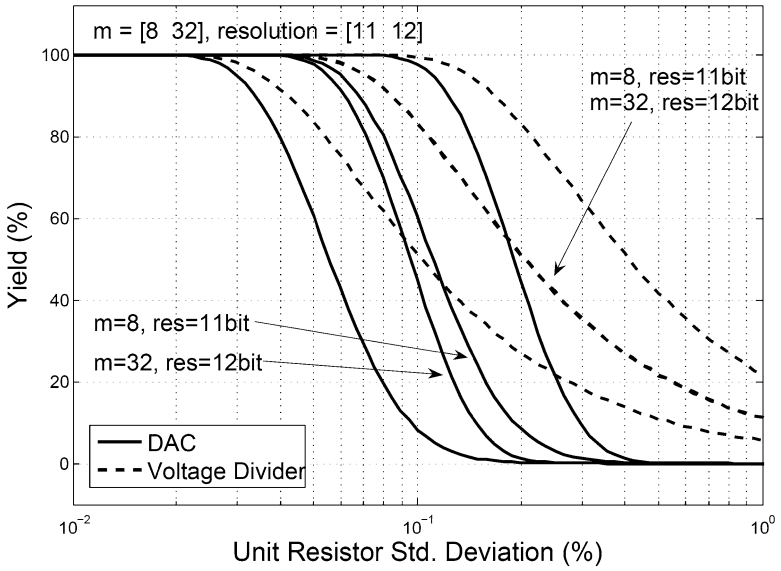


Fig. 5.6 Yield comparison: DAC vs. voltage divider

To calculate the yield function for a DAC consisting of m elements, an $(m-1)$-dimensional probability density function, covering all $(m-1)$ node voltages has to be integrated numerically, instead of solving only a one-dimensional integral (5.10) for the bivariate case of the voltage divider. The calculation effort for the multivariate case is growing drastically for higher numbers of unit resistor elements, restricting the usability of this approach for DACs with $m \leq 8$. For DACs containing a larger amount of unit elements, the Monte Carlo simulation method has to be used for yield analysis, as it can be simulated much faster, providing the same results. Figure 5.7 compares the yield functions of analytical approach and corresponding Monte Carlo result for a DAC consisting of four unit elements $(m = 4)$, proving the matching of both methods. Furthermore, it is shown that an assumption of independent DAC voltage nodes (correlation matrix $\boldsymbol{P}$ equal to unit matrix) leads to a too pessimistic yield plot, especially for higher number of unit resistor elements. In other words, the inherent correlation between the different DAC voltage taps is an improving factor in terms of yield.

The analytical approach can be used efficiently for the calculation of the voltage probability density function for any single tap of an R-String DAC to localize the critical taps, gaining more insight into the DAC topology. Figure 5.8 shows the relative width of the probability density functions of all taps (measured at the point, where the probability density is 1% of the maximum value at the corresponding median position) for R-Strings consisting of 8, 16, 32, and 64 of unit elements. The center tap of an R-String (zero position in Fig. 5.8) is the most critical node, where the probability density function has the largest width and this PDF width is getting smaller symmetrically in direction to the edges of the R-String.

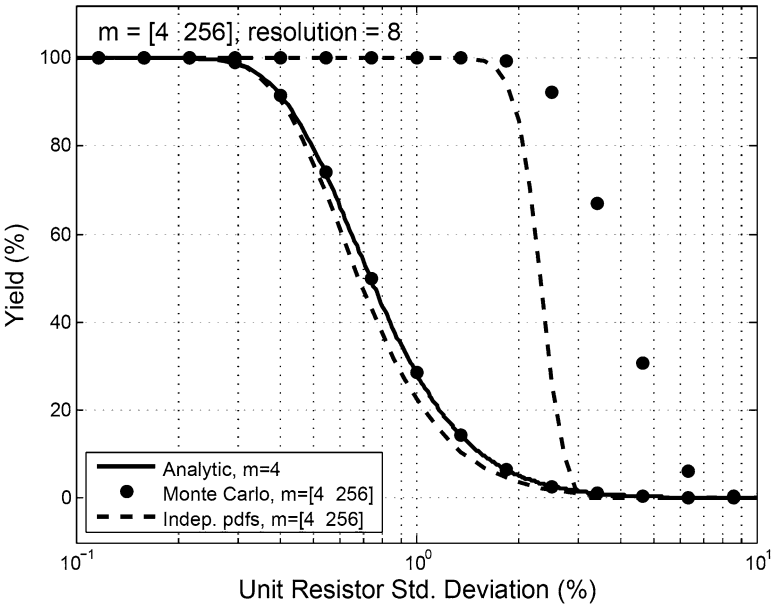


Fig. 5.7 Yield comparison: analytic vs. Monte Carlo vs. independent PDFs

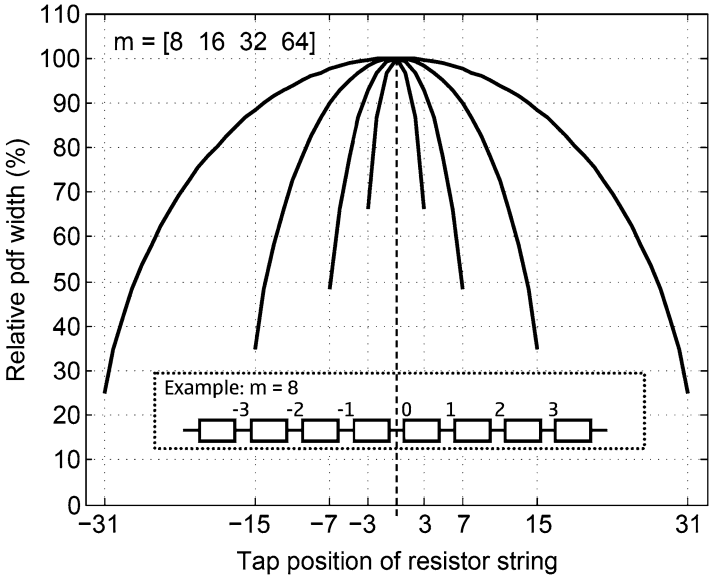


Fig. 5.8 Relative voltage PDF width of DAC taps

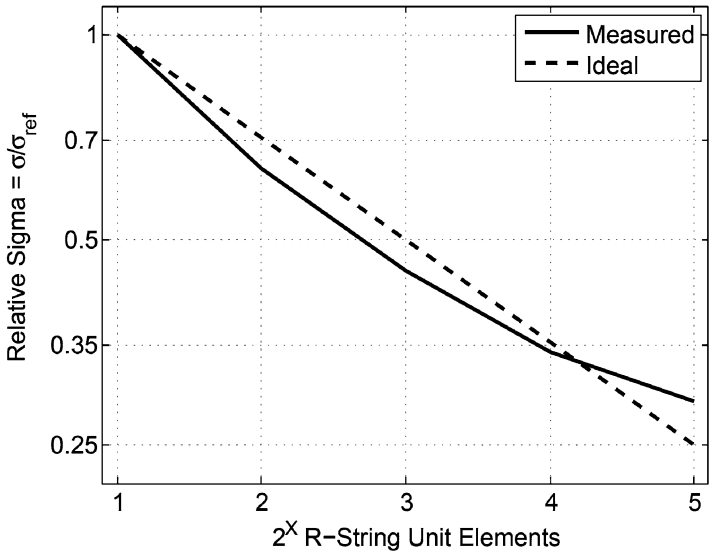


Fig. 5.9 Relative Sigma: measured vs. ideal

5.3.2.3 Comparison with Measurement Results

A 12-bit potentiometer DAC consisting of 32 coarse resistors and fine resistor strings (consisting of 128 serial unit resistors) in parallel to each coarse resistor has been designed using a 65 nm CMOS technology. The 15 measured samples show an average integral nonlinearity of 0.45 LSB and a differential nonlinearity of 0.1 LSB. The number of samples was too small for a statistical yield analysis, but as every DAC consists of 128 fine resistor chains, these data could be used to evaluate the accuracy of the square root law of information theory for a state-of-the-art CMOS process. Figure 5.9 shows the relative standard deviation of the average unit resistor value, calculated out of 480 bunches of serially connected unit resistors consisting of 2^1 up to 2^5 resistors. Each doubling of the number of unit resistors leads ideally to a decrease of the standard deviation by a factor of $1/\sqrt{2}$ related to the average unit resistor value. Figure 5.9 shows good agreement between measured and ideal curve.

5.3.2.4 Yield-Aware Analog Circuit Design

A general analytical yield calculation method has been applied to circuit topologies, based on unit resistor strings. The calculated yield-over-tolerance functions are in line with corresponding statistical Monte Carlo simulations. Measurements show that the square root law of information theory is valid even for current nanoscale process technologies. By arranging raw unit devices with poor standard deviation in the described way, the overall accuracy of these compound devices can be increased arbitrarily, depending on the number of used unit devices. The knowledge about

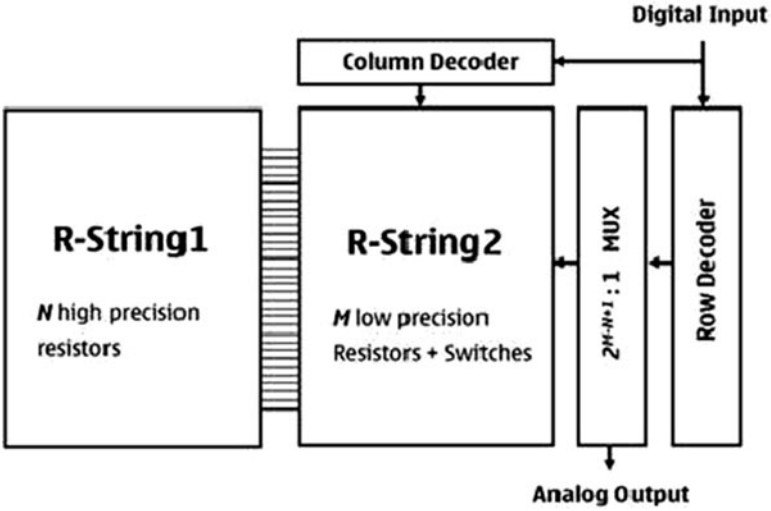


Fig. 5.10 Structure of an *M* bit potentiometer DAC

yield property and yield optimization in combination with the statistical averaging principle is the enabling factor to create competitive analog circuits for mass production using latest nanoscale processes.

5.3.3 Yield Model for Intermeshed R-String Architecture

Digital-to-Analog Converters (DACs) and their counterparts are vital in every modern transmission system by providing the interface between analog and digital domains. The focus in the following section is set on DACs, consisting of two resistor chains, built up by regularly resistive unit elements. Analog passive devices in nanoscale processes suffer from large device tolerances, which make it difficult to fulfill given accuracy requirements. The technology-related possibilities for accuracy improvement are restricted, as the processes have to be price-competitive. Alternatively, the statistical averaging approach can be utilized to increase the overall accuracy of an arrangement of raw unit devices with poor standard deviation, leading to improved yield and reliability. This enables the creation of competitive analog circuits for mass production using latest nanoscale processes [31].

5.3.3.1 Potentiometer DAC Topology

Figure 5.10 shows the block diagram of an *M* bit potentiometer DAC. R-String1 consists of *N* high-precision resistors, providing high-accurate reference node voltages. This R-String determines the overall accuracy of the potentiometer DAC.

Therefore, the area of each single unit resistor is large to achieve a small standard deviation. Furthermore, R-String1 has low impedance, determining the driving capability of the overall DAC. R-String2 consists of N subchains of 2^{M-N} serial low-precision unit resistors (M unit resistors in total). Each of the single R-String2 subchains is attached to one R-String1 resistor. The area efficiency of R-String2 can be increased strongly using small unit resistors with poor standard deviation, as they only have to meet an accuracy criterion of $(M-N)$ bits. This DAC topology enables an area- and power-efficient design, as one R-String of the DAC is fully in charge of overall accuracy and driving capability utilizing large low-impedance unit resistors, and the other R-String has relaxed accuracy requirements and provides the needed voltage nodes using very small unit resistors.

5.3.3.2 Behavioral DAC Model

The resistive string of the potentiometer DAC is modeled using Matlab. The following performance degradation effects are modeled:

Systematic errors: The gradient effect is modeled as well as effects of additional impedances at the connection points between R-String1 and R-String2 and at the reversal point of each R-String2 subchain, which are leading to INL[2] and DNL[3] discontinuities. These systematic errors can be reduced by proper layout.

Statistical errors, based on the normally distributed unit resistors of R-String1 and R-String2. This type of error can be reduced utilizing the statistical averaging principle. In other words, by using unit resistors in a compound way the overall accuracy of the compound resistor can be increased. It is therefore possible to buy accuracy (in terms of smaller standard deviation) at the cost of an increase of layout area.

Figure 5.11 shows the intermeshed resistor string used as voltage reference for a potentiometer DAC. For the case of a DAC, every node of the resistor string has to fulfill the given accuracy requirements. All DAC unit resistors are assumed to be normally distributed as discussed in the previous section.

The statistical averaging principle makes use of the square-root law. Related to a resistor string consisting of unit resistors this means: Each doubling of the number of unit resistors leads ideally to a decrease of the standard deviation by a factor of $1/\sqrt{2}$ related to the overall average unit resistor value. Figure 5.12 shows an example: The overall resistance shown in Fig. 5.12b has the same expectation value μ_R like the resistance of Fig. 5.12a, but the standard deviation is halved, as it consists of 4 unit resistor elements.

