“Accurate Power Estimation of Deep-Submicron VLSI Circuits Considering Delay Effects and Glitches”

von der Fakultät für Elektrotechnik und Informationstechnik der
Rheinisch-Westfälischen Technischen Hochschule Aachen zur Erlangung des
akademischen Grades eines
Doktors der Ingenieurwissenschaften
genehmigte Dissertation

vorgelegt von

Michael Meixner, Dipl.-Ing.
aus
Krefeld

Berichter: Universitätsprofessor Dr.-Ing. Tobias G. Noll
Universitätsprofessor Dr. sc. techn. Renato Negra
Universitätsprofessor Dr.-Ing. Tobias Gemmeke

Tag der mündlichen Prüfung: 25. Januar 2019

Contents

1 Motivation

In the early days of integrated circuits, silicon area and circuit performance were the restricting cost metrics of VLSI design. Continuing advances in CMOS technology have since lead to ever increasing transistor densities allowing for unprecedented levels of integration. The resulting challenges of power delivery and heat dissipation on the one hand as well as the rising demand for mobile computing performance on the other hand have caused the focus to shift towards power efficiency during VLSI design.

The power consumption is affected by a large number of design decisions on all levels of abstraction. Starting from architectural optimizations and operation scheduling, examples for power optimization continue down to clock or even power gating on the gate-level and device sizing on the physical level. Depending on the design method and the level of abstraction that is targeted, there are various approaches towards power estimation that allow evaluation of possible design alternatives leading to well-grounded decisions. Naturally, the more physical information of the circuit is known, the higher the confidence in the power estimation can be. At the same time, while the estimation methods allow for higher accuracy at lower levels of abstraction, the process of power estimation typically becomes more complex and therefore slower.

Due to this trade-off between estimation accuracy and complexity there is no single optimal estimation approach but rather a selection of possible methodologies suitable for a specific design stage and accuracy requirement. These might include coarse spreadsheet based power budgeting on system level during design specification on one end of the spectrum and physical level SPICE-compatible circuit simulation on the other end. Physical circuit simulation is typically the most reliable source for power estimation prior to actual measurement of fabricated silicon. It can be applied to any type of circuit without the need for pre-characterization or knowledge from previous designs since it is based on solving of differential equations governing the currents and voltages on all circuit nodes. Not surprisingly, this physical analysis results in huge sets of equations even for medium sized subcircuits. Full-chip physical simulations for current levels of integration are therefore

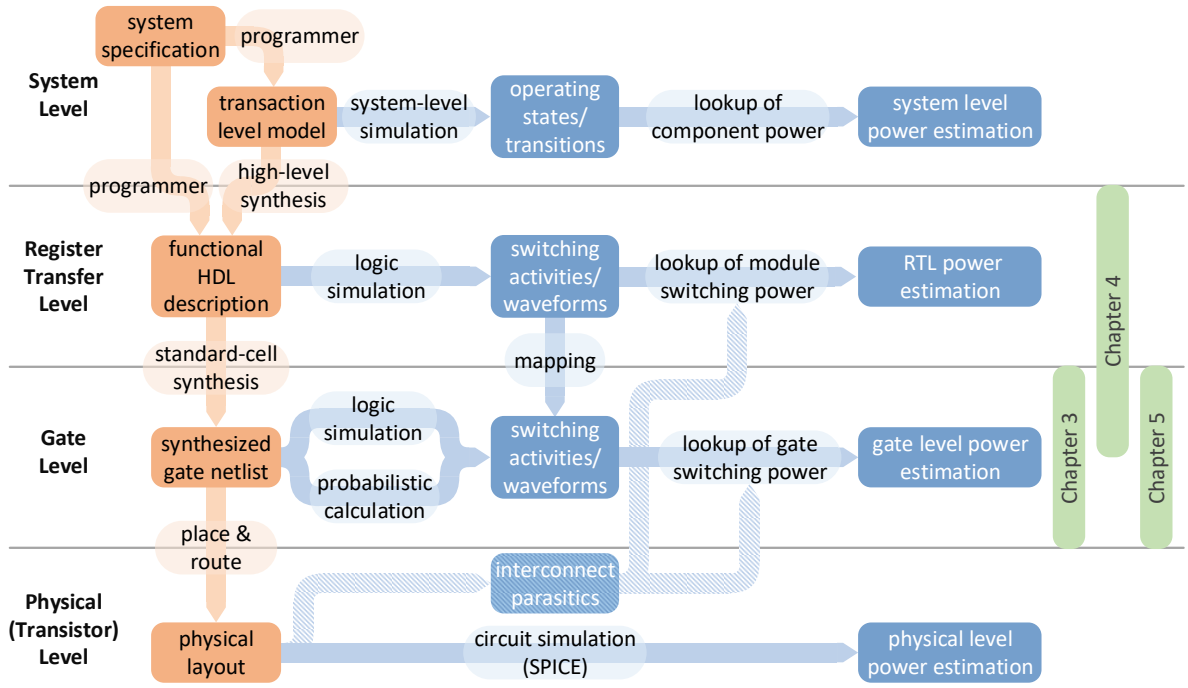


Figure 1.1: Schematic overview of power estimation methodologies on different levels of abstraction

unfeasible.

Due to the huge complexity of today's fabricated circuits, sign-off level power estimations are commonly performed on higher levels of abstraction particularly on the gate level where multiple established estimation flows exist. These analyses on the gate level are suitable in particular to top-down design flows employing logic synthesis tools because the precharacterized gate data is required for synthesis optimizations anyway and is distributed by standard cell developers. The power analysis on gate level abstracts from actual currents and voltages by simply considering switching events and boolean logic states. The related power consumption is derived from characterized lookup tables.

With increasing levels of integration even the derivation of meaningful switching activities for gate-level power estimation is becoming costly in terms of runtime. Consequently, a large number of power estimation methodologies on even higher levels of abstraction have been proposed. The general approach is based on encapsulation of larger logic blocks into macro-models that are characterized for suitable input parameters. On register transfer level possible model boundaries are function slices like adders or memories and parameters during power lookup might be statistical metrics of the input vectors. Even higher levels of abstraction might use models that estimate the power of a System-on-Chip by only considering the time each processing element spends in each operating state. An overview

of different approaches towards power estimation is shown in Fig. 1.

Apart from possible runtime benefits, power estimation approaches on higher level of abstraction also feature the advantage that power consumption can be estimated at early stages of design implementation even before detailed information is known about lower levels. This is possible because the pre-characterization that is commonly required for these methodologies allows for the transfer of knowledge about the power consumption from earlier designs to new circuits. While considering advantages of high level power estimation it is imperative to keep in mind that they are traded for a general loss in accuracy. The desired abstraction from numerous physical effects that allows for generalization and speedup leads to considerable discrepancies with actual power measurements.

This work aims at improving power estimation accuracy on selected levels of abstraction. The main focus is on including effects due to signal delays that were previously disregarded during power estimation but often contribute significantly to total power consumption. Despite the rising influence of leakage power, for most applications the main contribution to power consumption is still caused by the switching of circuit nodes. An accurate prediction of switching activities on all nodes therefore holds promise for high accuracy of power estimation. Unfortunately, due to different delay paths that lead to unaligned transitions on different inputs of gates, the gate outputs might feature multiple transitions in the same clock cycle. Depending on circuit topology, these spurious transitions, known as glitches, can form a significant contribution to the power consumption not least because once a glitch is created it might propagate through downstream gates. At the same time, accurately predicting the number and properties of these glitches on higher levels of abstraction is challenging, as they are strongly dependent on physical circuit properties.

Two novel methodologies for glitch-aware power estimation will be evaluated in this work. The first approach extends the classic macro-modeling approach on register transfer level or gate level by suitable glitch metrics that allow for the estimation of the effect on power consumption as well as the prediction of propagated glitch metrics on the macro outputs. The accurate propagation of glitch metrics is important in order to evaluate the power model iteratively across the circuit hierarchy. The second approach focuses on the gate level and demonstrates that more physical information needs to be included in the precharacterized lookup library in order to allow accurate power estimation considering real delay effects like glitches. Both approaches are aimed at exploiting the faster estimation runtimes of higher abstraction levels while improving accuracy compared to previous approaches by utilizing selected information from lower abstraction levels.

Before these proposed approaches are introduced, the terminology used throughout this work as well as previous work will be discussed in Chapter 2. Due to its prevalent usage as signoff-level reference, the accuracy that can be expected from commercial gate-level power estimation tools as well as shortcomings inherent to these estimation approaches are determined Chapter 3. Chapter 4 proposes novel glitch metrics to accurately capture glitch properties and to allow for power estimation as well as propagation of these glitch metrics by lookup of precharacterized models. An alternative approach is presented in Chapter 5 which solves inaccuracies of traditional gate level power estimation flows by substituting logic simulation with concatenation of looked-up analog waveforms and handling transition events that were previously neglected. Finally, the benefits and suitability of both propositions are evaluated in Chapter 6.

2 Introduction

When power consumption of digital circuits is to be determined, in the majority of cases the goal is the estimation of *average* power consumption. One of the few exceptions is the design of the power supply network which requires consideration for the tolerable supply voltage drop and electromigration both of which are determined by the maximum instantaneous power consumption. For important power related challenges like thermal package design or battery-constrained low power operation the maximum power consumption is of little concern because it only applies to a negligible amount of operating time. For these applications the average power consumption for a typical work load or selected input stimuli needs to be determined. Because of this focus on the mean power, many approaches in the field of power estimation are of probabilistic nature in order to capture typical behavior as will become apparent in the following chapter.

This fact is related to the problem of defining how "mean power consumption" is to be determined. Because of the strong dependence on switching activity, the characterization of a circuit macro requires the definition of typical workloads which is challenging for many applications. Changes in the operating environment often result in drastic changes in workload. On top of that, the trend of growing process variations in deep-submicron CMOS technologies also affects power consumption which will be discussed in more detail at the end of this chapter. All of these uncertainties add up to the result that a single figure for mean power consumption is only marginally valuable if no knowledge about the spread of the power consumption under expected operating conditions is available. Due to the strong dependency on workload- and variation-specific switching activity, classical corner-based characterization is not sufficient to determine this spread. Instead, detailed characterization of power consumption calls for statistical approaches like Monte-Carlo simulation and mean estimation. The requirement of analysis for differing input parameters in order to determine power variations increases the need for accelerated evaluation of power consumption. At the same time, this challenge lends itself to probabilistic approaches that are inherently capable of handling some form of variation.

Before discussing previous work related to power estimation and specifically the handling of glitch based contributions, a brief introduction to sources of power estimation of static CMOS circuits and the related terminology will be given.

2.1 Contributions to CMOS power consumption

The power consumption of a CMOS circuit can be separated into a *dynamic* and a *static* component. While the dynamic power, P_{dyn}, is caused by the loading and unloading of circuit nodes during logic evaluation of the circuit, static power, P_{stat}, is mostly independent from switching as it is caused by undesired leakage currents from supply to ground nodes that are quasi constant during one logic state:

$$P = P_{dyn} + P_{stat}. \tag{2.1}$$

The main contribution to these leakage currents in technologies below 45 nm is *subthreshold leakage*, which results from the fact that the drain current cannot be fully turned off in real transistors. *Gate leakage* which had gained importance due to shrinking gate oxide thickness resulting in direct tunneling has been reduced again by the adoption of high-k dielectric materials in recent CMOS technologies [1]. The gradual replacement of planar transistors by multigate devices at advanced technology nodes offers the promise of decreased leakage current by regaining control over the channel [2]. While the contribution of static power can be significant, its estimation complexity is relatively low because of the sole dependency on logic states.

Dynamic power consumption on the other hand depends on the switching of logic states over time which leads to both *switching power*, $P_{dyn,sw}$, when charging capacitive loads and *short circuit power*, $P_{dyn,sc}$, during the short period when both pull-up and pull-down networks of a CMOS gate are partially conducting:

$$P_{dyn} = P_{dyn,sw} + P_{dyn,sc}, \tag{2.2}$$

$$P_{dyn,sw} = \alpha \cdot f \cdot C_{load} \cdot V_{DD}^2, \tag{2.3}$$

$$P_{dyn,sc} = 2 \cdot \alpha \cdot f \cdot \overline{Q_{sc}} \cdot V_{DD}. \tag{2.4}$$

In this equation f specifies the clock frequency, $\overline{Q_{sc}}$ is the average charge that is dissipated during each switching window due to short circuit paths and α is the *activity factor* which can also be regarded as the probability that the capacitive load, C_{load}, is

charged in a clock cycle. This simplified view on dynamic power consumption abstracts from the effect of internal node capacitances inside the pull-up and pull-down networks that could be approximated by an average effective increment on the capacitive output load. Because both components of P_{dyn} are directly proportional to clock frequency it is often advantageous to consider the average energy dissipation per clock period E_{dyn} which eliminates this dependency:

$$E_{dyn} = \frac{P_{dyn}}{f} = \alpha \cdot C_{load} \cdot V_{DD}^2 + 2 \cdot \alpha \cdot \overline{Q_{sc}} \cdot V_{DD}. \tag{2.5}$$

The dynamic energy dissipation stays constant for varying operating frequencies as long as the same operations are performed and the maximum operating frequency that still results in full-swing transitions on all nodes is not exceeded. Because the static power consumption, in contrast, is only marginally influenced by the frequency of switching events, this also introduces an elegant way of separating both components. Assuming that the operating frequency can be manipulated during measurement or simulation, the power consumption of a circuit macro performing the same work load at two distinct frequencies f_1 and f_2 can be determined as

$$P_1 = E_{dyn} \cdot f_1 + P_{stat}, \tag{2.6}$$

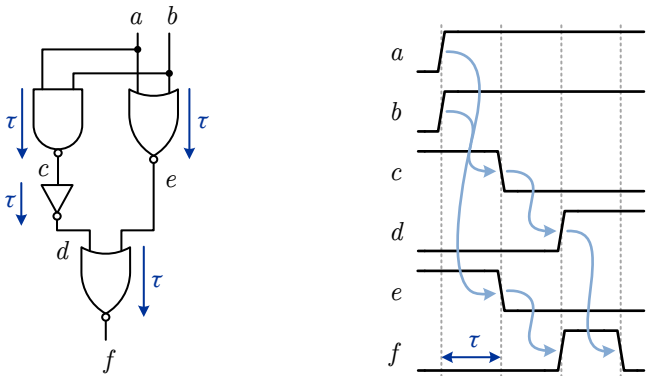
$$P_2 = E_{dyn} \cdot f_2 + P_{stat}. \tag{2.7}$$

Solving these equations for static power and dynamic energy therefore results in

$$P_{stat} = \frac{f_1 \cdot P_2 - f_2 \cdot P_1}{f_1 - f_2} \quad \text{and} \tag{2.8}$$

$$E_{dyn} = \frac{P_1 - P_2}{f_1 - f_2}. \tag{2.9}$$

Dynamic energy dissipation strongly depends on the probability of switching at individual circuit nodes which is defined by the switching activity α. This factor is generally subject to large variation because it strongly depends on the workload which is carried out by the circuit under test. A clock has an activity factor of $\alpha = 1$ caused by deterministic rising edges in each cycle while a random signal would feature $\alpha = 0.25$ resulting in charging of the load in every fourth clock cycle on average, which is equivalent to a toggling of the logic state in every second cycle. This consideration indicates that no logic signal carrying information could feature an activity $\alpha > 0.5$ because even toggling of the logic state in every clock cycle would only result in $\alpha = 0.5$. Unfortunately, for real signals



(a) circuit example (b) logic waveforms

Figure 2.1: Generation of a glitch pulse due to unbalanced paths at gate inputs.

featuring transport delays this does not need to be true, because spurious transitions due to unbalanced path delays can lead to additional switching before a signal settles to its stable state. The creation of one of these spurious pulses commonly known as glitches is demonstrated in Fig. 2.1 for a simple circuit example implementing the exclusive OR function of two inputs, a and b. For simplification, all gates are assumed to feature the unit delay of τ. When both inputs change at the same time, output f is not supposed to switch at all according to the logic function. However, due to the circuit structure in the assumed implementation, the transitions on the inputs d and e of the bottom NOR-gate are not aligned which results in a rising logic state for f before it settles to the low state again. Because of such glitches, the switching activity of a circuit node may exceed the limit of zero-delay evaluation of $\alpha = 0.5$.

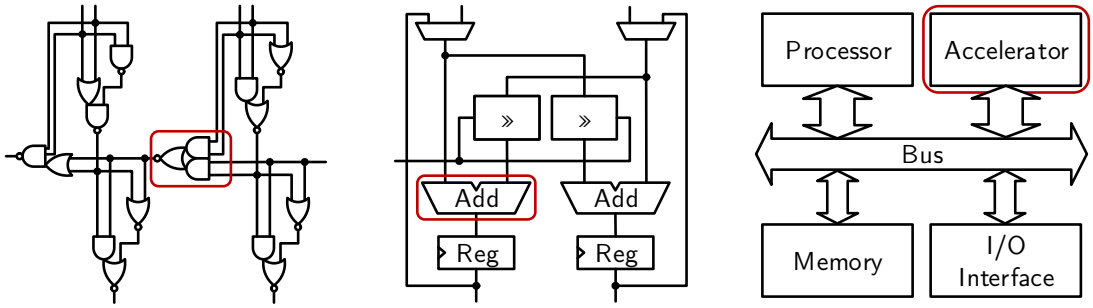
Depending on the circuit topology, glitching can be responsible for a significant amount of the power consumption ranging from 20 – 30 % for a large number of circuits but even reaching contributions of approximately 70 % of the total power for selected circuits like combinatorial adders or multipliers [3, 4]. These types of circuits typically feature topologies with a large number of gates that are connected in series, which is measured as the logic depth of a circuit. Therefore, when a glitch is created in an early state of these circuit structures it can potentially influence the switching activity of a large number of gates by propagating through the circuit hierarchy. The contribution of glitching to total power consumption is therefore becoming even more significant in designs that employ a high logic depth between register stages in order to mitigate increased variability or in applications using subthreshold operation for ultra-low power consumption due to growing delay variations [5, 6].

2.2 Power estimation methodologies

The development of both fast and accurate power estimation methodologies has been in the focus of research for more than 20 years to this date. While in the early days little to no CAD tool support was available for power estimation [7], suitable estimation tools on different levels of design abstraction have long since been published by all leading electronic design automation companies. As already stated in the motivation, there is no single optimal power estimation approach. Instead, the suitable method needs to be selected regarding the design style and required accuracy.

The most accurate way to evaluate power consumption of digital circuits prior to fabrication remains physical SPICE-compatible circuit simulation at transistor level. The need for accelerated power estimation arises due to the complexity linked to solving the systems of differential equations that are a fundamental part of physical simulation. Over the years, computing power has increased exponentially and numerous *fast-SPICE* simulators have been developed. They rely on techniques like parasitic reduction, partitioning, isomorphic matching, piecewise linearization or parallelization to deliver results that are hardly distinguishable from full SPICE accuracy for many applications despite dramatically reduced runtimes [8]. However, at the same time circuit sizes have likewise grown, following the trend for higher levels of integration, so the challenges of power estimation remain similar.

The basic principle behind virtually all accelerated power estimation approaches is to employ knowledge about the power consumption of existing circuit components to estimate the power of a yet to be realized implementation that is build from the same or similar components. A potentially costly precharacterization step is required to extract this knowledge and generate the component macromodels. However, once this power consumption data has been collected, all subsequent power estimations can profit from accelerated runtimes since evaluation of these macromodels is often as simple as a series of table lookups. In order to obtain the required accuracy, power models are usually parameterized which results in an estimation flow that is typically separated into two steps. During the first step the parameter values that apply in the analyzed implementation are determined. These values are employed in the second step to look up the power consumption for all macromodels. The granularity of the models and the relevant evaluation parameters vary in a wide range depending on the targeted abstraction level. Possible boundaries for these models on different levels of abstraction are exemplified in Fig. 2.2.



(a) gate level (b) register-transfer level (c) system level

Figure 2.2: Examples for boundaries of macromodels on different levels of abstraction.

2.2.1 Macromodels on gate level

Gate-level power models were the earliest approaches to accelerate power estimation. Traditionally, dynamic switching power, $P_{dyn,sw}$, according to (2.3) was considered the dominating contribution so the power model was essentially reduced to a switched capacitance and the goal of power estimation was an accurate estimation of the switching activity on all nodes of a gate level circuit. This could be achieved by performing logic simulation while monitoring signal states [7]. Modified simulators improved the handling of finite signal slopes [9, 10, 11, 12] as well as considering additional power contributions like short circuit power and internal gate capacitances [13]. These effects are particularly relevant for gates of higher complexity where internal power consumption can be the dominating contribution compared to charging of the output node. All of these effects can be easily captured by characterizing the power consumption related to all combinations of transitions from one set of input states to another [14]. For an N-input gate, this requires characterization of 2^{2N} transitions to be stored in a lookup table. This complexity is further increased when the effect of varying output loads and transition slopes is considered [15]. Established commercial gate-level power estimation flows used during sign-off level power estimation are still based on this characterization of transitions but are limited to events featuring the toggling of only one input in order to balance accuracy and complexity [16].

Irrespective of the specific modeling approach, the input parameters for gate-level models always include switching activity on all circuit nodes of the gate-level netlist. When logic simulations are used to determine these switching events and average power consumption is to be estimated, sequences of input vectors featuring the desired statistics need to be applied until convergence can be observed [17]. This potentially time-consuming step can be replaced by probabilistic propagation of signal statistics. Starting from signal

statistics on the circuit inputs like probabilities for signals to be in high state and probabilities for signals to switch during a specified time interval, the resulting probabilities at the remaining circuit nodes can be analytically determined in one sweep over the circuit hierarchy [18, 19]. Improvements to this propagation of signal statistics added support for handling of temporal [20, 21] and spatial correlation [22, 23] between signals.

2.2.2 Macromodels on register-transfer level

By encapsulating larger circuit blocks in macromodels, low level effects like capacitive coupling or internal timing hazards can be abstracted. Functional blocks like adders, shifters or register slices are natural boundaries for power macromodels on this abstraction level. The choice of model parameters that promise the highest accuracy are the individual states of all macro inputs before and after a switching event. Unfortunately, for typical macroblocks on register-transfer level the exponential dependency of characterization complexity on the number of input signals renders this approach infeasible. Efforts have been made to reduce the size of this characterization table by clustering of selected states assuming relatedness of neighboring inputs [24] or analyzing similarities in switching of selected internal nodes [25]. Another approach that is based on switching of individual inputs (and potentially outputs) without full characterization builds the regression model in the form of

$$\text{Power}(k) = c_0 + c_1 \cdot \Big(i_1(k-1) \oplus i_1(k)\Big) + c_2 \cdot \Big(i_2(k-1) \oplus i_2(k)\Big) + \ldots, \tag{2.10}$$

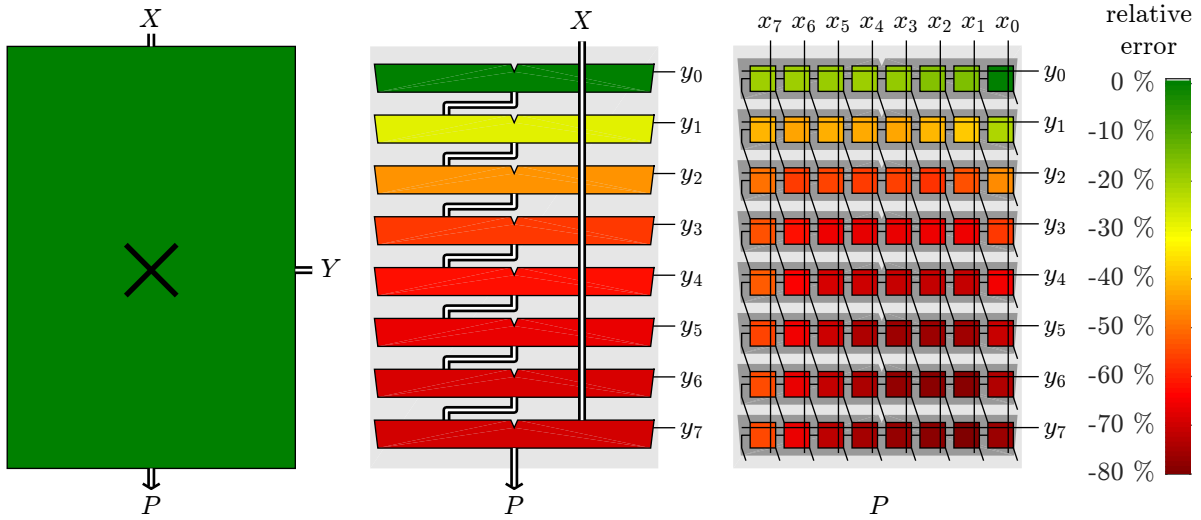
where $i_j(k)$ is the logic state of input j at time k and "$\oplus$" is the exclusive-or operation indicating a change of the logic state per input [26, 27]. The coefficients c_j can be obtained by linear regression analysis from the characterized power consumption of a representative number of switching cycles applying random input vectors. A number of refinements to this approach were proposed to automatically determine special inputs [28] or conditions [29] which select between fundamentally different operating states in order to build separate regression models for each state. Correlations between switching of multiple inputs can be captured by expanding of the regression model to include terms indicating pair- or group-wise simultaneous switching [30]. This promises a significant gain in accuracy but causes the number of regression coefficients to increase exponentially. A reduction in the number of parameters can be achieved by employing signal statistics that are averaged over selected inputs. Typical statistical parameters are the

average probability of all inputs to be in high logic state and the averaged probability for switching on inputs or outputs in a specified time interval [31, 32, 33]. Due to the reduced number of parameters, a full characterization of selected input vector streams featuring all desired statistics can be performed, which allows for direct table lookup possibly involving interpolation during estimation. Additional dimensions can be added to the lookup table to consider supplementary statistics like correlation between bits [34]. Due to the averaging of switching statistics over all input bits, information on differing behavior of individual bits is discarded, which might degrade estimation accuracy. The observation that input vectors in arithmetic circuits often show strongly correlated behavior for the most-significant bits and quasirandom switching for the lowest bits motivated an approach that builds separate models for lower and upper bit weights of word-level models [35]. The boundary between both bit types can be calculated from word-level statistics like mean, variance and autocorrelation of the input vectors [36]. Alternatively, these characteristics can directly be used as model parameters for components featuring inputs that have associated word-level values [37].

Evaluation of these models can be accelerated by employing simulations on the functional level instead of relying on gate-level simulators or probabilistic propagation. Selected high-level metrics can even be propagated throughout the circuit hierarchy by building of additional lookup tables [38, 37].

2.2.3 Macromodels on system level

On system level a model boundary might enclose whole submodules that are simulated using transaction level modeling. Due to the limited visibility of internal functionality, especially when components are implemented as black-boxes, possible input parameters during estimation are the time spent in coarse functional states (e.g. *idle*/*active*) [39, 40, 41] or the number of accesses on the outside ports [42, 43, 44]. Without knowledge of the implementation, power consumption in different *power states* can be characterized by measurement or from datasheets. During estimation, a transaction-level simulation that determines the state of the system component by monitoring its inputs and outputs can be performed which allows for accumulation of power consumption. When more detailed information on a module is available, the state of selected internal registers or signals indicating a specific event can be included in the parameter space [45, 46]. Particularly for processor cores, a multitude of specialized power models were proposed. They rely on additional parameters like instruction traces [47, 48, 49], cache hits/misses [50, 51] or



(a) full multiplier array (b) separate multiplier rows (c) separate multiplier cells

Figure 2.3: Error of zero-delay characterization of dynamic energy dissipation due to glitches inside an 8x8 array multiplier for different choices of component boundaries.

pipeline hazards [52].

2.3 Power estimation considering real delay effects and glitches

The correct handling of glitches during power estimation is challenging because glitching is highly dependent on accurate gate delays [53] which are in turn defined by physical parameters like parasitic capacitances or cross-coupling. The power consumption caused by glitches that are generated inside of a macromodel is automatically captured during characterization. Power consumption due to propagation of glitches which appear on the inputs of the circuit macro on the other hand needs to be estimated accordingly. Following this reasoning, it would be advantageous with respect to errors due to glitching to choose the boundaries of macromodels as large as possible, preferable as large as full pipeline states to guarantee glitch-free inputs. However, macros of this size might not be generic enough to be reusable in a large number of power estimations. On top of that, model parameters that still result in feasible characterization effort become increasingly inaccurate with growing macro size as discussed in the previous section.

Consequently, if glitch-free operation cannot be guaranteed, macromodels need to consider occurrence of glitches on the macro inputs. This need to address glitching on model

inputs is demonstrated in Fig. 2.3 for different choices of model boundaries when regarding an 8x8 array multiplier macro based on carry-ripple adders where the inputs, X and Y, are assumed to be glitch-free. This example was specifically chosen due to the high degree of glitching inside the regular array structure. Nevertheless, characterization of the whole multiplier array using glitch-free stimuli for both inputs results in perfect power estimation because the glitches that appear inside the macro are inherently captured during the low-level characterization simulation. In contrast, when all constituting adder rows inside the multiplier macro are stimulated separately using the same zero-delay stimuli that would result from the original input vectors, a growing error of the dynamic energy dissipation can be observed starting at the second adder row. When glitches that appear at the output of the first adder row are neglected while determining the switching activity at the inputs of the second row, the dynamic energy dissipation of the second row is underestimated by 28 %. The error in following adder rows is even larger because a certain number of glitches at the input of each row propagates to the output resulting in an additional increase in switching. Because of the dominating contribution of glitching to the overall power consumption for the analyzed multiplier macro, the total dynamic energy is underestimated by 57 % compared to the simulation of the full macro. While this error seems excessive, at least the effect of glitches that are generated inside the carry paths of the adder rows is captured when characterizing functional slices like adders. When the model granularity is increased further down to the gate-level not even these glitches are observed when employing zero-delay switching activity. This results in an overall error of 64 % for the estimation of dynamic energy.

