UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

IGOR REIS PEDROSA

Power Estimation Methodology for Pass Transistor Logic Circuits

Work presented in partial fulfillment of the requirements for the degree of Bachelor in Computer Engineering

Advisor: Prof. Dr. Ricardo Reis
Co-advisor: André Lucas Chinazzo, M.Sc.*

Porto Alegre
August 2024

"Die Sprache ist ein Labyrinth von Wegen.
Du kommst von einer Stelle und kennst dich aus;
du kommst von einer anderen zur selben Stelle,
und kennst dich nicht mehr aus."

— Ludwig Wittgenstein

ABSTRACT

In the current mobile and Internet of Things era, energy-efficient chip design is crucial due to battery limitations and the prevalence of dark silicon. Pass Transistor Logic (PTL) offers potential energy savings but lacks tailored power estimation methods, as traditional approaches designed for complementary static CMOS circuits are shown to be inadequate as a result of distinct electrical behavior. This work addresses this gap by proposing a gate-level power estimation methodology for PTL circuits, employing an event-driven simulator that provides input slope and capacitance information, as well as lookup tables containing cell characterization data. The objective of this work was to enable average power estimation of PTL circuits with accuracy within 10 % from SPICE simulation, a goal that was successfully achieved, with a total estimation error of $-4.34\,\%$ demonstrated for a benchmark circuit with PTL cells. This work thus contributes to the need for specialized power estimation techniques to support the adoption of PTL as an alternative logic style in contemporary chip design, advancing the broader research program of low-power digital designs.

Keywords: Pass Transistor Logic. Energy consumption. Power estimation. Power model. Event-driven simulation. Cell characterization. Physical design. Microelectronics.

[*] Microelectronic Systems Design Research Group, Rheinland-Pfälzische Technische Universität Kaiserslautern-Landau, Germany

Metodologia de Estimativa de Potência para Circuitos com Lógica de Transistores de Passagem

RESUMO

Na atual era dos dispositivos móveis e da Internet das Coisas, o projeto de chips energeticamente eficientes é crucial devido a limitações de bateria e à prevalência do *dark silicon*. Lógica de Transistores de Passagem (PTL) oferece uma potencial redução de energia, porém carece de métodos específicos de estimativa de potência, dado que as abordagens tradicionais projetadas para circuitos CMOS estáticos complementares se revelam inadequadas devido a diferenças elétricas. Este trabalho aborda esta lacuna propondo uma metodologia de estimativa de potência em nível de portas lógicas para circuitos com PTL, fazendo uso de um simulador orientado a eventos que fornece informações de *slope* de entrada e capacitância, assim como tabelas de consulta contendo dados de caracterização de células. O objetivo deste trabalho era de estimar a potência média de circuitos PTL com erro de menos de 10 % em relação a simulação SPICE, meta que foi alcançada com sucesso, como demonstrado por um erro total de −4,34 % para um circuito de referência com células PTL. Este trabalho contribui, portanto, para a necessidade de técnicas especializadas de estimativa de potência para amparar a adoção de estilos lógicos alternativos como PTL no projeto de chips modernos, desenvolvendo o programa de pesquisa geral de concepção de chips digitais de baixa potência.

Palavras-chave: Lógica de Transistores de Passagem. Consumo de energia. Estimativa de potência. Modelo de potência. Simulação baseada em eventos. Caracterização de células. Projeto físico. Microeletrônica.

LIST OF ABBREVIATIONS AND ACRONYMS

CMOS	Complementary Metal–Oxide–Semiconductor
DUT	Device Under Test
FinFET	Fin Field-Effect Transistor
IC	Integrated Circuit
IoT	Internet of Things
LDPC	Low-Density Parity-Check
LUT	Lookup Table
nMOS	n-type Metal-Oxide-Semiconductor
pMOS	p-type Metal-Oxide-Semiconductor
PTL	Pass Transistor Logic
PVT	Process-Voltage-Temperature
SAIF	Switching Activity Interchange Format
SPICE	Simulation Program with Integrated Circuit Emphasis
VCD	Value Change Dump
VLSI	Very-Large-Scale Integration

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

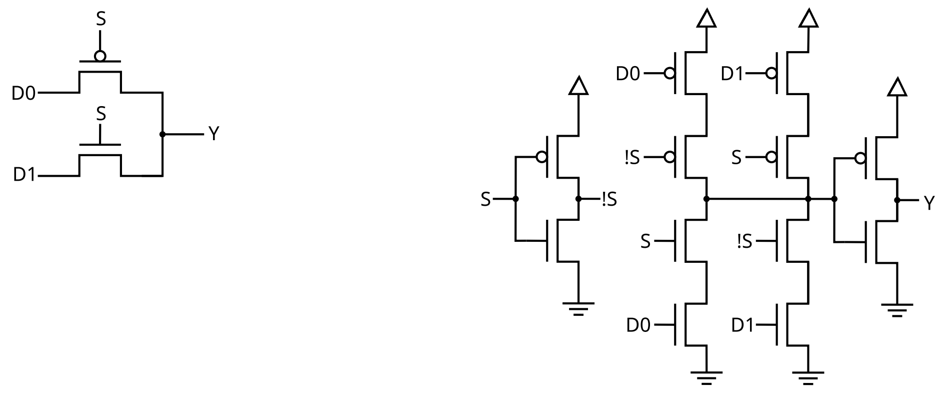
As transistors continued to shrink in size, the rate at which their power decreased began to slow down due to the growing impact of leakage currents. This has led to an increase in chips' once-constant power density and marked the breakdown of the Dennard scaling. Since packaging and cooling systems can only dissipate a finite amount of heat, often referred to as thermal design power, it eventually became necessary to turn off or underclock different areas of an integrated circuit (IC) to adhere to power constraints. These turn-off areas became known as dark silicon (Shafique et al., 2014).

Furthermore, the emergence of mobile and Internet of Things (IoT) chips introduced another major constraint to IC design: battery life. As these devices became an essential part of our daily lives, there was a surge in demand for extended battery life without compromising performance. These power and energy constraint challenges were labeled the power wall (Patterson; Hennessy, 2014) and made power and energy two key metrics when designing ICs, along with delay and area.

Many approaches at different design levels have been studied to make these integrated circuits meet today's tight power budgets. Architectural shifts such as heterogeneous and domain-specific computing resulted in innovations such as ARM's big.LITTLE architecture and Google Cloud's Tensor Processing Units. At different levels, power management techniques like dynamic voltage and frequency scaling, multi-voltage domain, and clock and power gating have become widely used in IC design (Weste; Harris, 2011).

At the circuit level, some power reduction methodologies involve using Pass Transistor Logic (PTL), frequently with custom design flows, such as those compared by Zimmermann and Fichtner (1997). PTL is an alternative logic style to the conventional complementary static CMOS that dates back to the early days of VLSI technology. PTL gained significant popularity in the '90s when many variations were developed, such as Complementary Pass Transistor Logic and Double Pass Transistor Logic, both designed by Hitachi (Kuroda; Sakurai, 1999). In light of new power-related challenges and the characteristics of contemporary technology nodes, this logic style is being revisited by researchers (Chinazzo et al., 2022; Lappas et al., 2022).

Figure 1.1 – Two different implementations of a 2-input multiplexer.
(a) Using pass transistors (b) Using complementary CMOS logic



Source: Author's own elaboration.

This logic style works by utilizing pass transistores to implement multiplexer-like gates, where both diffusion and control pins of the transistors may be connected to input signals. Differently from complementary CMOS, PTL gates may operate with passive behavior, directly passing input signals instead of implementing inverting logic with connection to V_{DD} and GND.