Resistance gradients in any direction over the chip layout cause DAC performance degradation. This incident is called gradient effect. Main cause for the

[2]Integral nonlinearity

[3]Differential nonlinearity

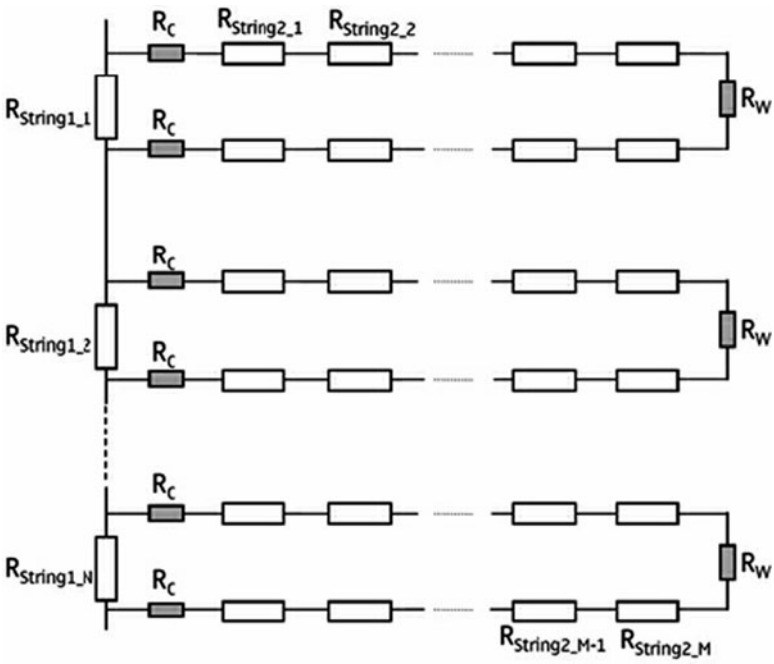


Fig. 5.11 Structure of a intermeshed R-String

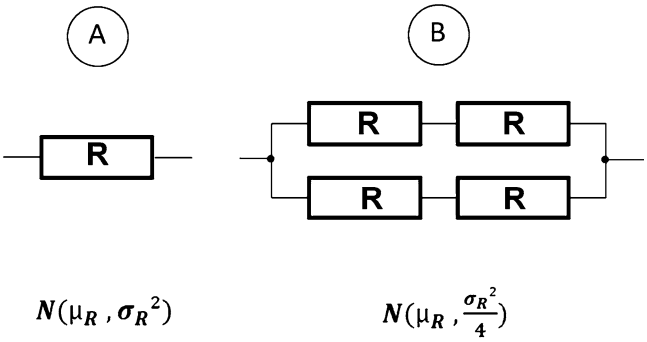


Fig. 5.12 Example of statistical averaging principle

gradient effect is the local variation of the square resistance, which depends on the thickness of the resistive material, on the doping profile, etc. Only gradients in direction along the R-String1 (vertical direction in Fig. 5.11) are critical, as the R-String1 determines the overall DAC accuracy. A gradient along the R-String2 can be neglected, as R-String2 has relaxed accuracy requirements. The following general formula is used for calculating the resistor values according to the gradient effect:

$$R_{\text{String1}}(i) = R_\mu + (i-1) \cdot G \cdot R_\mu \,, \tag{5.12}$$

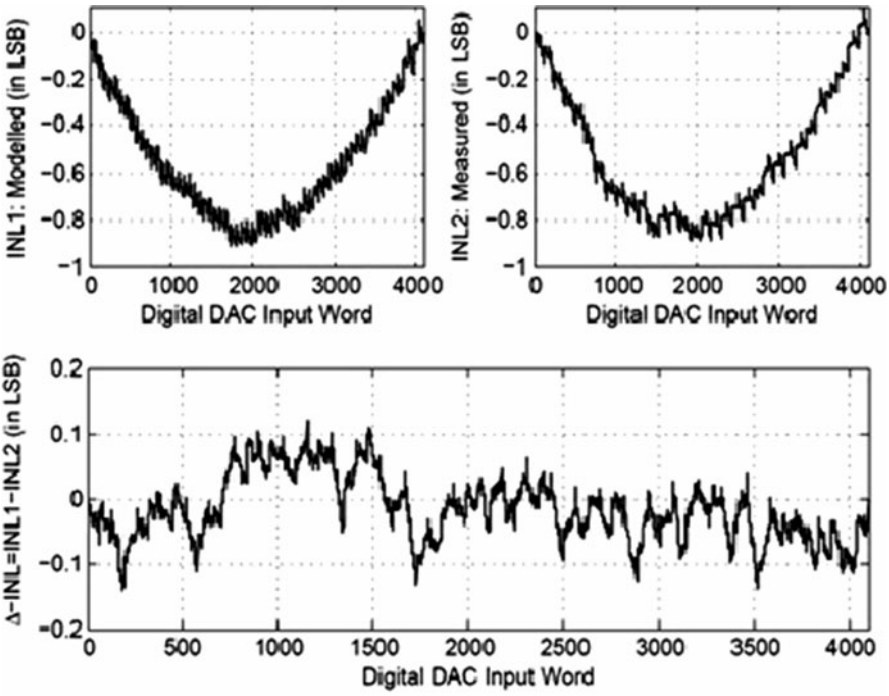


Fig. 5.13 Gradient effect, 12 bit DAC (180 nm)

where G is the resistance gradient from one unit resistor of R-String1 to the adjacent unit resistor, and R_μ is the expectation value of the R-String1 unit resistors. Figure 5.13 shows the INL performance comparison between measurement and model ($G = 5 \cdot 10^{-5}$) of a 12-bit DAC using 180 nm CMOS process. The resistance gradient modeling shows good agreement with measurement results.

Besides the wanted unit resistor elements, layout-related additional resistances are unavoidable which decrease the overall DAC performance. Contact resistances R_C and wiring resistances R_W (see Fig. 5.11) are the dominating error sources for the DNL performance of the discussed DAC topologies. These additional resistances are not normally distributed and are set to constant values in the DAC model as a first-order approximation. Figure 5.14 shows the modeled and measured (12-bit DAC, 65 nm) INL results visualizing the degradation effect caused by additional resistances $R_C = R_W = 1\Omega$. Good agreement between measurement and model is visible.

Figure 5.15 shows the modeled yield function of the 12-bit potentiometer DAC (65 nm) as function of the unit resistor tolerances of R-String1 and R-String2, based on extensive Monte Carlo simulations. The black square in the plots shows the indirectly measured yield point based on estimations of the standard deviations of the measured unit resistors of both R-Strings. This result shows the optimization potential of the DAC in terms of area efficiency, as the unit resistor tolerances of

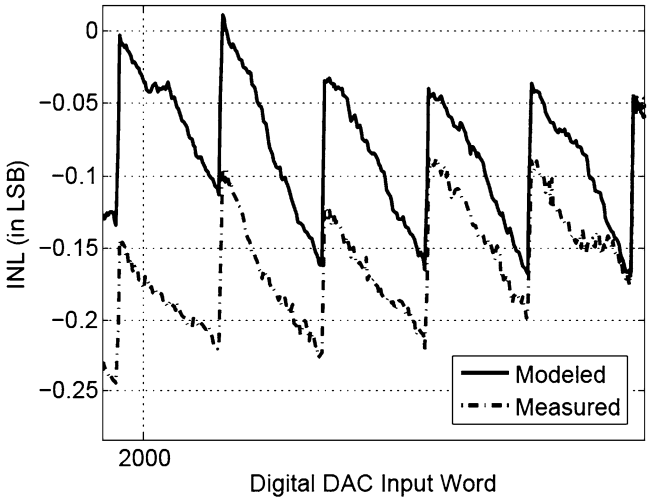


Fig. 5.14 Effects of R_C and R_W, 12-bit DAC (65 nm)

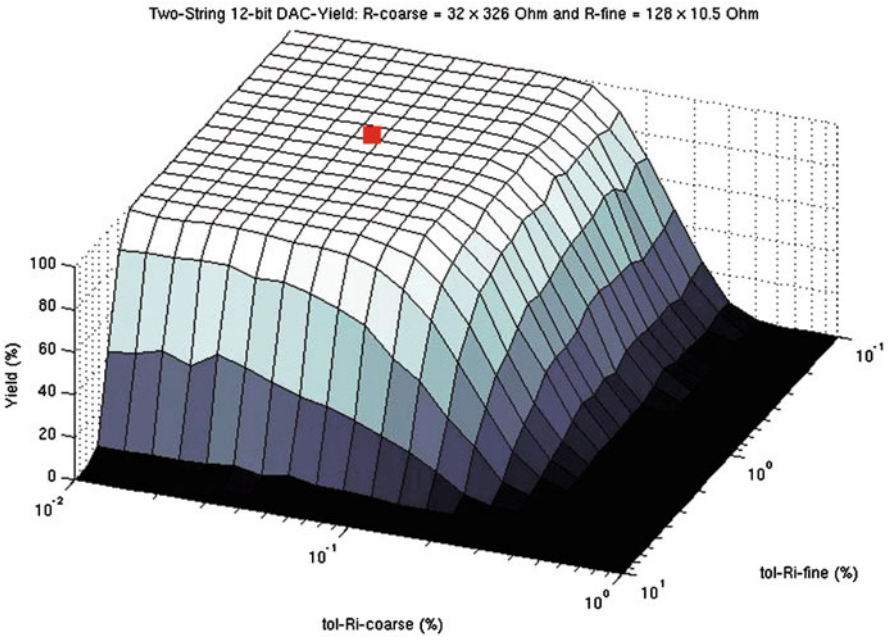


Fig. 5.15 Yield as function of $\sigma_{R-String1}$ and $\sigma_{R-String2}$

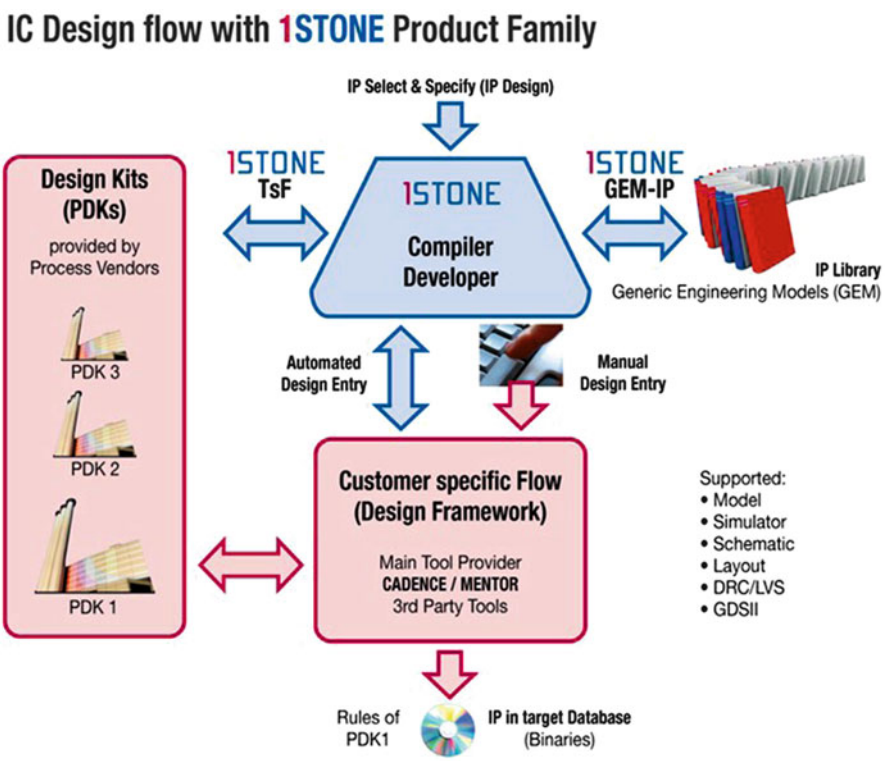


Fig. 5.16 Design flow extension that enables IP retargeting and design porting [32]

both R-Strings could be smaller without degrading the yield performance of 100%, leading to a smaller physical size of the DAC layout area. Of course, a safety margin is necessary to take all other technology-related variations into account.

5.3.4 Executable Engineering Models

The benefits of Executable Engineering Descriptions have been successfully demonstrated for different design examples recently [32]. Excellent results were achieved with respect to the increased engineering efficiency and reuse capability. A language-based design entry for analog circuit design, in addition to the established graphical one, offers the required flexibility to realize complex full custom designs, especially for the portable and configurable DCP architecture considered in this section. Main idea is to pay special attention to the IP engineering process itself instead of looking mainly to the result of it. Design parameters, PDK selections, and even design frameworks can be exchanged easily during the IP engineering process. Figure 5.16 shows the professional development platform

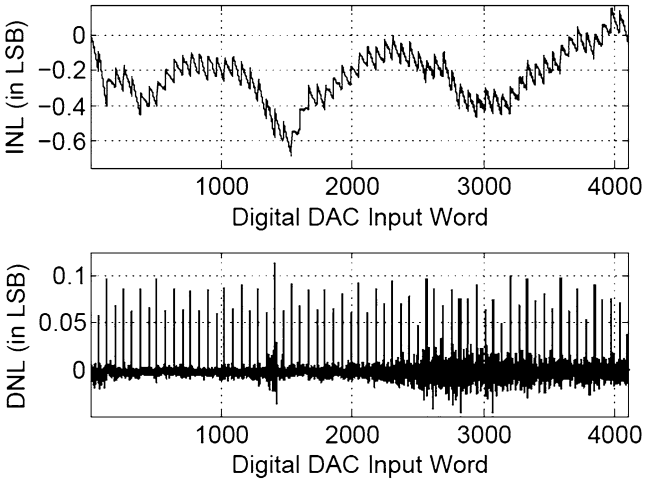


Fig. 5.17 INL and DNL of 12 bit DAC (65 nm CMOS)

1Stone® (IPGEN) as a valuable extension to a Mentor or Cadence-based design framework. The generic engineering model (GEM) design approach can be followed without exiting or shortening the already qualified design flows and by supporting the original PDKs from the process vendors. The graphical design entry stays available. A compiling process allows to execute the structured design descriptions and to compile the result into the database of the design framework. This includes schematics, layout, and testbenches. The GEM design flow supports decoupling of design and process-related data and is therefore an enabler to process portability.

1STONE® allows to organize engineering steps for hierarchical designs in an efficient, reliable manner and enables to execute it automatically. The design stays a full custom design since no common circuit synthesis takes place. Each action has to be defined, like in handcrafted design. But it can be done independently from a process technology and with a remaining high variability of the design parameters for an unlimited number of circuit hierarchies. In addition, new algorithm-based approaches can be addressed to optimize yield and reliability of the circuit design.

5.3.5 Measurement Results

Figure 5.17 shows the INL and DNL performance of a 12-bit DAC with 32-unit resistors for R-String1 using 65 nm CMOS technology [31]. The 15 measured samples show an average INL of 0.45 LSB and a DNL of 0.1 LSB.

Figure 5.18 shows the INL and DNL performance of one R-String2 subchain and the corresponding node numbers. The first and the last unit resistor of each R-String2 subchain is directly connected to the R-String1, leading to visible INL discontinuities and to sharp DNL peaks. These discontinuities are systematic errors,

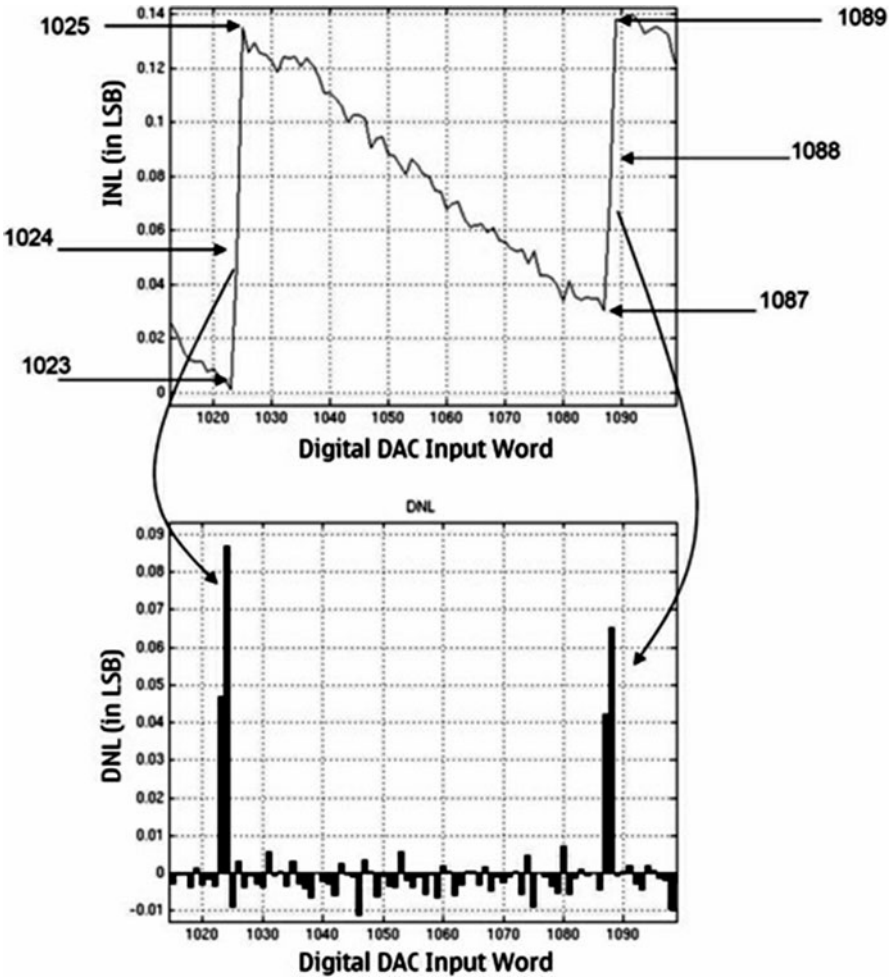


Fig. 5.18 INL and DNL of R-String2 sub-chain

being in a range of 0.1 LSB and therefore not dominating the INL performance. Nevertheless, these discontinuities are dominating the DNL performance, which is still superior, but could be even better. This optimization potential can be utilized to enable high performance potentiometer DACs with up to 16-bit resolution in near future.