Early gate-level estimation approaches that recognized the importance of glitching for power estimation focused on the use of accurate gate delays during simulative evaluation of switching activities in order to capture the increase in state toggling [7]. Limited output slew of circuit components in combination with nonuniform gate delays lead to attenuation of glitch pulses that are shorter in time than a gate-dependent threshold during propagation through the circuit hierarchy. To consider this glitch filtering characteristic of circuit gates, postprocessing of logic simulator results [12, 10] or custom event based simulators considering the finite signal transition times [9, 11, 15] were proposed. The thresholds for propagated pulse widths or load dependent transition times as well as power consumption due to partial-swing glitch pulses need to be determined by characterization or estimated from technology features. If probabilistic analysis is employed to analyze switching activities instead of logic simulation, similar considerations lead to concepts like

probability waveforms that consider gate delays during propagation of switching probabilities and consequently allow for filtering of non propagating glitch pulses [22, 6, 54, 23]. Alternatively, empirical correction factors that are calculated from topology properties or input vector characteristics have been proposed which are used as weighting factors during calculation of power consumption for individual gates [55, 56]. All of the above approaches solely focus on the correct analysis of switching activities which only form the input parameters to the required model lookup translating the switching events into power consumption. Due to the prevailing digital viewpoint, filtering of pulses is limited to a binary decision of propagating or blocking. When they are considered at all, partial-swing transitions are only determined in a postprocessing step to derive scaling factors to the power consumption of the related full-swing transitions.

By increasing the abstraction of macromodels to register-transfer level, the inherent consideration of glitches internal to the model boundary might appear to lessen the need for accurate handling of glitches. However, pure functional simulation which is commonly employed at this level of abstraction to accelerate model evaluation offers no possibility for propagation of glitches because only cycle-accurate timing can be observed. Instead, handling and propagation of metrics determining the glitching activity on input signals to the macromodel must be performed by means of additional lookup tables. The metric chosen by Raghunathan et al. [37], denoted *glitching activity*, defines the rate of glitch pulses appearing on the inputs of circuit components without specifying their shape or the time of their appearance. Liu and Papaefthymiou introduce an additional parameter which specifies both the pulse width of each glitch as well as the spacing between glitch pulses in case of multiple glitches [57]. However, partial swing glitches are not considered in their model and all glitches are assumed to occur synchronously at the beginning of each clock cycle.

Accurate estimation of power consumption due to glitching is clearly more challenging than estimation of functional switching power. On top of zero-delay functional simulations, interrelated delay effects which are defined by the physical level of the design need to be taken into account. To determine the state of the art concerning power estimation intended to capture glitching power, chapter 3 will evaluate established gate-level power estimation approaches. The insights gained regarding potential shortcomings related to these methodologies will serve as a motivation for novel estimation techniques proposed in subsequent chapters.

2.4 Power consumption and variability

Because of the direct dependency of the threshold voltage on leakage currents, static power consumption is strongly affected by process and temperature variations. Dynamic power consumption due to functional circuit evaluation is less susceptible to process variations or changes in temperature when the dominating contribution can be assumed to be the charging of capacitances. However, when glitches are considered, the influence of variability on dynamic power consumption becomes less predictable. This is due to the fact that the generation and propagation of glitches is determined by the delays of individual circuit components which in term might vary considerably due to local or global variation.

With the notable exception of the introduction of multigate structures, the continuing technology scaling is accompanied by increasing variability. Therefore, this influence of process variations on dynamic power consumption forms a considerable challenge for suitable estimation methodologies.

An early method for calculating coarse bounds of the dynamic energy dissipation when variation cannot be neglected calculates *ambiguity intervals* at the outputs of gates based on minimum and maximum delays of all gates during which the logic state might vary depending on delay variations [58]. The lower bound of the dynamic energy dissipation is derived from a gate level estimator supplied with switching activities under the assumption that there is at most one transition during each ambiguity interval. In contrast, for the estimation of the upper energy bound the maximum number of full-swing transitions which fits inside the ambiguity interval when spaced by at least the gate delay is assumed. This simplistic method assumes uniformly distributed gate delays and allows no conclusions regarding the expected distribution of dynamic energy inside the derived bounds. Higher accuracy for the estimation of switching activity variations can be achieved by extending the concept of probabilistic transition waveforms that list possible events on circuit nodes with a given probability. Each event is not only annotated by its (mean) occurrence time but also by the standard deviation assuming normally distributed gate delays [54]. The delay and energy dissipation of gates including variation is derived from a precharacterization employing Monte-Carlo simulations. The effect of spatial correlation of parameters related to dynamic power consumption can be considered in order to improve estimation accuracy [23]. In order to limit the estimation complexity, no information about switching direction or intersignal correlation is considered for this approach. A fundamentally different approach proposes to employ Monte-Carlo Analyses based on fast

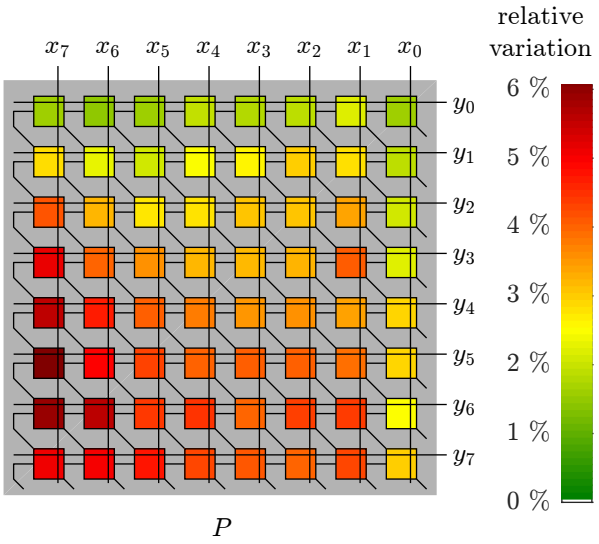


Figure 2.4: Variation of energy dissipation per cell measured as 1σ quantile bounds relative to mean energy dissipation per cell for an 8×8 unsigned array multiplier.

gate-level simulation by annotating precharacterized lookup tables for delays, slopes and switching energies as well as their sensitivities regarding selected variation parameters to the descriptions of basic cells [15]. During elaboration phase all instances of each basic cell are assigned an updated set of lookup tables according to the random variation parameters. Partial-swing glitch pulses are considered in a postprocessing step by deriving pulse heights from the assumed linear signal slopes.

The effect of delay variation on energy dissipation is demonstrated by a Monte-Carlo analysis of an array multiplier which exhibits both a high logic depth and a large number of reconverging delay paths. For 1000 circuit simulations both global and local variation parameters are varied for the same stream of input operands. The variation per multiplier cell is illustrated in Fig. 2.4 by comparing the relative deviation of the lower or upper 1σ quantile bounds from the mean dynamic energy dissipation per cell. This variation metric specifies the minimum deviation interval around the mean which contains 68.3 % of energy dissipation results. The dependency of the observed variation on delay variations becomes apparent in the increasing relative variation of the energy dissipation towards the lower left corner of the array. This effect can be explained by the growing number of possible delay paths terminating at these multiplier cells. In contrast, static effects like varying load capacitances or short circuit currents during switching affect all multiplier cells in the same way. Due to averaging across the multiplier array the 1σ bound of the total dynamic energy dissipation deviates from the nominal dynamic energy by only 3 %. The maximum total dynamic energy dissipation for the full array recorded during

Monte-Carlo analysis is 7.3 % higher than the analysis result employing nominal models.

This result which was obtained without consideration for varying operating voltage and temperature demonstrates the impact of variations of the propagation delays on dynamic energy dissipation. The effect is exacerbated by applications which adopt subthreshold operation in order to increase energy efficiency because of increased delay variations at these operating points. As a consequence, in addition to the discussion of the accuracy at the nominal operating point the following discussion of power estimation methodologies needs to evaluate the ability of each approach to consider and report variation effects.

However, the largest variations in dynamic energy dissipation are caused by the choice of input stimuli that are applied to the circuit under test. The input stimuli for the demonstration of variation effects were sampled from uncorrelated uniform distributions spanning the full wordlength of eight bit. This results in independent toggling of each input bit in every second cycle on average. This assumption might be far from the workload that could actually be observed during functional operation of a certain multiplier macro. Moderate modifications of the input statistics can result in significant deviations in energy dissipation. For example, the reduction of the switching activity for input Y by a factor of two decreases dynamic energy dissipation of the multiplier macro by 10 %. In many applications it is hardly possible to specify a single stream of input stimuli that captures typical circuit behaviour. In that case, the stimuli-dependend variation of energy dissipation should be considered in order to derive well-grounded design decisions.

3 Limits of Commercial Gate Level Power Estimation

Due to ever increasing levels of integration physical circuit simulation has long since ceased to be able to support full-chip power estimations. As an alternative, gate level estimation flows that fit well into top-down design methodologies employing standard cell synthesis have been developed. A number of commercial tools from established vendors in the field of electronic design automation on this level of abstraction have reached a reputation for accuracy that allows their usage as sign-off level estimators. In fact, many proposals for power estimation methodologies on higher levels employ these gate-level estimations as a reference to compare against [49, 59, 60]. Because of their importance in digital circuit design in combination with the lack of comparative studies targeting the accuracy to expect, the following discussion will focus on the work flow of commercial gate-level power estimation tools and possible sources of errors related to them. This in turn will help to identify the requirements for improved power estimation approaches that will be proposed in subsequent chapters.

Gate-level power estimation as performed by commercial state-of-the-art tools typically operates by accumulating precharacterized energy values associated with single-signal switching on individual gate inputs. As discussed previously, the information on signal switching on all gate inputs that is required for this approach can either be determined by probabilistic propagation of switching activities throughout the circuit or by logic simulation using the gate-level netlist and a testbench supplying suitable input stimuli. In order to be able to consider delay effects like glitches on the input of gates, the evaluation of switching activity needs to take realistic gate delays into account. These gate delays, that are strongly dependent on the output loads of the individual gates, can for example be estimated by static timing analysis (STA) and annotated to the gate-level netlist during logic simulation or analytical propagation of switching probabilities. This standard work flow for gate-level power estimation is shown in Fig. 3.1.

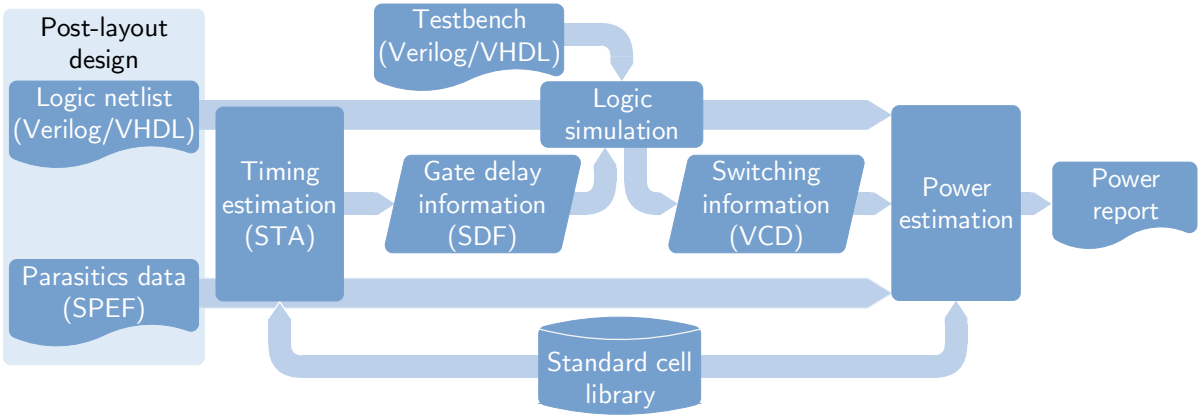


Figure 3.1: Standard gate-level power estimation flow.

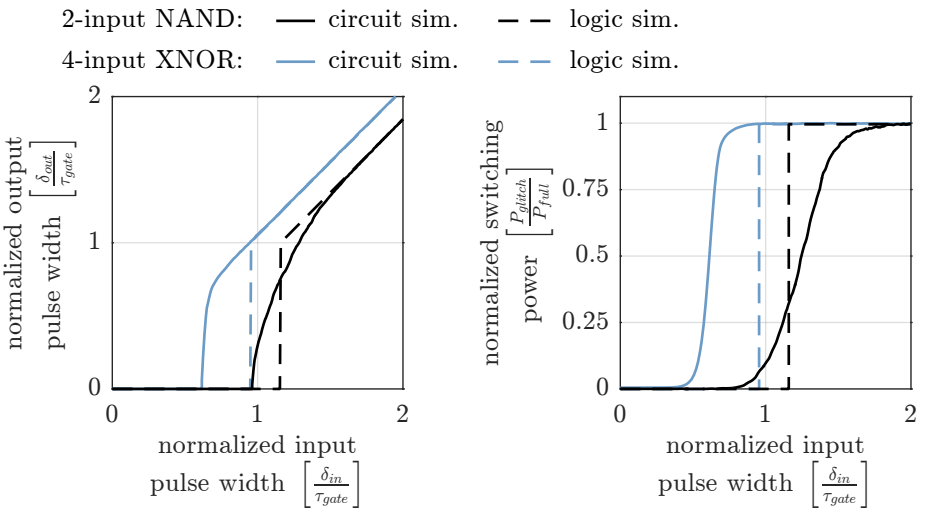
The depicted situation is that of postlayout power estimation where information from the physical level can be annotated to the gate-level view in the form of parasitic capacitances of interconnects. While gate-level estimation can also be performed at prelayout design stages, the accuracy of power consumption estimations will deteriorate accordingly.

While the standard use model of commercial gate-level power estimators is focused on classic corner-based design, the effect of process variation could be incorporated by a large number of repeated evaluations of the power consumption for varying gate delays. The required information on delay variation could for example be determined by statistical static timing analysis (SSTA). However, since the focus of the chapter is the evaluation of the overall accuracy of state-of-the-art power estimators for specific operating conditions and workloads, the effect of variation is not considered in the following discussion. The potential sources of error which will be discussed remain the same for repeated evaluation of the work flow.

Deviations of the estimated power consumption compared to physical-level power simulations might be introduced by any of the steps involved in gate-level power estimation as depicted in Fig. 3.1 [4]. The following sections will discuss selected steps in the work flow in order to highlight shortcomings and evaluate their impact on accuracy of gate-level power estimation.

3.1 Errors due to logic simulation

Power estimation on gate-level requires accurate knowledge of the switching activity on all circuit nodes. The most common method of determining this switching activity on all circuit nodes is to perform a logic simulation of the design under test using either the



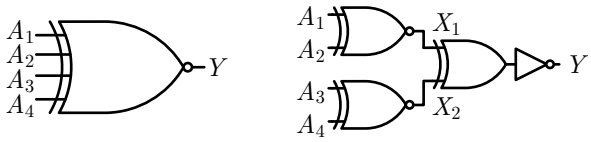
(a) output pulse width (b) power consumption

Figure 3.2: Example of glitch propagation and related switching power

gate-level netlists themselves or the RTL sources followed by a mapping step. Simulation on register-transfer level features the advantage of significant speedup compared to the gate-level but introduces additional errors because accurate timing properties of the gate-level implementation cannot be considered. In order to reduce the possible error sources, the following discussion will therefore focus on the gate-level instead. As an alternative to logic simulation, probabilistic methods for the calculation of switching activity like *tagged probability waveforms* [6] or *transition waveforms* [54, 23] could be used. These probabilistic propagation approaches suffer from similar problems as logic simulation when handling glitches but introduce additional sources of error due to the challenge of handling signal correlation due to reconvergent fanout during propagation of signal statistics.

The main contribution to errors during logic simulation is the handling of gate delays which lead to glitches on the circuit nodes. In order to be able to observe glitches, it is necessary to annotate accurate gate delays to the gate-level netlist during simulation. The gate delays, that are themselves dependent on the transition slopes on the gate inputs and the capacitive load at the gate outputs, can be estimated using the static timing analysis functionality that is typically provided by complementary tools to the gate-level power estimators.

A limitation that is inherent to logic simulation is the restriction to two signal levels (*true* and *false*) and the abstraction from finite signal transition slopes. In contrast, glitches occurring during actual operation of the circuit do not necessarily feature full-swing transitions due to limited signal slopes that cause the output to be discharged



(a) symbol (b) schematic

Figure 3.3: Multistage 4-input XNOR gate from a commercial standard-cell library

before it has been fully charged. The second physical effect that is hard to model in logic simulation is the impact of a glitch on the input of a gate on the output of that gate. In logic simulators this effect that is known as *glitch filtering* is commonly implemented by using inertial gate delays. The effect of this filtering is to prevent pulses that are shorter than the gate delay from propagating to the gate output. This is a simplification since the physical effect of a short pulse at the input is often visible as a partial swing pulse at the output. Fig. 3.2a shows the relationship between the input and output pulse widths for a simple NAND gate and a more complex multi-stage XNOR gate. As opposed to logic simulation with glitch filtering, circuit simulation does not feature and abrupt cut-off of the propagation for short input pulses but shows a smooth transition instead. Fig. 3.2b demonstrates that pulses that do not feature full swing transitions nevertheless cause significant power consumption. Inertial glitch filtering as employed in logic simulators would overestimate power consumption resulting from small glitches applied to the NAND gate while it would potentially lead to underestimation of the XNOR gate by rejecting a large number of valid glitches. This is a typical result which demonstrates that the gate delay is not a good measure for glitch filtering.

The inertial glitch filtering fails completely for complex multistage gates like full-adders or gates like the 4-input XNOR gate shown in Fig. 3.3. The delay of these gates that is looked up from the design libraries depends on the sum of stage delays. In contrast, for correct application of the pulse filtering using the inertial gate delay model the delay of the slowest stage would be significant. Since no information about the internal structure of the basic cells is known to the gate-level power estimation flow, the default glitch filtering tends to be too aggressive for complex multistage gates. This effect is demonstrated in Fig. 3.4 showing waveforms from physical circuit simulation and logic simulation alongside cycle accurate power estimations for a compound 4-input XNOR gate that features pulses of increasing width on a single input. In circuit simulation, the pulse in the second cycle can already be observed on the internal node X_1 and the pulse of the fourth cycle propagates to the output demonstrating the multistage nature of this gate. In contrast,

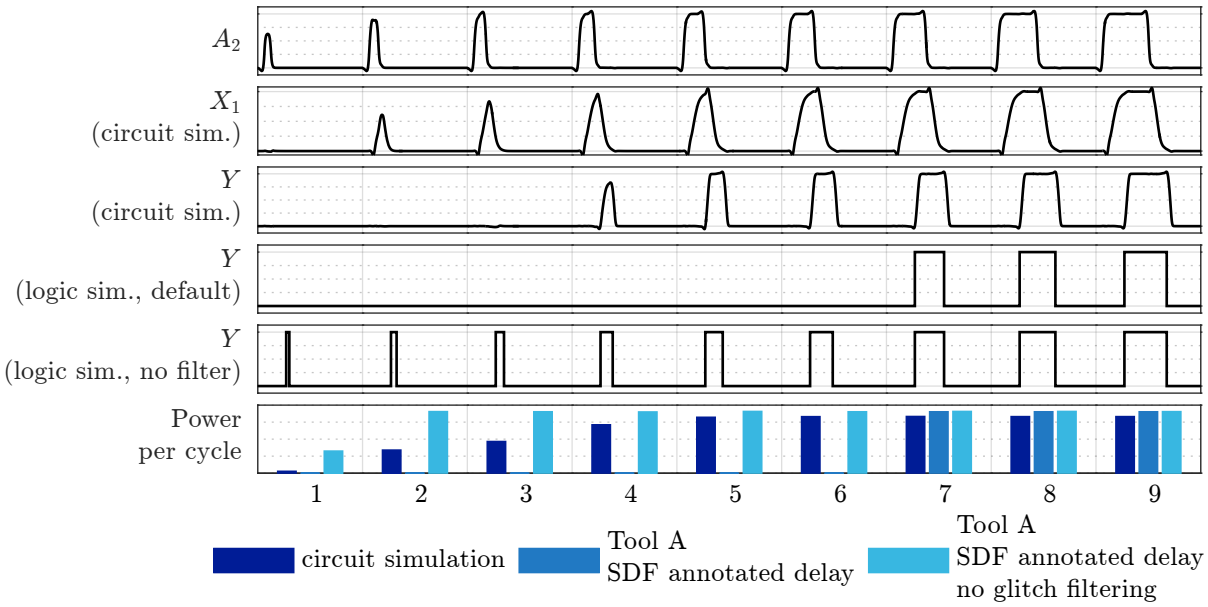


Figure 3.4: Glitch filtering and estimated glitch energy for pulses on a single input of a 4-input XNOR gate ($A_1 = 1, A_3 = A_4 = 0$)

logic simulation erroneously rejects all pulses until the seventh cycle when the default pulse filtering depending on the inertial gate delay model is employed.

Logic simulation without glitch filtering on the other hand results in gross overestimation of switching power as demonstrated in Fig. 3.4 despite efforts of power estimation tools to recognize glitches in the logic waveforms and apply a scaling factor to the power consumption of related events. The threshold for glitch detection is typically derived from the transition times, t_{rise} and t_{fall}, at the gate output as a pulse width

$$t_{pulse} < \frac{t_{rise} + t_{fall}}{2}. \tag{3.1}$$

Once a glitch has been detected the power associated with the pair of constituting transitions, P_{trans}, is reduced by a factor to approximate the power consumption of a partial-swing pulse as

$$P_{glitch} = \left(\frac{2 \cdot t_{pulse}}{t_{rise} + t_{fall}} \right)^2 \cdot P_{trans}. \tag{3.2}$$

This approximation clearly fails for the considered XNOR gate where the short transition times of the output stage are only marginally related to the minimum propagating pulse width. Consequently, the scaling factor is only applied to the power related to the erroneously predicted glitch in the first cycle of Fig. 3.4 during the logic simulation without glitch filtering. All subsequent pulses are not recognized as glitches and are therefore estimated as separate transitions.

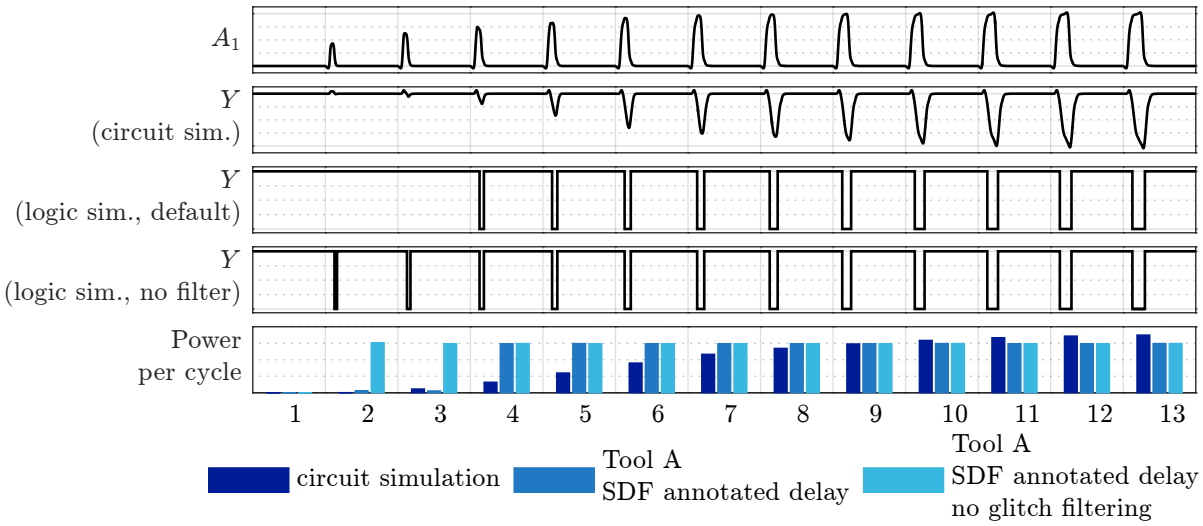


Figure 3.5: Glitch filtering and estimated glitch energy for pulses on a single input of a 2-input NAND gate ($A_2 = 1$)

While for complex gates the default glitch filtering routinely rejects valid pulses which leads to an underestimation of switching power, the opposite can be true for simple gates with gate delays that may well be shorter than the transition time of the output. This is demonstrated in Fig. 3.5 for a 2-input NAND gate featuring pulses of increasing width on one input. Inertial gate delay filtering predicts the third pulse to propagate to the output while in physical circuit simulation the first output pulse to show 90 % output swing is in cycle nine. Due to the steep transition slopes of the output, no pulses are recognized as glitches during power estimation preventing any possibility of correction of this overestimation.

3.2 Errors due to energy lookup

Apart from the errors caused by inaccurate switching activities that serve as inputs to the gate-level power estimation tools, there are shortcomings which are directly dependent on the estimation methodology itself. While in general it is hard to separate the contributions to the total error observed during power estimation, during an analysis of benchmark circuits from the ISCAS85 suite the power estimation tools were supplied with switching activities that were extracted from outputs of low-level circuit simulations [4]. The power estimation using these “perfect” switching activities still showed deviations of up to 16 % from circuit simulation references.

These errors are mainly caused by simplifications in the format of the characterized cell libraries that were introduced to reduce the characterization complexity and the size of the

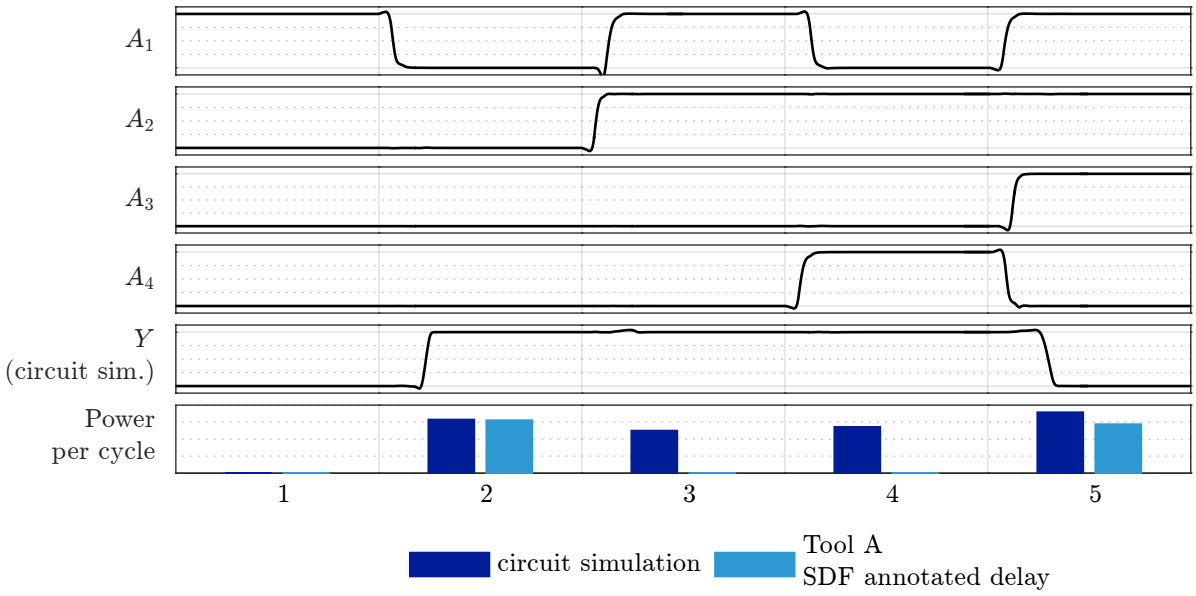


Figure 3.6: Estimated power for switching of multiple inputs of a 4-input XNOR gate

library itself. Apart from the restriction to full-swing transitions, the most fundamental simplification is the omission of multiple-input events from the library [61]. Because of this, the cell library used during lookup only contains power values related to the toggling of a single input. Power consumption related to the simultaneous or quasisimultaneous switching of multiple inputs can only be approximated. The situation is even more severe for gates where switching of the individual inputs always results in switching of the output while quasisimultaneous switching of multiple inputs does not cause the output state to change. The 4-input XNOR gate which has already been regarded in the last section can again be used as an example to demonstrate this shortcoming. In Fig. 3.6 the cycle accurate power estimation for selected events on single or multiple inputs is compared to the circuit simulation reference. The power consumption of the single-input event in the second cycle is estimated perfectly but for switching of three inputs in the fifth cycle the power is underestimated by 20 %. The lookup based power estimation fails completely in the third and fourth cycle that both show two input events which cancel out the output effect that would have been related to each single-input event. As the lookup tables in the cell library that are dependent on both the transitions on the input and output do not contain events for switching of inputs without activity on the output, no matching power values can be found for either of the input switching events and power cannot be estimated at all. Consequently, this effect is most pronounced for gates like XOR gates or full adders where all single input events result in output switching while quasisimultaneous switching of multiple inputs might not cause the output state to change.

name	circuit function	number of gates		
		CMOS40	CMOS28a	CMOS28b
C432	Interrupt Controller	87	75	75
C499	SEC	154	185	157
C880	ALU	150	158	153
C1355	SEC	131	194	149
C1908	SEC/DED	180	190	184
C2670	ALU/Controller	342	352	358
C3540	ALU	454	477	444
C5315	ALU	586	635	639
C6288	16-bit Multiplier	599	623	609
C7552	Adder/Comparator	729	766	693

Table 3.1: Characteristics of the ISCAS85 circuits

3.3 Benchmarking of estimation accuracy

In order to evaluate the accuracy of gate-level power estimation, a range of typical circuits was analyzed using two commercial gate-level power estimation flows. The reference power consumption data against which the estimations are compared was generated using physical-level circuit simulation running the same testbench as during estimation. In order to be independent of discrepancies caused by specific technologies or library files, this analysis was carried out using three different advanced commercial CMOS technologies: *CMOS40* is a 40-nm CMOS technology while *CMOS28a* and *CMOS28b* are both 28-nm CMOS technologies from two different foundries. All benchmark circuits were implemented using the foundry-supplied general purpose standard cell libraries.