In Figure 1.1a, for example, a PTL multiplexer is implemented using two pass transistors. When S is high, the nMOS pass transistor will be on, passing D1 to the output Y; when S is low, the pMOS pass transistor will be on, propagating D0 to the output. This logic function is best represented by the if-then-else operator: **if** S **then** Y := D1 **else** Y := D0. In Boolean algebra, it can be represented as $Y = (\bar{S} \cdot D0) + (S \cdot D1)$.

PTL's commonly cited advantages over complementary CMOS are the smaller number of transistors to implement gates, smaller associated capacitances, and high operating speed (Zimmermann; Fichtner, 1997), which are typically presented as a smaller energy-delay product. To illustrate this point, Figure 1.1 shows two implementations of a multiplexer: the CMOS one is implemented with 12 transistors, while the PTL multiplexer comprises only two transistors. This reduction in transistor count makes it an attractive contender for the adopted logic in low-power circuits (Conceição; Reis, 2019).

Power being arguably PTL's greatest strength, it is only natural to expect that fast and accurate power estimation tools have been studied and developed for this specific logic style. As far as we know, however, this is not true. Many works on PTL compare a complementary CMOS implementation against a PTL one using data from SPICE simulation. This type of simulation numerically solves the differential-algebraic system of equations of the circuit and is known to be accurate. On the other hand, it is very computationally intensive, achieving quadratic complexity for some circuit topologies (Rewieński, 2011). This may not be a problem when characterizing cells or estimating the power of small circuits, but for larger circuits, these electrical simulations are prohibitively slow, especially during the circuit optimization phase. To be able to estimate the power of present-day ICs, it is necessary to look for alternatives.

Power models are embedded into the traditional design flow at multiple design levels and stages (such as design and validation) and are capable of outputting an estimation for PTL circuits. The problem with these traditional approaches is that they are generally designed to work with complementary CMOS and, therefore, do not account for several electrical characteristics of PTL circuits. This usually results in them being unreliable and having low-accuracy estimations when it comes to this alternative logic style.

Due to their passive behavior as a switch, pass transistors may resistively and capacitively couple multiple stages in a circuit, which means that the load capacitance of a gate might depend on stages other than its successor stage. And depending on the circuit topology, designs using PTL may also suffer from level degradation due to threshold drop: nMOS transistor passing a weak logic 1 and pMOS a weak logic 0 (Weste; Harris, 2011). This results in additional static power dissipation, which needs to be addressed by power models. Another characteristic is that this logic style is more electrically sensitive when compared to complementary CMOS, Process-Voltage-Temperature (PVT) variations having a greater effect on the circuit (Chinazzo et al., 2022).

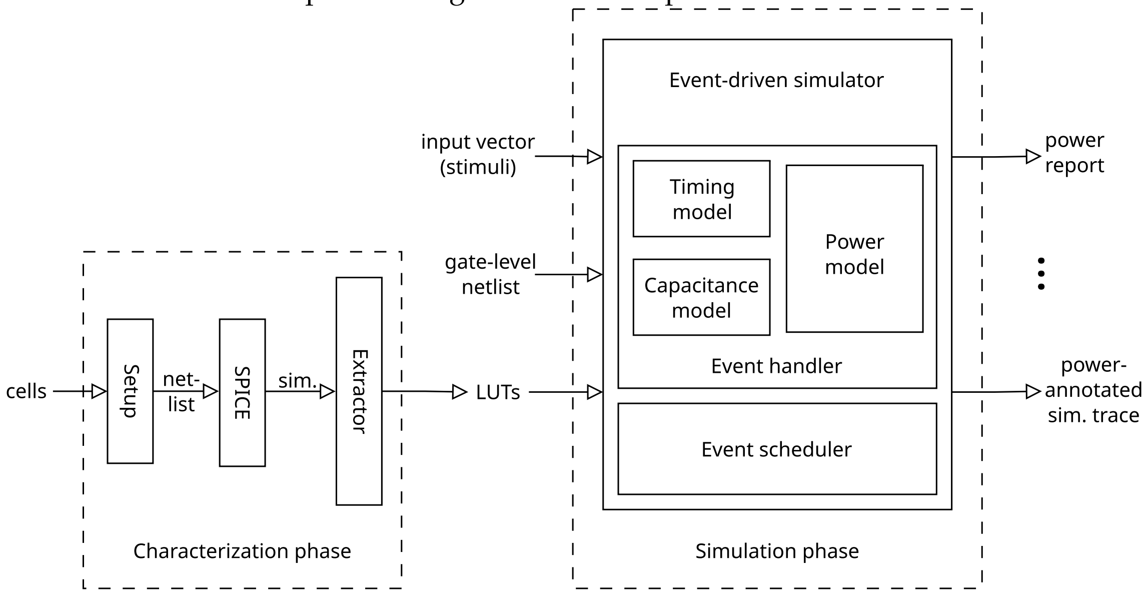
It becomes clear that for PTL to become feasible in larger designs, these electrical characteristics must be considered in the design and verification flows. While a great deal has been said about design methodology, formal methods, and logic synthesis for PTL (e.g. in Yano et al. (1996), Morgenshtein, Fish and Wagner (2002), Shelar and Sapatnekar (2005)), not as much effort has been put into the power analysis of circuits employing this logic. In fact, for all we know, no study has presented a power estimation methodology specifically tailored for PTL.

To tackle this knowledge gap, we propose a power estimation methodology that makes use of a gate-level, event-driven simulator with timing and capacitance models and PTL cell characterization data in the form of lookup tables (LUTs) to estimate the energy consumed in the events in each gate of a circuit. These estimations can then be used to generate a power report containing multiple metrics about the circuit.

Our methodology follows the usual two-step paradigm of first characterizing the standard library cells and then simulating the circuit in an event-driven fashion. This means that it is only necessary to characterize the library and generate the LUTs a single time. After this, the LUTs can be used in the power model embedded into the simulator.

The components of the methodology are related as illustrated in Figure 1.2. This work mainly addresses the characterization phase and the power model embedded in the event-driven simulator. In order to decouple the power model's error from the timing simulator error, we decided to assume ideal timing and capacitance models by extracting an event-driven trace from SPICE simulation instead of implementing the simulator itself.

Figure 1.2 – Block diagram of the power estimation methodology consisting of a preliminary characterization phase, followed by a simulation phase which estimates power using the embedded power model.



Source: Author's own elaboration.

This work is structured as follows. Chapter 2 introduces the theoretical foundations of power estimation techniques, while Chapter 3 investigates related works and influences, focusing on gate-level power analysis techniques based on simulation. In Chapter 4, we detail the methodology and implementation of the components. Chapter 5 validates and benchmarks our methodology by applying it to various circuits, including a module in a state-of-the-art low-density parity-check (LDPC) decoder. In Chapter 6, we draw our conclusions, compare them to our initial expectations, and discuss potential extensions of this work.

2 FOUNDATIONS OF POWER ESTIMATION

The total power P_{total} of a digital integrated circuit consists of dynamic and static dissipation, which can be further decomposed in the following way (Weste; Harris, 2011):

$$P_{dynamic} = P_{switching} + P_{short-circuit} \tag{2.1}$$

$$P_{static} = P_{leakage} + P_{contetion} \tag{2.2}$$

When a rising transition occurs, energy is delivered to charge the output capacitance of the gate. Half of this energy is stored in the capacitor, and half is dissipated by the transistors in the conducting path. The energy $E_{switching}$ required for a full switching cycle, i.e. a rising transition followed by a falling one, is given by:

$$E_{switching} = C_{out} {V_{DD}}^2 \tag{2.3}$$

where C_{out} is the output net capacitance.