Figure 5.19 shows the automatically generated layout of the 12-bit resistor chain DAC (65 nm), consisting of the resistor matrix covering R-String1 and R-String2, X- and Y-decoder logic and the additional wiring. Table 5.3 provides a summary about all three processed and measured DACs.

The tolerances σ of the unit resistors of R-String1 and R-String2 have been estimated out of the measured node voltages, assuming a constant current through

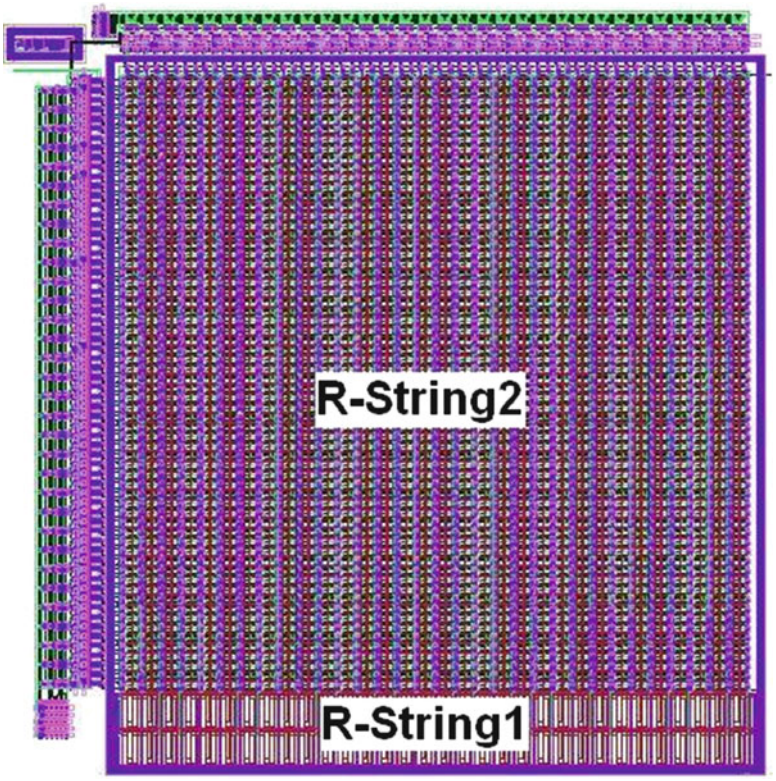


Fig. 5.19 Layout of 12bit DAC (65 nm)

R-String1 and R-String2. Comparing the DAC areas shows a factor of 4 from 12-bit to 14-bit and a factor of approx. 8 from 12 bit/65 nm to 12 bit/180 nm, which is in line with technology scaling principles. The measured standard deviation of R-String2 for the 14-bit DAC case is about 3%. This is not critical in this architecture for related INL and DNL values. Comparing the 12-bit DAC accuracy in terms of INL and DNL between 65 and 180 nm shows an accuracy improvement after downscaling, taking the LSB size into account.

To reduce the gradient effect, the design of the critical R-String1 should be made as compact as possible. Alternatively also a linear error compensated layout approach can be applied [33]. As the intensity of the DNL peaks depends on the resistor ratio between unit resistors and R_C or R_W, this degradation effect can be reduced in two ways:

- Reduction of the contact impedances at the connection point of R-String1 and R-String2 by placing a sufficient amount of contact holes in parallel.
- Reduction of the wiring impedance at the reversal point of each substring of R-String2, e.g., by increasing the width of the wire.
- Increase of the resistive value of the unit resistor elements.

Table 5.3 DAC performance summary

	Digital-to-analog-converter		
CMOS-process	65 nm		180 nm
	Vendor 1		Vendor 2
Resolution n (bit)	12	14	12
States $m = 2^n$	4,096	16,384	4,096
LSB $= V_{DD}/m$ (μV)	293	73	610
INL_{max} average	$\pm$0.446	$\pm$0.95	$\pm$0.332
DNL_{max} average	0.100	0.15	0.073
DAC-area (mm^2)	0.095	0.38	0.72
Power (μW)	665	296	740
@V_{DD}	@ 1.2 V	@ 1.2 V	@ 2.5 V
$R_{\square}$ (Ω)	8.5	9	6.5
$R_{String1}$: Number of unit resistors	32	128	32
$R_{String1}$ (Ω)	70	38.75	326
$R_{String2}$: Number of unit resistors	32×128	128×128	32×128
$R_{String2}$ (Ω)	15.2	15	10.5
σ (%) of $R_{String1}$ unit resistors	0.073	0.069	0.0358
σ (%) of $R_{String2}$ unit resistors	0.541	≈ 3	0.387

The statistical averaging principle has to be utilized in a reasonable way. It does not make sense to build unit resistors (especially of R-String2) more accurate than needed for meeting a required overall accuracy performance. One pre-condition to achieve accuracy and area-effective potentiometer DAC designs is accurate knowledge about the statistical properties of the relevant resistor material, provided by the technology vendor. The technology vendor has to characterize its process variability by using unit resistor matrices to find out realistic standard deviation behavior of resistor material in dependence of the unit resistor area. The discussed DAC topology can be very well used as testing structure for accurate process characterization of upcoming state-of-the-art nanoscale processes. The use of design automation is highly recommended: As the potentiometer DAC has a regular structure, script-driven design generation for schematic and layout can be applied efficiently. The complete layout of the DAC can be generated automatically in a few minutes using the Generic Engineering Model (GEM) approach enabling the possibility to optimize DAC accuracy by setting up the values and physical sizes of the unit resistor elements in a flexible and fast way.

5.3.6 Conclusion

We presented an approach to improve the reliability of potentiometer DACs consisting of unit resistors by using a combination of behavioral modeling and the knowledge about the statistical properties of the DAC topology. All qualitative results can be applied also to ADCs consisting of regular sets of unit elements.

The data converter reliability is optimized to effectively balance the tradeoff between DAC yield and DAC layout area, taking all relevant statistical and systematic error sources into account. This approach in combination with the generic engineering methodology gives full design flexibility and reduces the time to market significantly enabling fast creation of competitive analog circuits for mass production using latest nanoscale processes. The potentiometer DAC topology is also well suited as testing structure for accurate process characterization of upcoming state-of-the-art nanoscale processes.

References

1. Bowman, K., Duvall, S., Meindl, J.: Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration. IEEE Journal of Solid-State Circuits **37**(2), 183–190 (2002)
2. Tschanz, J., Kao, J., Narendra, S.: Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage. IEEE Journal of Solid-State Circuits **37**(11), 1396–1402 (2002)
3. Taur, Y., Ning, T.: Fundamentals of modern VLSI devices. Cambridge University Press (1998)
4. Wong, A.K.K.: Resolution Enhancement Techniques in Optical Lithography. SPIE Press (2001)
5. McCormick, C., Weber, C., Abelson, J., Gates, S.: An amorphous silicon thin film transistor fabricated at 125 C by dc reactive magnetron sputtering. Applied Physics Letters **70**, 226 (1997)
6. Liang, X., Brooks, D.: Mitigating the impact of process variations on processor register files and execution units. In: MICRO 39: Proceedings of the 39th annual ACM/IEEE International Symposium on Microarchitecture (2006)
7. Tiwari, A., Sarangi, S.R., Torrellas, J.: ReCycle: pipeline adaptation to tolerate process variation. In: ISCA '07: Proceedings of the 34th annual international symposium on Computer architecture, pp. 323–334 (2007)
8. Kheterpal, V., Rovner, V., Hersan, T.G., Motiani, D., Takegawa, Y., Strojwas, A.J., Pileggi, L.: Design methodology for ic manufacturability based on regular logic-bricks. In: DAC '05: Proceedings of the 42nd annual conference on Design automation, pp. 353–358 (2005)
9. Kobayashi, T., Sakurai, T.: Self-adjusting threshold-voltage scheme (SATS) for low-voltagehigh-speed operation. Proceedings of the IEEE Custom Integrated Circuits Conference (1994)
10. Razavi, B.: Design of Analog CMOS Integrated Circuits. McGraw-Hill (2001)
11. Oowaki, Y., Noguchi, M., Takagi, S., Takashima, D., Ono, M., Matsunaga, Y., Sunouchi, K., Kawaguchiya, H., Matsuda, S., Kamoshida, M., Fuse, T., Watanabe, S., Toriumi, A., Manabe, S., Hojo, A.: A Sub-0.1 m Circuit Design with Substrate-over-Biasing. In: Digest of Technical Papers of the Solid State Circuits Conference, pp. 88–89. IEEE INC (1998)
12. Miyazaki, M., Ono, G., Ishibashi, K.: A 1.2-GIPS/W microprocessor using speed-adaptive threshold-voltageCMOS with forward bias. IEEE Journal of Solid-State Circuits **37**(2) (2002)
13. Tachibana, F., Sato, H., Yamashita, T., Hara, H., Kitahara, T., Nomura, S., Yamane, F., Tsuboi, Y., Seki, K., Matsumoto, S., Watanabe, Y., Hamada, M.: A process variation compensation scheme using cell-based forward body-biasing circuits usable for 1.2 V design. In: CICC '08: Proceedings of the 2008 IEEE Custom Integrated Circuits Conference, pp. 29–32 (2008)
14. Narendra, S., Keshavarzi, A., Bloechel, B., Borkar, S., De, V.: Forward body bias for microprocessors in 130-nm technology generation and beyond. IEEE Journal of Solid-State Circuits **38**(5) (2003)

15. Zhao, W., Cao, Y.: New generation of predictive technology model for sub-45nm design exploration. IEEE Transactions on Electron Devices **53**(11), 2816–2823 (2006)
16. Clark, L., Hoffman, E., Miller, J., Biyani, M., Liao, L., Strazdus, S., Morrow, M., Velarde, K., Yarch, M.: An embedded 32-b microprocessor core for low-power and high-performance applications. IEEE Journal of Solid-State Circuits **36**(11), 1599–1608 (2001)
17. Ditzel, D.: Power Reduction using LongRun2 in Transmetas Efficeon Processor. In: Spring Processor Forum (2006)
18. Narendra, S., Haycock, M., Govindarajulu, V., Erraguntla, V., Wilson, H., Vangal, S., Pangal, A., Seligman, E., Nair, R., Keshavarzi, A., Bloechel, B., Dermer, G., Mooney, R., Borkar, N., Borkar, S., De, V.: 1.1 v 1 ghz communications router with on-chip body bias in 150 nm cmos. In: ISSCC '02: IEEE International Solid-State Circuits Conference Digest of Technical Papers, p. 270 (2002)
19. Borkar, S., Karnik, T., Narendra, S., Tschanz, J., Keshavarzi, A., De, V.: Parameter variations and impact on circuits and microarchitecture. In: Proceedings of the 40th conference on Design automation, pp. 338–342. ACM New York, NY, USA (2003)
20. Ono, G., Miyazaki, M.: Threshold-voltage balance for minimum supply operation [LV CMOS chips]. IEEE Journal of Solid-State Circuits **38** (2003)
21. Teodorescu, R., Nakano, J., Tiwari, A., Torrellas, J.: Mitigating parameter variation with dynamic fine-grain body biasing. In: MICRO 40: Proceedings of the 40th annual ACM/IEEE International Symposium on Microarchitecture (2007)
22. Ghosh, A., Rao, R., Kim, J., Chuang, C., Brown, R.: On-Chip Process Variation Detection using Slew-Rate Monitoring Circuit. In: Proceedings of the 21st International Conference on VLSI Design, pp. 143–149. IEEE Computer Society (2008)
23. Hazucha, P., Karnik, T., Bloechel, B.A., Parsons, C., Finan, D., Borkar, S.: Area-efficient linear regulator with ultra-fast load regulation. IEEE Journal of Solid-State Circuits **40**(4) (2005)
24. Herbert, S., Marculescu, D.: Variation-aware dynamic voltage/frequency scaling. In: High-Performance Computer Architecture (2009)
25. Hardavellas, N., Somogyi, S., Wenisch, T., Wunderlich, R., Chen, S., Kim, J., Falsafi, B., Hoe, J., Nowatzyk, A.: Simflex: A fast, accurate, flexible full-system simulation framework for performance evaluation of server architecture. ACM SIGMETRICS Performance Evaluation Review **31**(4), 34 (2004)
26. Skadron, K., Stan, M.R., Huang, W., Velusamy, S., Sankaranarayanan, K., Tarjan, D.: Temperature-aware microarchitecture. In: ISCA '03: Proceedings of the 30th annual international symposium on Computer architecture, pp. 2–13 (2003)
27. Kosakowski, M., Wittmann, R., Schardein, W., Jentschel, H.J.: Yield prediction and optimization to gain accurate devices for analog design in nonideal nanoscale processes. In: 10th International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD '08). Erfurt (2008)
28. Böker, F.: Multivariate Verfahren. Universität Göttingen, Institut für Statistik und Ökonometrie (2006)
29. Hinkley, D.V.: On the ratio of two correlated normal random variables. Imperial College, Biometrica (1969)
30. Mühlbach, G.: Repetitorium der Wahrscheinlichkeitsrechnung und Statistik. Binomi (2002)
31. Kosakowski, M., Wittmann, R., Schardein, W.: Statistical averaging based linearity optimization for resistor string DAC architectures in nanoscale processes. In: 21st Annual IEEE International SOC Conference. Newport Beach (2008)
32. Wittmann, R., Kakerow, R., Bothe, H., Schardein, W.: A multi-purpose digital controlled potentiometer IP-core for nano-scale integration. In: IP08. Grenoble (2008)
33. Shi, C., Wilson, J., Lsmail, M. Design techniques for improving intrinsic accuarcy of resistor string DHLS, IEEE international symposium on circuits and systems, 2009

Chapter 6
Conclusion

Manfred Dietrich

6.1 Application of Statistical Methods in Design

A two step process has dominated the digital design flow a long time. That means, a design was fixed and afterwards it was checked whether it is signoff clean regarding the manufacturing issues and operating conditions. The nominal values of the design parameters are chosen in a way such that the design goals are achieved with these values. Extreme deviations between actual and nominal design parameters and limits of operating conditions are described by corners. These corners characterize Process, Voltage and Temperature (PVT) limits. The signoff step of the design flow checks whether or not the design goals are violated by applying extreme values.