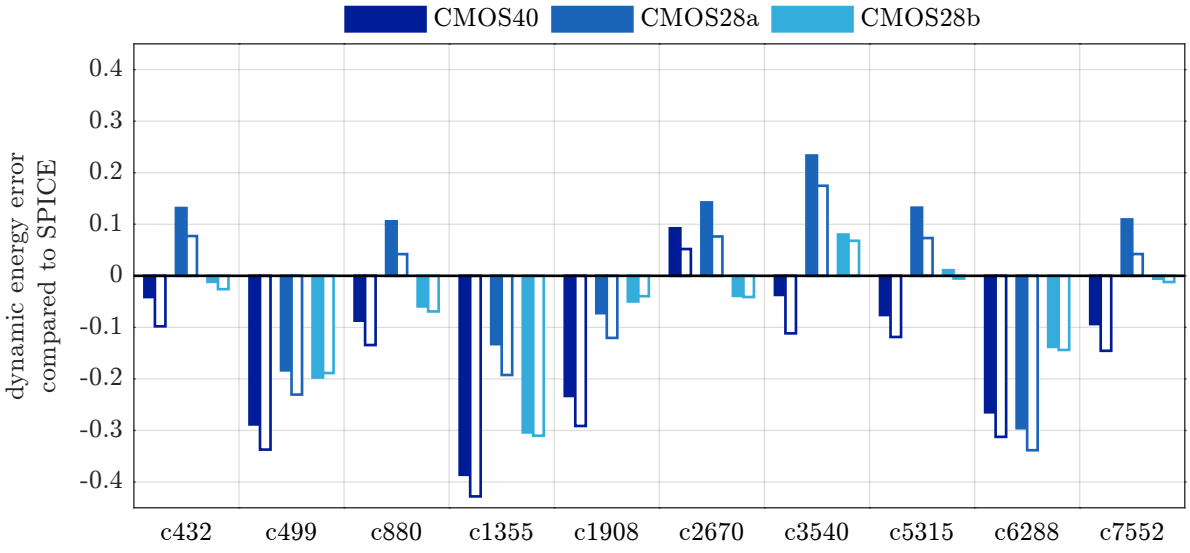
3.3.1 ISCAS85 circuits

One of the most widely used benchmark suites which is commonly known as *ISCAS85* stems from the *International Symposium of Circuits and Systems* in 1985 [62]. This package consists of 10 purely combinatorial industrial designs that were originally published as flattened gate netlists without description of the implemented function. Over the years these circuits have been used in a variety of fields from logic synthesis to test pattern generation. Because of their wide acceptance, individual circuits from this benchmark set have also quite commonly been used to demonstrate power estimation methodologies. For the purpose of this comparison, the original netlists where implemented using standard cell synthesis. The characteristics of the results after place and route for all three technologies are summarized in Table 3.1.

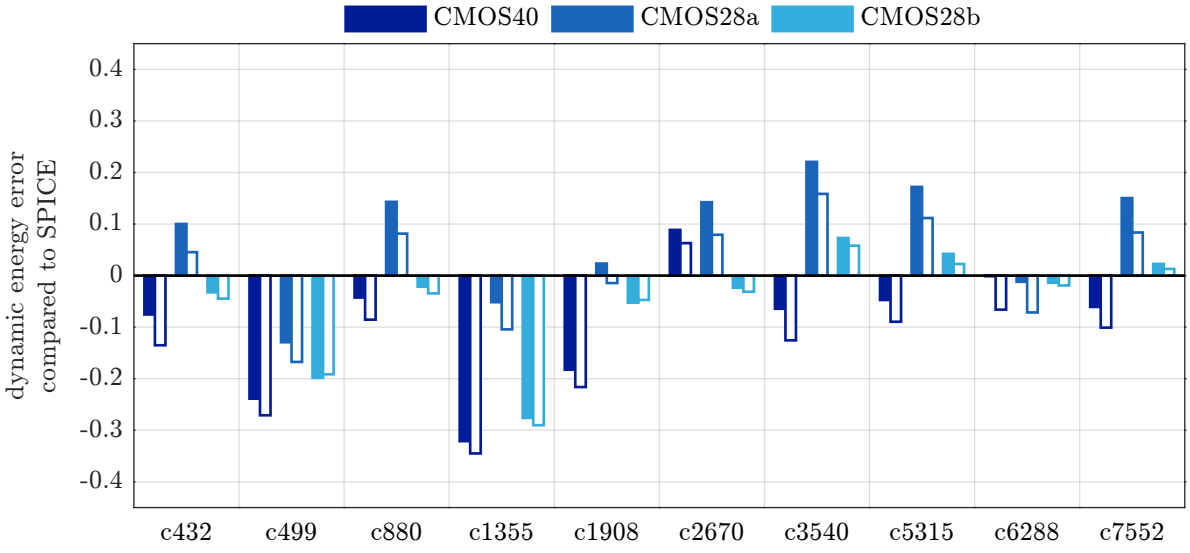
The ISCAS85 circuits were analyzed using both physical circuit simulation and gate-level power estimation using two different commercial tool suites. The power consumption observed during circuit simulation is regarded as the reference in the following comparison. As no realistic testbenches are published for the ISCAS85 circuits, the circuit simulation as well as the gate-level power estimations assume the same stream of uniformly distributed random vectors on the circuit inputs for a duration of 1000 cycles. The main focus of this analysis is the handling of effects due to real gate delays which have little to no impact on the static power consumption. Therefore, the total power consumption was separated into dynamic and static power. Gate-level power estimation reports separate figures for dynamic and static power by default. In circuit simulation this separation was performed according to (2.8) and (2.9) by varying the operating frequency while executing the same testbench.

The estimates for dynamic energy dissipation of both commercial tool suites, denoted as *Tool A* and *Tool B*, are compared to the reference from SPICE-level circuit simulation in Fig. 3.7a. The mean deviation from the circuit simulation result is as high as 14.6 % or 15.2 % for Tool A and Tool B respectively. Even more interestingly the dynamic energy is overestimated by up to 23 % for some circuits while it is underestimated by up to 43 % for others. The highest errors can be observed for circuits *c499*, *c1355* and *c6288*. Upon closer analysis it is found that this error is mainly caused by the use of large XNOR gates with three or four inputs for *c499* and *c1355*, which implement error detection functions, and full-adders in *c6288*, which represents a 16-bit multiplier. These types of gates are examples for compound gates that suffer both from excessive pulse filtering when using the default inertial gate delay model as described in section 3.1 as well as from missing library table entries for multi-input events as elaborated in section 3.2.

While the errors due to simplifications of the provided library are inherent to the state-of-the-art power estimators, the misestimation of switching activities can be reduced by more complex glitch filtering. Since no information on the minimum pulse width on gate inputs that leads to propagation is included in the default characterization library, a custom precharacterization of all cells used in the netlist is required in order to determine the propagation properties. This additional information can be used to pre- and postprocess the data related to the logic simulator, resulting in more realistic glitch filtering [4]. As shown in Fig. 3.7b, the estimation error can be reduced for most circuits by inclusion of extended glitch filtering. In particular, the errors during the original estimation of circuit *c6288* appear to be almost exclusively caused by over-aggressive pulse filtering



(a) default inertial delay pulse filtering



(b) extended glitch filtering based on custom precharacterization

Figure 3.7: Estimation error of dynamic energy dissipation for two commercial gate-level power estimators (filled bars: Tool A, empty bars: Tool B) compared to SPICE-level circuit simulation for three CMOS technologies.

in the inertial delay model. In contrast, the estimation for the problematic circuits containing large XNOR gates still shows underestimation of up to 34.5 % despite significant improvements. An explanation of this discrepancy is found when the maximum logic depth, which measures the number of gates that are connected in series, is compared for both circuit types. For the multiplier circuit *c6288* the logic depth is found to be 36 or 37 for all technologies while a maximum of only 14 gates are connected in series for the error detection circuits. Resulting from the three times higher logic depth, the accurate propagation of glitches becomes critical for *c6288*, because a large number of gates are

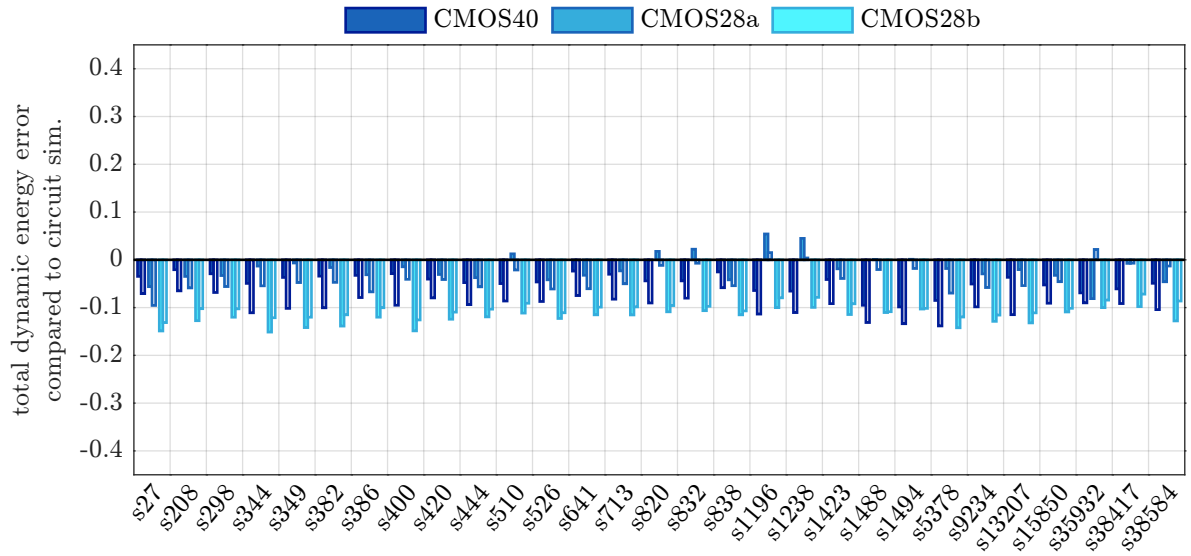
name	number of gates (number of registers)		
	CMOS40	CMOS28a	CMOS28b
s27	14 (3)	20 (3)	13 (3)
s208	45 (8)	52 (8)	56 (8)
s298	76 (14)	88 (14)	86 (14)
s344	87 (15)	126 (15)	100 (15)
s349	89 (11)	117 (15)	105 (15)
s382	126 (21)	148 (21)	128 (21)
s386	78 (6)	80 (6)	74 (6)
s400	127 (21)	136 (21)	124 (21)
s420	101 (16)	105 (16)	110 (16)
s444	128 (21)	143 (21)	128 (21)
s510	123 (6)	126 (6)	130 (6)
s526	119 (21)	129 (21)	134 (21)
s641	111 (14)	127 (14)	110 (14)
s713	110 (14)	130 (14)	108 (14)
s820	136 (5)	137 (5)	127 (5)
s832	129 (5)	132 (5)	131 (5)
s838	183 (32)	208 (32)	215 (32)
s1196	271 (18)	270 (18)	271 (18)
s1238	288 (18)	274 (18)	272 (18)
s1423	458 (74)	592 (74)	475 (74)
s1488	284 (6)	301 (6)	294 (6)
s1494	296 (6)	297 (6)	297 (6)
s5378	776 (162)	966 (162)	911 (162)
s9234	655 (132)	898 (132)	705 (132)
s13207	1009 (213)	1365 (213)	1111 (213)
s15850	585 (128)	675 (128)	655 (128)
s35932	8096 (1728)	12427 (1728)	9488 (1728)
s38417	6589 (1462)	8912 (1462)	8200 (1462)
s38584	7166 (1159)	9428 (1159)	7498 (1159)

Table 3.2: Characteristics of the ISCAS89 circuits

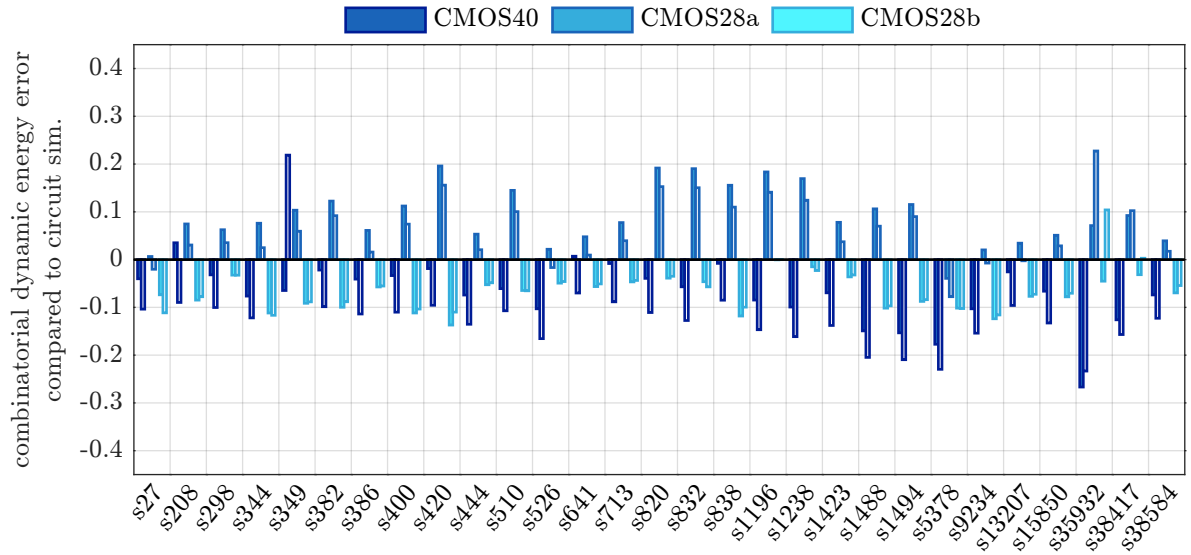
potentially affected by glitch pulses propagating along logic paths. Quasisimultaneous switching resulting in errors during energy lookup on the other hand is more likely for shorter and more balanced signal paths as found in the error detecting circuits.

3.3.2 ISCAS89 circuits

In order to extend the complexity of the ISCAS85 benchmarks as well as to provide the opportunity to benchmark scan-based test generation algorithms, during the *International Symposium of Circuits and Systems* in 1989 a set of sequential circuits was distributed [63] that was compiled from industrial and university sources and is commonly known as



(a) estimation accuracy of total dynamic energy



(b) estimation accuracy of dynamic energy excluding registers

Figure 3.8: Estimation error of dynamic energy dissipation for two commercial gate-level power estimators (filled bars: Tool A, empty bars: Tool B) compared to SPICE-level circuit simulation for two CMOS technologies.

ISCAS89. As with the ISCAS85 circuits, the benchmark circuits were published as gate-level netlists without logical hierarchy and mostly without functional description. The gate counts resulting from implementation of the ISCAS89 benchmark circuits in the analyzed technologies as well as number of internal states are summarized in Table 3.2. In addition to the inclusion of sequential cells, this benchmark set contains significantly more complex circuits than could be found in the ISCAS85 set.

Analogous to the analysis of the purely combinatorial circuits the placed and routed netlists including parasitic components were employed in an evaluation of the dynamic

energy estimated using two leading gate-level estimators. The resulting deviations from the reference obtained by physical-level circuit simulation, which are plotted in Fig. 3.8a, show considerably smaller maximum errors than could be observed for the ISCAS85 circuits. In particular, the implementations in technology *CMOS28a* seem to exhibit a high estimation accuracy, while circuits implemented in technology *CMOS28b* are seemingly prone to underestimation of about 10 %. Upon closer analysis of the estimation results these observations can be attributed to the fact, that the energy dissipation of the registers contained in the ISCAS89 circuits is generally underestimated for all technologies. This cancels out part of the overestimation of combinatorial gates for implementations in technology *CMOS28a* which is common for this technology as shown in Fig. 3.8b. In addition, the maximum overestimation of up to 23 % of combinatorial gates which is observed for the implementation of circuit *s35932* only marginally contributes to the overall estimation error because the power of sequential cells dominates the overall energy dissipation for this circuits. This effect can also be observed for the maximum underestimation of combinatorial gates of 27 % for the same circuit in technology *CMOS40*.

The compensation of overestimation of the energy dissipation for some components or modules of a circuit with underestimation of other parts is a typical feature which can be observed for the power estimation of larger digital circuits. However, this averaging effect which results in apparently good estimation accuracy for a majority of circuits cannot be guaranteed. As a worst case assumption the possibility of mutual reinforcement of individual errors needs to be considered.

This consideration motivates the search for modifications or full replacements of the prevalent gate-level power estimation methodologies. A major goal will be an improved handling of delay effects which enable glitch creation and propagation in order to mitigate one of the biggest deficiencies that was identified in this chapter.

4 Probabilistic macromodels considering glitch parameters

Motivated by the challenge of accounting for glitches using logic simulation, this chapter explores alternative ways of representing the underlying increase in switching activity. In the scope of macromodels as defined in section 2.2 this requires the definition of suitable parameters that can be used as inputs to a precharacterized lookup table to determine the effect of glitches at the input of the circuit macro of interest. Switching induced by the desired evaluation of the logic function is thereby separated from glitches caused by unbalanced delays. As depicted in Fig. 4.1 a general power estimation flow following the macromodeling approach starts by extracting these glitch parameters from the primary inputs of the circuit to be analyzed. Using the precharacterized models of circuit components, the power consumption induced by the evaluated values for glitch parameters can be looked up. At the same time, the model can be used to look up the values for the selected glitch parameters that could be observed at the output of the circuit component if the inputs would be stimulated according to the extracted input parameters. This

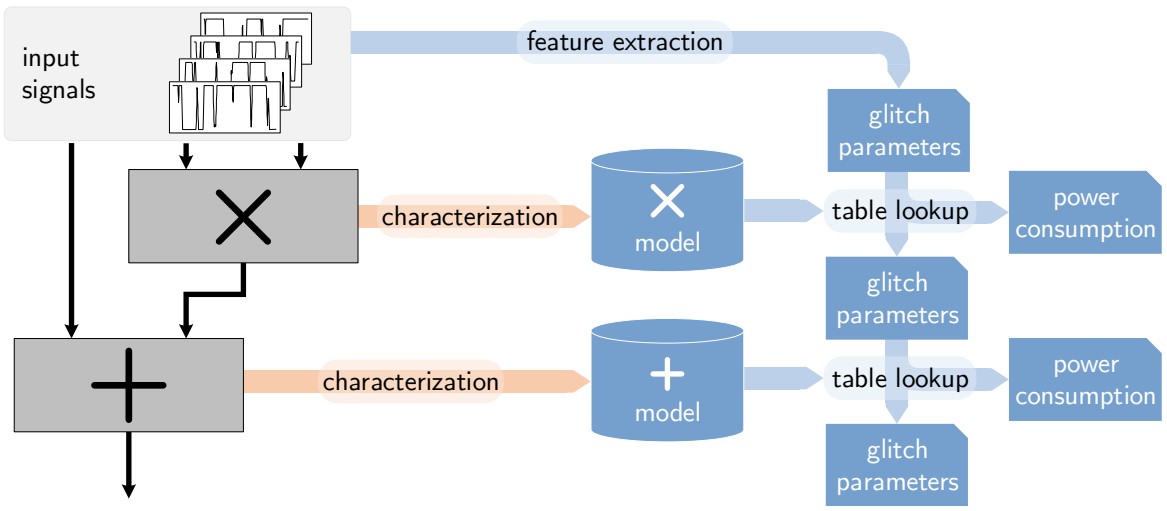


Figure 4.1: Schematic view of macromodel evaluation with integrated propagation of parameter values.

method of parameter propagation replaces time-consuming delay annotated logic simulation and holds promise for high accuracy because the parameter responses stored in the characterization library are based on accurate circuit simulation. The propagated glitch parameters subsequently form inputs to the power models of downstream components in the circuit hierarchy allowing for iterative analysis of all circuit components.

As glitching is accounted for by specialized model parameters, the amount of switching activity due to logic evaluation can easily be determined by high-level functional simulation without consideration of variable gate delays. This activity can for example be captured by traditional metrics like zero-delay switching activity, D_0, which specifies the probability of toggling on a circuit node during a switching period excluding the effect of glitches. When information on lower levels of abstraction is available, the transition slopes of signals, t_{slope}, or the capacitive output load, C_{load}, of macros can be included as model parameters to increase accuracy. A macromodel based on the proposed propagation of glitch parameters could therefore be implemented by a lookup table approximating the function

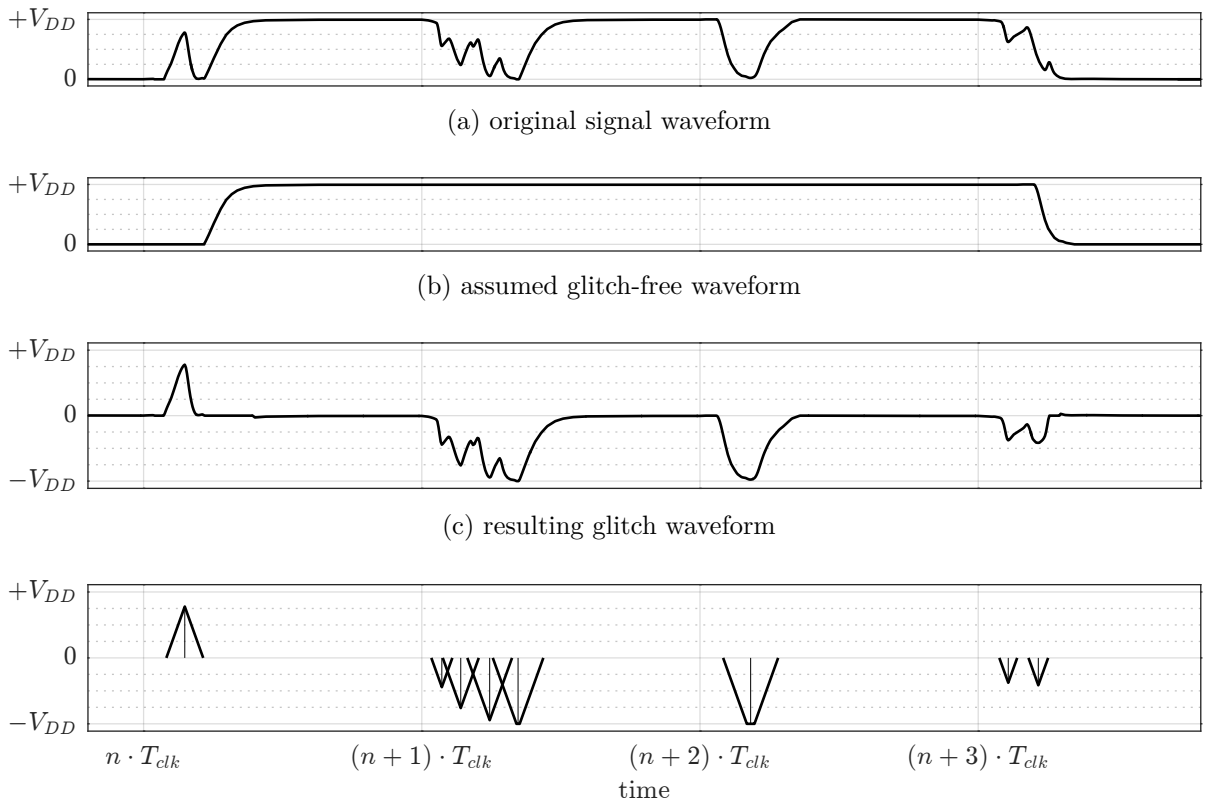
$$P_{macro} = f(D_0, t_{slope}, C_{load}, m_{glitch}), \tag{4.1}$$

which depends on additional parameters, m_{glitch}, capturing the glitching activity on the inputs of the macro that are yet to be determined. The derivation of suitable glitch parameters as well as challenges arising from their inclusion into the macromodeling approach will be discussed in the following sections.

4.1 Definition of quantitative glitch metrics

Previously proposed macromodel parameters which capture the effect of glitches at macro inputs on the power consumption are limited to counting the number of glitches in selected time windows [37, 57]. This metric, denoted as *glitching activity* or *glitch frequency*, shares the deficiencies of logic simulation in that glitch pulses are assumed to span the full signal swing. This prohibits realistic glitch filtering and neglects the effect of partial-swing glitches on power consumption. In addition, time offsets between switching on different macro inputs which are the major source of glitch generation are not taken into account.

To counteract these shortcomings, more detailed glitch metrics that capture the essential characteristics of switching pulses are required. The following discussion will focus on metrics promising the highest accuracy with only limited regard for associated complexity in order to determine the possible benefits of such an approach. Subsequent sections will



(a) original signal waveform

(b) assumed glitch-free waveform

(c) resulting glitch waveform

(d) approximation of extracted glitches

Figure 4.2: Example of extraction of glitch parameters from a signal waveform.

focus on the analysis of complexity and illustrate selected methods in particular for the reduction of characterization complexity.

The derivation of accurate glitch characteristics will be demonstrated using the waveform example in Fig. 4.2a. The depicted segment spanning four clock cycles features excessive glitching activity consisting of both partial-swing and full-swing pulses in addition to two nonglitch transitions. By subtracting the assumed glitch-free activity of Fig. 4.2b from the original signal, the glitch pulses are isolated from desired node transitions as shown in Fig. 4.2c. Successive approximation of this waveform results in the glitch pulses plotted in Fig. 4.2d. Each pulse can be unambiguously characterized by its position in time relative to the start of the clock cycle and its width, which is proportional to the signal swing for partial-swing glitches, if constant transitions slopes can be assumed. The switching direction of each glitch pulse on the other hand can be deducted from information on the glitch-free transitions. Simply counting the number of glitches exceeding a signal swing threshold per period would result in a metric comparable to *glitching activity* as introduced previously. To retain essential information on smaller partial-swing pulses and on the occurrence time, additional metrics could record the mean

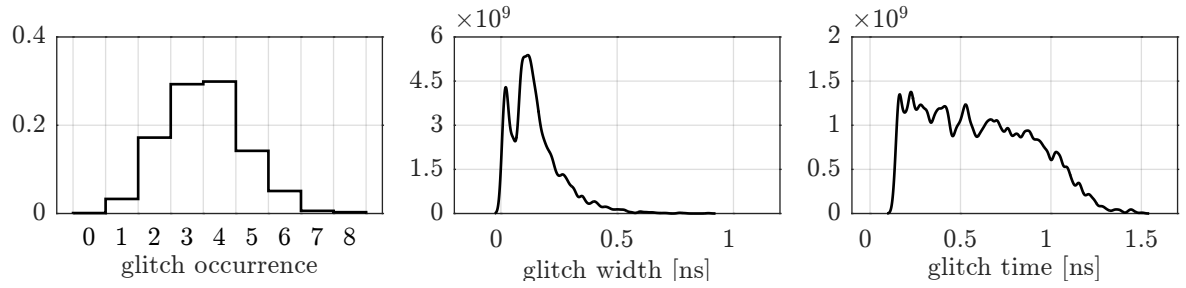


Figure 4.3: Example of characterized probability distributions for proposed glitch metrics.

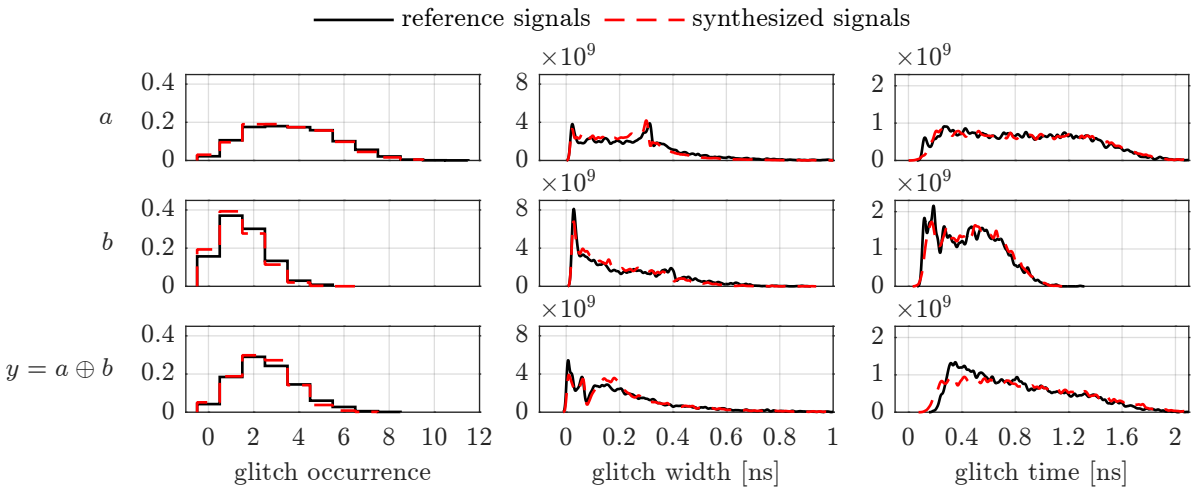
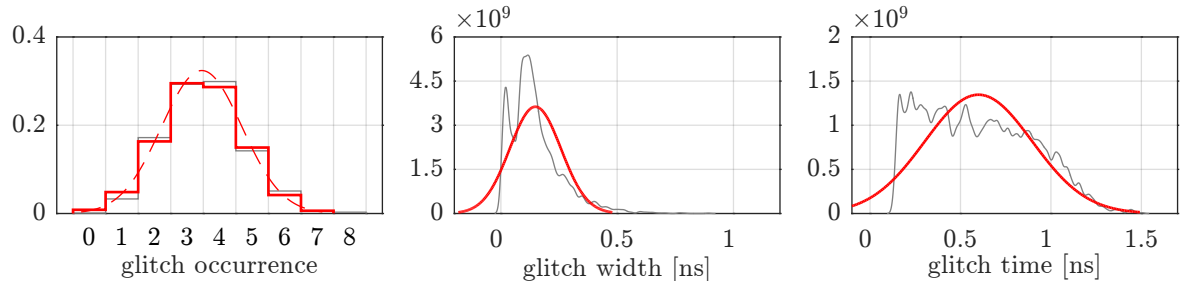


Figure 4.4: Demonstration of waveform synthesis results of glitch metric distributions for inputs a and b as well as for output y of 2-input XOR gate.