The power associated with this energy is referred to as switching power and is governed by the following equation:

$$P_{switching} = \alpha\, C_{out} {V_{DD}}^2 f = C_{eff} {V_{DD}}^2 f \tag{2.4}$$

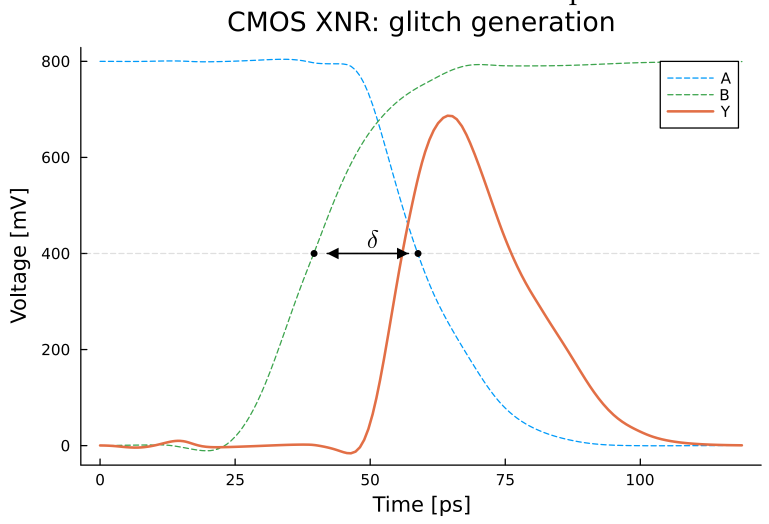
being α the switching activity of the gate, f the frequency, and C_{eff} the total output effective capacitance, which is equal to the product $\alpha \cdot C_{out}$.

Short-circuit power, on the other hand, is not related to the capacitance switching but to the low resistance path between V_{DD} and GND that is formed due to the finite slope of switching input signals. When a rising input signal has surpassed the threshold voltage of an nMOS transistor, V_{th_n}, but is still below $V_{DD} - V_{th_p}$, both pMOS and nMOS transistors controlled by this signal will be turned on, possibly causing a short circuit and drawing a large current.

Static power is consumed when no events are happening in the circuit. Transistors experience leakage currents on multiple fronts while it is quiescent. Subthreshold leakage, for instance, occurs due to the transistor operating in the weak inversion region. There are also leakage currents in the gate and junction areas. Some logic families, such as pseudo-nMOS, draw contention current due to their intrinsic functioning. However, this is not the case for complementary CMOS nor PTL, so we will not go into further details on this component.

Glitches are phenomena that greatly impact the power of some digital circuits. A glitch is a set of partial transitions generated by skewed input transitions or propagated from glitchy input signals. In Figure 2.1, for example, a glitch is generated in an XNOR gate due to the delay difference δ between inputs A and B. One of the challenges of gate-level power estimation is not overestimating glitches, as traditional simulators generally tend to consider them as complete, full-swing transitions. Considering that glitches make up to 20 % of a circuit's power (Favalli; Benini, 1995), it is essential not to overlook them when analyzing power.

Figure 2.1 – Occurrence of a glitch in the output Y of a CMOS XNOR gate due to the time difference δ between the arrival time of inputs A and B.



Source: Author's own elaboration.

The power analysis of a circuit can be carried out at varied abstraction levels, from electrical through gate to register-transfer level. At the electrical level, SPICE simulators are considered the golden standard for power analysis, but, as previously mentioned, are limited in terms of memory and runtime complexity (Rewieński, 2011). Techniques at the gate level are not as accurate but are usually several orders of magnitude faster than those at the electrical level.

At the gate level, power estimation techniques can be classified into three main categories: probabilistic, statistical, and simulation-based techniques (Nasser et al., 2021). Probabilistic techniques estimate power by using probabilistic information about a circuit's signals, such as signal probability and transition probability.

This information is calculated by traversing the circuit while evaluating the cell's logic function. With statistical techniques, the circuit is simulated by feeding selected input stimuli until the average power converges with respect to a certain threshold, e.g. following a Monte Carlo approach (Burch et al., 1993). As the name suggests, simulation-based techniques directly depend on circuit simulation to estimate power.

Commercial tools such as Synopsys' PrimePower (Synopsys, 2020) offer different kinds of power analysis flows, including vector-ful, activity-based, and vector-less analysis. Vector-ful analysis requires stimuli as input, typically in the Value Change Dump (VCD) format, while vector-less analysis can report power consumption without input vectors. Activity-based analysis is an intermediate approach, receiving switching activity information, such as a Switching Activity Interchange Format (SAIF) file, as input.

This work proposes a gate-level methodology that falls into the simulation-based category, as it is rooted in an event-driven simulator that provides timing and capacitance information. Therefore, only gate-level and simulation-based works were selected to be examined. Since our methodology requires input stimuli to be fed into the simulator, it facilitates a vector-ful power analysis. A distinguishing factor that places this work in uncharted territory is its focus on PTL circuits. We are not aware of any other work on power estimation that considers the distinct characteristics of PTL.

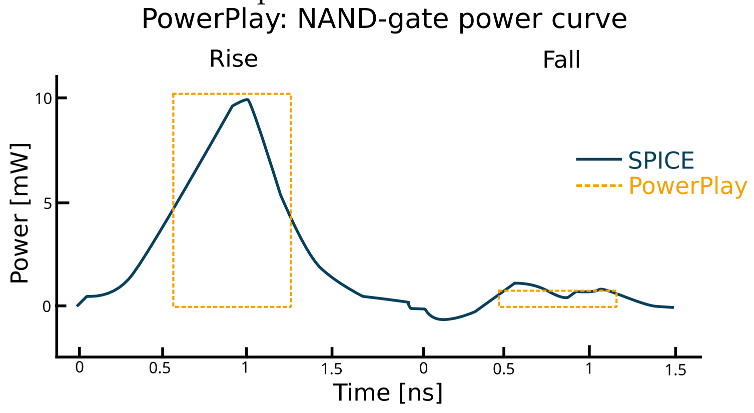
3 RELATED WORKS

In this chapter, we begin by investigating a few relevant works. Afterward, we discuss and compare their solutions to the problem of power estimation.

3.1 Investigation

Krodel (1991) implemented PowerPlay, a dynamic power estimation tool that reconstructs a BiCMOS circuit's instantaneous power waveform using a pre-characterized database and the circuit's logic trace, thus following the traditional two-step paradigm. The tool approximates an event's power curve as a rectangle whose height is equal to the transition's pre-characterized peak power, while its width is selected so that the rectangle's area is equal to the energy associated with the transition, e.g. as in Figure 3.1.

Figure 3.1 – Estimation of the power waveform of a transition in a NAND gate.



Source: Adapted from Krodel (1991).

In addition to depending on the transition and input pin, the pre-characterized power values of a cell are also variable with respect to the output load capacitance. In PowerPlay, this dependency between power and load capacitance was confirmed to be linear or nearly linear, and therefore, two parameters were used in the characterization to model it. With this dependency curve defined, power values for any load capacitance can be estimated.

George et al. (1994) introduced a two-step power analysis framework based on two tools. The first tool, Entice, characterizes library cells under multiple input slope and output capacitance conditions and manages a characterization database, which is used as input to the second tool, named Aspen, when estimating a circuit's power. This last tool depends on Verilog-XL as a simulator and estimates each event's dynamic and static power using transition- and state-dependent information from the characterization database. After estimating individual events, Aspen generates power and activity reports that can be used at different design and verification phases.