To carry out this step, the semiconductor foundries provide slow, typical, and fast device models. These limits define slow, typical and high speed nMOS and pMOS transistors that are used for the corner analysis. The transistor parameters are in general based on limits of the I_d saturation currents. Corners of interconnects can also be described. With these parameter sets, digital cells and cell libraries can be described for typical temperatures and supply voltages and their corners. Based on these libraries, typical PVT corners can be checked. Thus for instance, a slow process together with the lowest supply voltage at the highest temperature delivers the worst-case slow behavior in traditional CMOS designs. Assuming a fast process at highest voltage and temperature gives the maximal leakage. If all of such checks are successful the design is accepted.

Design margins assure that the temperature range in which a circuit is to be used and the fluctuations of supply voltages are considered. This approach has proven itself over many years. However, it might meet its limits if local fluctuations have

M. Dietrich (✉)
Fraunhofer Institute for Integrated Circuits IIS, Design Automation Division EAS,
Zeunerstraße 38, 01069 Dresden, Germany
e-mail: manfred.dietrich@eas.iis.fraunhofer.de

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6_6,

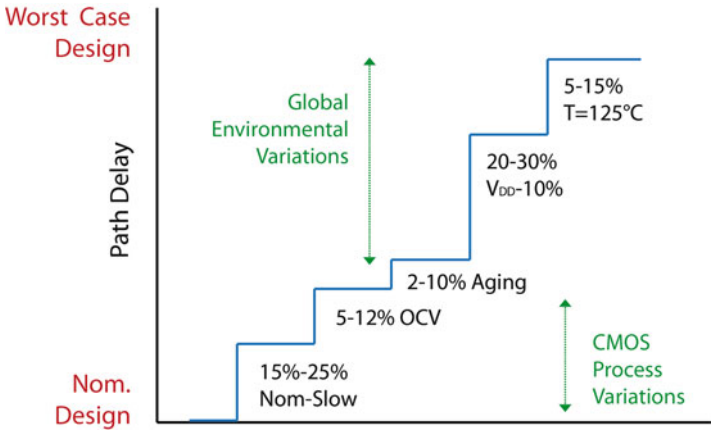


Fig. 6.1 Contributions to path delay in sub 100 nm CMOS digital circuits [3]

a greater impact than global ones and besides delay times and the dynamic power consumption of a circuit other properties such as leakage currents and slopes shall be optimized. Moreover, the demands of the manufacturing process have to be taken into account at any point of the the digital design process. The number of coners increases if different operating modes as full power, reduce power, and stand-by have to be checked.

There have been worries that in the context of the transition to smaller feature sizes the ratings obtained from corner analysis in digital design would become too pessimistic. It was expected that full statistical properties of fluctuations have to be taken into consideration as to avoid too pessimistic statements. In particular, rejecting designs that would be suitable or making insufficient use of the advances of new technologies should be avoided. Therefore, it was expected that instead of corner-based methods novel design approaches that make use of full statistical analysis techniques were necessarily going along with the transition to smaller feature sizes.

Although many academic publications emphasize a significant increase in process related fluctuation, in particular many industry publications do not report a significant increase of process variations up to the 32/28 nm technology node (see in this context for instance [3]). Rather, a constant or slightly increasing range of relative variation has been observed. A deeper understanding of the causes of variations in conjunction with improved process control and well-directed monitoring allows for mastering scaling up to the 28 nm CMOS node.

Figure 6.1 shows the influence of process and environment related variations on the path delay of clocked digital circuits. It becomes apparent that in particular operational variations such as variations of the supply voltage VDD and the temperature T have the largest portion of delay variations in such CMOS digital circuits. It seems that process fluctation do not dominate path delay variations compared to the influence of environment.

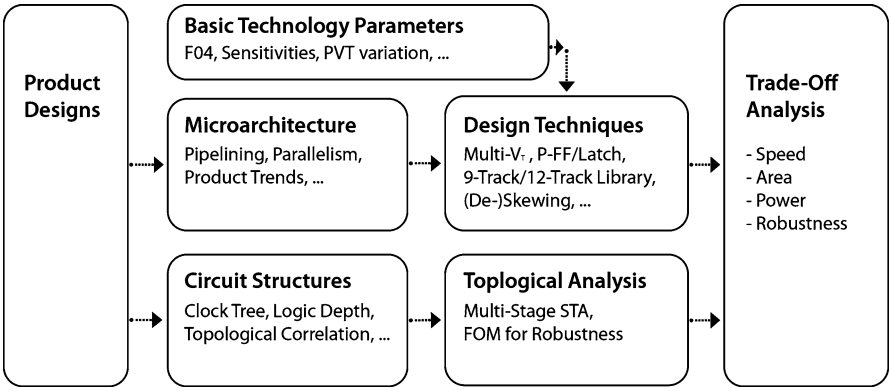


Fig. 6.2 Methodology for quantification and compensation of variation induced delay variations [3]

The process variations are not new effects in principle. When designing analog circuits they have been considered for a long time. A challenge in digital design is the accurate quantification of the scaling behavior of the variation effects. However, design techniques may help to reduce the influence of variations. This includes improvements and new techniques in digital design such as deeper pipelining and parallel super scalable processors for instance. Figure 6.2 shows various aspects in this context.

For a robust design, the interactions between technology, circuits and micro architectures have been considered. The reduction of static and dynamic power losses and the achievement of the desired performance are partially contradictory objectives if variations and reliability requirements have to be considered. Thus, a compromise has to be made.

In order to evaluate the effects of variations at the level of basic logic elements, a number of approaches has been presented in Chapter 4. The application of these approaches requires both the availability of sufficiently accurate models and also the characterization of the relevant design parameters, their nominal values and a description of their statistical properties. This is often a difficulty that should not be underestimated. The higher costs of preparing the application of statistical methods are another problem. For example, the characterization of cell libraries for the application of the Statistical Static Timing Analysis SSTA is much more expensive than providing Non-Linear Delay Model NLDM libraries.

The design and optimization of the basic components of digital circuits increasingly requires the use of sophisticated statistical analysis techniques. However, the evaluation of the system behavior is often affected by pragmatic solutions in order to keep the effort to analyze the effects of parameter variations within limits.

Thus, for instance, the efforts to use the Advanced On Cip Variation AOCV method for assessing critical path delay variations with sufficient accuracy can be significantly lower than the surplus of using SSTA instead of STA [2, 3]. However,

it must be still expected that this approach comes quickly to its limits when approaching even smaller structures. Only the boundaries have been moved to a different location than it was expected a few years ago.

After all these more or less problematic sounding remarks, a slightly more optimistic comment should be added: Process variations are a relevant challenge in design. However, they also open possible alternatives to existing procedures in some cases. The fact that the effects of independent fluctuations may cancel out each other has already been mentioned in the introductory Chap. 1. As with the joining of many individual components the overall error of an arrangement can be reduced has been shown in Sect. 5.3.

6.2 Forecast

The detection of manufacturing variations has to be outlined on different levels to develop high quality circuits in future applications. Different experts are going to reduce the influence of the variations, and several ways of handling are to be pursued.

6.2.1 Technological Development

Looking forward, the miniaturization will continue. The latest 20 nm technology node is going to be implemented quite soon. The planning for the next technology nodes has been made and first concrete ideas are considered. While the structures shrink down, new causes of variations arise. The following questions are just examples for what could be asked for:

- Will the variations sensitivity keep on increasing?
- Are new dependencies going to be detected?
- Can we disregard present dependencies?

Another important aspect is the introduction of the 450 mm-Wafer. The larger the wafer, the more difficult the consistent development on the whole face. Bending and other mechanical loadings could influence the structure. For example, gas flows are not consequently equal over the whole wafer. Tiny rotators above the wafer might affect it negatively. Thus local variances could gain importance.

Otherwise, new ideas and developments work against this process. Present procedures and equipment are being kept up to date and external influences on the production are reduced. A main step for the production is lithography, which is the most important procedure for structuring. To perform exact exposures with the 193 nm-lithography far below the wave length, the immersion lithography has been established. For the first time, ultrapure water is being used as working substance for light instead of air. In the next step, double lithography is established. By using twice

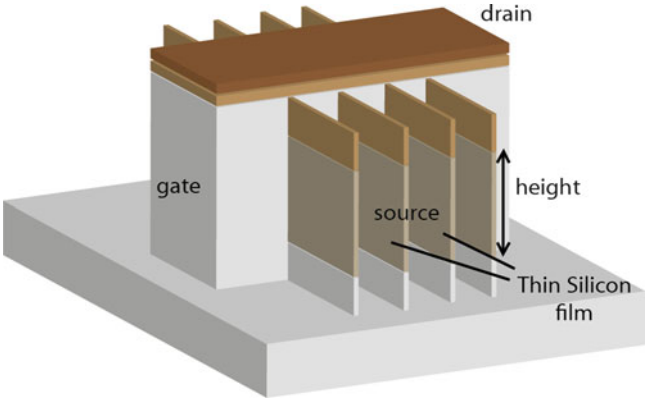


Fig. 6.3 FinFET

as much exposure in one level, the distances between the exposure areas expand whereby the interferences and exposure variances decrease. This results in exact angles and their roughness becomes minimized. The masks are improved constantly, so fewer variances are fabricated while picturing structures.

Besides, new developments are able to do qualitative switches. Concerning the lithography, it is the EUV lithography, which is near its introduction. A 13 nm wave length is used to allow a seriously advanced and exact structuring. This might cause a better angle roughness within the connections.

Using the miniaturization, the doping areas decrease. This causes a reduction of the number of doping atoms per doping area. If there are just 10–20 atoms for doping instead of several hundred, then the loss of few atoms is able to significantly influence the features and performance of the doping area. Another approach is focused on the use of new materials. Among the other things, it includes the usage of a new oxide (hafnium-oxide) for isolation. Hafnium-oxide responds less sensitive to variations of the layer thickness, because it is much more resistant to carrier tunneling. Low-key-materials decrease the capacitances between the connections and with it the influence of the roughness behavior of the connections.

6.2.2 New Devices

New active devices such as new production variations transistors will be developed to protect rightly operating of the integrated circuits. This new devices are more robust against, and the miniaturization can continue. One topic is the use of isolator materials to avoid high leakage currents. In the future, two advanced device are the favorite (Figs. 6.3 and 6.4, see also [4]).

In addition to the evolutionary development, there are some revolutionary approaches. The use of carbon with its different crystal structure delivers huge

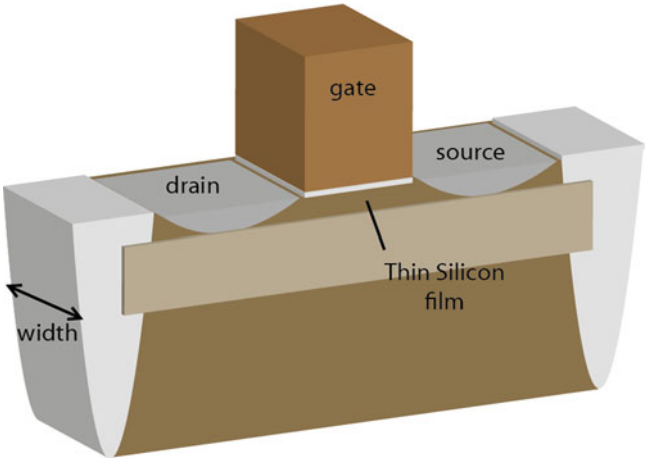


Fig. 6.4 FDSOI

number of possibilities to design new devices. Another approach is the integration of special devices such as mechanical, optical sensors (more than Moore). This approach opens the integration of big electronic systems on the chip. The placement of III/V semiconductor blocks on silicon is a possibility to generate new features and new applications.

6.2.3 New Circuits Techniques and Future Architecture

The circuit designers deliver a large input to reduce the influence of the production variations. They develop new approaches of circuit architectures. Two methods will be followed:

- Development of architectures which are robust against variations and avoid faults;
- Development of systems which can live with variations and hence resulting faults.

The use of these two approaches depends on the special application. Safety critical systems have to avoid faults and will use the first method (fault avoiding). Single mistakes during the image processing can be accepted (fault toleration). That means in the future the special requirements of the application will get more important during the design process. The designer will use techniques like:

- Usage of redundancy
- Self-repairing architectures
- Use of so-called added control bits.

The adoption of these techniques has to consider the technology possibilities and the integration of the integrated circuit in the entire system.

In addition, advanced approaches will be introduced besides the reduction of the variances, e.g., ultra low power technology. But the variation of the production will stay in consideration. New modeling and design methods have to be developed for these mentioned design approaches.

6.3 Future Tasks

The dealing with variations will involve in future design methods too. The introduction of new production methods and new technology nodes will change the approaches. It is necessary to analyze the new problems and to ask the right questions. New causes of variations can appear and already existed causes can disappear. In addition, new methods of circuit production (e.g. carbon technique, optical signal transmission on chip) will deliver a review of the well-known design methods.

Another aspect is the functions of the integrated circuits over the whole lifetime. During the operation, the internal and external influences change the characteristic of the circuit (aging of the circuit). This aging has to integrate in the design flow to get the best solution over the whole lifespan. The regard of the variations and of the reliability has to be merged to one design goal.

The short forecast demonstrates that the statistical methods will get more importance for the design of electronic circuits and systems. There will be a lot of tasks, which are waiting for a solution.

A variation aware design will play a central role in this context. Smaller structures, 3D orientation, more layers, ... - All these aspects are going to affect fluctuations of electrical and physical parameters even further. Therefore, one goal is to make the dependencies between the variations of design, component and process parameters easier to manage and better to handle. In this context, the interdependence of the variations of design parameters and performance parameters has to be captured. They have to be described in an appropriate manner, for instance with multi-dimensional composite probabilities. In addition to known sources of variations associated with further shrinking of structural widths, the effect of fluctuations due to the atomic structure of dopants has to be taken into account. The lower the average number of atoms the greater is the effect of fluctuations on the relative error. The relative error increases significantly with each missing and each needless atom.

Last but not least, it is of general interest to investigate how process-related fluctuations can be transferred to descriptions at the system level and thus affect the yield. Such methods are increasingly required for the comparison of system design alternatives, the optimization regarding contradictory design goals and the determination of safety margins for a Design for Manufacturability DFM. These requirements are also key issues that are addressed in the International Technology Roadmap for Semiconductors ITRS.

Manufacturing and designing are increasingly no longer performed within the same company. Production requirements already have to be considered in the design and they should be described using appropriate interfaces. The development of appropriate standards, such as extensions of the OpenDFM of the Design for Manufacturability Coalition DFMC of the Silicon Integration Initiative Si2 [1] is an approach that goes into this direction.