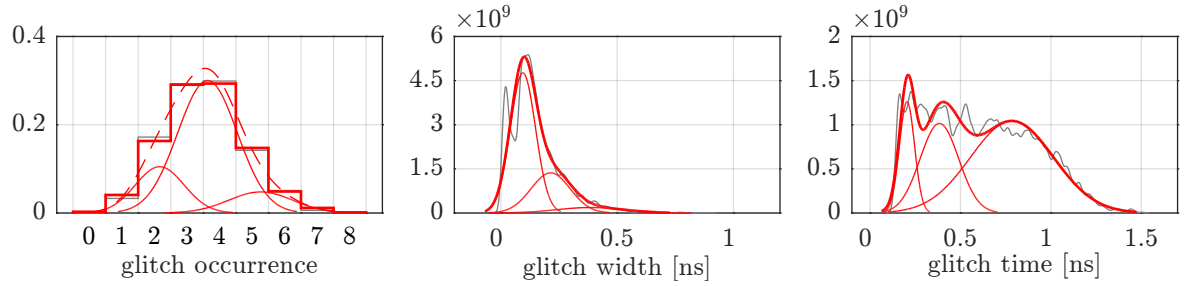
signal swing and mean time offset of glitches relative to the clock signal. However, the reduction to averaged parameters discards all information about the spread of these metrics. As a consequence, waveforms featuring only half-swing pulses would feature the same glitch parameter as waveforms with both full-swing and extremely small glitches despite the fact that power consumption and glitch propagation probability for both waveforms differs considerably. This deficiency is solved by the introduction of glitch metrics that are defined by probability distributions for all parameters.

The metric distributions resulting from characterization of an example waveform are plotted in Fig. 4.3. The probability distribution for *glitch occurrence* specifies the probability of any single cycle to feature a specific number of glitches. For any glitch that occurs, the distributions of *glitch width* and *glitch time* indicate the probabilities for the pulse width assuming a fixed transitions slope or the occurrence time relative to the start of a switching cycle to fall in any particular interval.

An indication whether these proposed metrics based on probability distributions are suitable for capturing glitching characteristics can be obtained by the comparison of the



(a) Fitted single normal distributions



(b) Fitted normal mixture distributions

Figure 4.5: Parameter extraction by fitting of different distribution templates.

effect of two unrelated sets of waveforms that exhibit the same metric distributions as demonstrated in Fig. 4.4. The first set, denoted as *reference signals* consists of two signals a and b that were recorded during the simulation of an arithmetic macro as well as of the output signal y of a gate evaluating the exclusive-OR operation of a and b. Using random sampling techniques it is possible to synthesize two signals that share the statistical metrics of the reference signals a and b without showing any correlation to the original signals. When these two *synthesized signals* are applied to the same XOR gate as the reference signals, the metric distributions at the output closely matches the effect caused by the original signals. The related dynamic power consumption measured during both simulations differs by 2.8 %.

4.2 Extraction of power macromodel parameters

While the proposed metric distributions allow detailed characterization of glitching activity on circuit nodes, their original representation is not suitable for macromodel-based power estimation which links combinations of scalar parameters to the related power consumption. Because both characterization and lookup costs are exponentially dependent on the number of model parameters, the probability distributions proposed as glitch metrics need to be specified with as few scalar parameters as possible while maintaining the desired level of accuracy. A possible approach would be to approximate each metric distri-

bution using a fitted normal distribution as demonstrated in Fig. 4.5a. These probability distributions can be uniquely defined by the mean μ_m and variance σ_m^2 for each metric:

$$F(\text{glitch occurrence}) = F_1 = \mathcal{N}\left(\mu_1, \sigma_1^2\right), \tag{4.2}$$

$$F(\text{glitch width}) = F_2 = \mathcal{N}\left(\mu_2, \sigma_2^2\right), \tag{4.3}$$

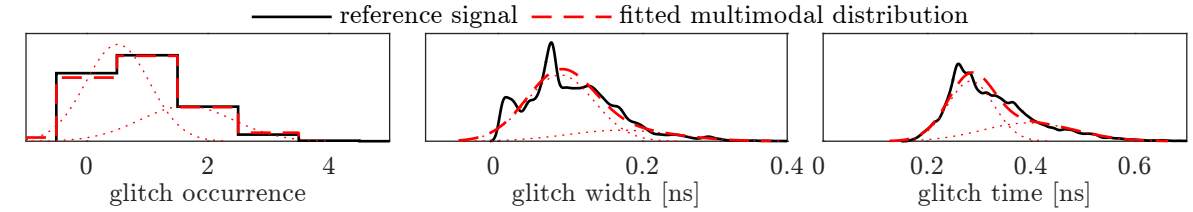
$$F(\text{glitch time}) = F_3 = \mathcal{N}\left(\mu_3, \sigma_3^2\right). \tag{4.4}$$

Following a precharacterization of the macro library encompassing a sufficient large number of combinations for these six glitch parameters (μ_1, σ_1, μ_2, σ_2, μ_3, σ_3), the effect of glitching on the power consumption and the output glitching activity can be determined in two steps. First, the input metric distributions need to be determined and fitted to normal distributions. The resulting set of parameters can subsequently be fed into the macromodel to simultaneously look up the power consumption and the related glitch metric distributions at the outputs of the component. However, by using statistical moments of fitted normal distributions as parameters, this approach assumes a certain normality of the metric distributions which does not appear to be appropriate for the example distributions of glitch width and time in Fig. 4.5a. Experimental experience shows that the empirical probability distributions obtained by characterization of circuit nodes vary significantly in shape precluding accurate fitting of any standard distribution template. Instead, the fitting accuracy can be improved considerably by employing templates based on mixture distributions that consist of weighted sums of standard distribution kernels. Fig. 4.5b demonstrates the increased accuracy when fitting is based on mixture distributions consisting of three normal component distributions. Each mixture distribution is specified by

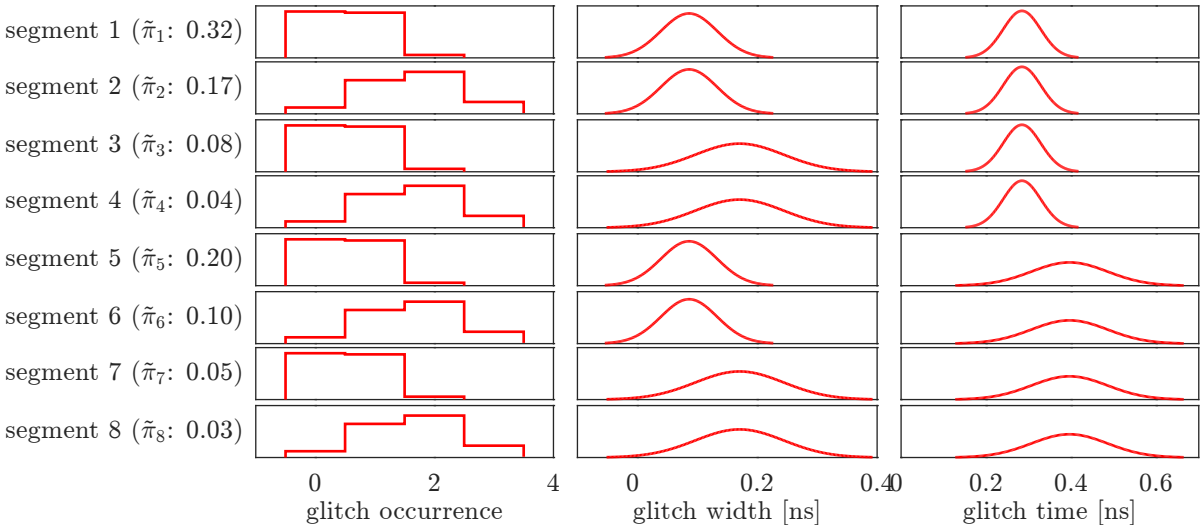
$$F_m = \sum_{i=1}^{n_m} \pi_{mi} \cdot \mathcal{N}\left(\mu_{mi}, \sigma_{mi}^2\right) \quad \text{with} \sum_{i=1}^{n_m} \pi_{mi} = 1, \quad m = 1, 2, 3, \tag{4.5}$$

where the weights π_{mi} define the mixing proportions of the n_m component distributions of each metric m. This increasing goodness of fit which can intuitively be evaluated based on the plotted probability distributions in Fig. 4.5b can formally be derived by suitable metrics like the Kolmogorov-Smirnov statistic. The fitting of these mixture distributions is performed using the Expectation-Maximization algorithm [64] which finds the maximum likelihood estimates for the means μ_{mi} and variances σ_{mi} as well as the weights π_{mi} for a mixture model given a fixed number of component distributions.

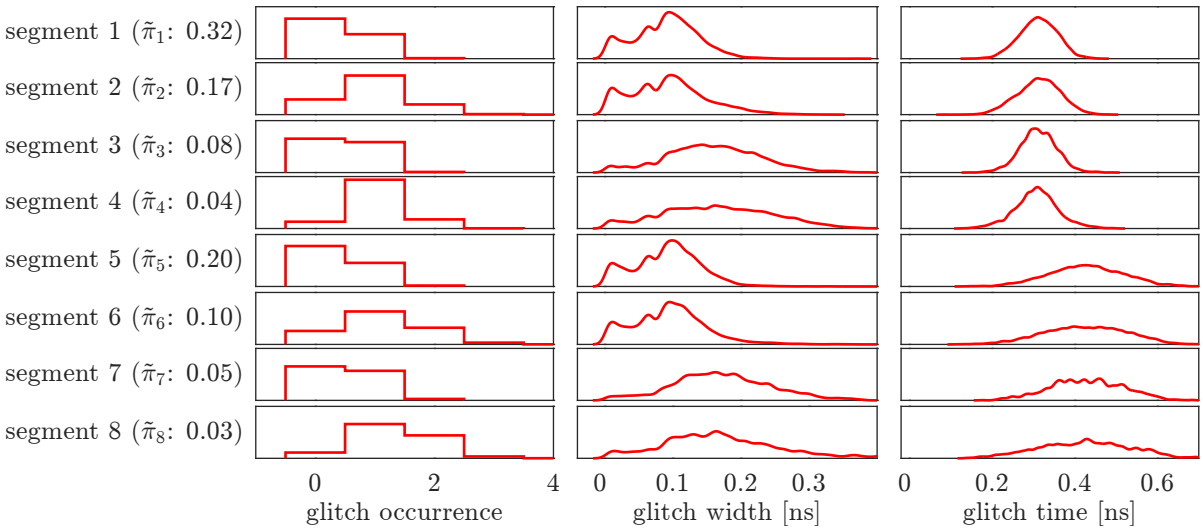
The obvious disadvantage of this approach is the considerable increase in the number of



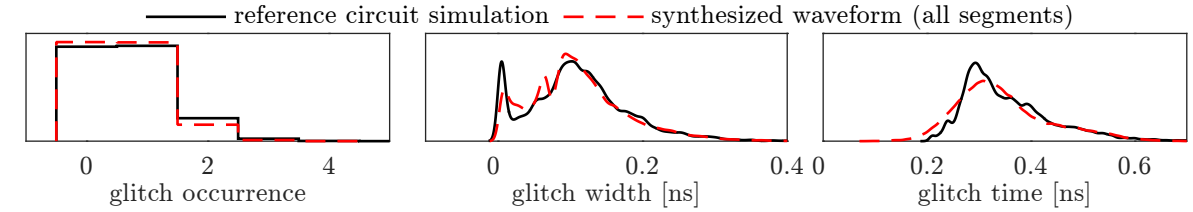
(a) metric distributions at gate input



(b) combinations of single normal component distributions



(c) effects of component distribution combinations



(d) metric distributions at gate output

Figure 4.6: Lookup of glitch metric distribution based on segmentation of single component distributions.

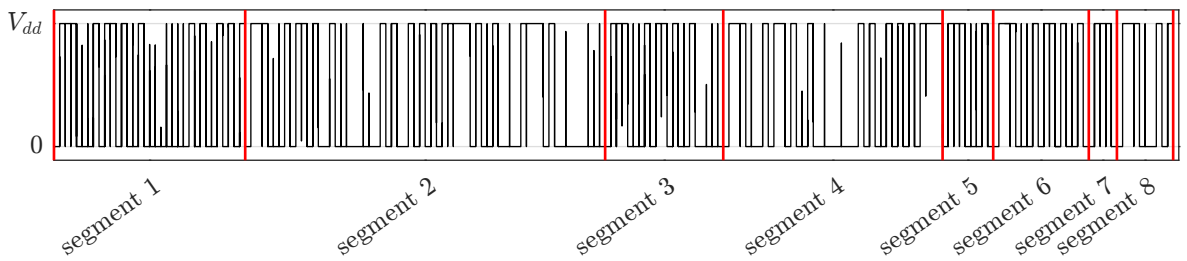


Figure 4.7: Synthesized waveform example reordered into segments that are defined by single component distributions.

model parameters. Instead of six parameters, fitting using mixture distributions results in $2 \cdot (n_1 + n_2 + n_3)$ scalar parameters that would need to be swept during building of the library in order to span the characterization space. As this results in an exponential increase in required low-level simulation runs, direct characterization based on mixture distributions remains unfeasible. However, this can be worked around by the introduction of an approach that is inspired by the way a hypothetical characterization waveform defined by metrics based on mixture distributions would be synthesized. To determine the number of glitches to generate in an individual period of the synthesized waveform one of the component distributions is randomly selected with a probability proportional to its mixing weight π_{1i}. In a second step, the actual number of required glitches is then drawn from this selected normal distribution. Afterwards, the shape and time parameters of each glitch are randomly chosen in the same way by first selecting a component distribution and then drawing a random sample from this distribution. Because it is reasonable to assume that the effect of glitches is only weakly dependent on the order they appear in, the synthesized waveform could just as well be reordered into segments that are defined by only one of the component distributions per metric. This approach will be demonstrated for a gate with one glitching input. Fig. 4.6a shows the metric distributions at the input of the gate as well as the fitted mixture distributions consisting of two normal components each ($n_1 = n_2 = n_3 = 2$). Each glitch is defined by one of the $2^3 = 8$ combinations of single component distributions that are plotted in Fig. 4.6b where the distribution $\tilde{F}_m(k)$ for metric m and combination k is defined by

$$\tilde{F}_1(k) = \mathcal{N}\left(\mu_{1x_k}, \sigma^2_{1x_k}\right), \quad \text{with } x = \{1,2,1,2,1,2,1,2\}, \tag{4.6}$$

$$\tilde{F}_2(k) = \mathcal{N}\left(\mu_{2y_k}, \sigma^2_{2y_k}\right), \quad \text{with } y = \{1,1,2,2,1,1,2,2\}, \tag{4.7}$$

$$\tilde{F}_3(k) = \mathcal{N}\left(\mu_{3z_k}, \sigma^2_{3z_k}\right), \quad \text{with } z = \{1,1,1,1,2,2,2,2\}. \tag{4.8}$$

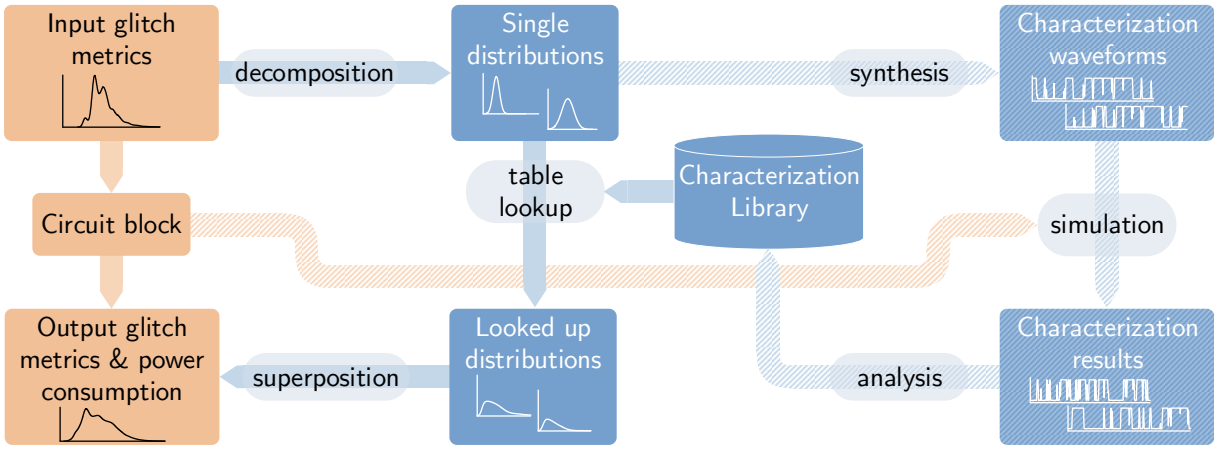


Figure 4.8: Schematic workflow of proposed power and glitch metric estimation.

The sequences x, y and z determine which component distribution contributes to each combination. Each of these combinations describes the metrics of one segment in a hypothetical synthesized waveform as plotted in Fig. 4.7. The glitching in the first segment is defined by the first component distribution of each of the metrics ($\tilde{F}_1(1) = \mathcal{N}(\mu_{11}, \sigma_{11}^2)$, $\tilde{F}_2(1) = \mathcal{N}(\mu_{21}, \sigma_{21}^2)$, $\tilde{F}_3(1) = \mathcal{N}(\mu_{31}, \sigma_{31}^2)$) whereas the glitch occurrence in the next segment follows the second component distribution ($\tilde{F}_1(2) = \mathcal{N}(\mu_{12}, \sigma_{12}^2)$) while glitch width and glitch time are still defined by their first component distributions. The weight of each combination k, which corresponds to the length of each segment in the synthesized waveform of Fig. 4.7 is calculated as the product $\tilde{\pi}_k = \pi_{1x_k} \cdot \pi_{2y_k} \cdot \pi_{3z_k}$ of the weights of participating component distributions (e.g. $\tilde{\pi}_2 = \pi_{12} \cdot \pi_{21} \cdot \pi_{31}$). Separate characterization of all individual waveform segments results in metric distributions at the output of the gate as depicted in Fig. 4.6c which can be superimposed according to their associated combination weight $\tilde{\pi}_k$. The resulting estimated metric distributions at the output, which are plotted in Fig. 4.6d, closely match the results of circuit simulation stimulated by the reference waveform.

This method of decomposition into combinations of single normal distributions can directly be applied to the proposed estimation of power consumption and output glitch metrics. As shown in Fig. 4.8, the combinations of normal metric distributions resulting from the described decomposition of input metric distributions are used as input parameters to the precharacterized lookup tables stored in the library. After lookup, the individual effects are combined by weighted superposition to arrive at the estimated metric distributions at the outputs and the power consumption of the circuit macro. The generation of the characterization library using synthesized waveform stimuli and accurate circuit simulations at transistor level depicted in the right side of the workflow needs

to be performed only once for all relevant circuit blocks. The lookup accuracy can be considerably increased by employing a suitable interpolation scheme during lookup.

The proposed power macromodel based on the lookup of decomposed mixture distributions does not need to make any assumption about the shape of metric distributions and still retains the lookup complexity of single normal distributions. However, the decomposition described so far neglects correlations between glitch metrics when combination weights $\tilde{\pi}_k$ are simply calculated as the product of component weights. As an intuitive example of this shortcoming, compare segments 3 and 4 in Fig. 4.6, which feature the same component distributions for glitch width and glitch time but different distributions for glitch occurrence. The decomposition approach so far assumed that the width and time attributes of glitches could be assumed to be the same irrespective of the numbers of glitches that occur during a period. This assumption can easily be refuted by showing that the number of distinguishable wide glitches that can physically occur during a short time window is limited which would result in a higher probability for narrow glitches in periods featuring a large number of glitches. Without increasing the complexity of the estimation approach by accurate handling of correlation between glitch metrics, this inaccuracy can be mitigated by empirical correction factors modifying the combination weights based on the intuitive interrelation of glitch count, glitch width and duration of switching time window.

4.3 Building of the characterization library

Like all power estimation methodologies following the macromodeling approach, the proposed power model requires preparatory characterization of the regarded library components on lower levels of abstraction. To capture the influence of the proposed glitch parameters based on moments of normal metric distributions, physical level implementations of the components need to be stimulated with input vector streams that were synthesized to follow certain combinations of input parameters. For each combination of input parameters a circuit simulation is required to determine the dynamic energy dissipation and the output glitch metrics related to the glitching activity at the inputs. As the synthesized simulation stimuli are randomly sampled from the selected metric distributions, the energy dissipation and the output glitch metrics analyzed from simulation are statistical properties and the characterization becomes a Monte-Carlo process. Therefore, the quality of the predictions for the output metrics and energy dissipation

can be expected to increase with growing number of considered characterization stimuli. The length of the characterization simulation required to estimate the desired parameters with desired confidence cannot accurately be determined beforehand, because it strongly depends on circuit structure and shape of input distributions. However, statistical inference theory enables the calculation of confidence intervals for selected parameters after a specified number of samples have been collected.

As an indicator that approximate convergence has been reached the sample mean of dynamic energy dissipation is tracked until convergence can be assumed with a certain level of confidence. Assuming that N independent normally distributed energy samples $E_1, E_2, \ldots, E_N$ have been observed without knowledge about the variation, statistical estimation theory predicts the mean energy dissipation E_μ underlying the sampling distribution to fall within the interval

$$\bar{E} - t_{\alpha/2}\frac{s}{\sqrt{N}} < E_\mu < \bar{E} + t_{\alpha/2}\frac{s}{\sqrt{N}} \tag{4.9}$$

with $100 \cdot (1-\alpha)\,\%$ confidence [65]. This interval depends on the number of samples N, the sample mean $\bar{E} = \frac{1}{N}\sum_{k=1}^{N} E_k$, the sample variance $s^2 = \frac{1}{N-1}\sum_{k=1}^{N}\left(E_k - \bar{E}\right)^2$ and the t-value with $N-1$ degrees of freedom $t_{\alpha/2}$ leaving an area of $\frac{\alpha}{2}$ to the right under the t-distribution.

If each sample E_i is obtained from the energy dissipation caused by an individual period during circuit simulation applying randomly sampled stimuli, it can hardly be assumed to be normally distributed. However, by using energy samples that are averaged over several subsequent periods, according to the Central Limit Theorem approximate normality can be assumed. As a general guideline, the averaging of energy dissipation over $n \geq 30$ periods provides samples for which the distribution is reasonably close to normal [65]. Statistical independence of the individual samples can be enforced by insertion of setup intervals between sampling periods that are long enough to ensure that any potential internal states of the characterized component are uncorrelated between subsequent samples.

From (4.9) the relative error of the sample mean $\bar{E}$ can be calculated as

$$\varepsilon_E = \left|\frac{E_\mu - \bar{E}}{\bar{E}}\right| < \frac{t_{\alpha/2} \cdot s}{\bar{E} \cdot \sqrt{N}}. \tag{4.10}$$

As a consequence, during characterization of the model library, glitching input stimuli are applied until the estimated relative error ε_E of the mean dynamic energy dissipation

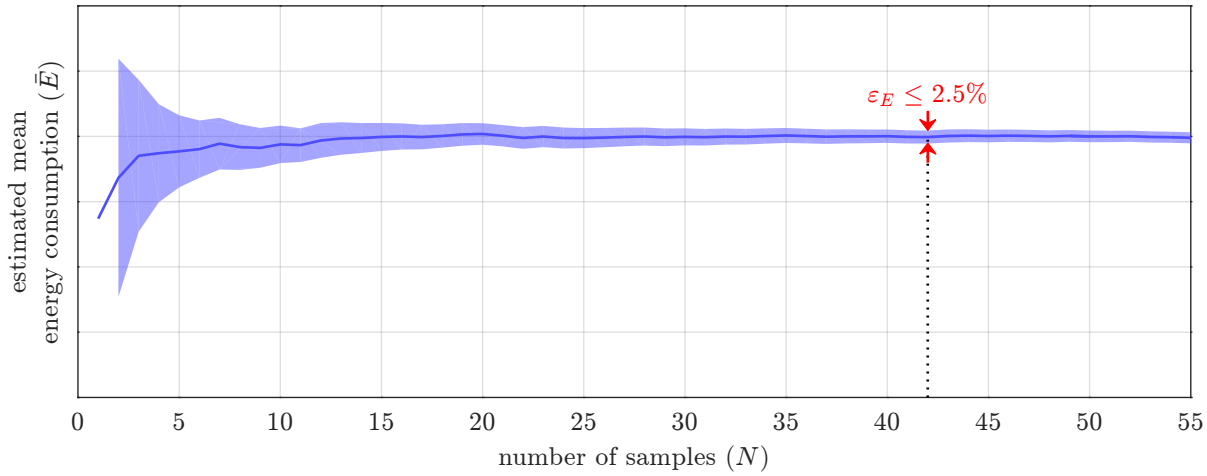


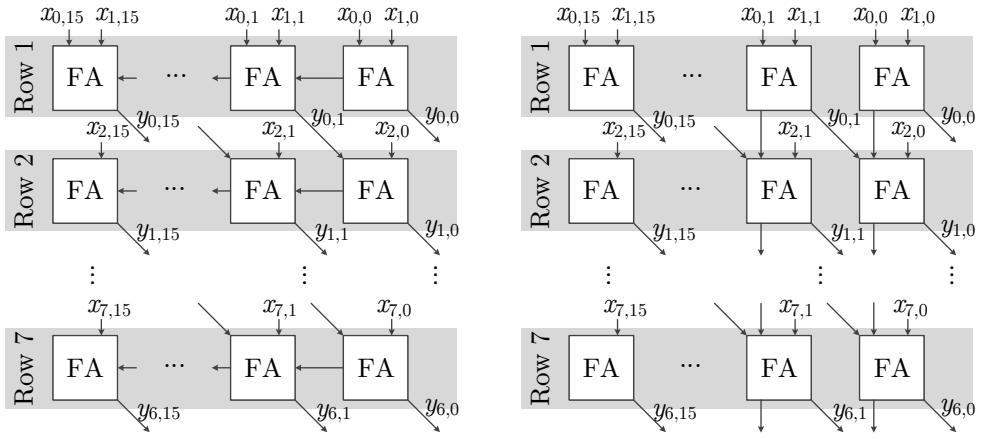
Figure 4.9: Error bounds during estimation of mean energy dissipation from simulation of random input samples (confidence: 99 %).

is smaller than a predefined threshold for the chosen confidence level. This approach is demonstrated in Fig. 4.9, which plots the estimations of mean energy dissipation resulting from increasing numbers of energy samples as well as the error bounds derived from (4.9). Assuming a targeted error threshold of 2.5 %, and a confidence level of 99 % this simulation could have been terminated after 42 samples. The sample mean $\bar{E}_{N=42}$ at this point would be stored in the characterization library with the knowledge that it is highly unlikely that it deviates from true mean E_μ of the underlying energy distribution by more than 2.5 %.

4.4 Application example

The proposed probabilistic macromodel approach will be demonstrated using regular multioperand adder macros as shown in Fig. 4.10 which can represent the essential part of array multipliers. These circuits were chosen due to the high glitching activity which can be expected at the adder outputs. Both circuits were implemented in a 40-nm CMOS technology and the analysis is performed on post-layout data including parasitic circuit components. The inputs $x_{i,j}$ are expected to be glitch-free sequences of uncorrelated bits with a mean probability of toggling of 50 % per cycle that might for example be directly driven by registers. During a circuit simulation on physical level both the power consumption of each adder row as well as the glitch metrics of the adder outputs were analyzed. These simulation results will form the reference against which the proposed macromodel approach is compared.

The first circuit shown in Fig. 4.10a is based on adder rows built from carry-ripple

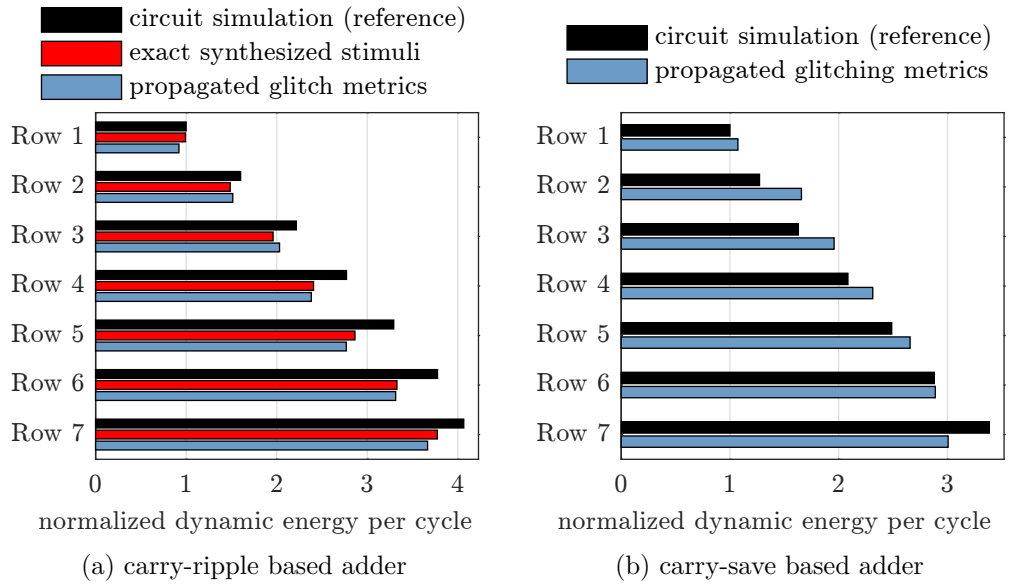


(a) carry-ripple based adder array (b) carry-save based adder array

Figure 4.10: Schematics of testbenches for the proposed probabilistic macromodel.

adders. The macromodel analysis is based on the characterization of one of these rows for a sufficient number of combinations of normal metric distributions as described in the previous section. Thereby the effect of glitching generated and propagated inside the adder row is intrinsically captured during characterization. To simplify the characterization, all bits of one input word, y_i, are assumed to feature the same glitch parameters. Because at least one of the adder inputs is connected to a primary input that is assumed to be glitch-free, this results in six glitch parameters that need to be varied during characterization.