One interesting contribution of this work is that it estimates static power. It does so by characterizing the static power consumed for every input state of a cell (i.e. 2^n for n inputs) and using this information when the simulation has no dynamic power activity. Many works of the same era disregard static power because it used to be negligible when compared to dynamic power for CMOS processes with feature size above 90 nm (Weste; Harris, 2011). However, this component has become significant for newer processes.

Kruse, Rabe and Nebel (1997) presented a power analysis tool as an add-on to Cadence's Leapfrog simulator, a discontinued event-driven simulator for VHDL netlists. The authors modified the simulator's event handler to support the propagation of linear ramps, allowing the tool to also estimate glitches. With common authors, Rabe et al. (1998) developed GliPS, a standalone event-driven power simulator that accurately models glitches, also propagating transitions as ramps. The methodology of both these works employs the two-step paradigm. When executing the simulation, a full-swing event's power P_e in a gate G due to a transition on output O caused by a slope on input I is given by:

$$P_e = \frac{V_{DD}}{T} \mathcal{Q}_{G,I,O}(C_O, t_{RF_I}) \tag{3.1}$$

where $\mathcal{Q}$ is a function of output load capacitance and input slope that returns the charge drawn from the power supply in these specific conditions. C_O is the capacitance connected to the output O, and t_{RF_I} is the slope of the input I.

Effectively, what Equation 3.1 is modeling is the lumped switching and short-circuit power of a non-glitchy event. Naturally, the charge $\mathcal{Q}$ has to be characterized for the specific electrical conditions of the dynamic power event. In Rabe et al. (1998), obtaining the charge in arbitrary electrical conditions is done through linear interpolating the values obtained in the characterization phase. t_{RF_I} and C_O are also determined by the simulator using information extracted in the characterization phase.

To take glitches into account, it is necessary to introduce a correction factor to the energy estimation, as hazards generally consume less power than full transitions. In the aforementioned work, this is done by estimating the voltage peak ΔV of the transition and normalizing it with respect to V_{DD}:

$$P_{e_{glitch}} = \frac{\Delta V}{V_{DD}} P_e \tag{3.2}$$

Although this correction factor was shown to be a good heuristic for correcting the estimation of glitches, the tool has the shortcoming of not estimating static power. In contrast, George et al. (1994) estimates static power but does not take glitches into account. These are two essential features that cannot be ignored by a modern power estimation tool.

Boliolo et al. (1997) proposed PPP, a power and current simulation tool based on custom LUTs and Verilog-XL as a simulation platform. Energy is estimated by keeping track of the charge status in every node, while current pulses are approximated as triangles defined by parameters extracted from the characterization phase. Thus, the tool not only provides global and local estimations for power or energy but also does the same for time-continuous current flows, which is particularly useful for reliability analysis.

As with some of the previous works, this one only addresses dynamic power. On the other hand, this work handles glitches, and it does so differently from Kruse, Rabe and Nebel (1997). Instead of correcting a partial event's energy using its output voltage swing, it estimates the energy of a glitch by linearly interpolating it using pre-characterized information about the transition of each individual partial transitions and also the multi-bit transition equivalent to the entire glitch.

Let us take the glitch in the CMOS XNOR gate of Figure 2.1 as an example. Input A is falling while input B is rising. Ideally, both signals arrive instantaneously and the output stays at 0. However, with a delay between the signals, there is an intermediate input pattern, 11, that changes how the output behaves. There are then two input transitions, $10 \rightarrow 11 \rightarrow 01$, and the output partially transitions from 0 to 1 and back to 0.

In PPP, this glitch's energy will be composed of the energy of $10 \rightarrow 11 \rightarrow 01$ and $10 \rightarrow 01$, with different weights based on the delay δ between the input signals and the first signal's transient time T. In this case, the energy of the glitch would be equal to:

$$E = (E_{10\rightarrow11} + E_{11\rightarrow01})\frac{\delta}{T} + E_{10\rightarrow01}\left(1 - \frac{\delta}{T}\right) \tag{3.3}$$

This heuristic is based on the observation that when a glitch occurs, at some point in time a multi-bit input transition is happening. Thus, its energy should be composed of the individual single-bit transitions and the grouped multi-bit transition.

In Meixner and Noll (2017), the authors presented a sophisticated power analysis methodology focusing on the dynamic power consumption of glitches. Similar to the previous works, it is based on the characterization and subsequent lookup of events. In contrast to them, the pre-characterized LUTs contain information about multi-bit events with varying skews between the inputs, where events include full-swing transitions as well as glitches with specific pulse durations. Consequently, the characterization phase is considerably more complex than usual, characterizing events not only with respect to input slope and output capacitance but also input skew and pulse width for each input of the gate.

Moreover, the information that is retrieved from the LUTs is the output signal waveform and the power supply current waveform of an event. They are used to reconstruct the gate's overall output waveform and power supply current waveform. While the latter is directly employed to calculate the gate's power, the former serves as input for the estimation of the subsequent stage. This approach eliminates the need for a logic or timing simulator, as analog signal waveforms are estimated and propagated across the circuit.

With the analog waveforms of each input signal, event information (including time, slope, and transition) is extracted. Before looking up each one of these events in the LUT, a preliminary step is employed to group temporally adjacent events into multi-bit events or glitches, thus incorporating skew and pulse width parameters to these events.

3.2 Discussion

One common theme among these works is how they estimate dynamic power. In general, a cell is characterized under multiple electrical conditions for every possible transition arc, and then this information is looked up when estimating energy. PVT parameters fixed, the most important electrical characteristic upon which a cell's switching energy depends is its load capacitance, as supported by Equation 2.3. The preceding works all take this characteristic into account in different ways, e.g. by characterizing cells for every load in a circuit, interpolating missing power values, or assuming a linear dependency between power and load capacitance.

For short-circuit power, the relevant parameter is the slope of the input signal. Many works do not consider this parameter when estimating energy because of two main reasons. First, event-driven simulators working at the logic level do not provide timing information at all, as signals transition instantaneously. Second, even if simulators provided this information, short-circuit power is not the dominant component of dynamic power (usually less than 10 % when rise and fall times are equalized) and is even insignificant for some modern technology nodes (Rabaey; Chandrakasan; Nikolić, 2003).

As there is no contention current in the logic families relevant to this work, the estimation of static power is reduced to estimating leakage, which seems to be relatively straightforward at the gate level. It is necessary to characterize leakage for every input state of a cell and use this information to compute the gates' energy when no event is happening. One obstacle may be defining whether an event is happening or not, which depends on the gates' delays. George et al. (1994), for example, uses delay information from characterization to predict when events end and leakage currents start being drawn. However, as we are assuming an event-driven simulator with timing information, this information is directly available.

How glitches should be handled, as seen previously, is more controversial. As these works do not simulate the circuit at the electrical level, it is necessary to rely upon additional heuristics to predict glitches. Even after predicting the existence of a hazard, how to estimate its power is not so clear. Kruse, Rabe and Nebel (1997) and Rabe et al. (1998) both utilize ramp timing information, along with pre-characterized voltage values, to estimate the peak voltage of a partial transition and scale the energy down. Meanwhile, Boliolo et al. (1997) and Meixner and Noll (2017) estimate a hazard's energy value based on the delay between inputs. Therefore, a common consensus among them is that timing information is essential.

4 POWER ESTIMATION METHODOLOGY

This chapter describes the proposed power estimation methodology. First, we present an overview of the methodology. In the subsequent sections, we give more details on specific parts of the methodology: the characterization phase, algorithms for parameter extraction, the event-driven simulator, and the power model.

4.1 Overview of the methodology

The power estimation methodology, as illustrated in Figure 1.2, consists of two main steps: characterizing the cells used in the circuit and simulating the circuit while dynamically estimating its energy. An additional final step involves generating a power report from the accumulated energy data.