References

1. Si2 silicon integration initiative. URL http://www.si2.org
2. Advanced on-chip-variation timing analysis. Tech. rep., Incentia Design Systems Inc. (2007)
3. Technology based modeling and analyzing methods considering variations within 65nm technology node (in german). Tech. rep., TIB Hannover (2010). URL http://edok01.tib.uni-hannover.de/edoks/e01fb10/638174686l.pdf
4. J. Rabaey, *Low Power Design Essentials* Springer, Boston, MA, (2009). DOI 10.1007/978-0-387-71713-5

Appendix A
Standard Formats for Circuit Characterization

A.1 Standard Parasitic Exchange Format SPEF

Standard Parasitic Exchange Format (SPEF) is an ASCII format that represents data of wires in a chip. The latest definition of the format is given by the IEEE Std 1481–2009 [1]. It defines syntactical forms to describe resistances (*RES), capacitances (*CAP), and inductances (*INDUC) of wires that are considered as parasitics. SPEF information can also capture dependency of parasitics on technology parameters (*VARIATION_PARAMETERS). The data can be determined by layout parasitic extraction or package/board extraction tools. These tools evaluate the floorplanning information. They can consider sheet resistances of each interconnect layers and contact resistances of each via cell. Usually, capacitance tables are used to determine the area capacitances of nets to substrate, the fringe or side-wall capacitances to substrate and the cross-coupling capacitances between nets.

Based on SPEF data, parasitic information can be exchanged between different tools. A general SPEF description is structured into

- Header definition
- Name map definition
- Power and ground net definition
- External definition
- Hierarchical SPEF (entities) definition
- Process and temperature variation definition
- Internal definition

Header and internal definitions are mandatory.

The header definition describes administrative issues as used SPEF version (*SPEF), design name (*DESIGN) and others. It contains information about used lexical elements (*DIVIDER, *DELIMITER) and scaling factors and units for time (*T_UNIT) and capacitances (*C_UNIT), resistances (*R_UNIT), and inductance (*L_UNIT). The name map (*NAME_MAP) allows to map a positive integer number to a name that may be used multiple times in an SPEF file. The power

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6,

and the ground net definitions (*POWER_NETS, *GROUND_NETS) may specify references to power and ground nets.

The hierarchical SPEF definition (*DEFINE, *PDEFINE) can be used to reference entity instances within the current SPEF file. This allows merging multiple SPEF files.

The external definition defines external logical ports (*PORTS) and physical-only ports (*PHYSICAL_PORTS) of the nets that are described.

The internal definition of the nets that represent the wires can be done in detailed (*D_NET, *D_PNET) or reduced form (*R_NET, *R_PNET). The connection section (*CONN) of a detailed net describes the external (*P) and internal (*I) connection points of a net and capacitances, resistances, and inductances that built up the net. The reduced form is based on a pi-model that consists of two capacitances and a resistance (*C2_R1_C1). Distributed models are reduced to equivalent reduced models by asymptotic waveform evaluation (AWE). The reduced models represent an admittance model seen by the driving cell. The description of a reduced net requires information on the driver (*DRIVER, *CELL).

The SPEF data can be used for circuit simulation, power calculation, and crosstalk analysis for instance. In digital circuit verification, delays are determined based on these data. SPEF information can be used to do postlayout Static Timing Analysis. Rule checks and power calculation are other areas of application.

Example:

```
// Header definition

*SPEF         "IEEE 1481-2009"
*DESIGN       "Sample"
*DATE         "Monday December 18, 1995"
*VENDOR       "Sample Tool"
*PROGRAM      "Sample Generator"
*VERSION      "1.1.0"
*DESIGN_FLOW "Sample Flow"
*DIVIDER /
*DELIMITER :
*BUS_DELIMITER [ ]
*T_UNIT 1 NS
*C_UNIT 1 PF
*R_UNIT 1 OHM
*L_UNIT 1 HENRY

// Name map definition

*NAME_MAP

*1  in1
```

```
*2  out1
*3  net1
*4  net2
*5  net3
*6  net4


// External ports

*PORTS

*1 I
*2 O

// Detailed of one net description

*D_NET  *4  0.287695

*CONN
*I *5:Z  O  *D DRIVER_CELL_1
*I *6:A  I  *D DRIVER_CELL_1

*CAP
1 *5:Z      0.189802
2 *6:A      0.097893

*RES
1 *5:Z *6:A 1.054678
*END

...
```

Listing A.1 Part of a SPEF file based on [1]

The detailed net net2 (identified by *4) connects point Z of net3 with point A of net4. The parameters are determined for a connection with DRIVER_CELL_1. The letters I and O in the definition of ports and connection points of nets characterize inputs and outputs.

The variation definition (*VARIATION_PARAMETERS) is new in the IEEE Std 1481–2009 compared to older versions of the standard. It defines the process parameters that affect capacitances, inductances, and resistances of interconnects. Furthermore, first-order and second-order coefficients (CRT1, CRT2) for the temperature dependencies of resistances are defined as well as the nominal temperature for the extraction. The variation effects are described by

$$C\left(\underline{p}\right) = C_0 \cdot \frac{1+\sum_j cn_j \cdot v_j}{1+\sum_i cd_i \cdot v_i} \tag{A.1}$$

$$L\left(\underline{p}\right) = L_0 \cdot \frac{1+\sum_j ln_j \cdot v_j}{1+\sum_i ld_i \cdot v_i} \tag{A.2}$$

$$R\left(\underline{p},T\right) = R_0 \cdot \left(1+a\cdot(T-T_0)+b\cdot(T-T_0)^2\right) \cdot \frac{1+\sum_j rn_j \cdot v_j}{1+\sum_i rd_i \cdot v_i}. \tag{A.3}$$

$\underline{p}$ is a vector of process parameters. p_i is a component of this vector. T_0 is the nominal temperature for the extraction of the parameters. T is the current temperature. C_0, L_0, T_0 are capacitance, inductance, and resistance values at nominal values of the process parameters and nominal temperature.

$cn_j = \frac{1}{C_0} \cdot \frac{\partial C(\underline{p})}{\partial p_j} \cdot NF(p_j)$, $ln_j = \frac{1}{L_0} \cdot \frac{\partial L(\underline{p})}{\partial p_j} \cdot NF(p_j)$, $rn_j = \frac{1}{R_0} \cdot \frac{\partial R(\underline{p})}{\partial p_j} \cdot NF(p_j)$ are sensitivity coefficients for so-called N-type variation parameters. These coefficients characterize the numerators in the expressions that describe the parameter dependencies of capacitance, inductance, and resistance. $NF(p_j)$ is an optional normalization factor of the process parameter p_j.

$cd_i = C_0 \cdot \frac{\partial C^{-1}(\underline{p})}{\partial p_i} \cdot NF(p_i)$, $ld_i = L_0 \cdot \frac{\partial L^{-1}(\underline{p})}{\partial p_i} \cdot NF(p_i)$, $rd_i = R_0 \cdot \frac{\partial R^{-1}(\underline{p})}{\partial p_i} \cdot NF(p_i)$ are sensitivity coefficients for so-called D-type variation parameters. These coefficients characterize the denumerators in the expressions that describe the parameter dependencies of capacitance, inductance, and resistance. A process parameter is either a N type or an D-type parameter for the description of capacitances, inductances, and resistances resp. It is also possible that a variation of a process parameter does not influence neither numerator nor denumerator. Then it is called X-type variation parameter.

$a = \frac{1}{R_0} \cdot \frac{\partial R}{\partial T}$ and $b = \frac{1}{R_0} \cdot \frac{\partial^2 R}{\partial T^2}$ are sensitivity coefficients that characterize the temperature dependency of resistances (CRT1, CRT2).

The worst case variation $\Delta v_i = VC(p_i) \cdot VM(p_i)$ of a process parameter is given by its variation coefficient $VC(p_i)$ and the variation multiplier $VM(p_i)$. Nominal values of the variation parameters p_i are assumed to equal the mean values $\mu(p_i)$ of the associated probability distribution. The relation $VC(p_i) = \frac{\sigma(p_i)}{NF(p_i)}$ gives the standard deviation σ of the distribution.

Example:

The subsequent example is taken from [1]. The variation definition is given by

```
*VARIATION_PARAMETERS
0 "field_oxide_T"  D X X   0.080 1
1 "poly_T"         D X X   0.030 1
2 "poly_W"         D X X   0.023 1
3 "Diel1_T"        X X D   0.050 1
4 "metal1_T"       X N X   0.050 1
5 "metal1_W"       X N X   0.030 1
6 CRT1
```

```
7 CRT2
27.0000
```

Listing A.2 Variation parameters definition in a SPEF file [1]

The first lines characterize process parameters as thickness, width, and permittivity of dielectric layers by its parameter index i, a string and the variation parameters types (N, D, or X) for capacitance, resistance, and inductance followed by the variation coefficient ($VC(p_i)$) and the normalization factor ($NF(p_i)$). Afterward, the parameter indices for first- and second-order temperature sensitivities of resistances are given. The last value is the nominal temperature.

The characterization of capacitances, inductances, and resistances can now be extended by its sensitivities. The variation description follows after *SC. It is given by pairs of parameter index and then associated sensitivity coefficient.

```
*CAP
1 *5:Z       0.189802  *SC 0:-0.005 1:0.029 2:0.026
...

*RES
1 *5:Z *6:A 1.054678  *SC 4:0.900  5:0.531 6:0.00321
7:-0.00021
```

Listing A.3 Sensitivity description in an SPEF file based on [1]

A.2 Standard Delay Format (SDF)

The Standard Delay Format (SDF) is a standardized format to describe delay and timing information of a digital design [2]. It describes path delays, timing constraint values, interconnect delays, and high level technology parameters in a tool independent way. It is commonly used to calculate data from postlayout information. However, it can also provide constraints for a prelayout design step. One of the original intentions of this development was to provide delay information for the digital simulation, for instance, using Verilog.

The SDF file is an ASCII file. It starts with a header section where some administrative information as SDFVERSION, DESIGN name, DATE, VENDOR and the name of the PROGRAM name that created the file are given. PROCESS, VOLTAGE, and TEMPERATURE in degrees Celsius are specified. All these values do not affect the meaning of the data. Thus, this part of the header section is often used to characterize the Process-Voltage-Temperature (PVT) conditions. They are provided for documentation purposes The TIMESCALE defines the unit of timing information.

IOPATH delays characterize the input–output path delays of a device (CELL). An IOPATH is defined by an input (or bidirectional) port and an output (or bidirectional) port of a device followed by a list of delays. The delay values are given in general by triples characterizing the minimum, typical, and maximum value of a delay.

The list of delays contains either one, two, three, six, or twelve elements. Each token corresponds to a special output transition (see [2, Table 1]). The delay is applied when the output port (for IOPATH or INTERCONNECT) or in the case of two elements, the first element describes the $0 \rightarrow 1$ transition. The second element characterizes the $1 \rightarrow 0$ transition. The delay is considered at the end of the path. Furthermore, it is possible to specify delays regarding special conditions (COND) at the input port of a device. On the opposite side, all delays of a device can be described by only one list that starts with the keyword DEVICE. Delays can be given by ABSOLUTE or INCREMENT values. Incremental delays are added to existing values in the same SDF file whereas absolute values replace old values. This may be used to characterize several instances (INSTANCE) of the same CELLTYPE by describing the differences between the instances.

Interconnect delays characterize the wires between outputs and inputs of devices. They can be given as PORT delays, NETDELAYs, or INTERCONNECT delays. Port delays are considered as delays at input (or bidirectional) ports of a device. Net delays specify the delays from all drivers to all loads of a net in the same way. A more detailed description is given by INTERCONNECT delays. In this case, the first port is a driver port (output or bidirectional port) and the second port is an input (or bidirectional) port of a driven device.

Furthermore, SDF files can contain statements considering timing constraints of pulses as setup and hold times, pulse width, period and others.

Example:

```
(DELAYFILE
   (SDFVERSION "4.0")
   (DESIGN     "Test")
   (DATE       "Tue Sep 16 15:45:26 2008")
   (VENDOR     "Test vendor")
   (PROGRAM    "Test Tool")
   (VERSION     "1.0")
   (DIVIDER /)
   (VOLTAGE     1.00::1.00)
   (PROCESS     "typical")
   (TEMPERATURE 25.00::25.00)
   (TIMESCALE 1ns)
   (CELL
     (CELLTYPE "SYSTEM")
     (INSTANCE)
     (DELAY
        (ABSOLUTE
           (INTERCONNECT a0    c1/a  (0.004::0.005)
             (0.004::0.005))
           (INTERCONNECT c1/z  c2/a  (0.002::0.003)
             (0.002::0.003))
```

```
            (INTERCONNECT c2/z  b0    (0.001::0.001)
              (0.001::0.001))
          )
       )
     )

    (CELL
      (CELLTYPE "INV")
      (INSTANCE c1)
      (DELAY
         (ABSOLUTE
             (IOPATH A Z (0.143::0.167) (0.032::0.152))
          )
       )
     )

    (CELL
      (CELLTYPE "INV")
      (INSTANCE c2)
      (DELAY
         (ABSOLUTE
             (IOPATH A Z (0.129::0.157) (0.122::0.142))
          )
       )
     )

)
```

Listing A.4 Principal structure of an SDF file

The SDF information can be used to specify delay times of digital models used in VHDL or Verilog simulations. It can also be used to investigate critical paths applying STA tools.

The current version of the file format [2] does not support dependencies of delays on process parameters as well as dependencies on supply voltages of a cell. It only considers worst-case conditions. Using minimal and maximal delay times, the estimation of delays of interesting paths of a design may be either too optimistic or too pessimistic if inter-die variations have to be taken into account. The "engineering approach" to solve this problem is to provide derating tables that correct worst-case values. This approach is known as On-Chip Variation (OCV) analysis. The improved version is called Advanced On-Chip Variation (AOCV) that uses variable derating factors based on the digital levels and location of a cell [3].

A.3 Nonlinear Delay Model NLDM

The nonlinear delay model is widely used to characterize the propagation delay through digital cells and blocks to their outputs depending on the load capacitances and input rise and fall times. It also describes how output rise and fall times depend on these values (see also Sect. 4.1). NLDM was introduced in the early 1990s [4]. The model characterizes the propagation delay through a cell and the rise and fall times at the output ports.

The general model description is part of the Liberty format specification that is an industry's used library modeling standard [5, 6]. Members of the Liberty Technical Advisory Board (LTAB) are EDA vendors such as Magma Design Automation, Mentor Graphics and Synopsys and semiconductor companies such as AMD, IBM, Infineon and Texas Instruments.

The Liberty description starts with library-level attributes as, for instance, the `technology`, the `delay_model`, the units of pulling resistances, currents, time, voltages, and capacitances. The nonlinear delay model is a `table_lookup` model. Information is given concerning the operating conditions. Different library files can be created for best and worst cases. Default values for pin attributes as `default_inout_pin_cap` are provided. The `wire_load` group information may be used to estimate interconnect parameters depending on the fanout of a cell.

Furthermore, the threshold levels of the input (`input_threshold_pct_rise`, `input_threshold_pct_fall`) and output signals (`output_threshold_pct_rise`, `output_threshold_pct_fall`) are given and are used to determine delays provided by the library. The level of the threshold points that are used to determine rise and fall times is also given by `slew_lower_threshold_pct_rise` and `slew_upper_threshold_pct_rise` for a rising edge and `slew_upper_threshold_pct_fall` and `slew_lower_threshold_pct_fall`, resp.