Two sets of estimation results are compared to the circuit simulation reference. The first set, called *propagated glitch metrics*, is derived from the proposed macromodel based estimation that determines the glitch metrics at the adder outputs by table lookup allowing for iterative estimation of the circuit hierarchy. The second set of results, which will be denoted *exact synthesized stimuli*, serves as a means to distinguish errors introduced by the propagation of glitch metrics from errors related to the simplifications introduced during characterization. It is generated by synthesizing glitching stimuli that adhere to the glitch metrics observed at the adder inputs of each row during circuit simulation. These synthesized stimuli are subsequently applied to a single adder row featuring the same capacitive output load as the reference circuit. The two sets of dynamic energy estimations per adder row for the carry-ripple based adder array are compared to the circuit simulation reference in Fig. 4.11a. The proposed approach based on propagation of glitch metrics by table lookup shows a maximum error of 11.7 % in the last adder row. The results based on reproduction of the averaged glitch metric distributions per adder row are only slightly better at a maximum error of 9 % indicating that the largest contribution

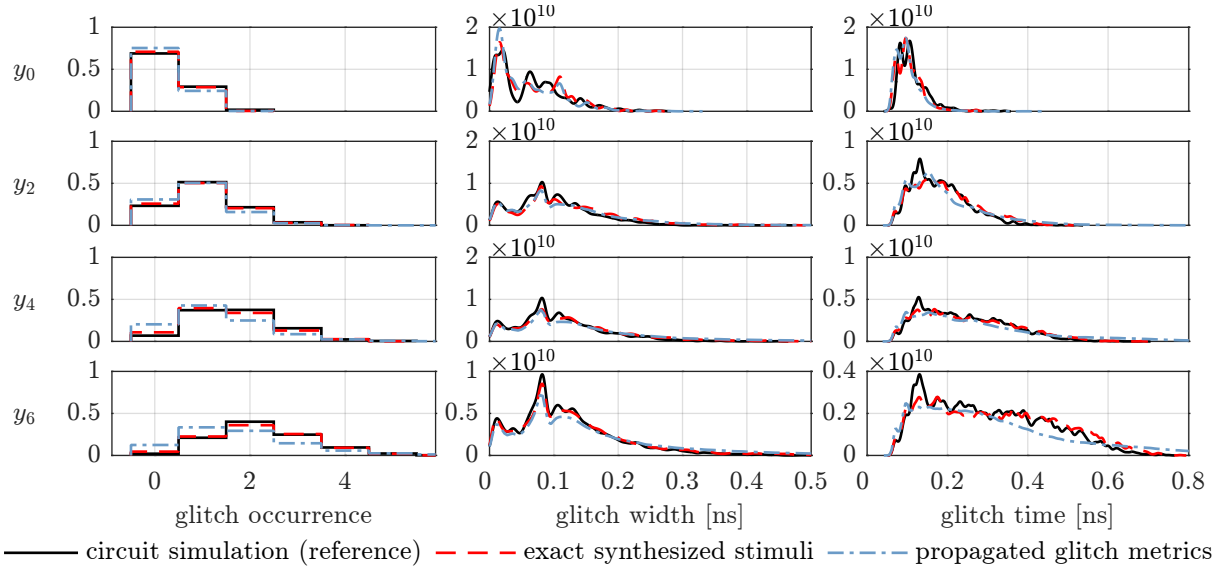


(a) carry-ripple based adder (b) carry-save based adder

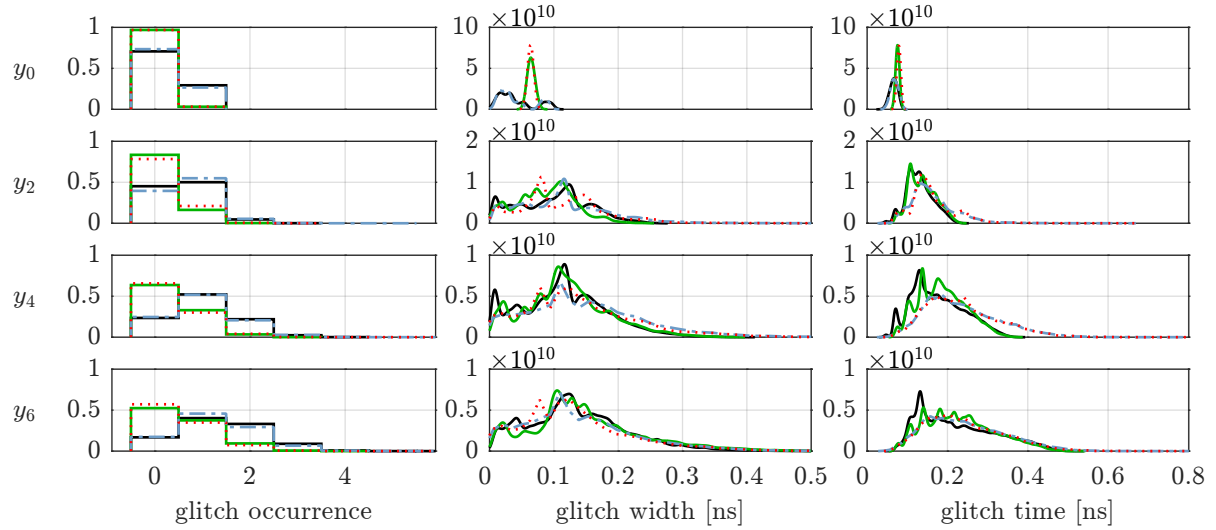
Figure 4.11: Average dynamic energy dissipation per row for multioperand adder benchmarks.

to the estimation error is caused by the simplifying assumption that all adder inputs per row share the same metric distributions. It is also worth noting that the dynamic energy consumed in the last row is four times higher than that of the first row which means that the energy dissipation would have been underestimated by a factor of 3 if glitching on the adder inputs had not been considered. The metric distributions at the adder outputs resulting from both sets of estimations plotted in Fig. 4.12a for every second row show a close match of both estimations to the reference distributions observed during circuit simulation.

In a second experiment, the adder array was modified as shown in Fig. 4.10b to feature carry-save arithmetic. In contrast to the carry-ripple based multioperand adder, two input words to each adder row might exhibit glitching and the related metric distributions might not be correlated. Consequently, the characterization effort increases dramatically because a total of twelve parameters need to be varied during building of the macro library. The estimation of dynamic energy dissipation performed according to the proposed macromodel results in a maximum error of 11.1 % as shown in Fig. 4.11b. The estimated glitch metrics of both outputs words for every second row are compared to the observed metrics during circuit simulation if Fig. 4.12b confirming the high degree of correlation to circuit simulation that can be expected of the proposed approach.



(a) propagation of glitch metrics in carry-ripple based adder



(b) propagation of glitch metrics in carry-save based adder

Figure 4.12: Demonstration of glitch metric propagation for multioperand adder testbenches.

4.5 Challenges of the proposed approach

While the results discussed in the previous section attest a general suitability of the proposed probabilistic macromodel for capturing glitching effects on energy dissipation, the approach is limited by the complexity of its precharacterization. Before an attempt is made to mitigate this disadvantage, in this section its impact as well as other factors potentially limiting accuracy are identified.

4.5.1 Size of parameter space

Even the simpler of both testbenches in the previously discussed application example features a six-dimensional parameter space. This complexity severely restricts the number of different values which can be characterized per parameter due to the exponential growth of possible parameter combinations. This situation can be somewhat mitigated by intelligently selecting the parameter combinations to be characterized based on prior knowledge about inter-parameter correlations but even such sparse characterization results in up to 4000 parameter combinations for the characterization of the single adder macro of the application example. Due to the statistical nature of the proposed parameters, each circuit simulation evaluating a single combination needs to be continued until convergence to typical results can confidently be assumed which takes several hundred simulation cycles. To make matters worse, this complexity is not restricted to the one-time effort of precharacterization but also affects the storage requirements for the lookup table and the runtime of the actual energy estimation due to the increased interpolation complexity when considering large numbers of dimensions. When estimating arbitrary circuits that are not as regular as the multioperand adders chosen in the previous section, the number of parameters and therefore characterization dimensions is even further increased by crucial metrics like transition slope and capacitive output load.

For such a circuit block with N independent inputs the exhaustive number of parameter combinations can be given as

$$n_{comb} = \mathrm{n}(C_{load}) \cdot \left[\mathrm{n}(t_{slope}) \cdot \prod_{m=1}^{3} \mathrm{n}(\mu_m) \cdot \mathrm{n}(\sigma_m) \right]^N \tag{4.11}$$

where the operator $\mathrm{n}(\cdot)$ denotes the number of possible values for each parameter, t_{slope} and C_{load} specify the transition slope and the capacitive output load, respectively, and μ_m and σ_m are the moments of the three glitch metric distributions. If only three possible

values for each parameter were selected, this would result in over 14 million possible combinations for a circuit block with two independent inputs when no pruning to sensible combinations is considered.

4.5.2 Averaging of glitch metrics for word-level signals

While the proposed macromodeling approach allows the handling of different glitch statistics on individual input bits of word-level component, this possibility is not even considered in the application example presented before because of its implications regarding the number of input parameters. Instead, the input parameters to the presented macromodel are chosen as the averaged glitch metric distributions of all bit weights to keep the number of parameter combinations in a manageable range. This simplification inevitable decreases accuracy especially for low bit weights. Again focusing on the carry-ripple adder, it can be shown that glitching activity saturates relatively fast due to averaging effects. In contrast, applying averaged glitch statistics to the lowest bits, which often show different activity statistics, will result in significant errors for those bit weights which may then propagate to higher weights of the adder. Additionally, the shift of occurrence times to higher values along the adder weights cannot be captured by averaged statistics.

One possible strategy to solve this problem could be an approach similar to the dual bit type modeling [35] where low bit weights and high bit weights are handled separately. On the other hand this approach would again increase the number of parameters. Another possibility would be to reduce the size of the characterized macros in order to reduce the potential differences of its inputs. This can in an extreme case lead to the modeling of single gates where no averaging is required but the advantage of encapsulating physical effects of internal interconnects is mostly lost.

A related problem is the fact that glitches on neighboring inputs and outputs in word-level signals are often correlated in time. This can easily be demonstrated by the example of the carry-ripple adder where glitches tend to ripple though multiple full adders which results in delayed glitches along neighboring outputs.

4.5.3 Correlation between glitch metrics

An additional source of errors is caused by the decomposition of mixture distributions into their components in order to build combinations of single normal distributions for table look-up. Since no correlation between glitch metrics is recorded, this routinely leads

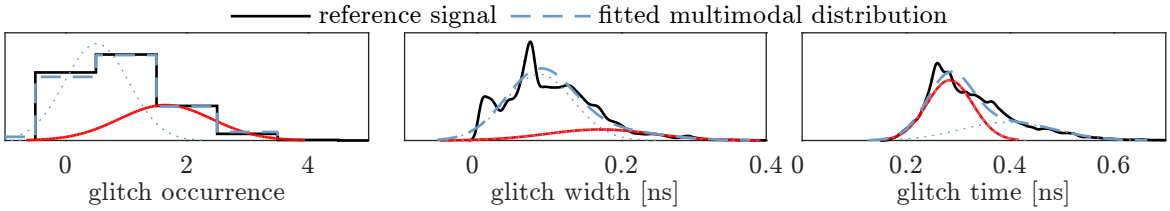


Figure 4.13: Example of decomposition of glitch metrics that results in physically impossible combinations.

to combinations that are physically impossible. Considering for example the metrics from Fig. 4.13 where each metric was fitted using two normal components, during table look-up, these mixture distributions are decomposed into eight combinations of single normal distributions. One of the problematic combinations is highlighted in red in Fig. 4.13. This combination specifies that approximately 2 glitches with a mean width of 0.2 ns are expected to occur within 0.2 ns from each other. While this in itself is highly unlikely, the tails of these distributions contain combinations that are virtually impossible. This problem increases with the number of components in each mixture distribution, as the components themselves tend to get narrower.

For the case of few component distributions it is possible to post-process the combinations to reduce the effect of the unlikely combinations. This post-processing step follows an algorithm that tries to retain the total probability for each component but distribute the probability of wide glitches towards combinations with small number of transitions and wide time distributions. That way improbable combinations are assigned less weight and are ideally ignored in the final recombination step of glitch estimation.

A more accurate alternative is to consider additional parameters that record correlation between the chosen metrics. As this would further increase the modeling complexity, this approach does not seem attractive. Instead, a desired feature is a reduced significance of correlation between different glitch metrics.

4.6 Reduction of modeling complexity

Obviously, while the proposed glitch metrics are accurate at capturing the characteristics of glitching, the huge complexity related to their direct implementation in macromodels calls for simplifications that trade accuracy for modeling complexity. There are several options, some of which will be discussed in the following chapter.

The most basic option is to discard the additional information resulting from the usage of distributions as metrics and use simple mean parameters instead. While this allows

a drastic reduction of parameters, the resulting accuracy would be poor. Depending on the metrics, it might be possible to use this simplification only on some (less important) metrics, but the benefit seems small nevertheless when compared to the expected impact on accuracy.

Instead, if selected metrics are determined to be marginally important, it might be a better choice to discard them completely. Unfortunately for the glitch metrics presented in the last sections all three metrics have significant influence on the energy dissipation and especially on the glitch metrics at the outputs of a macro block, so there is no potential for reducing parameters. A transformation of the parameter space using principal component analysis which tries to utilize the correlation between glitch metrics also fails at providing simplified metrics.

Therefore, one strategy is to find alternative metrics that might not be as accurate as the original ones but allow easier simplification. Such approximate metrics will be discussed in the next section.

Section 4.6.2 presents a simplification for discrete distributions which does not require fitting of mixture distributions and manages to improve accuracy while at the same time reducing the number of parameters.

An approach that keeps most of the information of the original distributions while requiring far less parameters tries to describe the distributions as a sequence of discrete windows that are uniformly distributed each. This method will be discussed in more detail in Section 4.6.3.

4.6.1 Alternative glitch metrics

The metrics presented so far are based on *glitches*, which consist of (undesired) signal pulses. The activity due to glitches increases the dynamic power consumption that would be observed if each node would flip at most once during each period. In order to estimate the total power consumption not only the glitch parameters have to be varied during characterization but the switching activity due to desired logic evaluation has to be considered as well which adds another dimension to the characterization space.

Therefore, a way to reduce the number of required parameters is to extend the metrics to include both desired and undesired logic switching. This can be accomplished by considering *transitions* instead of *glitches*, where two transitions that are close enough

can be regarded as a glitch. This means, that the original power function

$$P_{macro} = f(D_0, t_{slope}, C_{load}, m_{glitch}) \tag{4.12}$$

is simplified to

$$P'_{macro} = f(t_{slope}, C_{load}, m_{transition}), \tag{4.13}$$

where D_0 is the zero delay transition probability at the input nodes, t_{slope} is the signal slope at the inputs, C_{load} specifies the capacitive loading at the output and the remaining parameters describing the glitches or transitions are summarized as m_{glitch} and $m_{transition}$ respectively. As an additional benefit the generation of glitches due to delayed arrival of inputs can be modeled more intuitively if the characteristics of desired logic switching are captured by distributions as well.

Possible parameters based on transitions closely relate to the parameters introduced for glitches. Probably the most important parameter is the number of transitions per period which will be called *transition occurrence* and combines the glitch occurrence with transitions due to logic evaluation. The *transition time* is measured as the point in time the transition starts relative to the start of the period and is equivalent to the *glitch time*. The third parameter which was previously defined as the *glitch width* can be replaced by the *signal swing* of each transition which for the case of a glitch pulse is defined by the distance between neighboring transitions, assuming an approximately constant signal slope.

An alternative to counting the number of transitions in each period is to accumulate the total signal swing traversed in each period or alternatively only the total signal swing of rising transitions. This metric would directly relate to the power consumed due to loading and unloading of node capacitances

$$P_{node,i} = \alpha \cdot f \cdot C_{node} \cdot V_{DD} \cdot \Delta V_i, \tag{4.14}$$

where ΔV_i is the accumulated signal swing of rising transitions.

Fig. 4.14 gives examples for extracted distributions of all proposed metrics for the same input signal. Both the distribution of signal swing and that of accumulated signal swing per period feature the disadvantage of narrow peaks. These peaks are caused by transitions due to logic evaluation which are included in the statistic predominantly feature the full signal swing. These peaks cannot be modeled adequately by mixture distributions

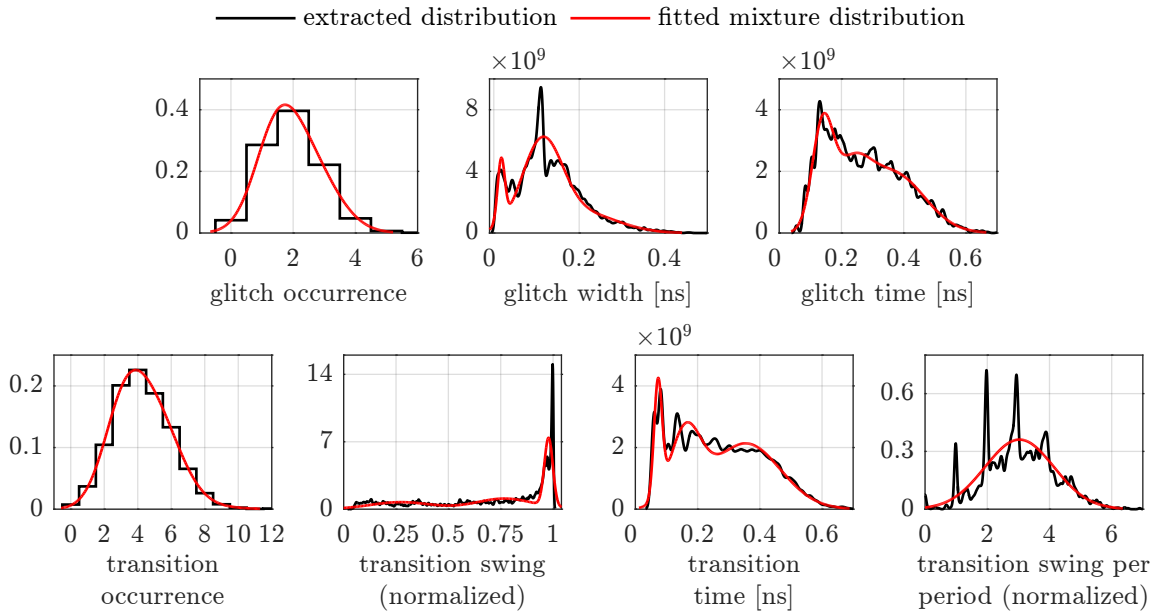


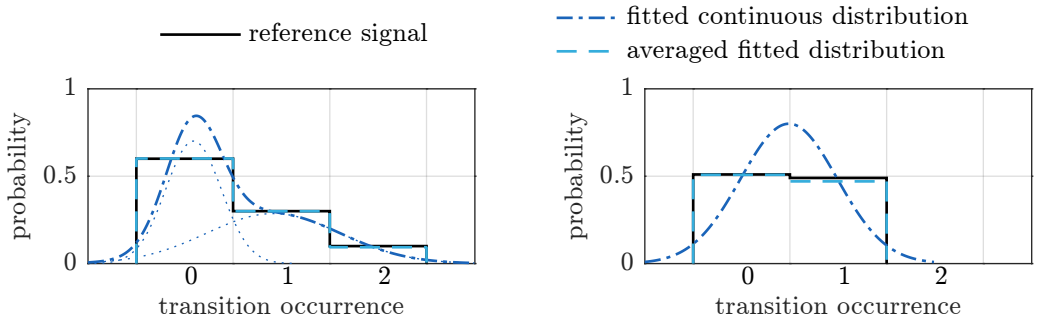
Figure 4.14: Example glitch and transition metrics for the same signal.

which tend to feature low pass characteristics. This disadvantage in combination with the overall goal to minimize the amount of parameters considered in the macromodel motivates the decision to completely ignore the parameters related to the signal swing. This does not necessarily hurt accuracy if there is a strong correlation between the metrics. This correlation that led to errors during decomposition of the glitch-based metrics can reasonably be assumed to determine the signal swing of transitions to a certain degree when transition occurrence and transition time are known.

4.6.2 Simplification of occurrence metrics

The metric distributions proposed so far are approximated using fitted mixture distributions. This is appropriate for occurrence times or glitch widths, which feature continuous distributions. In contrast, glitch or transition occurrence specify the discrete number of glitches or transitions that occur in a specified period. While it is possible to handle these discrete distributions in the same way as the continuous distributions by fitting normal components and averaging over intervals of width “1” as has been done in Fig. 4.15, this might not be the most efficient method.

If both signals from Fig. 4.15 are applied to a characterized gate, according to the decomposition approach described in Section 4.2 all combinations of single normal distributions are assembled and the resulting power consumption and glitch metrics at the output of the gate are looked up (or interpolated) from the characterization table. In



(a) input a (b) input b

Figure 4.15: Example of occurrence distributions for a two-input gate.

μ_a	σ_a	μ_b	σ_b	% cycles
0.86	0.70	0.49	0.5	54.3 %
0.07	0.25	0.49	0.5	45.7 %

(a) combinations based on fitted normal distributions

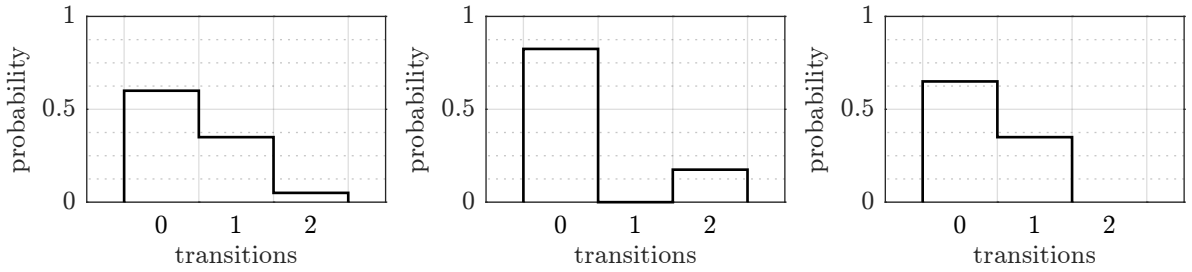
# transitions a	# transitions b	% cycles
0	0	30.6 %
0	1	29.4 %
1	0	15.3 %
1	1	14.7 %
2	0	5.1 %
2	1	4.9 %

(b) combinations based on number of transitions

Table 4.1: Comparison of transition parameter combinations.

order to simplify this example, only the transition occurrence distribution is considered. As shown in the figure, input a is fitted using two normal components while input b can accurately be described by only one component. This results in the two combinations consisting of four parameters (two parameters per metric) as shown in Table 4.1a. The weight of each combination is computed as the product of the mixing factors of the normal components used in the combination. This weight is equivalent to the percentage of switching cycles in a synthesized waveform that would feature the given normal distributions.

As the distributions are discrete, a more intuitive way of representing them is to simply list the full *probability mass function* (PMF). The PMF of input a consists of three states (0, 1 or 2 transitions) with their respective probabilities while the PMF of signal b has only two possible states (0 or 1 transition). Building combinations of possible numbers of transitions on input a and input b as shown in Table 4.1b results in the joint probability for the inputs to feature the specified number of transitions in the same period under the



(a) input b (b) output c_{out} $(a = 0, c_{in} = 0)$ (c) output c_{out} $(a = 1, c_{in} = 0)$

Figure 4.16: Dependence of number of transitions on output c_{out} on signal state of non-switching inputs a and c_{in} for a full-adder.

assumption that they are uncorrelated.

Compared to the representation using fitted mixture distributions this results in more combinations. On the other hand, each combination is defined by only two instead of four parameters. Due to the exponential dependency of the characterization space on the number of parameters this advantage results in huge savings during precharacterization. In addition, because the parameters based on the PMF are discrete values on a very limited range, all possible parameter states can be characterized exhaustively. During evaluation table lookup can be performed without requiring interpolation which increases accuracy and speeds up the runtime. This compensates the disadvantage of more combinations. In contrast, the parameters resulting from fitted mixture distributions feature rational values which require interpolation in virtually all lookups because only a selected number of parameter values are included in the characterization library.

Another advantage of PMF-based parameters is the fact that the exact probabilities can be used during evaluation. Fitting of mixture distributions (with a limited number of components) always leads to some fitting errors that degrade accuracy.

An important point which has to be considered for characterization using PMF-based parameters is the *signal state* of the inputs during characterization simulations. Contrary to characterization using fitted distributions, the number of transitions of each signal is constant in each period for the characterization of PMF-based parameters. This means that a signal remains in the same steady state for the whole simulation, if an even number of transitions (or no transitions at all) are performed in each period. This can lead to significant errors if the steady signal state during evaluation differs from that assumed during characterization as both the power consumption and the output transitions might depend on the state of the inputs. Fig. 4.16 demonstrates this dependency for a single full adder where only input b is switching according to the distribution shown in Fig. 4.16a

while the remaining inputs a and c_{in} remain constant. Depending on the state of the constant inputs, the distribution of the transitions on output c_{out} is fundamentally different as can be seen in Figs. 4.16b and 4.16c.

Simply assuming equal probability for steady signal states of all signals and partitioning the characterization simulations accordingly will lead to errors for all cases where the probability for either one state is significantly higher than for the other. This can easily be accommodated into the modeling approach by introducing a new parameter per input that defines the steady state for this signal. Even though this increases the size of the characterization space, the overhead is acceptable because each of the additional parameters can only assume two distinct values (steady state of "0" or "1"). The characterization can even be optimized further as the definition of a steady state is only required for an even number of transitions per period.

In order to generate reliable estimates, these steady state probabilities are also required during evaluation of the model. Fortunately, steady state probabilities can easily be calculated in a simple zero delay analysis of the circuit under investigation.

To demonstrate the improvement in terms of complexity due to PMF-based occurrence parameters, Table 4.2 lists example parameters for both fitted normal distributions and a PMF-based analysis of a 3-input gate. Each input is assumed to exhibit at most four transitions in one period. In order to span this region with mixture distributions, four values for mean as well as for standard deviation have been selected. As it is most unlikely that a fitted distribution matches these parameters exactly, interpolation techniques are used during look-up to evaluate points in the characterization space that were not simulated directly. For PMF-based parameters on the other hand only one parameter is required to determine the fixed number of transitions per period, but the unlikely case of four transitions per period has to be considered in a separate characterization run resulting in five possible values. The signal states of each signal form additional parameters that can assume two values each. Signal states as additional parameters could also be introduced for parameters based on fitted distributions but are not as important as for PMF-based parameters because in most characterizations the input nodes feature varying numbers of transitions in different periods.

In order to populate the complete look-up table for this simple example, four times the number of simulations are required for fitted distributions than for PMF-based parameters. At the same time the characterized grid for fitted distributions remains coarse and thus requires interpolation schemes during evaluation while the characterization for

fitted normal distributions	fixed number of transitions
occurrence distribution μ	*transitions per time interval*
signal a: 0, 0.5, 1, 2	signal a: 0, 1, 2, 3, 4
signal b: 0, 0.5, 1, 2	signal b: 0, 1, 2, 3, 4
signal c: 0, 0.5, 1, 2	signal c: 0, 1, 2, 3, 4
occurrence distribution σ	*signal state*
signal a: 0, 0.25, 0.5, 1	signal a: 0, 1
signal b: 0, 0.25, 0.5, 1	signal b: 0, 1
signal c: 0, 0.25, 0.5, 1	signal c: 0, 1
$\Rightarrow$ 4096 parameter combinations	$\Rightarrow$ 1000 parameter combinations

Table 4.2: Comparison of transition occurrence parameter spaces.

PMF-based parameters includes all possible combinations.

4.6.3 Simplification of time metrics

The discrete nature of the transition metric allows for considerable reduction of the model complexity without sacrificing accuracy by focusing on the probability mass function. In contrast, the occurrence time of a transition is a continuous metric which precludes the simplification using an analogous approach. Instead, an alternative approximation of the transition time distribution is introduced which allows further simplification of the characterization process. The accurate approximation using fitted normal mixture distributions which was introduced at the beginning of this chapter is replaced by a piecewise linearization of the cumulative probability function. This results in an approximation of the probability density function by segments that are uniformly distributed. In consequence, transition times are assumed to be evenly distributed inside each *time window* which results in a simplified representation of the original distribution as demonstrated in the upper row of Fig. 4.17.