An external user estimating the power of a combinational circuit with a tool based on this methodology would primarily interact with the simulator's interface. The user would input the circuit netlist and stimuli into the tool, which would in turn output a power report containing all relevant power information about the simulation.

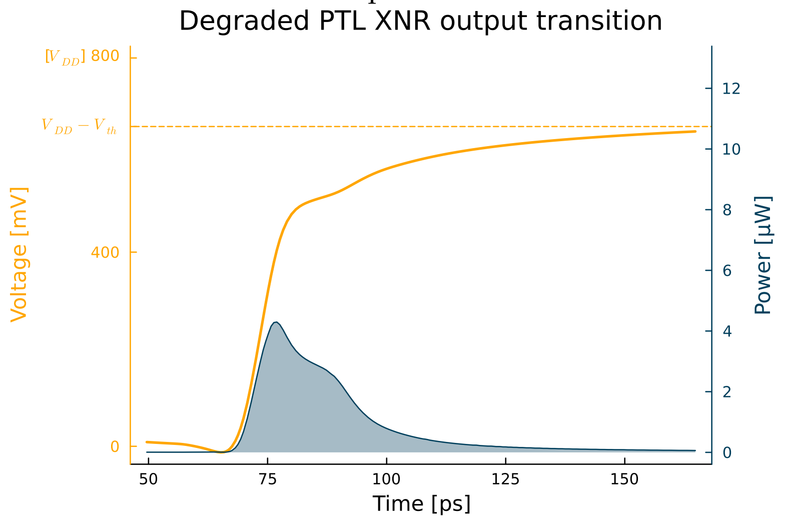
This simulation would be conducted at the gate level, providing event information about the circuit, i.e. change of the logic state of gates' inputs, including input slope and output capacitance information. Although the event-driven simulator is a critical component of the methodology, it is not the focus of this work. The timing engine presented in Lappas et al. (2024) is an example of what could be used. In this work, instead of developing a simulator from scratch, we mock its behavior using data extracted from SPICE simulations. This approach allows us to evaluate the accuracy of the power model independently of the event-driven simulator's error, as the SPICE data will be considered error-free.

Therefore, this work only covers the characterization of the cells and the power model integrated into the simulation phase. The characterization phase is a preliminary step that involves electrically simulating the cells with selected drivers and loads, extracting relevant parameters from the simulation, such as dynamic energy per input transition, and organizing them in a convenient format. In this case, a LUT indexed by input transition, input slope, and output capacitance will store dynamic energy information, while another LUT indexed by input transition will store the static power of the state after the transition.

During the gate-level simulation, the slope and capacitance information provided by the simulator will be used to access the event's energy in the first LUT. Ideally, each event in the simulation would map to a corresponding entry in the LUT. In practice, a one-to-one mapping is unfeasible because an event is subject to many variables, many of which are continuous, making it impossible to characterize every possibility. Therefore, we rely upon linear interpolation techniques to obtain continuous energy data given discrete data points.

If an event is considered a partial transition, its interpolated energy must be corrected. In this work, we will correct it with respect to the voltage swing of the associated output transition, similar to the approach in Kruse, Rabe and Nebel (1997) and Rabe et al. (1998). Given that we are assuming a simulator capable of predicting glitches, estimating the voltage swing is as simple as multiplying the associated output event's slew rate by its duration.

Figure 4.1 – Output voltage and power waveforms of a degraded output transition. Reconstructed from the SPICE simulation of a PTL XNOR gate transitioning with a $10 \rightarrow 11$ input transition.



Source: Author's own elaboration.

As explained before, PTL logic gates may pass a degraded signal, where the output only swings to within V_{th} of V_{DD} or GND, forming a horizontal asymptote, as illustrated in Figure 4.1. This means that, unlike previous works, we cannot normalize the output voltage swing with respect to V_{DD}, as these degraded transitions are considered complete even though they do not reach V_{DD} or GND. Instead, we characterize a cell's degraded transitions and normalize the voltage swing with respect to $V_{DD} - V_{th}$ when faced with one.

Following every transition, there is an interval of static dissipation until the start of the next event. This is estimated through the lookup of static power values in a similar approach to George et al. (1994). After estimating the total energy consumed by each logic gate, we generate a power report containing the total consumed energy and average power of the circuit, as well as detailed information for individual gates and cell types.

4.2 Cell characterization

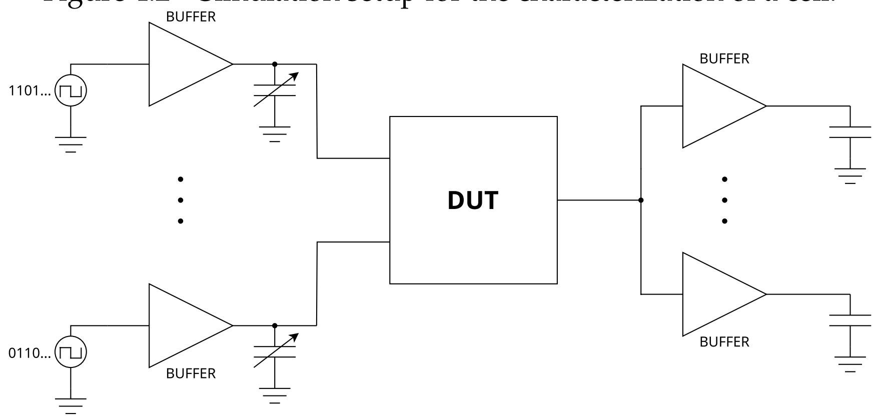
The objective of the characterization phase is to obtain the relevant electrical characteristics of cells and arrange them in a convenient format for power estimation. But first, we need to define what characterization information is required by our power model. As introduced previously, the model accumulates the energy of each event occurring in a gate, requiring that we characterize the energy consumed in each associated input transition. It is also necessary to extract the static power (not energy) dissipated after each event. These are the two main parameters that are required to estimate dynamic and static energy.

Under the single-input switching assumption of the event-driven simulator, it is unnecessary to simulate and extract these parameters for every possible input transition. For a cell with n inputs, we only need to simulate $n \cdot 2^n$ possible single-bit input transitions instead of the $4^n - 2^n$ possible n-bit transitions. To account for any potential noise, we repeat each input transition multiple times in random order and average their extracted values when extracting parameters.

As seen in Chapter 3, load capacitance and input slope are the two main variables influencing dynamic power. Therefore, the key to this characterization is to simulate the cell under different driving and loading conditions and extract dynamic energy values. These values must be generic enough to apply to arbitrary circuits containing the cell. For this to be possible, the characterization must incorporate realistic driver and receiver models, capturing the intricacies of the circuit.

One way to achieve realistic conditions is by using actual buffers as drivers and loads. We can simulate different load conditions by varying the number

Figure 4.2 – Simulation setup for the characterization of a cell.



Source: Author's own elaboration.

of parallel buffers connected to the output or by adjusting the buffer's driving strength, which in turn affects its input capacitance. To control the cell's input slope, we can add an ideal variable capacitance to the input or use buffers with different driving strengths.