The cell descriptions characterize electrical properties of input and output pins as well as timing conditions. The `rise_capacitance` and the `fall_capacitance` attributes allow to specify different capacitance values for input or inout pins for rising and falling waveforms, resp. The capacitances for output pins are not specified if their effects are considered in the timing information of a cell. The electrical configuration at the output (for instance, pull-up or pull-down resistances) can be described by the `driver_type` attribute. The nominal relations of the NLDM model are given by tables that characterize the delays in the case of rising and falling edges at outputs (`cell_rise`, `cell_fall`. `rise_transition` and `fall_transition` tables describe the dependencies of rising and falling times at output pins on input slopes and load capacitances. The tables allow to describe nonlinear relations by selected points.

New attributes can be created using the `define` statement. This property can be used to characterize first-order sensitivities of dependent table values on process parameters as for instance `cell_rise_sensitivity` (see also Sect. 4.5.2.2). This

offers the opportunity to consider parameter variations in the analysis. However, the accuracy is limited if only first-order sensitivities are used.

The delay information can be extended by a Nonlinear Power Model (NLPM) that was introduced in 1995 [7]. It describes the leakage power per state (`cell_leakage_power`). The power that is dissipated within a cell if the signal of an output port changes is considered as `internal_power`. Dependencies on process parameters can be taken into account by first-order sensitivities for instance (see also Sect. 4.5.2).

Examples of libraries are provided in [8].

Example:

```
library (MyCellLibrary) {

  /* general */
  technology(cmos);
  delay_model              : table_lookup;
  in_place_swap_mode       : match_footprint;
  library_features(report_delay_calculation,report_power_calculation);

  /* units */
  time_unit                : "1ns";
  leakage_power_unit       : "1nW";
  voltage_unit             : "1V";
  current_unit             : "1uA";
  pulling_resistance_unit  : "1kohm";
  capacitive_load_unit      (1,pf);
  /* internal power unit is 1pJ */

  /* operating conditions */
  nom_process              : 1.0;
  nom_temperature          : 25.0;
  nom_voltage              : 1.1;

  operating_conditions (typical) {
    process                : 1.00;
    voltage                : 1.10;
    temperature            : 25.00;
    tree_type              : balanced_tree;
  }
  default_operating_conditions : typical;

  /* thresholds */
  slew_lower_threshold_pct_fall : 30.0;
  slew_lower_threshold_pct_rise : 30.0;
  slew_upper_threshold_pct_fall : 70.0;
  slew_upper_threshold_pct_rise : 70.0;
  slew_derate_from_library      : 1.0;
  input_threshold_pct_fall      : 50.0;
  input_threshold_pct_rise      : 50.0;
  output_threshold_pct_fall     : 50.0;
  output_threshold_pct_rise     : 50.0;

  /* leakage */
  default_leakage_power_density : 0.0;
  default_cell_leakage_power    : 0.0;

  /* pin capacitances */
  default_inout_pin_cap       : 1.0;
  default_input_pin_cap       : 1.0;
  default_output_pin_cap      : 0.0;
  default_fanout_load         : 1.0;

  /* wire loads */
  default_wire_load_capacitance : 0.000177;
  default_wire_load_resistance  : 0.0036;

  power_lut_template (slp_load_pwr) {
    variable_1 : input_transition_time;
    variable_2 : total_output_net_capacitance;
    index_1 ("0.0060, 0.0200, 0.0400, 0.0800, 0.1400, 0.2800, 0.5600")
    index_2 ("0.0004, 0.0008, 0.0016, 0.0032, 0.0064, 0.0128, 0.0256")
  }

  lu_table_template (slp_load_tmg) {
    variable_1 : input_net_transition;
    variable_2 : total_output_net_capacitance;
    index_1 ("0.0060, 0.0200, 0.0400, 0.0800, 0.1400, 0.2800, 0.5600")
```

```
    index_2 ("0.0004, 0.0008, 0.0016, 0.0032, 0.0064, 0.0128, 0.0256")
  }


  /*****************************************************************
    BEGIN Additional Definitions for Process Parameters and Sensitivities   */

  power_lut_template (slp_load_pwr_sensitivity) {
    variable_1 : input_transition_time;
    variable_2 : total_output_net_capacitance;
    index_1 ("0.0060, 0.0200, 0.0400, 0.0800, 0.1400, 0.2800, 0.5600")
    index_2 ("0.0004, 0.0008, 0.0016, 0.0032, 0.0064, 0.0128, 0.0256")
  }

  lu_table_template (slp_load_tmg_sensitivity) {
    variable_1 : input_net_transition;
    variable_2 : total_output_net_capacitance;
    index_1 ("0.0060, 0.0200, 0.0400, 0.0800, 0.1400, 0.2800, 0.5600")
    index_2 ("0.0004, 0.0008, 0.0016, 0.0032, 0.0064, 0.0128, 0.0256")
  }

  define_group(cell_fall_sensitivity,       timing);
  define_group(cell_rise_sensitivity,       timing);
  define_group(fall_transition_sensitivity, timing);
  define_group(rise_transition_sensitivity, timing);
  define_group(fall_power_sensitivity,      internal_power);
  define_group(rise_power_sensitivity,      internal_power);

  define(param_name, cell_fall_sensitivity, string);
  define(values,     cell_fall_sensitivity, string);
  define(param_name, cell_rise_sensitivity, string);
  define(values,     cell_rise_sensitivity, string);
  define(param_name, fall_transition_sensitivity, string);
  define(values,     fall_transition_sensitivity, string);
  define(param_name, rise_transition_sensitivity, string);
  define(values,     rise_transition_sensitivity, string);
  define(param_name, fall_power_sensitivity, string);
  define(values,     fall_power_sensitivity, string);
  define(param_name, rise_power_sensitivity, string);
  define(values,     rise_power_sensitivity, string);

  define_group(process_parameter, library);
  define(parameter_type,     process_parameter, string);
  define(distribution_type,  process_parameter, string);
  define(nominal_value,      process_parameter, float);
  define(sigma,              process_parameter, float);

  /* process parameters used */
  process_parameter (NMOS_THKOX) {
    parameter_type    : inter_cell;
    distribution_type : normal;
    nominal_value     : 0.0;
    sigma             : 1.0;
  }

  process_parameter (NMOS_VTH) {
    parameter_type    : inter_cell;
    distribution_type : normal;
    nominal_value     : 0.0;
    sigma             : 1.0;
  }

  /* END   Additional Definitions for Process Parameters and Sensitivities
   *****************************************************************************/


cell (INV1) {
  cell_leakage_power : 2.75;

  leakage_power () {
    when        : "!A";
    value       : 2.0;
  }
  leakage_power () {
    when        : "A";
    value       : 3.5;
  }

  pin (A) {
    direction        : input;
    capacitance      : 0.00046;
    fall_capacitance : 0.00045;
    rise_capacitance : 0.00048;
    fall_capacitance_range(0.00042, 0.00054);
    rise_capacitance_range(0.00043, 0.00057);
    max_transition   : 0.56;
  }

  pin (Z) {
```

```
direction        : output;
max_capacitance  : 0.0256;
min_capacitance  : 0.0;
function         : "(!A)";

timing () {
  related_pin  : "A";
  timing_sense : negative_unate;

  cell_fall(slp_load_tmg) {
    values ("0.01041, 0.01289, 0.01753, 0.02698, 0.04588, 0.08364, 0.15914", \
            "0.01393, 0.01721, 0.02261, 0.03201, 0.05085, 0.08869, 0.16405", \
            "0.01637, 0.02087, 0.02837, 0.04026, 0.05933, 0.09693, 0.17235", \
            "0.01755, 0.02363, 0.03386, 0.05022, 0.07531, 0.11409, 0.18905", \
            "0.01514, 0.02325, 0.03705, 0.05922, 0.09361, 0.14526, 0.22330", \
            "0.00347, 0.01465, 0.03324, 0.06311, 0.10955, 0.18021, 0.28516", \
            "-0.02680, -0.01241, 0.01189, 0.05177, 0.11436, 0.21013, 0.35336");
  }
  cell_rise(slp_load_tmg) {
    values ("0.01745, 0.02223, 0.03174, 0.05069, 0.08852, 0.16404, 0.31499", \
            "0.02353, 0.02814, 0.03748, 0.05631, 0.09405, 0.16957, 0.32059", \
            "0.03144, 0.03757, 0.04776, 0.06620, 0.10366, 0.17900, 0.32979", \
            "0.04235, 0.05055, 0.06446, 0.08673, 0.12380, 0.19840, 0.34891", \
            "0.05863, 0.06915, 0.08730, 0.11734, 0.16420, 0.23887, 0.38794", \
            "0.08429, 0.09789, 0.12103, 0.15989, 0.22266, 0.31893, 0.46917", \
            "0.12912, 0.14531, 0.17433, 0.22362, 0.30471, 0.43312, 0.62828");
  }
  fall_transition(slp_load_tmg) {
    values ("0.00526, 0.00721, 0.01131, 0.01951, 0.03587, 0.06863, 0.13408", \
            "0.00815, 0.00989, 0.01242, 0.01955, 0.03590, 0.06864, 0.13404", \
            "0.01219, 0.01424, 0.01767, 0.02282, 0.03620, 0.06870, 0.13418", \
            "0.01903, 0.02161, 0.02578, 0.03285, 0.04392, 0.06965, 0.13433", \
            "0.03093, 0.03414, 0.03944, 0.04859, 0.06346, 0.08612, 0.13645", \
            "0.05212, 0.05624, 0.06353, 0.07526, 0.09402, 0.12459, 0.17059", \
            "0.09276, 0.09723, 0.10614, 0.12155, 0.14593, 0.18511, 0.24687");
  }
  rise_transition(slp_load_tmg) {
    values ("0.01144, 0.01602, 0.02496, 0.04296, 0.07869, 0.15051, 0.29392", \
            "0.01269, 0.01636, 0.02502, 0.04293, 0.07873, 0.15055, 0.29330", \
            "0.01752, 0.02062, 0.02692, 0.04298, 0.07884, 0.15062, 0.29354", \
            "0.02490, 0.02926, 0.03639, 0.04850, 0.07920, 0.15058, 0.29353", \
            "0.03628, 0.04177, 0.05146, 0.06731, 0.09168, 0.15170, 0.29424", \
            "0.05622, 0.06231, 0.07444, 0.09537, 0.12874, 0.17894, 0.29695", \
            "0.09468, 0.10007, 0.11377, 0.13889, 0.18245, 0.25140, 0.35281");
  }
  /* Additional Sensitivity Tables for Timing  */
  cell_fall_sensitivity (slp_load_tmg_sensitivity) {
    param_name : NMOS_VTH;
    values : " 0.00042152, 0.00061292, 0.00082434, 0.00123816, 0.00195448, 0.00344520, 0.00667040, \
               0.00081422, 0.00095084, 0.00108636, 0.00144650, 0.00214434, 0.00355960, 0.00667260, \
               0.00194458, 0.00219032, 0.00246400, 0.00277860, 0.00332640, 0.00454740, 0.00736340, \
               0.00353100, 0.00379060, 0.00386320, 0.00464420, 0.00524260, 0.00627000, 0.00869220, \
               0.00646360, 0.00691240, 0.00724460, 0.00771100, 0.00869880, 0.01028940, 0.01232440, \
               0.00911680, 0.00953040, 0.01027400, 0.01072060, 0.01156980, 0.01416360, 0.01634160, \
               0.01423840, 0.01553860, 0.01612600, 0.01684980, 0.01824020, 0.01956240, 0.02402400";
  }
  cell_rise_sensitivity (slp_load_tmg_sensitivity) {
    param_name : NMOS_VTH;
    values : " 0.00003597, 0.00003012, 0.00002226, 0.00001047,-0.00000152,-0.00001102,-0.00001727, \
              -0.00000290, 0.00001552, 0.00002130, 0.00002171, 0.00001459, 0.00000237,-0.00000820, \
              -0.00047982,-0.00030338,-0.00020211,-0.00011458,-0.00005896,-0.00002807,-0.00001761, \
              -0.00160600,-0.00111320,-0.00086636,-0.00059796,-0.00033418,-0.00017963,-0.00010384, \
              -0.00372900,-0.00317240,-0.00272140,-0.00203962,-0.00137918,-0.00078716,-0.00043274, \
              -0.00603460,-0.00533500,-0.00481580,-0.00389620,-0.00286220,-0.00174218,-0.00096470, \
              -0.01102860,-0.00999460,-0.00937640,-0.00815320,-0.00630300,-0.00442860,-0.00259380";
  }
  fall_transition_sensitivity (slp_load_tmg_sensitivity) {
    param_name : NMOS_VTH;
    values : " 0.00005903, 0.00013996, 0.00024926, 0.00039578, 0.00076626, 0.00143066, 0.00218460, \
               0.00004536, 0.00017648, 0.00019587, 0.00043164, 0.00073766, 0.00128304, 0.00267300, \
               0.00000655, 0.00028688, 0.00021424, 0.00014293, 0.00060676, 0.00120560, 0.00256300, \
               0.00015235, 0.00000940, 0.00013138, 0.00047872, 0.00043626, 0.00106128, 0.00258060, \
               0.00013605, 0.00010076, 0.00014863,-0.00007583, 0.00093104, 0.00073040, 0.00226820, \
               0.00002886, 0.00021371, 0.00008452, 0.00039446, 0.00053548, 0.00152218, 0.00108592, \
               0.00021074, 0.00000191,-0.00001433,-0.00004354, 0.00005460, 0.00122914, 0.00221760";
  }
  rise_transition_sensitivity (slp_load_tmg_sensitivity) {
    param_name : NMOS_VTH;
    values : "-0.00000002,-0.00000033,-0.00000080,-0.00000015,-0.00000009,-0.00000013,-0.00000032, \
               0.00000040,-0.00000148,-0.00000045, 0.00000003,-0.00000024,-0.00000020,-0.00000044, \
               0.00009423, 0.00004180, 0.00003696, 0.00001804, 0.00000135,-0.00000039,-0.00000002, \
               0.00011414, 0.00014032, 0.00010179, 0.00010369, 0.00006039, 0.00000881, 0.00000013, \
               0.00009821, 0.00025454, 0.00027346, 0.00028336, 0.00024398, 0.00013671, 0.00002798, \
               0.00006888, 0.00033220, 0.00030118, 0.00052910, 0.00034628, 0.00031262, 0.00013435, \
              -0.00032978, 0.00027896, 0.00052360, 0.00069344, 0.00080300, 0.00063184, 0.00058080";
  }

...
```

```
    }

    internal_power () {
      related_pin : "A";
      fall_power(slp_load_pwr) {
        values ("0.00025, 0.00025, 0.00026, 0.00026, 0.00026, 0.00026", \
                "0.00025, 0.00025, 0.00025, 0.00026, 0.00026, 0.00026, 0.00026", \
                "0.00028, 0.00028, 0.00027, 0.00027, 0.00026, 0.00026, 0.00026", \
                "0.00041, 0.00038, 0.00038, 0.00032, 0.00020, 0.00028, 0.00028", \
                "0.00075, 0.00058, 0.00059, 0.00049, 0.00040, 0.00036, 0.00035", \
                "0.00155, 0.00122, 0.00127, 0.00103, 0.00075, 0.00062, 0.00049", \
                "0.00322, 0.00307, 0.00258, 0.00245, 0.00194, 0.00142, 0.00105");
      }
      rise_power(slp_load_pwr) {
        values ("0.00081, 0.00082, 0.00083, 0.00084, 0.00085, 0.00087, 0.00091", \
                "0.00083, 0.00083, 0.00083, 0.00084, 0.00085, 0.00087, 0.00091", \
                "0.00088, 0.00088, 0.00087, 0.00086, 0.00086, 0.00087, 0.00091", \
                "0.00102, 0.00100, 0.00098, 0.00095, 0.00092, 0.00091, 0.00093", \
                "0.00136, 0.00131, 0.00125, 0.00118, 0.00111, 0.00104, 0.00100", \
                "0.00212, 0.00203, 0.00191, 0.00177, 0.00160, 0.00144, 0.00129", \
                "0.00377, 0.00363, 0.00343, 0.00313, 0.00279, 0.00244, 0.00210");
      }
      /* Additional Sensitivity Tables for Dynamic Power */
      fall_power_sensitivity (slp_load_pwr_sensitivity) {
        param_name : NMOS_VTH;
        values : "-0.00000651,-0.00000314,-0.00000182,-0.00000097,-0.00000051,-0.00000026,-0.00000019, \
                  -0.00001711,-0.00001179,-0.00000882,-0.00000586,-0.00000341,-0.00000183,-0.00000107, \
                  -0.00006404,-0.00005019,-0.00004244,-0.00003198,-0.00002252,-0.00001430,-0.00000796, \
                  -0.00014674,-0.00012408,-0.00010868,-0.00008862,-0.00006475,-0.00004446,-0.00002727, \
                  -0.00030833,-0.00028468,-0.00026037,-0.00022770,-0.00017974,-0.00013266,-0.00008698, \
                  -0.00047267,-0.00044638,-0.00042537,-0.00037345,-0.00031603,-0.00024046,-0.00017006, \
                  -0.00080839,-0.00078749,-0.00075537,-0.00070235,-0.00061336,-0.00050006,-0.00036784";
      }
      rise_power_sensitivity (slp_load_pwr_sensitivity) {
        param_name : NMOS_VTH;
        values : "-0.00005020,-0.00005173,-0.00005377,-0.00005647,-0.00005915,-0.00006109,-0.00006223, \
                  -0.00005887,-0.00005519,-0.00005397,-0.00005392,-0.00005544,-0.00005806,-0.00006019, \
                  -0.00010746,-0.00009569,-0.00008856,-0.00007933,-0.00007048,-0.00006470,-0.00006202, \
                  -0.00018711,-0.00016786,-0.00015543,-0.00013706,-0.00011660,-0.00009611,-0.00007959, \
                  -0.00035024,-0.00032615,-0.00030701,-0.00027775,-0.00023694,-0.00019118,-0.00014619, \
                  -0.00052173,-0.00049181,-0.00046541,-0.00042779,-0.00037543,-0.00030789,-0.00023694, \
                  -0.00085954,-0.00082379,-0.00079651,-0.00074745,-0.00067078,-0.00057354,-0.00045628";
      }

    ...

    }
  }
}
```