While the linearization of the probability function itself does not feature significant advantages, this representation allows for separate analysis of each time window. Only transitions that occur very close the window borders might cause correlations between consecutive time windows. Therefore, a separate lookup of energy dissipation and output transitions for each window appears to be a reasonable simplification. Instead of a single distribution of transition occurrence for the complete switching cycle, separate distributions are defined for each time window as shown in the lower row of Fig 4.17. When the width of the time windows is fixed, this only requires characterization of different transition counts which considerably limits the size of the model library that needs to be

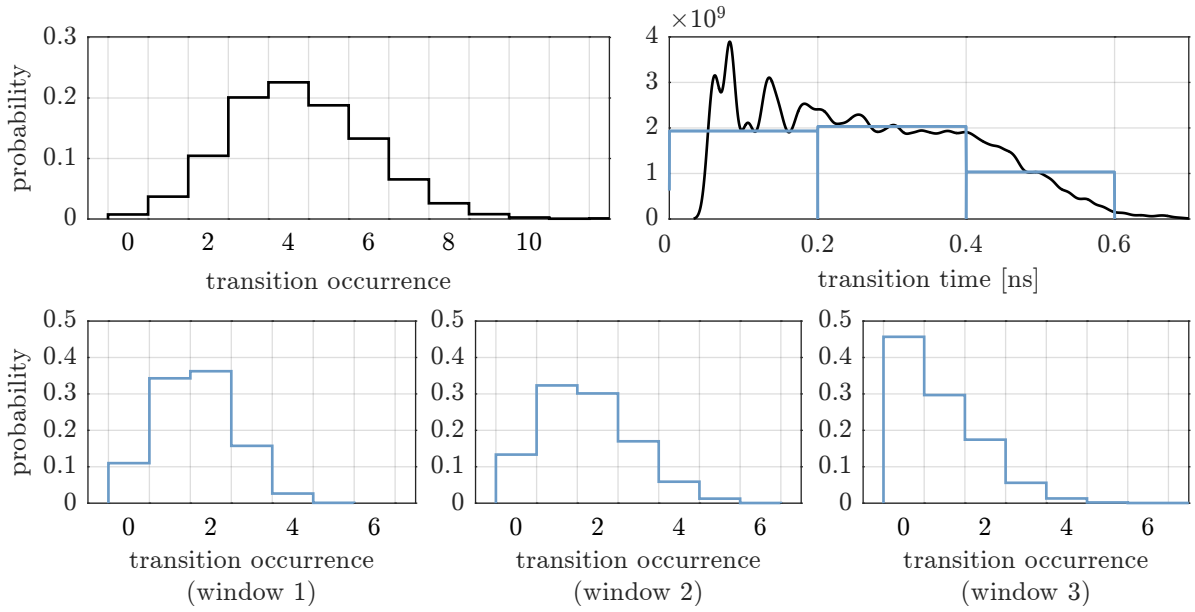


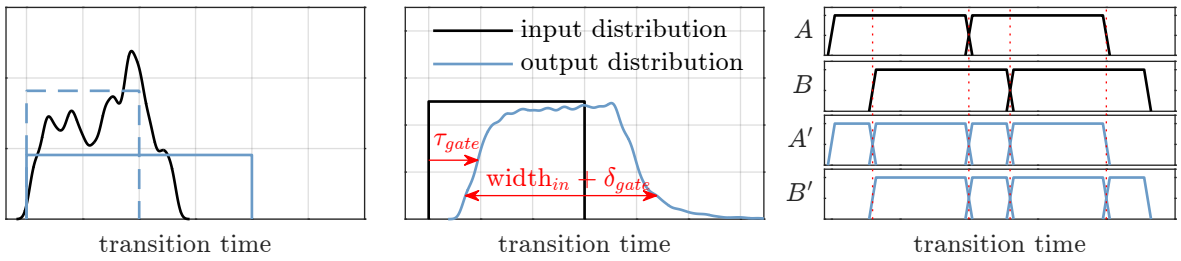
Figure 4.17: Demonstration of simplified statistics where transition time distribution is approximated by windows in which occurrence times are assumed to be uniformly distributed.

generated.

Challenges for windowed time statistics

The assumption of uniformly distributed occurrence times during a time window will not be satisfied in most cases. Fig. 4.18a demonstrates the important case that the original time distribution plotted in black is significantly narrower than the assumed uniform time window plotted as a solid blue line. This problem occurs in an extreme case for signals that switch at quasiconstant points in time which is the case for primary inputs and outputs of registers. Because only a limited number of window widths can be characterized, this problem will also occur for windows that feature widths in-between characterization points. However, this problem can be mitigated by characterizing different window widths in a representative range as suggested by the narrower time window plotted as a dashed blue line and interpolating the results during look-up.

A challenge that is more severe is the fact that the transition time distribution at the output typically does not share the same shape as the transition time distribution at the input as demonstrated in Fig. 4.18b. This transformation generally consists of a time shift τ_{gate} depending on the gate delay and a widening of the window by an offset δ_{gate} due to delay variations for different input combinations. As the signal statistics which are looked up for one component are inputs to another model in a subsequent iteration of the



(a) uncharacterized widths (b) transformation of windows (c) alignment of windows

Figure 4.18: Challenges for separation of the transition time distribution into windows

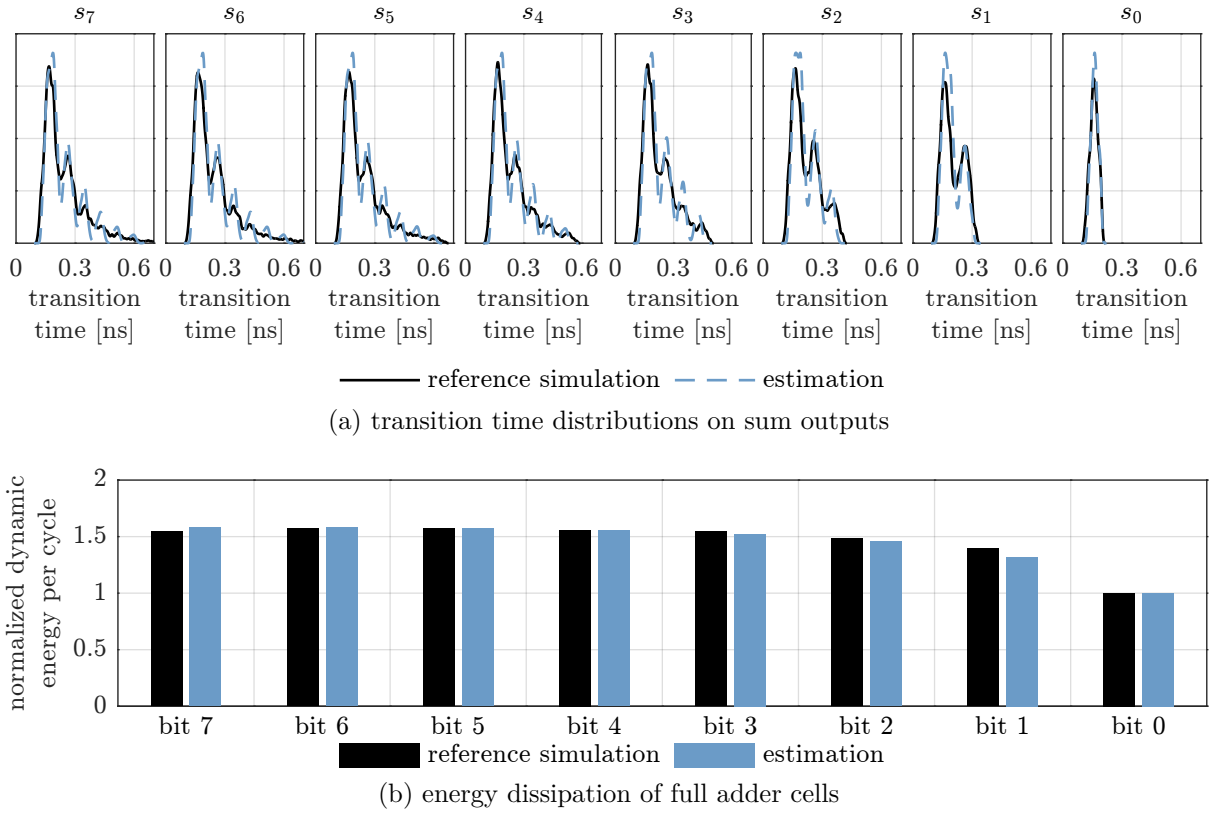
estimation, consecutive widening can result in time windows that exceed the characterized range. This needs to be prevented by splitting of wide windows into smaller parts.

The more dramatic effect of these transformations of time windows is the observation that windows on different inputs of the same gate cannot be expected to be aligned as exemplified in Fig. 4.18c for two signals A and B. This results in the need for a preprocessing algorithm that splits and merges windows on the individual inputs until the assumption of matching windows on all inputs is met as shown for the modified windows for A' and B'.

One important limitation of the windowed approach follows directly from the assumption that the effect of each window can be handled separately. In order to satisfy this assumption, the size of the macro to be modeled is severely restricted. The worst case delay needs to be relatively small compared to the window width because this limits the overlapping of transition time windows at the output due to signal delay variations. On the other hand, the window width needs to be kept small to allow for accurate approximation of the original distributions. These restrictions lead to the conclusion, that the approach based on windowed statistics is particularly useful if applied to individual gates instead of compound macros.

4.6.4 Demonstration of simplified macromodels

To demonstrate the modified estimation approach using occurrence parameters based directly on the probability mass functions and transition time distributions split into windows, the same multioperand adder that was evaluated for the original glitch metrics is analyzed. In contrast to the estimation based on fitted mixture distributions of glitch parameters, the size of the characterized macro was reduced to a single full-adder to restrict the influence of overlapping windows for the looked up statistics at the macro



(a) transition time distributions on sum outputs

(b) energy dissipation of full adder cells

Figure 4.19: Estimation results for a single carry-ripple adder using windowed statistics

outputs.

Analysis of a single carry-ripple adder

The first demonstration focuses on the topmost carry-ripple adder row of the multi-operand adder shown in Fig. 4.10a. All bits of both input operands are assumed to switch independently with a probability of 50 % per cycle in a short time window of width 50 ps at the start of each period. This way, the inputs are two uncorrelated streams of uniformly distributed numbers spanning the full wordlength. The results of this experiment are displayed in Fig. 4.19 for both the reconstructed transition time distributions of the sum outputs and the resulting energy dissipation of each full adder. The time distributions of higher bit weights exhibit transitions at increasingly later times due to carry propagation which is closely matched in the proposed estimation. The maximum error of the dynamic energy dissipation on cell-level is −5.1 % for *bit 1* while the full row is underestimated by only 0.7 %.

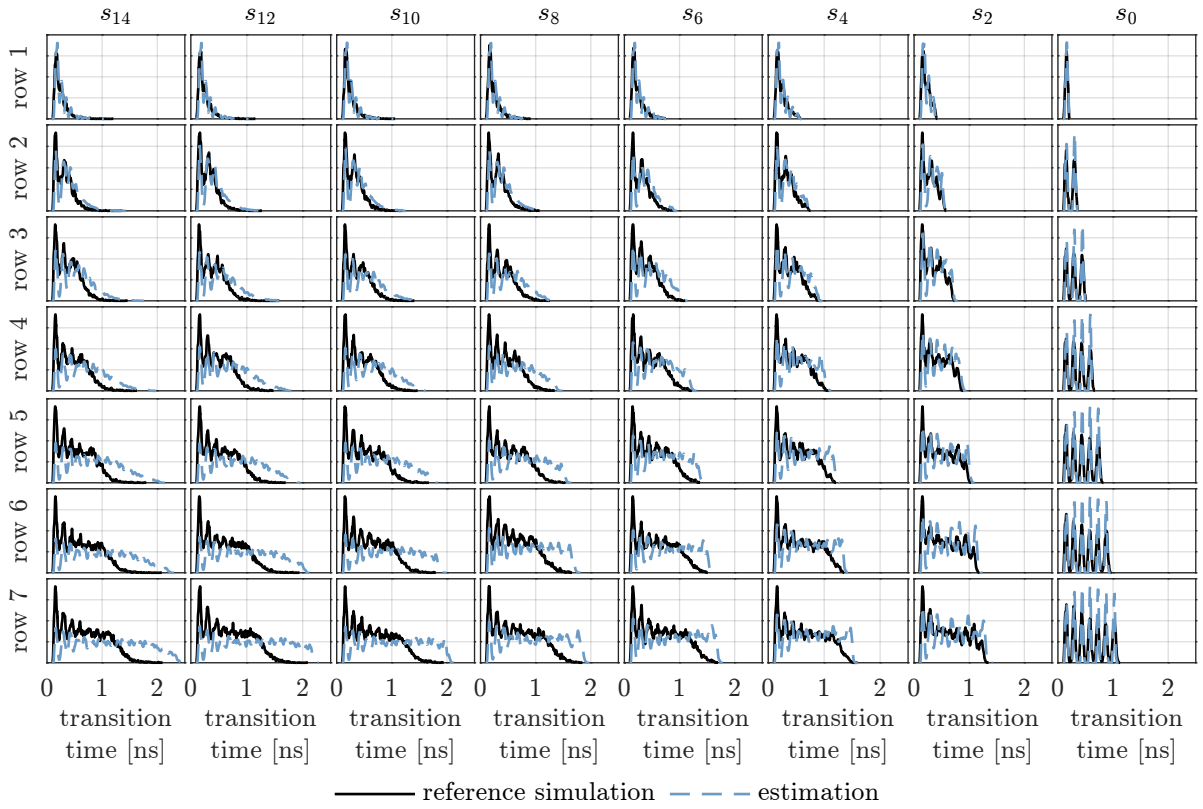


Figure 4.20: Simulated and estimated distributions of transition occurrence time for every second bit weight of sum output in a multioperand array adder.

Analysis of multioperand adder

In the next step, instead of a single carry-ripple adder, the full multioperand adder for 8 input words is analyzed. In addition to the number of rows, the wordlength is increased to 16 bit. Due to the higher logic depth the accurate propagation of transition metrics becomes more important.

In the carry-ripple adder analyzed in the last section, only the transition metrics of the carry signal needed to be propagated. When analyzing the full multioperand adder, the accumulated sum of each row features glitches and delayed transitions as well. In order to align the transition windows of signals on multiple inputs of the same gate, an algorithm that tries to minimize the error introduced due to splitting and merging of overlapping windows was implemented. This algorithm needs to consider both the case that two windows of the same input overlap due to widening of the windows in previous gates as well as the case that multiple inputs feature windows that are misaligned in time.

Fig. 4.20 shows the transition time distributions resulting from the approach employing windowed time statistics. This figure demonstrates the major weakness of the presented approach to capture the propagation of transition metrics over a sufficient logic depth.

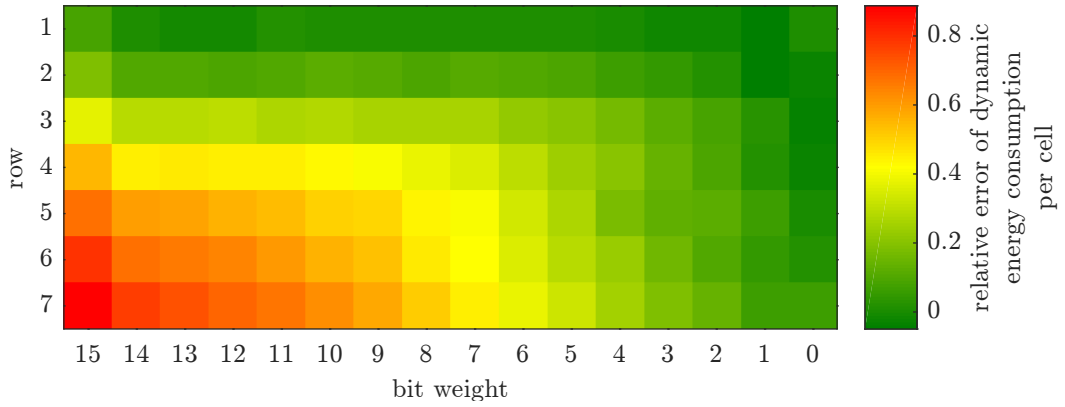


Figure 4.21: Relative error of estimated dynamic energy per full adder between estimation result and reference simulation.

While the estimated distributions of the first bit weights and the first adder rows closely match the reference statistics, starting at the third row the differences start to exceed the tolerable range. This is mainly due to the fact that the maximum transition occurrence time is continuously overestimated resulting in a self-accelerating flattening of the distributions. This overestimation of transitions also results in an excessive overestimation of dynamic energy dissipation for cells in higher adder rows and bit weights of up to 89 % as demonstrated in Fig. 4.21. These propagation failures cause the overall estimation error of the full multioperand adder to be as high as 33 %.

4.7 Handling of variability

The discussion of extended macromodels in this chapter did not consider variation so far. However, due to the probabilistic nature of the chosen parameters, these models lend themselves perfectly to the estimation of energy dissipation under the influence of process variation or variation of operating conditions. In fact, the estimation methodology itself does not need to be changed at all. The main changes are concerned with the characterization phase. Instead of determining the output signal parameters and energy dissipation only for the selected corner case, this characterization simulation needs to involve a number of separate runs for different choices of variation parameters. The output signal parameter distributions can be merged to reflect the possible variations in glitches or transitions without requiring changes in the internal representation. In contrast, the dynamic energy dissipation which was previously characterized as a scalar value needs to be replaced by a probability distribution as well under the influence of variations. These energy distributions can be handled in the same way as the parameter

distributions and, therefore, benefit from the full set of interpolation and transformation algorithms which are already developed.

This modified estimation approach results in a probability distribution for the energy dissipation of each analyzed macro block. From these individual energy distributions the bounds of dynamic energy dissipation of the full circuit can be calculated.

4.8 Comparison of probability macromodel approaches

The goal of this chapter was to improve library lookup based macromodels by the inclusion of accurate glitching parameters. Because of the statistical nature of glitches, these parameters themselves need to be probabilistic. The identified glitch metrics based on probability distributions of occurrence count per cycle, occurrence time and glitch width proved to allow accurate characterization. In combination with parameters based on fitted mixture distributions that where decomposed during library lookup this approach was shown to enable accurate estimation of dynamic energy dissipation considering glitches. However, generalization is hampered by the high complexity induced by the large number of model parameters which affects both model characterization as well as library lookup employing interpolation.

Aiming for a reduction of complexity, alternative metrics based on transitions instead of glitches were proposed. Additional simplifications targeting the representation of the probability distributions allowed further reduction of complexity at the cost of restricting the model size to individual gates. While simplifying both the characterization process as well as the estimation flow, these simplified metrics representations were shown to be applicable to the estimation of small macros. However, when targeting larger macros exhibiting higher logic depths, the accumulation of inaccuracies caused the proposed simplifications to fail at achieving sufficient estimation accuracy.

The consequent approach at this point would be to try to balance the simplifications with regard to the achievable accuracy by applying only selected changes as proposed in the last section. However, instead of this incremental improvement of the probabilistic macromodels, the next chapter will focus on a fundamentally different approach which is more closely related to conventional gate-level power estimation but aims at improving the handling of glitches.

5 Event-based lookup of power waveforms

As shown in chapter 3, current state-of-the-art gate-level power estimation flows suffer from a number of limitations that result in inaccuracies compared to physical circuit simulation. The main reasons for these inaccuracies are:

- Logic simulation which is performed to capture switching activity cannot account for partial swing glitches.
- Glitch filtering, especially for complex multistage gates, cannot accurately be implemented using standard logic simulators.
- Multi-input events are not included in the characterization library and therefore their effect can only be coarsely approximated.

The last shortcoming could be mitigated by the adoption of a more complex characterization that would not be restricted to single-input events but would include multi-input events as well. Because the timing offset between switching on multiple inputs is of crucial importance when estimating the energy consumed by such a multi-input event, a significant number of different offsets would have to be considered during characterization. This change would be costly in terms of characterization runtime and resulting size of the lookup library but could potentially reduce the estimation errors with only a small overhead in estimation runtime.

In contrast the simplified handling of glitches cannot be fully corrected by pre- or post-processing of the data relevant to the logic simulator. A method proposed by Dietrich and Haase [15] appends information on the transition slopes to each change of the signal states which requires deep changes to the hardware description of the circuit as well as the logic simulator. Using a customized precharacterization of the fundamental gates, this allows for more accurate filtering of glitch pulses. In addition, pulses that are prevented

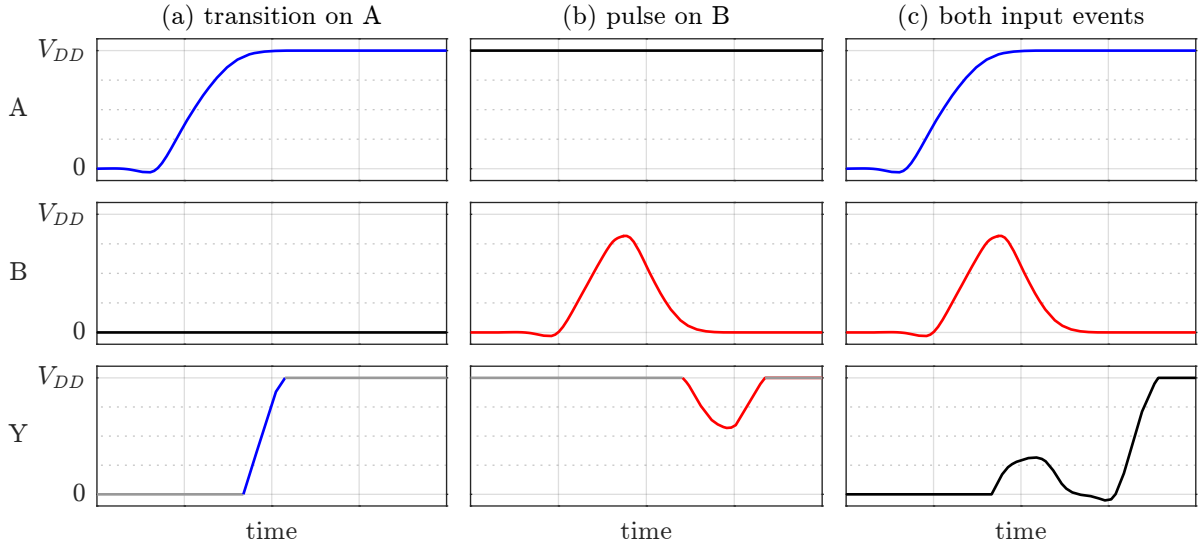


Figure 5.1: Output response of a two-input XOR gate to different input events demonstrating the need for multi-input characterization.

from propagation during logic simulation are nevertheless stored and used in a post-processing step which considers the energy consumed due to partial swing pulses. While this method significantly slows down the logic simulation due to the custom handling of signal transitions it still cannot adequately handle correlations between multiple quasi-simultaneous transitions on different inputs especially when one of the inputs features a partial swing glitch pulse. This problem is demonstrated in Fig. 5.1, which shows the inputs A and B of a two-input XOR gate along with the output Y for three different event combinations on the inputs. The first and second column show the gate response to a full transition or a short pulse on one of the inputs, respectively. The third column demonstrates that the response to near-simultaneous stimulation with both input events differs significantly from the superposition of the individual responses.

In order to capture complex delay effects leading to partial masking of output switching or the generation of glitches, more physical information on the switching waveforms is required than can be produced by purely logic simulation. On the other hand, physical-level circuit simulation at SPICE-level accuracy is much too time-consuming to be applicable to realistic circuit sizes. The approach proposed in this section [66] combines the runtime advantages of lookup-based power estimation relying on precharacterization with the accuracy of analog signal waveforms. It extends the power lookup tables by including multi-input events with varying switching offsets as well as events involving glitches on selected inputs. In addition, the analog waveform response at the output is stored in the library for each input event combination along with the supply current waveform which

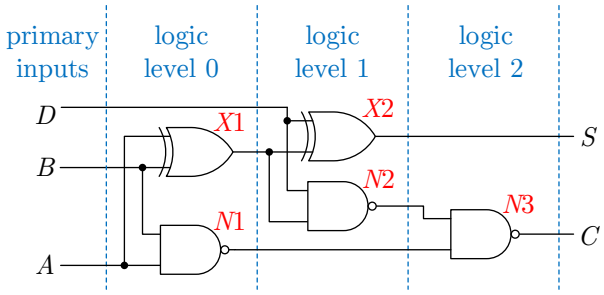


Figure 5.2: Decomposition of a simple circuit into logic levels.

defines the power consumption. This output waveform can be used as an accurate input to the power estimation of the next gate, thereby allowing iterative progression of the estimation across the whole circuit. This completely removes logic simulation from the estimation flow and, therefore, avoids all error sources related to suboptimal handling of glitch creation and propagation during simulation.

5.1 Lookup-based estimation of output waveforms and gate power

Once the characterization library is generated for a selected number of gates, it can be used for power estimation of all circuits that are constructed from any combination of gates in the library. While the following discussion will focus on purely combinatorial circuits, this approach can easily be extended to general circuits containing sequential cells, as will be demonstrated in chapter 5.5. For the proposed approach to work, the signal waveforms at the primary inputs need to be known. If only statistical information is known about primary inputs, suitable input waveforms can easily be constructed using random sampling before applying the estimation approach. This is the same requirement that is necessary for meaningful calculation of switching activity as performed during commercial gate-level power estimation.

Prior to the estimation, the circuit to be analyzed is preprocessed and all gates are sorted into *logic levels* depending on the signals connected to their input ports. Fig. 5.2 shows an example of this decomposition of a simple circuit. Gates in logic level 0 are only dependent on primary inputs. Logic level 1 contains gates with at least one input port connected to an output port of a gate in logic level 0. This scheme is continued until all gates have been assigned a logic level. This way, gates in the same logic level can be estimated in parallel while the logic levels are processed sequentially.

The estimation for each gate is performed in multiple steps:

1. The signal characteristics of all nets connected to the inputs of the current gates are retrieved from the local database. These characteristics list switching times together with the respective signal slope and the width of glitch pulses.

2. Switching events on different inputs that occur in close temporal proximity are grouped into event combinations.

3. The supply current as well as the output voltage response related to each event combination is looked up from the characterization library.

4. The potentially overlapping voltage and current responses are superimposed. The resulting supply current waveform can be used to calculate the dynamic power consumption of the current gate.

5. The voltage waveform at the output is analyzed to retrieve the relevant signal characteristics which are stored in the local database for use in subsequent iterations.

In the following section, these individual steps will be discussed in more detail.

5.1.1 Signal characteristics of event-based power waveform lookup

Contrary to the probabilistic approach described in chapter 4, the method discussed here is similar to the functionality of logic simulators in that individual *events* on circuit nodes are considered sequentially. However, in contrast to logic simulation, an event is not purely defined by a change in the logic state at a specific time, t_e, but is annotated with more detailed characteristics. One shortcoming of logic simulation that prevents accurate pulse filtering is the abstraction from the finite transition slope, s_e. In the proposed approach this information is retained and annotated to each switching event. When considering glitch propagation, the second important property is the shape of a pulse. In the proposed approach, the pulse shape is characterized by its width, w_e, because this way partial-swing glitches and full-swing glitches can be equally described and the pulse height can easily be determined if the transition slope is already known. The signal characteristics recorded for each event therefore consist of the logic state prior to the event, l_e, the time offset relative to a defined reference, t_e, the transition slope, s_e, and the pulse width, w_e, if applicable and will be denoted by the 4-tuple (l_e, w_e, t_e, s_e) where w_e is set to "-1" for full-swing transitions to differentiate from a glitch pulse with negligible width. An example of this

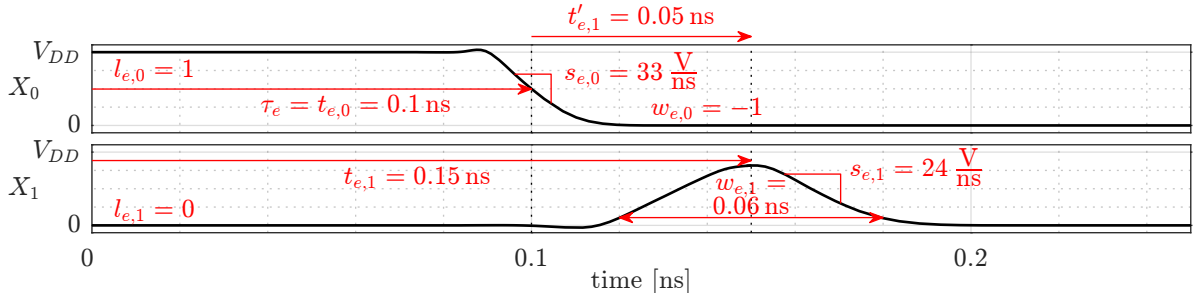


Figure 5.3: Annotated characteristics for two switching events.

annotation of characteristics to each switching event is depicted in Fig. 5.3 for both a full-swing transition and a partial-swing pulse. Event times are defined at 50 % signal swing or at the maximum of the pulse respectively, pulse widths are measured at 10 % of full signal swing and slopes are considered from 10 % to 90 % thresholds.

During the analysis of a signal, it is decomposed into switching events that correspond to the events considered during library characterization. A crucial parameter is the threshold that determines whether two consecutive full-swing transitions are considered as part of one glitch pulse or as two separate events. Especially for complex multistage gates and multi-input events the consideration of full-swing glitches compared to the concatenation of two separate transitions can result in a large gain in accuracy due to the increasing importance of interactions. This threshold must therefore be chosen carefully to achieve the desired accuracy without exceeding the tolerated characterization effort. A suitable compromise was found to be an upper glitch width of the maximum of either the gate delay or four times the transition time.

5.1.2 Grouping into event combinations

For a gate with multiple inputs all combinations of individual events on different inputs could be considered. Since the influence of each input event on the gate output and supply current is limited in time to a gate-specific interval δ_t, such interdependencies between events on multiple inputs only need to be considered for events with occurrence times within a maximum distance of δ_t from each other. Each event combination, K_e, is then defined as the ordered list of events on all N gate inputs

$$K_e = \Big((l_{e,0}, w_{e,0}, t_{e,0}, s_{e,0}), (l_{e,1}, w_{e,2}, t_{e,1}, s_{e,1}), \ldots, (l_{e,N-1}, w_{e,N-1}, t_{e,N-1}, s_{e,N-1})\Big) \quad (5.1)$$

The case that input i remains in a stable state during an event combination can be

conveniently denoted as $w_{e,i} = 0$ without changing the notation. In order to derive a unique representation of the event combination that corresponds to the characterized library, all time offsets are defined relative to the earliest event in the combination with the event combination offset, τ_e, defined as

$$\tau_e = \min_{i=0,\ldots,N-1} t_{e,i}, \tag{5.2}$$

$$t'_{e,i} = \begin{cases} t_{e,i} - \tau_e & \text{if } w_{e,i} \neq 0 \\ 0 & \text{if } w_{e,i} = 0 \end{cases} \qquad i = 0, \ldots, N-1. \tag{5.3}$$

Instead of N individual transition slopes, $s_{e,i}$, annotated to each event, the average transition slope, $\overline{s_e}$, of all switching inputs during an event combination is used for library lookup. This simplification results in a certain loss of accuracy for inputs featuring large differences in signal slope but allows for a significant reduction in the dimension of the lookup tables. Since the logic states, $l_{e,i}$, at the beginning of the event combination are limited to the parameter values "0" or "1", they can be used as coefficients in a binary weight vector to form the combined decimal equivalent input state

$$L_e = \sum_{i=0}^{N-1} l_{e,i} \cdot 2^i. \tag{5.4}$$

The simplified event combination used during library lookup is therefore defined as

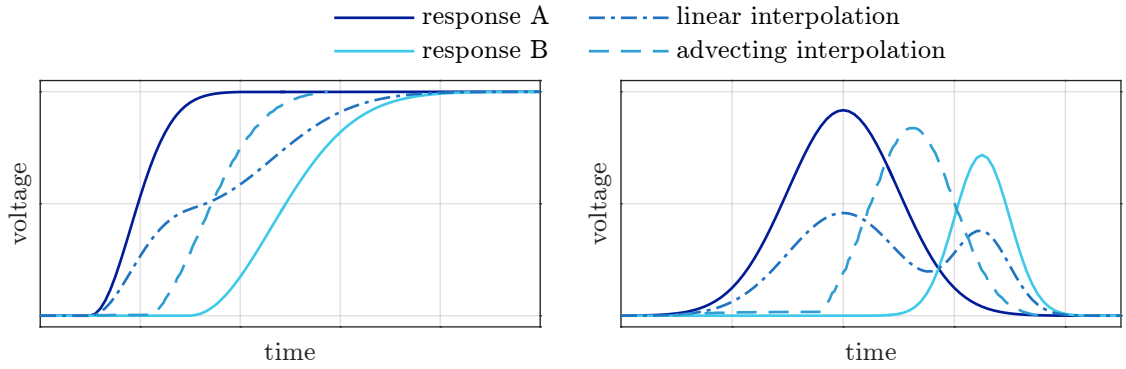
$$K'_e = \left[\tau_e, \overline{s_e}, L_e, \Big((w_{e,0}, t'_{e,0}), (w_{e,1}, t'_{e,1}), \ldots, (w_{e,N-1}, t'_{e,N-1})\Big)\right], \tag{5.5}$$

where the last parameter is a list of tuples specifying the pulse width, $w_{e,i}$, and event time offset, $t'_{e,i}$, for each of the N inputs.