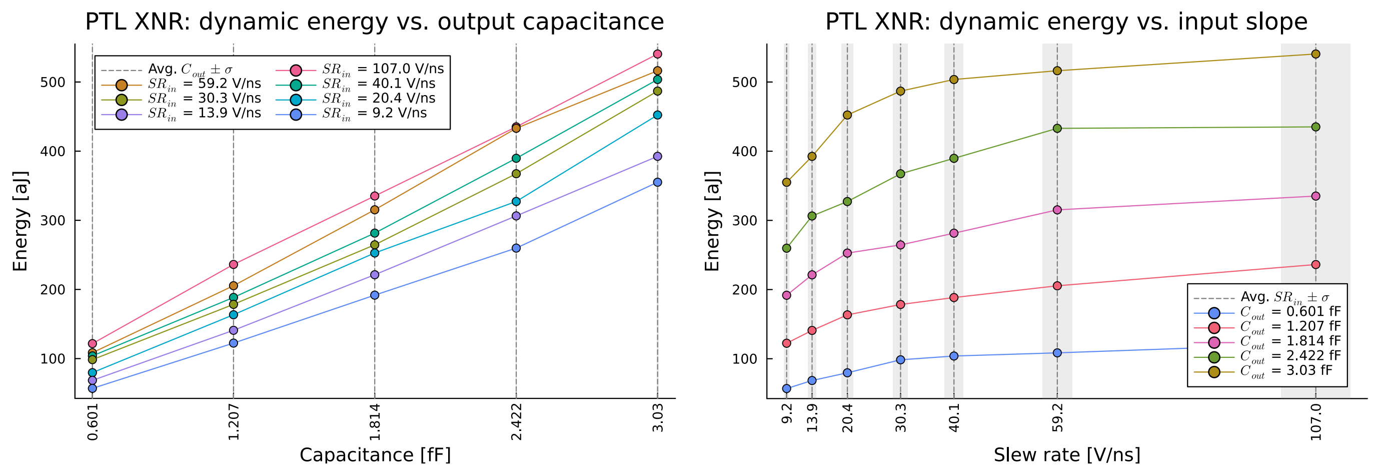
Figure 4.2 shows the testbench setup for characterization. The cell, referred to as the Device Under Test (DUT), is wrapped with minimal buffers, employing previously explained techniques to control input slope and load capacitance. The buffers working as drivers are fed stimuli to simulate all single-bit transitions of the cell.

To ensure the characterization is as realistic as possible, we extract the parasitics of both the DUT and buffer and include them in the simulation. The simulations are run on Cadence's Spectre using a 12 nm FinFET technology node with super-low $V_{\mathtt{th}}$ devices under the following PVT conditions: typical-typical corner, 0.8 V and 27 °C.

We selected the following driving and loading conditions to be characterized: drivers with ideal capacitances of 0, 1, 2, 3, 5, 8, and 13 fF; and loads consisting of 1, 2, 3, 4, and 5 minimal buffers in parallel. After running the simulations for these conditions, we extract the energy consumed by each single-bit transition, as well as their input slope and output capacitance. This data is sufficient to characterize the cell's dynamic power dissipation, as depicted in Figure 4.3 for a specific input transition in a PTL XNOR gate. To characterize its static power, we measure the average DUT power with respect to the energy that was consumed *after* each single-bit transition when the circuit is quiescent.

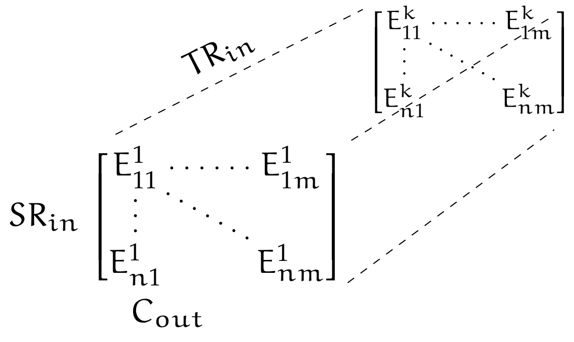
Figure 4.3 – Dynamic energy characterization of an 11 → 10 input transition in a PTL XNOR gate. Plotted against output capacitance and input slew rate.

(a) Against output capacitance (b) Against input slope



Source: Author's own elaboration.

Figure 4.4 – Three-dimensional array representation of dynamic energy values in the LUT. It is composed of input slew rate SR_{in}, output capacitance C_{out}, and input transition TR_{in} dimensions.



Source: Author's own elaboration.

If we characterize n driving and m loading conditions, there will be a total of $n \cdot m$ simulations, where each simulation covers every single-bit input transition. The dynamic energy information extracted from these simulations is arranged in a three-dimensional LUT, i.e. an array indexed by input slope, output capacitance, and input transition, as shown in Figure 4.4. Static power information is arranged in a similar way, but is only indexed by input transition.

It is important to clarify the distinction between driving and loading conditions versus actual input slope and output capacitance. Driving and loading conditions refer to the cells used as drivers and loads, while input slope and output capacitance are the actual values calculated using data from simulations. Although there are m simulation setups with the same driver, the actual slopes associated with the DUT inputs will not be the same due to the varying loads. For a complementary CMOS DUT, the variance in input slope is zero for all intents and purposes, because the output is decoupled from the input aside from parasitics. For PTL DUTs, this variance is more relevant due to possible paths directly coupling inputs to outputs.

When computing the LUT keys, we average these different values for the same condition. Thus, m simulations with a specific driver but multiple loads yield a single input slope value as key, corresponding to a line in Figure 4.4. In the same manner, n simulations with the same load but different drivers will yield a single output capacitance key, represented by a column in Figure 4.4. This averaging process is especially noticeable for PTL cells as illustrated by Figure 4.3b, where the gray bands represent the interval within one standard deviation σ of the mean (vertical dashed lines). For the PTL XNOR's 11 $\rightarrow$ 10 input transition in Figure 4.3, it is the cell's driver (instead of the cell itself) that is actively discharging the output capacitance through one pass transistor in the XNOR gate, resulting in the cell's input slope varying with different output capacitances. Although averaging these values introduces some error to the estimation of PTL cells, it was deemed acceptable based on the results presented in Chapter 5.

4.3 Extracting parameters from simulations

This section describes the algorithms used to extract output capacitance, input slope, and dynamic and static energy from SPICE simulations. First, we shall define a transition's start and end time points, here referred to as t_{start} and t_{end}, respectively. These time points are calculated relative to 2 % and 98 % of the transitions' minimum and maximum voltages, $V_{min} \geqslant GND$ and $V_{max} \leqslant V_{DD}$. We cannot calculate them relative to V_{DD} and GND because PTL gates may not swing to these voltages due to voltage degradation.

For rising and falling transitions (denoted by the subscripts R and F), t_{start} and t_{end} can be expressed as follows:

$$t_{start_R} = t_{end_F} = \text{cross}\left(V_{min} + \frac{2}{100} V_{max}\right) \tag{4.1}$$

$$t_{start_F} = t_{end_R} = \text{cross}\left(V_{min} + \frac{98}{100} V_{max}\right) \tag{4.2}$$

where $\mathrm{cross}(V)$ is a function that returns the time point at which the voltage V is crossed.

With these definitions, computing the dynamic energy associated with an input transition is straightforward. We integrate the logic gates' instantaneous power waveform from t_{start} to t_{end} of the output transition caused by the input transition. This time window captures the instantaneous power bell curve related to the event. By definition, energy consumed outside this window will be regarded as static dissipation.

Next, we define the equation for output capacitance. From its fundamental electrical definition, the output capacitance C_{out} associated with an input transition is the charge drawn through the gate's output divided by the voltage swing of the output net:

$$C_{out} = \frac{1}{\Delta V_{out}} \int_{t_{start}}^{t_{end}} I_{out}(t)\,dt \tag{4.3}$$

where t_{start} and t_{end} are the start and end time points of the associated output transition, ΔV_{out} is the voltage difference from t_{start} to t_{end} and $I_{out}(t)$ is the gate's output current.