Listing A.5 Liberty standard cell library example with extensions for process parameter sensitivities based on [5, 9]

References

1. IEEE Standard for Integrated Circuit (IC) Open Library Architecture (OLA). IEEE STD 1481-2009 pp. 1–658 (2009)
2. IEC/IEEE Delay and Power Calculation Standards - Part 3: Standard Delay Format (SDF) for the Electronic Design Process (Adoption of IEEE Std 1497-2001). IEEE STD 1497-2004 pp. 1–94 (2004)
3. Incentia: Advanced on-chip-variation timing analysis. Tech. rep., Incentia Design Systems Inc. (2007)
4. Croix, J.F., Wong, D.F.: A fast and accurate technique to optimize characterization tables for logic synthesis. In: IEEE/ACM Design Automation Conference (DAC 97), pp. 337–340 (1997)
5. Liberty technical advisory board (TAB). http://www.opensourceliberty.org
6. Synopsys, Inc.: PrimeTime User Guide, B-2008.12 edn. (2008)
7. Trihy, R.: Addressing library creation challenges from recent liberty extensions. In: 45th ACM/IEEE Design Automation Conference, 2008. DAC 2008., pp. 474 –479 (2008)
8. Nangate: Nangate 45nm Open Cell Library (2008). URL http://www.nangate.com
9. Dietrich, M., Eichler, U., Haase, J.: Digital statistical analysis using VHDL. In: Design, Automation & Test in Europe (DATE'10), pp. 1007–1010. Dresden, Germany (2010). URL http://www.date-conference.com/proceedings/PAPERS/2010/DATE10/PDFFILES/08.1_3.PDF

Appendix B
Standard Formats for Simulation Purposes

B.1 VHDL/VHDL-AMS Statistical Analysis Package

VHDL is a well-known behavioral modeling language for digital circuits and systems. It is extended to VHDL-AMS to model also analog and mixed-signal systems. In VHDL(-AMS) applications, it becomes interesting to make Monte Carlo features available where the following requirements should be fulfilled

- Usage of the same model for nominal and Monte Carlo analysis
- Assignment of different statistical distributions that are parameterizable to each constant
- Support of continuous and discrete distributions
- Possibility to specify correlation between constants

From a practical point of view, the following points should also be mentioned

- Independent random number generation for any constant
- Reproducibility of Monte Carlo simulation within the same simulation tool

The SAE J2748 Statistical Analysis Package [1] provides VHDL-AMS functions that can be used for describing the random behavior of parameters in a VHDL/VHDL-AMS description. At the beginning of each simulation run, the parameters are initialized using random values distributed in accordance with the associated probability density or cumulative density function (PDF or CDF respectively).

The fundamental function described by the SAE J2748 standard is a random number generator STD_UNIFORM that delivers (0, 1) distributed uniform numbers. The uniform random numbers can be transformed with respect to the required distribution function. The idea behind STD_UNIFORM is to access a random number generator for uniform values as it is given in the MATH_REAL package of the IEEE library by the UNIFORM procedure. The SEED values of this procedure UNIFORM must be handled using a global storage place. Without further requirements, this place can be a read/write position in a file. The function

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6,

Table B.1 Pre-defined functions in the statistical analysis package

Function name	Comment
STD_UNIFORM	Delivers a random value between 0 and 1
STD_NORMAL	Delivers a N(0,1) distributed random value
NORMAL	Delivers a Gaussian distributed random value with given mean value and standard deviation described by tolerances of min/max limits
BERNOULLI	Delivers Bernoulli distributed random numbers
PDF	Delivers a random value described by a piecewise linear description of a PDF
CDF	Delivers a random value described by a piecewise linear description of a CDF

STD_UNIFORM reads the SEED values from this file and determines the next (0, 1) distributed value calling the UNIFORM procedure that also delivers new SEED values. These new SEED values are written to the file and are used during the next activation of STD_UNIFORM.

Thus, every VHDL-AMS/VHDL simulator that allows for multiple runs can be used for setting up standard Monte Carlo experiments. It is also possible to replace the file by handling the administration of global SEED places inside a simulator.

The package is named STATISTICS and compiled into the VHDL_UTILITY library.

Example:

As an example, one of the regular distribution functions implementing the normal (or Gaussian) distribution for type REAL is declared as follows:

```
impure function NORMAL (
   NOMINAL:  REAL;        -- Nominal value
   TOL:      REAL;        -- Tolerance > 0.0
   TRUNCATE: BOOLEAN := TRUE;
   ZSCORE:   REAL    := 3.0;
   MODE:     STAT_MODE_TYPE := STAT_MODE
   ) return REAL;
```

Listing B.1 Declaration of function NORMAL in the STATISTICS package

In nominal mode, the function returns the nominal value. In statistical mode, and if TRUNCATE is FALSE, the function returns random values with a normal distribution with mean $\mu = NOMINAL$ and standard deviation $\sigma = \frac{|\text{NOMINAL}| \cdot \text{TOL}}{\text{ZSCORE}}$ Thus, ZSCORE describes how many standard deviations correspond to the absolute tolerance of the parameter. The default is 3, which means that 99.7% of all random values returned by the function will be within the limits of the tolerance range. If the value of TRUNCATE is TRUE, the normal distribution is truncated to the interval defined by the bounds of the tolerance range.

The function can be used to assign a value to a parameter during instantiation:

```
library VHDL_UTILITY, SPICE2VHD;
use VHDL_UTILITY.STATISTICS.all;

...

C1:  entity SPICE2VHD.CAPACITOR(SPICE)
        generic map (C => NORMAL(NOMINAL => 1.0E-9, TOL => 0.15)
        port map (P => N1, N => N2);
```

Listing B.2 Instantiation of a capacitor with a normal distributed value

Example Package: STATISTICAL_CORRELATED

The real-valued one-dimensional function STD_UNIFORM of the package STATISTICS allows also the declaration of user-specific functions. In this way, correlated random numbers can be generated. In the following, the basic functions of a package STATISTICS_CORELATED are explained.

Function STD_NORMAL:

The function creates a real vector with correlated normal distributed random numbers. The same identifier as in the one-dimensional case can be used because VHDL allows to overload functions and use the correct one regarding its arguments. The correlation is given by the correlation matrix $\boldsymbol{P}$. The correlation coefficients of two random variables are defined by the covariance of the two variables divided by the product of their standard deviations (Pearson's correlation coefficient, see (2.27)).

The function checks whether its parameter matrix *CORR* fulfills the characteristics of a correlation matrix $\boldsymbol{P}$. The diagonal elements of the matrix must be 1 and the others between -1 and 1. The correlation matrix must be a symmetric matrix. However, the fulfillment of these requirements does not guarantee that the matrix is positive-semidefinite. This is an additional condition that has to be fulfilled by a correlation matrix. The function STD_NORMAL only supports positive-definite correlation matrices *CORR*. Therefore, the correlation matrix can be decomposed using a Cholesky decomposition:

$$\boldsymbol{P} = \boldsymbol{L} \cdot \boldsymbol{L}^T, \tag{B.1}$$

where $\boldsymbol{L}$ is a lower triangular matrix (see function CHOLESKY of the package body). Afterward, a vector $\underline{Z}$ that consists of uncorrelated $N(0,1)$ normal distributed variables is generated. The function STD_NORMAL of the STATISTICS package is used for this purpose. The multiplication of $\boldsymbol{L}$ and $\underline{Z}$ delivers the vector $\underline{X}$ that contains correlated standard normal distributed random variables

$$\underline{X} = \boldsymbol{L} \cdot \underline{Z}. \tag{B.2}$$

Function STD_UNIFORM:

The function creates a real vector with correlated uniform distributed random numbers. The correlation is given by the rank correlation matrix $CORR = \boldsymbol{P}'$. The elements of the rank correlation matrix are the Spearman's rank correlation coefficients.

The Spearman's correlation coefficient ρ' is often explained as being the Pearson's correlation coefficient between the ranked variables. Rank means an integer number that characterizes the position of a value in an ordered sequence of the values. The "winner" is the smallest value. Thus, the rank 1 is assigned to the smallest value ($rg(smallestvalue) = 1$). The rank of identical values (so-called "ties") is the mean of the ranks that could be given to the equal values. Let (x_i, y_i) be pairs of n given values then the Spearman's rank correlation coefficient can be determined by

$$\rho' = \frac{\sum\limits_{i=1}^{n} \left(rg(x_i) - m_{rg_x}\right) \cdot \left(rg(y_i) - m_{rg_y}\right)}{\sqrt{\sum\limits_{i=1}^{n} \left(rg(x_i) - m_{rg_x}\right)^2} \cdot \sqrt{\sum\limits_{i=1}^{n} \left(rg(y_i) - m_{rg_y}\right)^2}}, \tag{B.3}$$

where $m_{rg_x} = \frac{1}{n} \sum\limits_{i=1}^{n} rg(x_i)$ and $m_{rg_y} = \frac{1}{n} \sum\limits_{i=1}^{n} rg(y_i)$ are the mean values of the ranks of the x and y values, resp. The formula can be simplified if there are no tie ranks. Using $\sum\limits_{i=1}^{n} i = \frac{n \cdot (n+1)}{2}$ and $\sum\limits_{i=1}^{n} i^2 = \frac{n \cdot (n+1) \cdot (2n+1)}{6}$, we get

$$\rho' = 1 - \frac{6 \cdot \sum\limits_{i=1}^{n} \left(rg(x_i) - rg(y_i)\right)^2}{n \cdot (n^2 - 1)}. \tag{B.4}$$

For two normal distributed random variables, the following relation between the Spearman's rank correlation coefficient S' and the Pearson's correlation coefficient ρ is valid [2]

$$\rho = 2 \cdot \sin\left(\frac{\pi}{6} \cdot \rho'\right). \tag{B.5}$$

The transformation of Spearman's correlation coefficients to Pearson's correlation coefficients is done by the function STD_UNIFORM. The possible infrequent problem that the target matrix of this transformation might not be positive definite is not considered at this place. It is obvious that rank correlation does not change using a monoton increasing transformation $G : \mathbb{R} \longrightarrow \mathbb{R}$ between two random variables. Each cumulative distribution function $F : \mathbb{R} \longrightarrow [0,1] \subset \mathbb{R}$ is monotonically increasing. Thus, the rank correlation of two random variables X and Y is the same as the rank correlation of $F(X)$ and $F(Y)$. Therefore,

$$\rho'_{F(X),F(Y)} = \rho'_{X,Y}. \tag{B.6}$$

This leads to the implemented approach. The matrix $\boldsymbol{P}'$ with Spearman's rank correlation coefficients is transformed into a matrix $\boldsymbol{P}$ with Pearson's correlation

coefficients using (B.5). Afterward, a vector $\underline{X}$ of correlated standard normal distributed random variables $N(\underline{0}, \boldsymbol{P})$ is created using the function STD_NORMAL. The components of $\underline{X}$ are transformed to uniform distributed components of $\underline{Y}$ using the CDF of the standard normal distribution. That means

$$Y_i = \Phi(X_i). \tag{B.7}$$

Φ is approximated based on [3, Algorithm 26.2.17]. The mapping is realized by the function STD_NORMAL_CDF_ROUND of the package body.

```
library IEEE, VHDL_UTILITY;
use IEEE.MATH_REAL.all;
use VHDL_UTILITY.all;

package STATISTICS_CORRELATED is

--/**
-- Declaration of a real-valued matrix.
--*/
type REAL_MATRIX is array (NATURAL range <>, NATURAL range <>) of REAL;

--/**
-- Correlated standard normal distributed random numbers.
--*/

impure function STD_NORMAL (CORR : REAL_MATRIX)
return REAL_VECTOR;

--/**
-- Correlated uniform distributed random numbers.
--*/

impure function STD_UNIFORM (CORR : REAL_MATRIX)
return REAL_VECTOR;

end package STATISTICS_CORRELATED;
```