For the example depicted in Fig. 5.3 the event combination features a time offset $\tau_e = \min\{t_{e,0}, t_{e,1}\} = 0.1\,\text{ns}$ with $t'_{e,0} = t_{e,0} - \tau_e = 0\,\text{ns}$ and $t'_{e,1} = t_{e,1} - \tau_e = 0.05\,\text{ns}$. The average transition slope for the example is calculated as $\overline{s_e} = \frac{1}{2}(s_{e,0} + s_{e,1}) = 28.5\,\frac{\text{V}}{\text{ns}}$ and the combined logic state is $L_e = l_{e,0} \cdot 2^0 + l_{e,1} \cdot 2^1 = 1$ which results in

$$K'_e = \Big[0.1, 28.5, 1, \Big((-1, 0), (0.06, 0.05)\Big)\Big], \tag{5.6}$$

where the units of all times and slopes are omitted for the sake of brevity.



(a) interpolation of full-swing transitions (b) interpolation of pulses

Figure 5.4: Comparison of methods for interpolation between two waveforms

5.1.3 Library lookup of event combination effects

The effect of each event combination K'_e on the supply current as well as on the gate output voltage can be looked up from the characterization library. Both effects are stored as piecewise linear approximation waveforms where the position and number of supporting points is optimized for accuracy and storage size. In addition to the event combination parameters (with exception of τ_e), both waveforms exhibit a strong dependency on the capacitive load, C_{load}, at the gate output which forms an additional lookup parameter. In order to accelerate library lookup, for all parameters except for the mean transition slope, $\overline{s_e}$, and the capacitive output load, C_{load}, nearest neighbor interpolation is used, which leads to acceptable deviations compared to more complex interpolation methods.

In contrast to these slight inaccuracies due to time offsets and pulse widths, the values used for gate output load or the transition slopes on the input nets during lookup of output effects require more care. Both parameters have a major influence on the supply current waveform and nearest neighbor interpolation would lead to significant errors.

The simplest method for interpolation of two waveforms would sample both waveforms at the same time grid and use the weighted sum of both values at each sample time as the result. This approach that is closely related to linear interpolation does not result in the desired properties as shown in Fig. 5.4 when trying to interpolate an intermediate state between two given waveforms. Fig. 5.4b shows two pulses that could have been observed at the same gate output for two different capacitive output loads. For an output load that falls in-between the characterized values, one would still expect a single pulse that is shifted in time and enlarged compared to pulse A towards pulse B. Instead, linear interpolation predicts two smaller pulses at the same position as the original pulses. In order to obtain a more suitable interpolation, an approach developed for the interpolation

of probability distributions called *distribution interpolation* [67] or *histogram matching* [68] is extended to general waveforms. The main advantage of this method that will be called *advecting interpolation* is that it morphs the waveform features instead of cross-fading them as in the linear interpolation. This method leads to the same results as a one-dimensional application of *displacement interpolation* [69], which is based on Lagrangian mass transport, but is significantly faster because no initial fitting of Gaussian kernels is required.

5.1.4 Superposition of looked up waveforms

The waveform segments of the gate output net and the supply current that correspond to the event combinations need to be superimposed in order to obtain the full waveforms spanning the whole estimation time. First, all looked up waveform segments are moved in time according to the offset, τ_e, corresponding to the respective event combination, K'_e. In case the resulting segments are nonoverlapping in time, the output and supply current waveforms can be obtained by simple concatenation of the segments. For closely spaced events it is possible for consecutive waveform segments to overlap in selected time windows. In that case the output waveform is constructed using the first intersection of both waveforms as the time where control over the output state switches from one segment to the next.

Due to this concatenation of analog signal waveforms this estimation retains full information about glitch creation and propagation. In order to perform analysis for gates in subsequent logic levels, the output voltage waveform is analyzed in the same way as the gate inputs before, resulting in a number of event tuples (l_e, w_e, t_e, s_e), consisting of the logic state, l_e, at the beginning of the event, the potential pulse width, w_e, the event time, t_e, and the signal slope, s_e. These events are stored in the local signal database until needed. The supply current waveform on the other hand can be integrated in order to obtain the power consumption of the current gate.

5.2 Characterization complexity

The description of the estimation approach using event-based lookup of waveform segments assumes that a sufficiently accurate characterization library containing information on all gates of the circuits of interest exists. The generation of this library is a one-time cost but nevertheless requires some attention in order not to restrict the overall accuracy

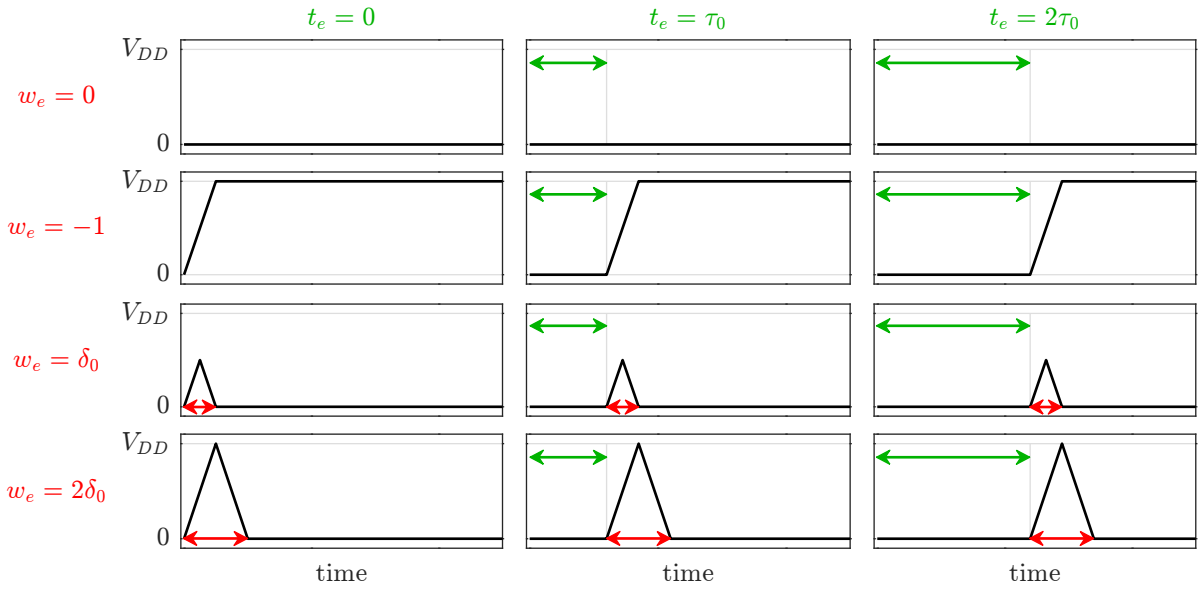


Figure 5.5: Combination of event type and event offset for single gate input ($\mathrm{n}(w_e) = \mathrm{n}(t_e) = 3, \mathrm{n}(s_e) = 1$).

and keep the storage size of the characterization library in manageable limits.

For a gate with multiple inputs all combinations of input events with different time offsets, t_e, on all inputs need to be considered. As discussed earlier, an event can either be a single full-swing transition ($w_e = -1$) or a pulse featuring a certain width ($w_e > 0$). Additionally, it is possible for an input to remain in a stable state ($w_e = 0$). Fig. 5.5 shows possible events on a single input starting at logic state "0" for $\mathrm{n}(w_e) = \mathrm{n}(t_e) = 3$ and a single transition slope, s_e, where the operator $\mathrm{n}(\cdot)$ denotes the number of possible parameter values (not considering the special case $w_e = 0$). For the case of no transition on an input port the time offset becomes meaningless and does not need to be varied. The number of combinations of pulse widths, time offsets and transition slopes for a gate with N inputs that drives a capacitive output load, C_{load}, can be written as

$$n'_{comb} = 2^N \cdot \mathrm{n}(C_{load}) \cdot \Big(\mathrm{n}(w_e) \cdot \mathrm{n}(t_e) \cdot \mathrm{n}(s_e) + 1\Big)^N. \tag{5.7}$$

This calculation considers that for the special cases where any signal remains in its stable state during an event combination, neither time offset nor transition slope need to be varied for this signal. Additionally, for each combination of pulse width, time offset and transition slope the N-input gate can be in any of 2^N input states at the start of the event combination. The actual number of physically distinct combinations is slightly reduced because a number of event combinations according to (5.7) will only differ in a

global time offset. The reasonable restriction to event combinations that feature at least one event at $t_e = 0$ and the simplification of only using the average signal slope $\overline{s_e}$, which was introduced before, reduce the number of event combinations to

$$n_{comb} = 2^N \cdot \mathrm{n}(C_{load}) \cdot \mathrm{n}(\overline{s_e}) \left[\Big(\mathrm{n}(w_e) \cdot \mathrm{n}(t_e) + 1 \Big)^N - \Big(\mathrm{n}(w_e) \cdot (\mathrm{n}(t_e) - 1) + 1 \Big)^N + 1 \right]. \tag{5.8}$$

This exponential growth of combinations severely limits the number of pulse widths and event times that could be characterized in reasonable time for gates with a larger number of inputs. Commercial standard cell libraries often feature combinatorial gates with up to six inputs, which even for the restriction of only three possible values per parameter would require the evaluation of approximately $5.6 \cdot 10^{12}$ event combinations. This number is prohibitory high, even when considering that event combinations are independent from each other and characterization could therefore apply arbitrary large degrees of parallelization. In order to enable the handling of high-fanin gates, the proposed estimation approach limits the number of inputs that are assumed to feature quasisimultaneous switching events to a parameter n_{sw}. This simplification is motivated by the reasoning that it becomes increasingly improbable for larger number of inputs to switch at the same time. This way, the number of event combinations to characterize for a gate with $N \geq n_{sw}$ inputs is reduced to

$$n_{comb,n_{sw}} = 2^N \cdot \mathrm{n}(C_{load}) \cdot \mathrm{n}(\overline{s_e}) \cdot \sum_{k=1}^{n_{sw}} \binom{N}{k} \cdot \Big(\mathrm{n}(w_e)^k \cdot \mathrm{n}(t_e)^k - \mathrm{n}(w_e)^k \cdot \Big(\mathrm{n}(t_e) - 1 \Big)^k \Big). \tag{5.9}$$

For the choice of $n_{sw} = 3$, that is used for the results presented here, (5.9) can be evaluated to

$$\begin{aligned} n_{comb,3} = 2^N \cdot \mathrm{n}(C_{load}) \cdot \mathrm{n}(\overline{s_e}) \cdot \Bigg(& \frac{N^3 - 3N^2 + 2N}{6} \cdot \mathrm{n}(w_e)^3 \cdot \Big(3\,\mathrm{n}(t_e)^2 - 3\,\mathrm{n}(t_e) + 1 \Big) + \\ & \frac{N^2 - N}{2} \cdot \mathrm{n}(w_e)^2 \cdot \Big(2\,\mathrm{n}(t_e) - 1 \Big) + \\ & \mathrm{n}(w_e) \cdot N + 1 \Bigg), \end{aligned} \tag{5.10}$$

which demonstrates the reduction of the inner term from exponential complexity to cubic complexity.

Each combination is simulated analogous to conventional library characterization using an automated process. The resulting waveforms can be compressed using the same base curve technology that is employed for compact-CCS models which already require analysis

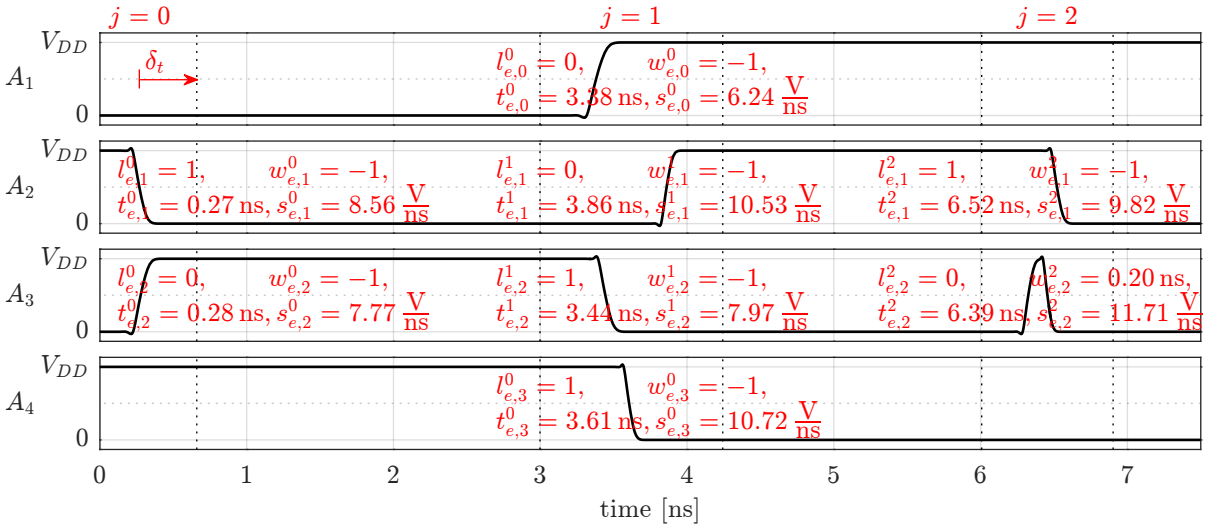


Figure 5.6: Example input signals to 4-input XNOR gate with annotated event characteristics.

and storage of current waveforms [70]. Although the number of switching events that need to be characterized is increased by up to four orders of magnitude, a large number of combinations result in either highly similar output waveforms or negligible output switching. This can be exploited for intelligent selection of switching combinations to characterize and for further reductions of the storage requirements. Without employing these compression techniques, the prototype implementation results in a characterization library that is approximately 800 times larger than a conventional library. As with conventional library characterization, the process can be parallelized to arbitrary degrees.

5.3 Demonstration of estimation flow

As a demonstration of the proposed power estimation based on waveform lookup, the workflow during estimation of a single gate will be examined in detail in the following example. The signals shown in Fig. 5.6 are assumed to be connected to the 4-input XNOR gate discussed in section 3.1. These signal waveforms could either form primary inputs to the circuit under test or could result from the concatenation of waveform segments in the estimation of a previous logic level. In both cases, prior to the actual estimation all waveforms are decomposed into switching events that can be written as 4-tuples (l_e, w_e, t_e, s_e) as defined in section 5.1.1, specifying the logic state, l_e, prior to the event, the pulse width, w_e, in nanoseconds, which is set to "-1" by definition for single transitions, the event time, t_e, in nanoseconds and the transition slope, s_e, in volts per nanosecond. The

lists of switching events for the example waveforms are given as

$$A_1: \quad \{(0,-1,3.38,6.24)\}, \tag{5.11}$$

$$A_2: \quad \{(1,-1,0.27,8.56),(0,-1,3.86,10.5),(1,-1,6.52,9.82)\}, \tag{5.12}$$

$$A_3: \quad \{(0,-1,0.28,7.77),(1,-1,3.44,7.97),(0,0.2,6.39,11.7)\}, \tag{5.13}$$

$$A_4: \quad \{(1,-1,3.61,10.7)\}. \tag{5.14}$$

In the next step, combinations of events within a maximum distance of δ_t are formed. The switching events of the individual inputs can be separated into three intervals that feature switching activity, which results in the following event combinations:

$$\begin{aligned}
K_{e,0} &= \big((0,0,0,0), && (1,-1,0.27,8.56), && (0,-1,0.28,7.77), && (1,0,0,0)\big),\\
K_{e,1a} &= \big((0,-1,3.38,6.24), && (0,0,0,0), && (1,-1,3.44,7.97), && (1,-1,3.61,10.7)\big),\\
K_{e,1b} &= \big((1,0,0,0), && (0,-1,3.86,10.5), && (1,0,0,0), && (1,-1,3.61,10.7)\big),\\
K_{e,2} &= \big((1,0,0,0), && (1,-1,6.52,9.82), && (0,0.2,6.39,11.7), && (0,0,0,0)\big).
\end{aligned} \tag{5.15}$$

As the combinations $K_{e,0}$ and $K_{e,2}$ only feature switching on two of the inputs, the remaining input events are set to $w_e = 0$. The second switching interval features switching on all four inputs. As only event combinations with three switching inputs were included in the characterization library in order to reduce complexity, this interval is separated into two event combinations. Combination $K_{e,1a}$ contains the earlier three switching events and assumes a stable state on input A_2, while combination $K_{e,1b}$ includes the last event on input A_2 in addition to the event on A_4 that was already included in the previous combination but is within the time window of influence, δ_t, for this combination as well. The earlier events on A_1 and A_3 can be assumed to have settled on their final value during $K_{e,1b}$.

Prior to library lookup all event combinations are shifted in time according to (5.2) and (5.3), the combined input state, $L_{e,j}$, at the start of the event combination is calculated from all $l_{e,i}$ according to (5.4) and the mean transition slope of all switching inputs, $\overline{s_{e,j}}$, is defined, which results in

$$\begin{aligned}
K'_{e,0} &= \big[0.27,\ 8.17,\ 10,\ \big((0,0), && (-1,0), && (-1,0.01), && (0,0)\big)\big],\\
K'_{e,1a} &= \big[3.38,\ 8.30,\ 12,\ \big((-1,0), && (0,0), && (-1,0.06), && (-1,0.23)\big)\big],\\
K'_{e,1b} &= \big[3.61,\ 10.6,\ 9,\ \big((0,0), && (-1,0.25), && (0,0), && (-1,0)\big)\big],\\
K'_{e,2} &= \big[6.39,\ 10.8,\ 3,\ \big((0,0), && (-1,0.13), && (0.2,0), && (0,0)\big)\big].
\end{aligned} \tag{5.16}$$

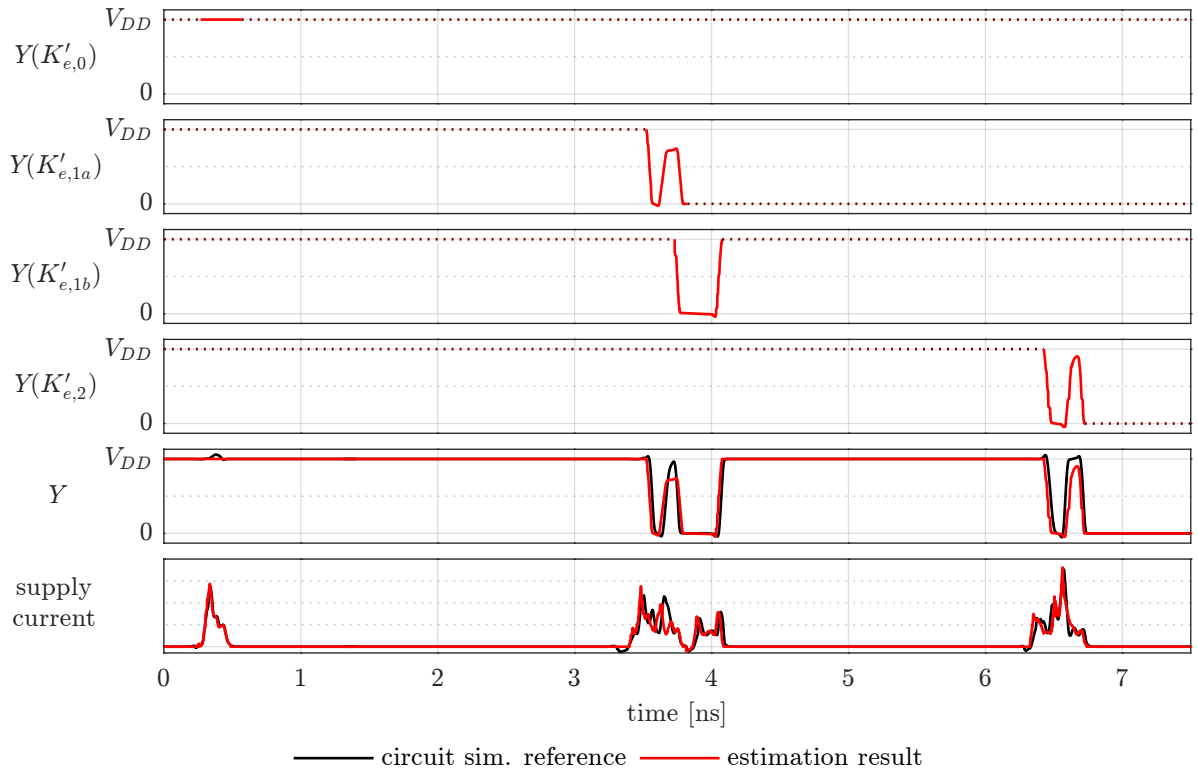


Figure 5.7: Waveforms at the output Y of the XNOR gate for each event combination in the estimation example as well as resulting output and supply current waveform (individual event contributions not shown) after concatenation in comparison to SPICE-level reference.

The first three entries per event combination specify the event combination offset, $\tau_{e,j}$, the mean transition slope, $\overline{s_{e,j}}$, and the input state, $L_{e,j}$, followed by event widths and times on individual inputs.

These normalized event combinations in combination with the capacitive output load, C_{load}, which is estimated from the netlist and parasitic extraction, correspond to the scenarios characterized in the library. This allows lookup of the voltage waveform at the gate output as well as of the supply current waveform for each combination. In order to balance runtime complexity and estimation accuracy, interpolation using the proposed *advecting* algorithm introduced in section 5.1.3 is only performed for output load and signal slope while event width and time parameters are fitted to the closest characterized scenario. This way, at most four data points need to be looked up per event combination. The first four rows in Fig. 5.7 correspond to the effect on the output waveform of each of the four event combinations which have already been shifted by $\tau_{e,j}$ in time to reverse the normalization. The estimation result of the full output waveform is obtained by concatenation of these looked up segments. For the effects of $K'_{e,1a}$ and $K'_{e,1b}$ this requires

additional attention since the looked up waveform segments are overlapping in time. This case is resolved by using the first intersection of both waveforms as the time when control over the output switches from the first to the second event combination. The resulting estimation waveform in the fifth row of Fig. 5.7 is compared to the result of a SPICE-level circuit simulation stimulated by the same input waveforms. After analysis of the switching events on this output node, which results in signal statistics to be used in a subsequent logic level, this voltage waveform is no longer required in the estimation flow. The supply current waveform that can be integrated to obtain the actual power estimation is plotted in the sixth row of the figure. It is constructed from looked-up waveform segments related to all event combinations in the same way as the voltage response and shows the same high level of estimation accuracy. In this admittedly tiny example the estimation error of the dynamic energy consumption compared to the circuit-level simulation is as low as 1.2 %.

5.4 Evaluation of estimation accuracy

To demonstrate the accuracy of the proposed power estimation methodology and allow a comparison to other approaches, the event-based waveform lookup is applied to representative benchmark circuits. Because the introduction of the proposed approach aims at mitigating shortcomings that are inherent to state-of-the-art gate-level power estimators, the implementations of the ISCAS85 benchmark set, which was discussed in detail in section 3.3.1 during the evaluation of commercial gate-level power estimators, are selected for the accuracy evaluation. Fig. 5.8 compares the accuracy of the proposed approach to that of a state-of-the-art gate-level power estimator for three different commercial CMOS technologies. Both estimation approaches as well as the physical SPICE-level circuit simulation which serves as the reference for the comparison employ the same stream of input stimuli and have access to postlayout parasitic data. With exception of the custom precharacterization library employed by the event-based waveform lookup both estimation approaches require the same set of input files.

Fig. 5.8 demonstrates a considerable increase of accuracy of the proposed event-based approach compared to state-of-the-art power estimation which becomes apparent in the mean error of only 4.7 % for all circuits and technologies compared to 13.1 % for state-of-the-art tools. Even more important is the reduction of the maximum error observed for this benchmark set to be 12.6 % for the proposed approach in contrast to 38.6 % for

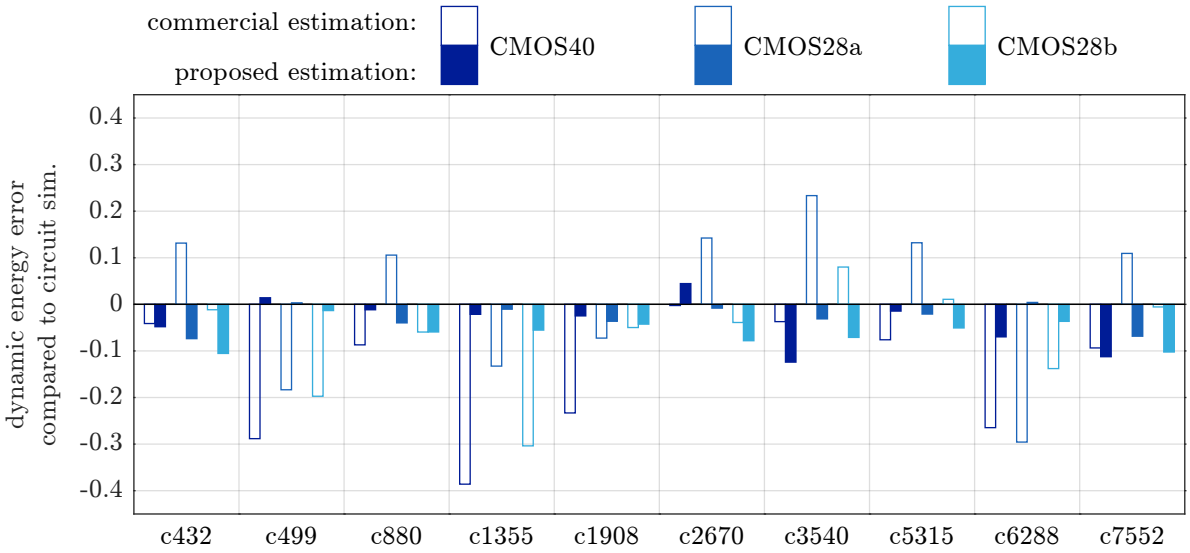


Figure 5.8: Comparison of estimation accuracy of proposed event-based waveform lookup compared to state-of-the-art commercial power estimator (results of Tool A as shown in Fig. 3.7a).

commercial power estimators. The highest gains in accuracy can be observered for circuits which exhibit complex signal delay effects that are handled poorly in current gate-level estimators. The circuits *c499* and *c1355*, where energy consumption is dominated by compound XNOR gates, can be shown to suffer from dramatic underestimation when handled by commercial power estimators mainly because of the importance of quasisimultaneous switching of multiple gate inputs that cannot be considered in current library formats. The circuit *c6288* on the other hand, which implements a 16-bit multiplier, features a high logic depth requiring accurate estimation of glitch propagation which cannot be provided by standard estimation workflows. Both of these shortcomings are mitigated by the proposed estimation approach which is demonstrated by the dramatic reduction of estimation errors for these circuits.

5.5 Estimation of sequential circuits

The introduction of the proposed power estimation methodology assumed purely combinatorial circuits. However, it can be generalized to circuits containing sequential cells. The extensions which need to be considered as well as the resulting accuracy will be discussed in the following section.

In contrast to combinatorial gates, the power consumption and output switching waveforms of registers or latches are not solely dependent on the input events but also on the

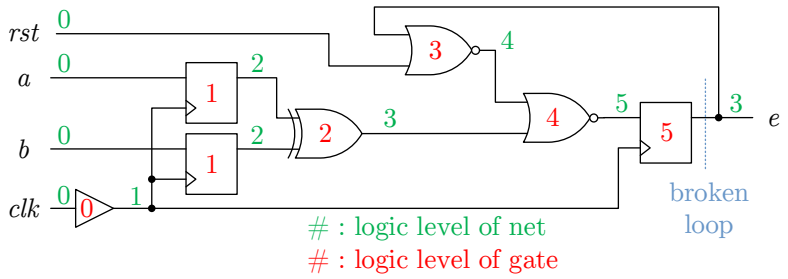


Figure 5.9: Assignment of logic levels for sequential circuit featuring a loop in the connectivity.

stored output state. This dependency needs to be considered during cell characterization by simulating each event combinations for both possible output states. Compensating this increasing characterization complexity is the fact that the clock input can be assumed to be free of glitches. During estimation the stored state at the outset of an event combination can be included in the combined state parameter, L_e, without modifying the lookup procedure.