Lastly, we calculate the input slope as a 20 %-to-80 % slew rate (unit of $\mathrm{V\,s^{-1}}$) instead of the traditional 20 %-to-80 % rise or fall time (unit of s). This approach is necessary because PTL cells may have a different voltage swing compared to normal transitions, and comparing rise or fall times with different voltage swings leads to significant estimation errors. Slew rate, on the other hand, is a metric that essentially measures the rate of voltage change over time, so different voltage swings are comparable.

Given that the 20 %-to-80 % voltage swing ΔV is:

$$\Delta V = \left(\frac{80}{100} - \frac{20}{100}\right) V_{max} \tag{4.4}$$

and the rise or fall time t_{RF}, using the same values of lower and upper threshold, is:

$$t_{RF} = \left| \mathrm{cross}\left(V_{min} + \frac{20}{100} V_{max}\right) - \mathrm{cross}\left(V_{min} + \frac{80}{100} V_{max}\right) \right| \tag{4.5}$$

then we define the input slew rate $\mathtt{SR}_{in}$ as:

$$\mathtt{SR}_{in} = \frac{\Delta V}{t_{RF}} \tag{4.6}$$

These algorithms are essential for both extracting DUT parameters during cell characterization and extracting the event-driven trace from the SPICE simulation, as outlined in the following section.

4.4 Event-driven simulator

In this work, the event-driven simulator is not directly implemented but instead emulated using data from SPICE simulations. In any case, it is crucial to clearly define the simulator's output trace, as it will be used as input to the power model, which expects parameters not traditionally provided by gate-level simulators. While most traditional simulators only evaluate logic functions and propagate stimuli as instantaneous transitions, the simulator in this work is assumed to have similar capabilities to the one developed in Lappas et al. (2024), specifically:

1. Modeling of transition waveforms with timing information. Although Lappas et al. (2024) proposes a novel exponential model, we opt for the simplicity of linear ramps, which is sufficient for the purposes of this work.
2. Dynamic modeling of input capacitance. Unlike static models that assume constant capacitance, this dynamic model uses a recurrent algorithm to compute capacitance on the fly based on previous input waveforms. This is essential for accurately estimating circuits with pass transistors, as it accounts for the potential coupling effects across multiple stages.
3. Generation and propagation of partial transitions, as well as modeling of degraded transitions, incorporating the event's end time as an additional parameter.

The simulator's trace is here considered a vector of events for each logic gate in the circuit. An event is defined by several parameters: input transition $\mathtt{TR}_{in}$, start time t_{start}, end time t_{end}, input slew rate $\mathtt{SR}_{in}$, and output capacitance C_{out}. The trace format and these parameters were chosen for their compatibility with the power model, facilitating the estimation, but, internally, alternative structures could have been employed for implementation purposes.

To mock the event-driven trace, we use the same algorithms described in the previous section to extract the necessary parameters from SPICE simulations. First, each input net in each gate is monitored, triggering the creation of an event when the voltage crosses 50 % of V_{DD}. The start and end times, t_{start} and t_{end}, are extracted as in Equation 4.1, using 2 % and 98 % of the voltage range, and the input slew rate SR_{in} is extracted as in Equation 4.6.

The aforementioned parameters are easily determined because they are computed relative to the *input* signals of a gate. In contrast, C_{out} must be calculated relative to the *output* events of the gate and then mapped to the input events that caused them. This input-to-output map is readily available in logic simulators, as they strictly adhere to cause-effect relationships between inputs and outputs. As we are emulating the simulator, a decision tree is used to extract this relationship from SPICE, possibly merging opposing input transitions when an output event is not found.

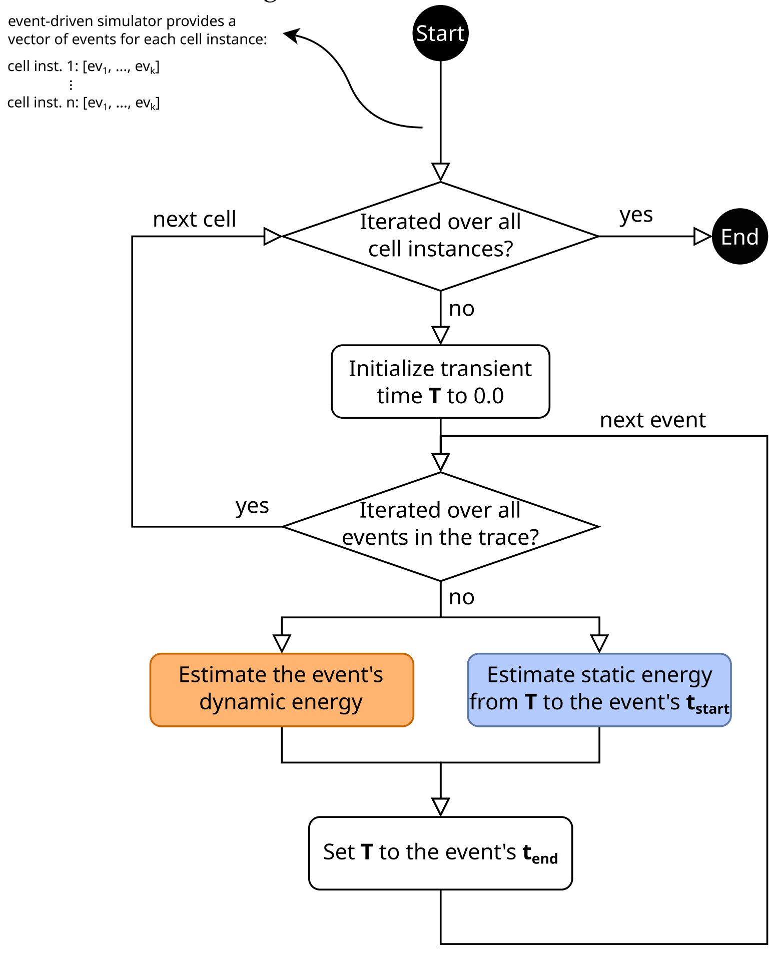
By using the same algorithms for both characterization and simulation phases, we ensure that characterized transitions are comparable to simulated events, allowing the power model to accurately estimate dynamic energy based on the pre-characterized data.

4.5 Power model

The general idea of the power model is to use characterized transition information stored in LUTs to estimate the energy consumed by events occurring in a gate, as well as the energy consumed when no events occur. The model's inputs include the LUTs and a vector of events for each logic gate in the circuit, while its output is the estimated energy for each gate.

The algorithm employed in the power model is illustrated by the flowchart in Figure 4.5. It consists of two nested loops: the outer loop iterates over logic gates, while the inner loop iterates over events for each specific gate. During iteration, the algorithm estimates not only the dynamic energy of each event but also the static energy consumed from the end of the previous event to the start of the current

Figure 4.5 – Flowchart of the general algorithm used to estimate the energy of a circuit given some stimuli.


event-driven simulator provides a vector of events for each cell instance:
cell inst. 1: [ev_1, ..., ev_k]
⋮
cell inst. n: [ev_1, ..., ev_k]
Start
Iterated over all cell instances?
next cell
yes
End
no
Initialize transient time **T** to 0.0
next event
yes
Iterated over all events in the trace?
no
Estimate the event's dynamic energy
Estimate static energy from **T** to the event's t_{start}
Set **T** to the event's t_{end}

Source: Author's own elaboration.

event. This is achieved by keeping track of the elapsed transient time, along with the previous event's information.