Listing B.3 Package header of STATISTCS_CORRELATED

```
package body STATISTICS_CORRELATED is

--/**
-- Cholesky decomposition.
--*/

function CHOLESKY (A : REAL_MATRIX)
return REAL_MATRIX is
    constant N   : INTEGER := A'LENGTH(1) - 1;
    variable SUM : REAL;
    variable L   : REAL_MATRIX (0 to N, 0 to N) := A;
begin

    for I in 0 to N loop
        for K in I to N loop
            SUM := L(I,K);
            for J in I-1 downto 0 loop
                SUM := SUM - L(K,J)*L(I,J);
            end loop;
            if I = K then
               if SUM <= 0.0 then
                  report "A not positive definite.";
               else
                  L(I,I) := SQRT(SUM);
               end if;
            else
               L(K,I) := SUM/L(I,I);
            end if;
        end loop;
    end loop;

    for I in 0 to N loop
        if I+1 <= N then
           for J in I+1 to N loop
               L(I, J) := 0.0;
           end loop;
        end if;
    end loop;

return L;
end function CHOLESKY;
```

```
--/**
-- Approximated real-valued CDF of standard normal distribution.
--*/

function STD_NORMAL_CDF_ROUND (X : REAL)
return REAL is
    constant A  : REAL                 := 1.0/SQRT(MATH_2_PI);
    constant B0 : REAL                 := 0.2316419;
    constant B  : REAL_VECTOR (1 to 5) := (
                                           0.319381530,
                                          -0.356563782,
                                           1.781477937,
                                          -1.821255978,
                                           1.330274429);
    variable T         : REAL;
    variable RESULT    : REAL;
begin
        T      := 1.0/(1.0 + B0*ABS(X));
        RESULT := B(5);
        for I in 4 downto 1 loop
            RESULT := RESULT*T + B(I);
    end loop;
    if  X >= 0.0 then
        RESULT := 1.0 - A*EXP(-X*X/2.0)*T*RESULT;
    else
        RESULT := A*EXP(-X*X/2.0)*T*RESULT;
    end if;
return RESULT;
end function STD_NORMAL_CDF_ROUND;


--/**
-- Correlated standard normal distributed random numbers.
--*/


impure function STD_NORMAL (CORR : REAL_MATRIX)
return REAL_VECTOR is
    constant CORR0             : INTEGER := CORR'LEFT(1);
    constant CORR1             : INTEGER := CORR'LEFT(2);
    constant N                 : INTEGER := CORR'LENGTH(1) - 1;
    variable SUM               : REAL;
    variable VALUE_1           : REAL;
    variable VALUE_2           : REAL;
    variable L                 : REAL_MATRIX (0 to N, 0 to N);
    variable STD_NORMAL_UNCORRELATED : REAL_VECTOR (0 to N);
    variable STD_NORMAL_CORRELATED   : REAL_VECTOR (0 to N);
begin
    assert N = CORR'LENGTH(2) - 1
      report "Matrix CORR not quadratic."
      severity ERROR;

    -- Special case (1-dimensional)

    if N = 0 then
       assert CORR(CORR0,CORR1) = 1.0
         report "In the one-dimensional case CORR must be 1.0"
         severity ERROR;

         STD_NORMAL_CORRELATED (0) := VHDL_UTILITY.STATISTICS.STD_NORMAL;
         return  STD_NORMAL_CORRELATED;
    end if;

    -- Test correlation matrix

    for I in 0 to N loop
        for J in 0 to I loop
            VALUE_1 := CORR (CORR0 + I, CORR1 + J);

            if I = J then
               if VALUE_1 /= 1.0 then
                 report "CORR(" & INTEGER'IMAGE (I) & ","
                                & INTEGER'IMAGE (J) & ") = "
                                & REAL'IMAGE(VALUE_1) & " unequal 1.0."
                 severity ERROR;
               end if;
            else
               if abs(VALUE_1) > 1.0 then
                 report "CORR coefficient not correct (|" & REAL'IMAGE(VALUE_1) & "| > 1.0)"
                 severity ERROR;
               end if;

               VALUE_2 := CORR (CORR0 + J, CORR1 + I);

               if VALUE_1 /= VALUE_2 then
                 report "CORR matrix not symmetric ["
                         & "CORR (" & INTEGER'IMAGE(I) & "," & INTEGER'IMAGE(J) &") = " & REAL'IMAGE(VALUE_1)
                           & " /= "
                         & "CORR (" & INTEGER'IMAGE(J) & "," & INTEGER'IMAGE(I) &") = " & REAL'IMAGE(VALUE_2)
```

```
                        & "]."
                  severity ERROR;
               end if;

            end if;
         end loop;

    end loop;

    -- Cholesky algorithm to determine lower tiangle matrix

    L := CHOLESKY (CORR);

    -- Test of result CORR = L * L^T required

    if NOW = 0.0 then
       for I in 0 to N loop
           for J in 0 to N loop
               SUM := 0.0;
               for K in 0 to N loop
                   SUM := SUM + L(I,K)*L(J,K);
               end loop;

               VALUE_1 := SUM;
               VALUE_2 := CORR(CORR0+I, CORR1+J);

               if abs(VALUE_1 - VALUE_2) > 1.0E-9 then
                  report "Difference in Cholesky results ["
                         & "L*Trans(L) (" & INTEGER'IMAGE(I) & "," & INTEGER'IMAGE(J) &") = "
                           & REAL'IMAGE(VALUE_1) & " /= "
                         & "CORR (" & INTEGER'IMAGE(I) & "," & INTEGER'IMAGE(J) &") = " & REAL'IMAGE(VALUE_2)
                         & "]."
                   severity WARNING;

                 report "Cholesky result is not correct."
                 severity WARNING;
               end if;
           end loop;
       end loop;
    end if;

    -- Uncorrelated STD normal distributed random values

    for I in 0 to N loop
        STD_NORMAL_UNCORRELATED (I) := VHDL_UTILITY.STATISTICS.STD_NORMAL;
    end loop;

    -- Correlated STD normal distributed random values

    for I in 0 to N loop
        SUM := 0.0;
        for J in 0 to I loop
            SUM := SUM + L (I,J)*STD_NORMAL_UNCORRELATED (J);
        end loop;
        STD_NORMAL_CORRELATED (I) := SUM;
    end loop;

return STD_NORMAL_CORRELATED;
end function STD_NORMAL;


--/**
-- Correlated uniform distributed random numbers.
--*/


impure function STD_UNIFORM (CORR : REAL_MATRIX)
return REAL_VECTOR is
    constant CORR0             : INTEGER := CORR'LEFT(1);
    constant CORR1             : INTEGER := CORR'LEFT(2);
    constant N                 : INTEGER := CORR'LENGTH(1) - 1;
    variable CORR_NORMAL       : REAL_MATRIX (0 to N, 0 to N);
    variable RESULT            : REAL_VECTOR (0 to N);
begin
    assert N = CORR'LENGTH(2) - 1
      report "Matrix CORR not quadratic."
      severity ERROR;

    -- Special case (1-dimensional)

    if N = 0 then
       assert CORR(CORR0,CORR1) = 1.0
         report "In the one-dimensional case CORR must be 1.0"
         severity ERROR;

         RESULT (0) := VHDL_UTILITY.STATISTICS.STD_UNIFORM;
         return RESULT;
    end if;
```

```
    -- Transformation of Spearman correlation matrix to Pearson correlation matrix
    -- Reason: Spearman correlation will be preserved   by strictly monoton transformation.

    for I in 0 to N loop
        for J in 0 to N loop
            if I = J then
               CORR_NORMAL (I,J) := CORR(CORR0+I, CORR1+J);
            else
               CORR_NORMAL (I,J) := 2.0*SIN(MATH_PI/6.0*CORR(CORR0+I, CORR1+J));
            end if;
        end loop;
    end loop;

    RESULT := STD_NORMAL (CORR_NORMAL);

    for I in 0 to N loop
        RESULT (I) := STD_NORMAL_CDF_ROUND(RESULT(I));
    end loop;

return RESULT;
end function STD_UNIFORM;

end package body STATISTICS_CORRELATED;
```

Listing B.4 Package body of STATISTICS_CORRELATED

B.2 Probalistic Distribution Functions in Verilog-AMS

Verilog-AMS is another behavioral modeling language to describe digital, analog, and mixed-signal circuits. The language provides built-in probabilistic distribution functions to describe the random behavior of parameters [4].

The function $random returns a 32-bit signed integer number each time it is called, 32-bit signed integers are between $-2.147.483.648$ and $2.147.483.647$. Thus, the generation of a random integer value between for example -100 and 100 is given by the following code fragment

```
integer rand_int;
...
rand_int = $random % 101;
```

$U(0,1)$ uniformly distributed random numbers can be generated by

```
real rand_real;
...
rand_real = ($random + 2147483648.0)/4294967295.0;
```

Verilog-AMS defines a set of predefined random number generator that can be used to initialize parameters. The functions are supported in digital and analog context. Table B.2 gives an overview on the real-valued versions of these function.

All these functions provide an additional string parameter. Its value can be "global" or "instance." This allows to characterize several instances of the same model considering global and local variations. If the value is "global," then in a Monte Carlo simulation run only one value is created that is used by different instances. If the value is "instance," then a new value is generated each instance that references the associated parameter.

The paramset statements in Verilog-AMS that is used in a similar manner as the model card in Spice-like simulation engines supports in this way the description of global and local varying technology parameters (see [4, Sect. 6.4.1] Paramsets statements).

Table B.2 Probabilistic distribution functions in Verilog-AMS

Function name Verilog-AMS	C function in Verilog HDL	Comment
$rdist_uniform	uniform	Delivers a uniformly random value in the interval from `start` and `end`
$rdist_normal	normal	Delivers a Gaussian distributed random value defined by `mean` value and `standard_deviation` (see (2.29))
$rdist_exponential	exponential	The PDF $f_X(x) = \frac{1}{\mu} \cdot e^{-\frac{x}{\mu}}$ for $x \geq 0$ (otherwise 0) is given by the `mean` value μ. See also notes on page 38. The exponential function is often used to characterize the time between failures.
$rdist_poisson	poisson	The function shall deliver the integer value k (here represented by a real number) with the probability $P(X = k) = \frac{\mu^k}{k!} \cdot e^{-\mu}$ with $k \geq 0$. It is defined by the `mean` value μ. In reliability analysis, it is often used to characterize the number of failures in a given time.
$rdist_chi_square	chi_square	The chi-square distribution is defined by its `degree_of_freedom` df. It is the distribution of the sum of squares of df independent standard $N(0,1)$ normal distributed random variables. It is widely used in statistical theory for hypothesis testing (see also page 34 and Sect. 4.7.2.2). The PDF of values less than 0 is 0.
$rdist_t	t	The Student's t distribution is defined by its `degree_of_freedom` df. It is the distribution of the quotient built up by a standard normal distributed random variable and the square root of a chi-square distributed random variable divided by the degrees of freedom (see also [5]). The Student's t distribution is used in statistical theory to describe confidence intervals for instance (see also page 57).
$rdist_erlang	erlang	The Erlang distribution is a special case of the gamma distribution. It can be used to describe queueing systems. Details can be found in [68].

References

1. SAE J 2748: VHDL-AMS Statistical Packages. Electronic Design Automation Statndards Committee (2006)
2. Fackler, P.L.: Generating correlated multidimensional variates. Online published: www4.ncsu.edu/~pfackler/randcorr.ps
3. Abramowitz, M., Stegun, I.A.: Handbook of mathematical functions with formulas, graphs, and mathematical tables. U.S. Govt. Print. Off., Washington (1964)
4. Verilog-AMS Language Reference Manual Version 2.3.1. Acellera Organization (2009)
5. Saucier, R.: Computer generation of statistical distributions. Tech. rep., Army Research Laboratory (2000). URL http://ftp.arl.mil/random/

Glossary

Analysis of variance (ANOVA) A method to detect significant differences between more than two samples (a generalization of the simple t-test).

Corner-case point A corner-case point is characterized by a combination of extreme values of parameters. Parameters or environmental conditions at their limits are applied to investigate the extreme behavior of a system.

Design flow Design flows are the explicit combination of electronic design automation tools to accomplish the design of an integrated circuit.

Design reuse The ability to reuse previously designed building blocks or cores on a chip for a new design as a means of meeting time-to-market or cost reduction goals.

Differential nonlinearity (DNL) The DNL is a measure to characterize the accuracy of a digital-to-analog converter. It is the maximum deviation between the analog values that belong to two consecutive digital input values and the ideal Least Significant Bit (LSB) value.

EDA Electronic design automation (also known as EDA or ECAD) is a category of software tools for designing electronic systems such as printed circuit boards and integrated circuits. The tools work together in a design flow that chip designers use to design and analyze entire semiconductor chips.

Generic engineering model (GEM) Description of a step-by-step circuit design creation process, considers different design views (specification, model, schematic, layout), executable within an EDA-Design-Framework, enables efficient design reuse principles for analog circuit design.

Integral nonlinearity (INL) The INL measures the maximum deviation between the actual output of a digital-to-analog-converter and the output of an ideal converter.

Intellectual property core (IP-Core) Reusable unit of logic, cell, or chip layout design that is the intellectual property of one party. IP cores are used as building blocks within chip designs.

IC layout Integrated circuit layout, also known IC layout, IC mask layout, or mask design, is the representation of an integrated circuit in terms of planar geometric

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6,

shapes, which correspond to the patterns of metal, oxide, or semiconductor layers that make up the components of the integrated circuit.

Monte Carlo (MC) methods/Monte Carlo simulation A random experiment, applied if an analytical description of the system seems to be hardly or not possible, (simulations of integrated circuits, e.g., transistors, library cells, chips). To investigate the behavior of a performance value y of interest, a great number n of simulations have to be made, where the process parameters x_i were changed at random following a given probability distribution.

Potentiometer A potentiometer is a three-terminal resistor with a sliding contact that forms an adjustable voltage divider. If only two terminals are used (one side and the wiper), it acts as a variable resistor.

Principal component analysis (PCA) A method to reduce the complexity. It transforms a number of possibly correlated random variables into a smaller number of uncorrelated random variables. These uncorrelated variables, called principal components. They are linear combinations of the original variables.

Response surface methods (RSM) A method to find relationships between performance and process characteristics $y = h(\underline{x})$, which allows easy predictions of y for given parameter configuration $\underline{x}$.

Statistical error Statistical or random errors are caused by unknown or unpredictable changes in parameters of a system.

Safe operating area (SOA) The SOA is defined as the region of voltages and currents or power where a device can safely operate over its lifetime without self-damage. The SOA has to consider degradation of parameters.

Systematic error Systematic errors result in deviations between expected and observed results that can be predicted.

Worst-case point The worst-case point is the parameter set of highest probability density that a parametric fault occurs under worst-case operating conditions..

Worst-case analysis Worst-Case Analysis determines, for every specification separately, the most likely process parameter set (i.e., the process parameter set "closest" to the nominal process parameter set), at which the value of the performance of interest is exactly the specification value under worst-case operating conditions.

Index

M. Dietrich and J. Haase (eds.), *Process Variations and Probabilistic Integrated Circuit Design*, DOI 10.1007/978-1-4419-6621-6,