A second point that needs to be addressed is the decomposition of the circuit under analysis into logic levels. The algorithm proposed in section 5.1 assumed a gate netlist without loops in the connectivity in order to assign unambiguous levels to all gates. While this assumption is valid for virtually all combinatorial circuits with the exception of oscillators, sequential circuits routinely feature internal states as part of feedback loops. Notable examples are state machines or the calculation of iterative algorithms. A simple example is depicted in Fig. 5.9. For the proposed estimation approach to handle these circuits, potential loops need to be broken. The solution to this challenge lies in the distinctive property of register outputs to toggle at most once during a clock cycle. In addition, the time offsets of these full-swing transitions at register outputs are determined with sufficient accuracy by the switching events on the clock input.

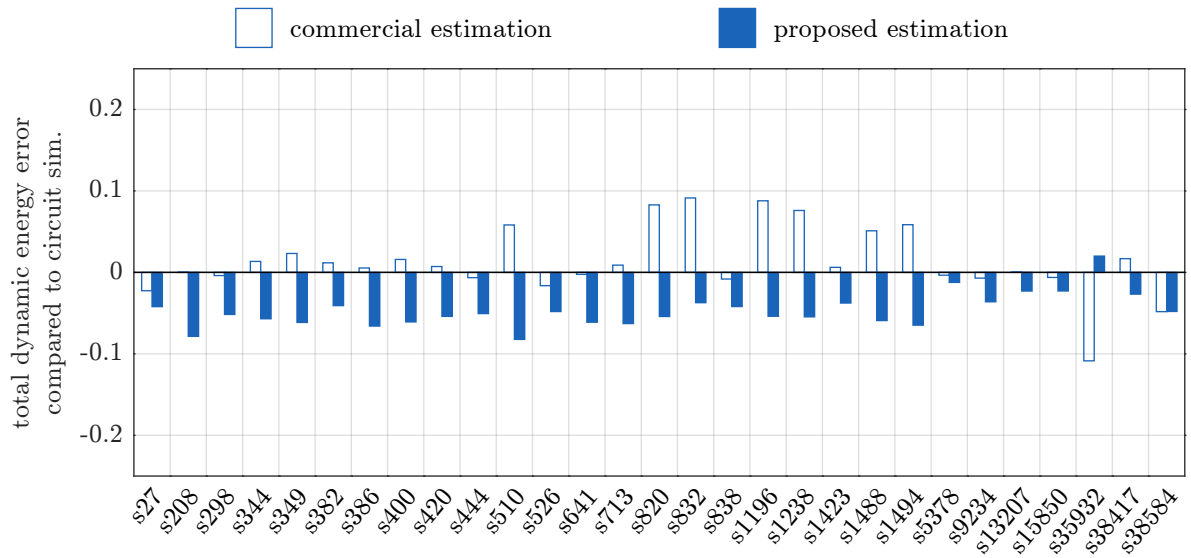
As a consequence, the preprocessing step which assigns an unique logic level to each gate is modified as given in Algorithm 1. When a loop in the connectivity prevents further assignment of logic levels, a register taking part in such a loop is heuristically selected from the set of gates which are not assigned a level yet by the procedure SelectRegister. By assuming that the switching at the circuit node connected to this register output was estimated in a previous iteration, a deadlock in the algorithm is prevented. In the example depicted in Fig. 5.9, neither of the *NOR* gates can be assigned a logic level by linearly parsing the hierarchy because the register output, e, is fed back thereby creating a loop. By assuming that the switching events on e were known, the remaining gates can

Algorithm 1 Assignment of logic levels

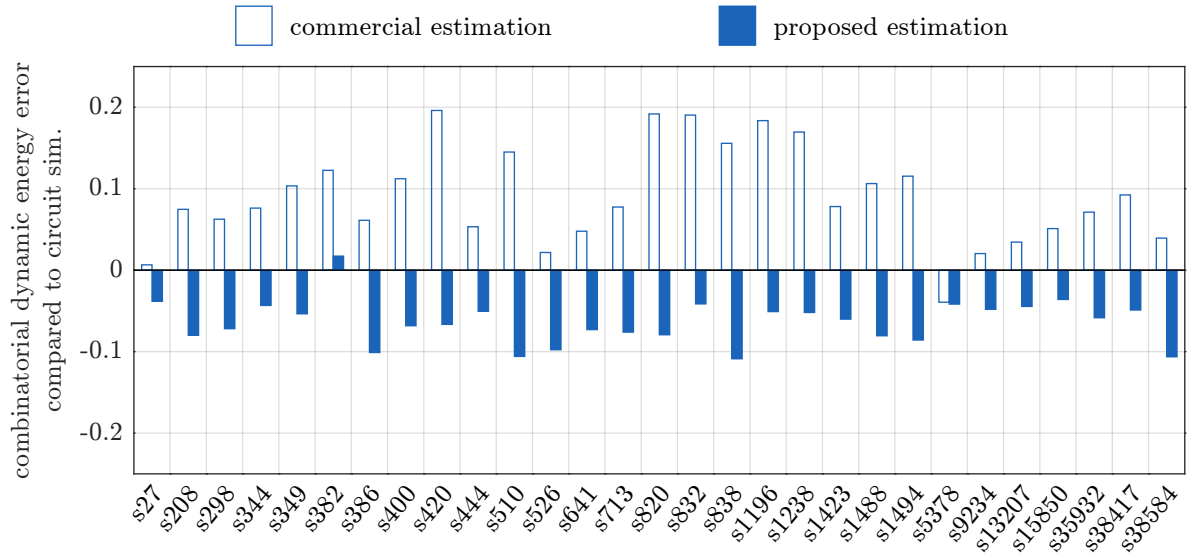
```
for all gate ∈ G do                                  ▷ initialize levels of gates to undefined value
    l_g(gate) = ∞
end for
for all net ∈ N do                                   ▷ initialize levels of nets to undefined value
    l_n(net) = ∞
end for
N_inputs ← PrimaryInputs(N)
for all net ∈ N_inputs do                            ▷ assign primary input nets to level 0
    l_n(net) ← 0
end for
G_unassigned ← G
G_assumed ← ∅          ▷ registers for which output is assumed to be known to break loop
level ← 0
while |G_unassigned| > 0 do                          ▷ continue until all gates are assigned a level
    for all gate ∈ G_unassigned do
        N_gate ← InputNets(gate)                     ▷ fetch set of current input nets
        if l_n(net) ≤ level ∀ net ∈ N_gate then      ▷ when all input nets are defined
            l_g(gate) ← level                        ▷ assign gate to current level...
            l_n(OutputNets(gate)) ← level+1          ▷ ...and output nets to next level
            remove gate from G_unassigned
        end if
    end for
    if ∄ l_g(gate) = level ∀ gate ∈ G then           ▷ if no gate was assigned to current level
        gate_reg ← SelectRegister(G_unassigned)      ▷ select suitable register to break loop
        l_n(OutputNets(gate_reg)) ← level+1          ▷ assume output to be defined
        add gate_reg to G_assumed                    ▷ remember assumed register outputs
    else
        level ← level + 1                            ▷ continue for next level
    end if
end while
```

be assigned suitable logic levels.

The switching events on the outputs of all registers in the set of gates, $G_{assumed}$, which were selected to break connectivity loops, need to be approximated prior to actual estimation of the register gate. The heuristic employed during loop-breaking guarantees that switching events on the clock input of the selected registers are estimated prior to the iteration during which the approximated register output is required. The gate delays and transition slopes of the registers in $G_{assumed}$ can be estimated from the default characterization library while considering parasitic node capacitances and circuit connectivity. The zero-delay switching activity at these nodes is recorded during the execution of the testbench that is required in any case to determine switching on the primary inputs. The recorded switching events are shifted in time by an offset depending on the estimated



(a) estimation accuracy of total dynamic energy



(b) estimation accuracy of dynamic energy excluding registers

Figure 5.10: Comparison of estimation accuracy of proposed event based approach to commercial gate-level estimator for sequential circuits.

gate delay and the timing of the clock input events. In addition, the looked-up transition slopes are annotated to these approximated full-swing switching events. This way, an accurate estimation of switching activity at the output of registers is available prior to the processing of the register themselves using the event based waveform lookup approach. However, the supply current waveforms needed to calculate the dynamic power consumption are still generated in a later iteration of the estimation algorithm once all input nodes have been annotated with accurate switching events.

The accuracy of the proposed event-based waveform lookup compared to circuit simulation resulting from this generalization to sequential circuits is evaluated using the

ISCAS89 benchmark set that was introduced in section 3.3.2. As before, the accuracy achieved by the proposed estimation approach is put into perspective by comparison to a state-of-the-art gate-level estimator. As shown in Fig. 5.10a the mean error of 4.8 % achieved by the proposed approach seems high compared to the apparently perfect estimation for a large number of circuits using the commercial estimation tool which results in a mean error of 2.9 %. However, as discussed in chapter 3.3.2, this seemingly good estimation accuracy results from compensation of relatively high error components featuring opposite signs. The higher dependability of the proposed approach is demonstrated in Fig. 5.10b which compares the estimation accuracy of the combinatorial circuit components. The relative energy errors for the proposed approach are in the same range as for the total error whereas the commercial state-of-the-art tool exhibits errors of up to 19.6 % which did not show up in the analysis of the total dynamic energy error due to compensation with underestimated sequential components.

5.6 Analysis of runtime complexity

Since the proposed algorithm traverses the circuit hierarchy by logic levels and performs power estimation for each logic gate separately, the computational complexity of the algorithm grows linearly with circuit size, similar to conventional gate-level power estimators. Concepts like caching of interpolated waveform results can be applied to further reduce the required runtime while the fact that all estimations at the same logic level are uncorrelated lends itself to parallel implementations.

The linear runtime complexity of the prototype implementation of the proposed algorithm based on MATLAB is demonstrated in Fig. 5.11. It outperforms highly optimized commercial circuit simulators by a factor of 50 for circuits with 1000 gates. The speedup increases with growing circuit size as expected. State-of-the-art gate-level power estimators exhibit runtimes that are faster by a factor of approximately 200. However, due to similar computational complexity it can be assumed that runtime-optimized implementations of the proposed approach will significantly reduce this gap without sacrificing the superior accuracy.

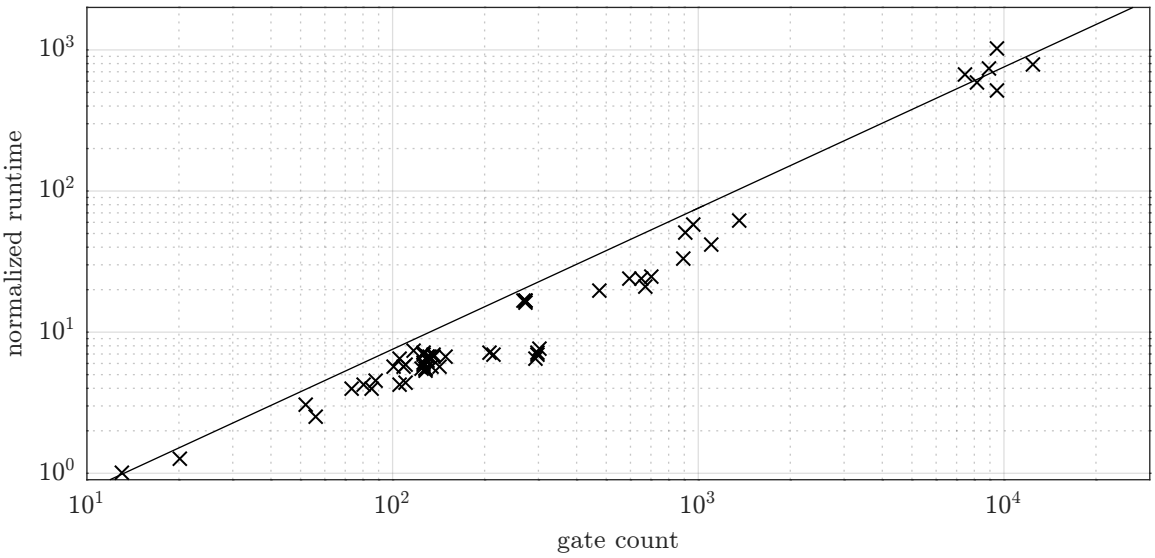


Figure 5.11: Runtime of proposed estimation algorithm for various circuit sizes.

6 Conclusion

The analysis of state-of-the-art gate-level power estimators that are commonly employed in current digital design work flows revealed considerable accuracy deficits when effects due to signal delays need to be considered. In particular, the generation and propagation of glitches cannot accurately be represented by the underlying digital switching activities. Further errors are caused by the restriction to the handling of single input events which become most pronounced for complex compound gates where switching of internal nodes is significant.

Motivated by these deficiencies of existing power estimation methodologies this work explores alternative approaches towards power estimation of digital circuits while considering complex delay effects like glitches. The stochastic nature of average power consumption analysis serves as the basis for macromodeling approaches that link statistical properties of input signals to the power consumption of circuit components without requiring error-prone analysis of switching activities on all nodes. This class of power estimators allows analysis on various levels of abstraction by relying on libraries of circuit components which were characterized on a lower level of abstraction in a preparatory phase. In order to accurately consider the increase of dynamic energy dissipation due to glitches on circuit nodes, novel metrics as additional parameters to circuit macromodels were introduced which capture the essential properties of glitching. The proposed glitch metrics are defined by probability distributions to retain information about their spread which is essential for accurate propagation of metrics. The parameters required for library lookup are defined as the moments of normal distributions obtained by fitting of mixture distributions to the empirical metric distributions. The resulting estimation methodology was shown to achieve good accuracy under the influence of glitches but was hampered by the high characterization complexity resulting from the large number of model parameters.

In an effort to reduce the model complexity alternative metrics were proposed that targeted single transitions instead of glitches and therefore allowed the unified handling of switching due to logic evaluation and undesired glitching. Further reductions of the

number of model parameters were achieved by piecewise linearization of the probability distribution of transition occurrence time which allowed the separate analysis of segments in time. During analysis of the modified modeling approach it was demonstrated to allow fast estimation that capture the power dependency on glitching for small circuit macros. However, this simplifying assumption of weakly correlated time windows was revealed to cause fatal estimation errors for large logic depths.

Subsequently, a second, fundamentally different approach towards accurate consideration of glitching effects during power estimation was introduced. Instead of applying statistical parameters to a fitted macromodel, this proposed approach targets individual switching events. In an effort to retain the runtime advantages of lookup-based power estimation while benefiting from the accuracy of analog switching waveforms as obtained during physical circuit simulation, this approach replaces inaccurate digital simulation by the lookup of precharacterized waveform segments related to switching events on the inputs of gates. By including partial-swing glitches as well as multi-input events in the characterization library, the complex behavior of glitch generation and propagation is accurately handled. The comparison of estimation accuracy of this approach based on lookup of waveform segments and a state-of-the-art gate-level power estimation demonstrates the significant improvements. In particular, the benchmark circuits which suffered from complex signal delay effects that cannot be modeled well in existing methodologies are handled well by the proposed approach.

The two methodologies proposed in this work demonstrate two distinct approaches aiming at the same goal. The probabilistic macromodels fit naturally into the statistical view of mean power estimation and could allow for the evaluation of power consumption in a single step without the need for lengthy input sequences. However, the derivation of simplified model parameters and the balancing of accuracy and evaluation complexity requires additional work. The fundamental idea underlying the event-based lookup of waveform segments on the other hand is closer to current gate-level estimators which facilitates incorporation into existing work flows. By targeting the known limitations of state-of-the-art estimators, the event-based approach achieves high accuracy even for circuits that are highly dependent on complex delay effects.

Bibliography

[1] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*. Pearson Education, 2011.

[2] A. N. Bhoj and N. K. Jha, "Design of Logic Gates and Flip-Flops in High-Performance FinFET Technology," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 11, pp. 1975–1988, Nov 2013.

[3] A. Shen, A. Ghosh, S. Devadas, and K. Keutzer, "On average power dissipation and random pattern testability of CMOS combinational logic networks," in *International Conference on Computer-Aided Design*. Santa Clara, CA , USA: IEEE Comput. Soc. Press, 1992, pp. 402–407.

[4] M. Meixner and T. G. Noll, "Limits of gate-level power estimation considering real delay effects and glitches," in *International Symposium on System-on-Chip*. IEEE, Oct 2014.

[5] D. Kamel, C. Hocquet, F.-X. Standaert, D. Flandre, and D. Bol, "Glitch-induced within-die variations of dynamic energy in voltage-scaled nano-CMOS circuits," in *European Solid-State Circuits Conference*. IEEE, Sep 2010, pp. 518–521.

[6] F. Hu and V. D. Agrawal, "Enhanced dual-transition probabilistic power estimation with selective supergate analysis," in *International Conference on Computer Design*. IEEE Comput. Soc, 2005, pp. 366–369.

[7] C. M. Huizer, "Power Dissipation Analysis of CMOS VLSI Circuits by means of Switch-Level Simulation," in *European Solid-State Circuits Conference*, 1990, pp. 61–64.

[8] M. Rewienski, "Simulation and Verification of Electronic and Biological Systems," in *Simulation and Verification of Electronic and Biological Systems*, P. Li, L. M. Silveira, and P. Feldmann, Eds. Springer Netherlands, 2011, ch. A Perspect, pp. 23–43.

[9] D. Rabe and W. Nebel, "New approach in gate-level glitch modelling," in *European Design Automation Conference*. IEEE Comput. Soc. Press, 1996, pp. 66–71.

[10] W.-C. Tsai, C. B. Shung, and D. C. Wang, "Accurate logic-level power simulation using glitch filtering and estimation," in *Asia Pacific Conference on Circuits and Systems*. Seoul, South Korea: IEEE, 1996, pp. 314–317.

[11] S. Gavrilov, A. Glebov, S. Rusakov, D. Blaauw, L. Jones, and G. Vijayan, "Fast power loss calculation for digital static CMOS circuits," in *Proceedings European Design and Test Conference*. IEEE Comput. Soc. Press, 1997, pp. 411–415.

[12] P. Israsena and S. Summerfield, "Novel pattern-based power estimation tool with accurate glitch modeling," in *International Symposium on Circuits and Systems*, vol. 4, 2000, pp. 721–724.

[13] A. Bogliolo, B. Ricco, L. Benini, and G. De Micheli, "Accurate logic-level power estimation," in *Symposium on Low Power Electronics*. IEEE, 1995, pp. 40–41.

[14] J. Lee, B. Vinnakota, and L. Lucke, "Power estimation using input/output transition anaylsis (IOTA)," in *International Symposium on Circuits and Systems*. Monterey, CA, USA: IEEE, 1998, pp. 49–52.

[15] M. Dietrich and J. Haase, *Process Variations and Probabilistic Integrated Circuit Design*. Springer, 2012.

[16] Synopsys, "CCS Power Technical White Paper," Tech. Rep., 2006.

[17] R. Burch, F. N. Najm, P. Yang, and T. N. Trick, "A Monte Carlo approach for power estimation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 1, no. 1, pp. 63–71, Mar 1993.

[18] F. N. Najm, "Transition density: a new measure of activity in digital circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 2, pp. 310–323, 1993.

[19] T.-L. Chou and K. Roy, "Estimation of activity for static and domino CMOS circuits considering signal correlations and simultaneous switching," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 10, pp. 1257–1265, 1996.

[20] R. Marculescu, D. Marculescu, and M. Pedram, "Probabilistic modeling of dependencies during switching activity analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 2, pp. 73–83, 1998.

[21] L. Wang, M. Olbrich, E. Barke, T. Buchner, M. Buhler, and P. Panitz, "A theoretical probabilistic simulation framework for dynamic power estimation," in *International Conference on Computer-Aided Design*. IEEE, Nov 2011, pp. 708–715.

[22] C.-S. Ding, C.-Y. Tsui, and M. Pedram, "Gate-level power estimation using tagged probabilistic simulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 11, pp. 1099–1107, 1998.

[23] Z. Hao, R. Shen, S. X.-D. Tan, B. Liu, G. Shi, and Y. Cai, "Statistical full-chip dynamic power estimation considering spatial correlations," in *International Symposium on Quality Electronic Design*. Santa Clara, CA , USA: IEEE, Mar 2011, pp. 1–6.

[24] H. Mehta, R. M. Owens, and M. J. Irwin, "Energy characterization based on clustering," in *Design Automation Conference*. Las Vegas, NV, USA: ACM, 1996, pp. 702–707.

[25] J.-Y. Lin, W.-Z. Shen, and J.-Y. Jou, "A structure-oriented power modeling technique for macrocells," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 7, no. 3, pp. 380–391, 1999.

[26] S. Ravi, A. Raghunathan, and S. T. Chakradhar, "Efficient RTL power estimation for large designs," in *International Conference on VLSI Design*. IEEE Comput. Soc, 2003, pp. 431–439.

[27] B. Carrion Schafer and K. Wakabayashi, "Precision tunable RTL macro-modelling cycle-accurate power estimation," *Computers & Digital Techniques*, vol. 5, no. 2, p. 95, 2011.

[28] L. Benini, A. Bogliolo, M. Favalli, and G. De Micheli, "Regression Models for Behavioral Power Estimation," *Integrated Computer-Aided Engineering*, vol. 5, no. 2, pp. 95–106, Apr 1998.

[29] N. R. Potlapally, A. Raghunathan, G. Lakshminarayana, M. S. Hsiao, and S. T. Chakradhar, "Accurate power macro-modeling techniques for complex RTL circuits," in *International Conference on VLSI Design*. IEEE, 2001, pp. 235–241.

[30] Q. Wu, Q. Qinru, M. Pedram, and C.-S. Ding, "Cycle-accurate macro-models for RT-level power analysis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 6, no. 4, pp. 520–528, 1998.

[31] S. Gupta and F. N. Najm, "Power Macromodeling For High Level Power Estimation," in *Design Automation Conference*. IEEE, 1997, pp. 365–370.

[32] M. Barocci, L. Benini, A. Bogliolo, B. Ricco, and G. De Micheli, "Lookup table power macro-models for behavioral library components," in *Alessandro Volta Memorial Workshop on Low-Power Design*. Como , Italy: IEEE, 1999, pp. 173–181.

[33] M. Anton, I. Colonescu, E. Macii, and M. Poncino, "Fast characterization of RTL power macromodels," in *International Conference on Electronics, Circuits and Systems*. IEEE, 2001, pp. 1591–1594.

[34] S. Gupta and F. N. Najm, "Power modeling for high-level power estimation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 1, pp. 18–29, 2000.

[35] P. E. Landman and J. M. Rabaey, "Architectural power analysis: The dual bit type method," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 3, no. 2, pp. 173–187, Jun 1995.

[36] B. Jovanovic, R. Jevtic, and C. Carreras, "Binary Division Power Models for High-Level Power Estimation of FPGA-based DSP Circuits," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2013.

[37] A. Raghunathan, S. Dey, and N. K. Jha, "High-level macro-modeling and estimation techniques for switching activity and power consumption," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 4, pp. 538–557, Aug 2003.

[38] G. Bernacchia and M. C. Papaefthymiou, "Analytical macromodeling for high-level power estimation," in *International Conference on Computer-Aided Design*. San Jose, CA , USA: IEEE, 1999, pp. 280–283.

[39] A. Ahmadinia, B. Ahmad, and T. Arslan, "Efficient High-Level Power Estimation for Multi-standard Wireless Systems," in *Symposium on VLSI*. Montpellier: IEEE, 2008, pp. 275–280.

[40] W.-T. Hsieh, J.-C. Yeh, S.-C. Lin, H.-C. Liu, and Y.-S. Chen, "System power analysis with DVFS on ESL virtual platform," in *International SOC Conference*. IEEE, Sep 2011, pp. 93–98.

[41] L. Kosmann, D. Lorenz, A. Reimer, and W. Nebel, "Enabling energy-aware design decisions for behavioural descriptions containing black-box IP-components," in *International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*. IEEE, Sep 2013, pp. 51–58.

[42] N. Dhanwada, I.-C. Lin, and V. Narayanan, "A power estimation methodology for systemC transaction level models," in *International Conference on Hardware/Software Codesign and System Synthesis*. New York, New York, USA: ACM Press, 2005, pp. 142–147.

[43] S. Schürmans, Diandian Zhang, D. Auras, R. Leupers, G. Ascheid, Xiaotao Chen, and Lun Wang, "Creation of ESL power models for communication architectures using automatic calibration," in *Design Automation Conference*, 2013, pp. 1–6.

[44] K. Gruttner, P. A. Hartmann, T. Fandrey, K. Hylla, D. Lorenz, S. Stattelmann, B. Sander, O. Bringmann, W. Nebel, and W. Rosenstiel, "An ESL timing & power estimation and simulation framework for heterogeneous SoCs," in *International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation*, 2014, pp. 181–190.

[45] C.-W. Hsu, J.-L. Liao, S.-C. Fang, C.-C. Weng, S.-Y. Huang, W.-T. Hsieh, and J.-C. Yeh, "PowerDepot: Integrating IP-based power modeling with ESL power analysis for multi-core SoC designs," in *Design Automation Conference*, 2011, pp. 47–52.

[46] D. Lorenz, K. Gruettner, and W. Nebel, "Data-and State-Dependent Power Characterisation and Simulation of Black-Box RTL IP Components at System Level," in *Euromicro Conference on Digital System Design*. IEEE, Aug 2014, pp. 129–136.

[47] R. Dochia, D. Bogdan, and C. Burileanu, "Model for software power estimation of an 8-bit microcontroller," in *International Semiconductor Conference*. IEEE, Oct 2011, pp. 443–446.

[48] S.-A. Wen, H.-L. Lin, C. Wu, C.-C. Chen, K.-H. Tsai, and W.-M. Cheng, "Power-aware design technique for PAC Duo based embedded system," in *International SOC Conference*. IEEE, Sep 2011, pp. 132–135.

[49] S. Hesselbarth, T. Baumgart, and H. Blume, "Hardware-assisted power estimation for design-stage processors using FPGA emulation," in *International Workshop on Power and Timing Modeling, Optimization and Simulation*. IEEE, Sep 2014, pp. 1–8.

[50] R. Ben Atitallah, S. Niar, and J.-L. Dekeyser, "MPSoC power estimation framework at transaction level modeling," in *Internatonal Conference on Microelectronics*, Cairo, 2007, pp. 245 – 248.

[51] W. L. Bircher and L. K. John, "Complete System Power Estimation Using Processor Performance Events," *IEEE Transactions on Computers*, vol. 61, no. 4, pp. 218–227, 2012.

[52] C.-K. Chen and R.-S. Tsay, "AROMA: A highly accurate microcomponent-based approach for embedded processor power analysis," in *Asia and South Pacific Design Automation Conference*. IEEE, Jan 2015, pp. 761–766.

[53] M. Favalli and L. Benini, "Analysis of glitch power dissipation in CMOS ICs," in *International symposium on Low power design*. New York, New York, USA: ACM Press, 1995, pp. 123–128.

[54] Q. Dinh, D. Chen, and M. D. F. Wong, "Dynamic power estimation for deep submicron circuits with process variation," in *Asia and South Pacific Design Automation Conference*. IEEE, Jan 2010, pp. 587–592.

[55] C. Henning and T. G. Noll, "A new power modeling approach for transversal filters based on physically oriented design," in *International ASIC/SOC Conference*. IEEE, 2002, pp. 114–118.

[56] J. H. Anderson and F. N. Najm, "Power estimation techniques for FPGAs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, no. 10, pp. 1015–1027, Oct 2004.

[57] X. Liu and M. C. Papaefthymiou, "Incorporation of input glitches into power macromodeling," in *International Symposium on Circuits and Systems*, vol. 4. IEEE, 2002, pp. IV–846–IV–849.

[58] J. D. Alexander and V. D. Agrawal, "Algorithms for Estimating Number of Glitches and Dynamic Power in CMOS Circuits with Delay Variations," in *Symposium on VLSI*. IEEE, 2009, pp. 127–132.

[59] J. Yang, L. Ma, Zhaom Kang, Y. Cai, and T.-F. Ngai, "Early stage real-time SoC power estimation using RTL instrumentation," in *Asia and South Pacific Design Automation Conference*. IEEE, Jan 2015, pp. 779–784.

[60] D. Lee, L. K. John, and A. Gerstlauer, "Dynamic power and performance back-annotation for fast and accurate functional hardware simulation," in *Design, Automation & Test in Europe*, 2015, pp. 1126–1131.

[61] Synopsys, "CCS Power Liberty Syntax," Synopsys, Tech. Rep., 2006.

[62] F. Brglez and H. Fujtwara, "A neutral netlist of 10 combinational benchmark circuits and a target translator in fortran," in *IEEE International Symposium on Circuits and Systems*, Jan 1985.

[63] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits," in *IEEE International Symposium on Circuits and Systems*. IEEE, 1989, pp. 1929–1934.

[64] M. R. Gupta and Y. Chen, "Theory and Use of the EM Algorithm," *Foundations and Trends in Signal Processing*, vol. 4, no. 3, pp. 223–296, 2010.

[65] R. E. Walpole, R. H. Myers, S. L. Myers, and K. Ye, *Probability & Statistics for Engineers & Scientists*, 9th ed. Prentice Hall, 2012.

[66] M. Meixner and T. G. Noll, "Accurate estimation of CMOS power consumption considering glitches by using waveform lookup," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 64, no. 7, pp. 787–791, July 2017.

[67] A. Read, "Linear interpolation of histograms," *Nuclear Instruments and Methods in Physics Research*, vol. 425, no. 1-2, pp. 357–360, Apr 1999.

[68] W. Matusik, M. Zwicker, and F. Durand, "Texture design using a simplicial complex of morphable textures," in *ACM SIGGRAPH*, vol. 24, no. 3. New York, New York, USA: ACM Press, Jul 2005, p. 787.

[69] N. Bonneel, M. van de Panne, S. Paris, and W. Heidrich, "Displacement interpolation using Lagrangian mass transport," *ACM Transactions on Graphics (TOG)*, vol. 30, no. 6, p. 158, Dec 2011.

[70] Synopsys, "Liberty User Guide, Volume 1," 2013.