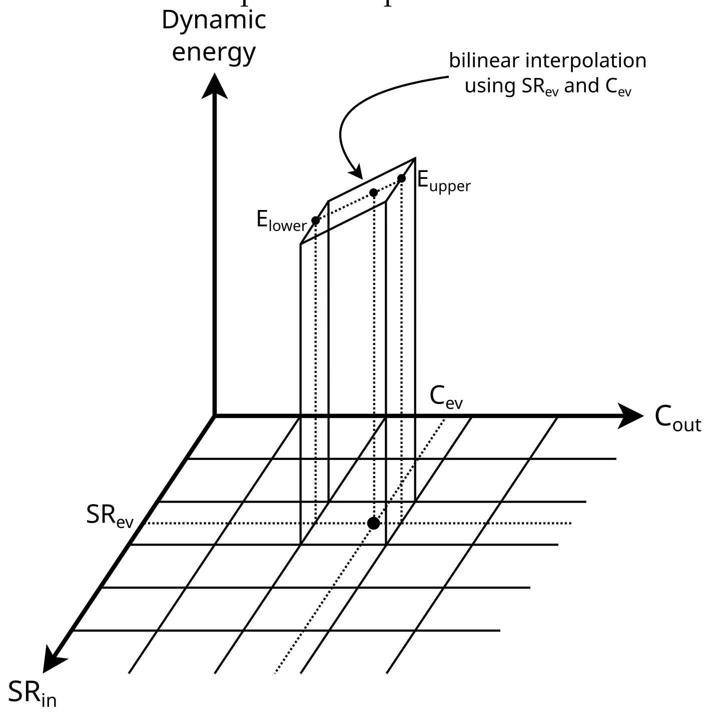
The orange and blue squares in Figure 4.5 represent the two main parts of the power model. The orange one on the left is the part that estimates dynamic energy, while the blue one estimates static energy. We will first explain the estimation of dynamic energy, which is the dominant component of power dissipation (Weste; Harris, 2011), followed by the estimation of static energy.

4.5.1 Dynamic energy estimation

We have the following information at our disposal to estimate the energy of an event: input transition TR_{in}, start time t_{start}, end time t_{end}, input slew rate SR_{in} and output net capacitance C_{out}. Since the LUT most certainly does not contain the exact match for our event's $\mathsf{SR_{in}}$ and $\mathsf{C_{out}}$, we access the neighboring keys.

We denote the lower and upper keys for the parameter P in the LUT by the symbols $\lfloor \mathsf{P} \rfloor$ and $\lceil \mathsf{P} \rceil$, respectively, such that $\lfloor \mathsf{P} \rfloor < \mathsf{P} < \lceil \mathsf{P} \rceil : \lfloor \mathsf{P} \rfloor, \lceil \mathsf{P} \rceil \in \mathsf{LUTKEYS}$. Given $\lfloor \mathsf{SR_{in}} \rfloor$, $\lceil \mathsf{SR_{in}} \rceil$, $\lfloor \mathsf{C_{out}} \rfloor$ and $\lceil \mathsf{C_{out}} \rceil$, we can access four energy values in the LUT that enclose the energy of the event.

Figure 4.6 – Bilinear energy interpolation scheme with respect to input slew rate and output load capacitance.



Source: Author's own elaboration.

As depicted in Figure 4.6, these four values will be used in a bilinear interpolation scheme. First, we linearly interpolate the energy values with respect to the input slew rate, fixing the output capacitance key to $\lfloor \mathsf{C_{out}} \rfloor$, which results in the energy lower bound $\mathsf{E_{lower}}$. We then do the same but this time fixing the capacitance key to $\lceil \mathsf{C_{out}} \rceil$ to obtain the energy upper bound $\mathsf{E_{upper}}$:

$$
\begin{aligned}
m_{lower} &= \frac{E\left[TR_{in}, \lceil SR_{in} \rceil, \lfloor C_{out} \rfloor\right] - E\left[TR_{in}, \lfloor SR_{in} \rfloor, \lfloor C_{out} \rfloor\right]}{\lceil SR_{in} \rceil - \lfloor SR_{in} \rfloor} \\
m_{upper} &= \frac{E\left[TR_{in}, \lceil SR_{in} \rceil, \lceil C_{out} \rceil\right] - E\left[TR_{in}, \lfloor SR_{in} \rfloor, \lceil C_{out} \rceil\right]}{\lceil SR_{in} \rceil - \lfloor SR_{in} \rfloor} \\
E_{lower} &= E\left[TR_{in}, \lfloor SR_{in} \rfloor, \lfloor C_{out} \rfloor\right] + m_{lower} \cdot (SR_{in} - \lfloor SR_{in} \rfloor) \qquad (4.7) \\
E_{upper} &= E\left[TR_{in}, \lfloor SR_{in} \rfloor, \lceil C_{out} \rceil\right] + m_{upper} \cdot (SR_{in} - \lfloor SR_{in} \rfloor) \qquad (4.8)
\end{aligned}
$$

where $E\left[TR_{in}, SR_{in}, C_{out}\right]$ is the dynamic energy lookup using TR_{in}, SR_{in} and C_{out} as input transition, input slew rate, and output capacitance keys.

Next, E_{lower} and E_{upper} are linearly interpolated with respect to the output capacitance to obtain E_{interp}:

$$
\begin{aligned}
m_{interp} &= \frac{E_{upper} - E_{lower}}{\lceil C_{out} \rceil - \lfloor C_{out} \rfloor} \\
E_{interp} &= E_{lower} + m_{interp} \cdot (C_{out} - \lfloor C_{out} \rfloor) \qquad (4.9)
\end{aligned}
$$

If the event is a complete transition, then our final energy estimation shall be E_{interp}. However, if it is part of a glitch, then it is necessary to correct its estimation. To do so, we multiply E_{interp} by a correction factor similar to the one in Equation 3.2, as described by Kruse, Rabe and Nebel (1997).

This correction factor is based on the principle that a partial transition's energy is proportional to its swing voltage ΔV. From Equation 2.3, we can infer that the energy of a single complete transition, without voltage degradation, is:

$$E_{switching} = \frac{1}{2} C_{out} V_{DD}{}^2 \qquad (4.10)$$

This equation can be generalized to partial transitions in the following way:

$$E_{switching_{glitch}} = \frac{1}{2} C_{out} V_{DD} \, \Delta V = \frac{\Delta V}{V_{DD}} E_{switching} \qquad (4.11)$$

where $\Delta V / V_{DD}$ is our correction factor. However, this is only valid for transitions without voltage degradation. For degraded transitions, the energy is not governed by Equation 4.10, as it does not swing up to V_{DD}. Therefore, we revise Equation 4.11 so that the correction factor also accounts for degraded transitions:

$$E_{switching_{glitch}} = \frac{\Delta V}{V_f} E_{switching} \tag{4.12}$$

where V_f is the final voltage swing of the complete transition. For a non-degraded transition, V_f should be V_{DD}; for a degraded one, it should be $V_{DD} - V_{th}$. Thus, our final dynamic energy estimation $E_{dynamic}$ is:

$$E_{dynamic} = \frac{\Delta V}{V_f} E_{interp} \tag{4.13}$$

4.5.2 Static energy estimation

The method for estimating static energy shares similarities with the estimation of dynamic energy. It involves monitoring the occurrence of events to determine when the circuit is quiescent and the duration of these periods of static dissipation. During these quiescent periods, we access the static power entries in the LUT and calculate the energy dissipated over the period.

As illustrated in Figure 4.5, the algorithm iterates through events and estimates the static energy dissipated from the previous event's t^{n-1}_{end} to the current event's t^{n}_{start}. The energy dissipated in this interval is associated with an input state effectuated by the previous event's input transition TR^{n-1}_{in}. This input transition is the sole key used to access the static power data in the LUT.

We can formally define the static energy E_{static} dissipated in an interval with duration ΔT as follows:

$$E_{static} = P\left[TR_{in}\right] \cdot \Delta T \tag{4.14}$$

where $P\left[TR_{in}\right]$ is the static power lookup using TR_{in} as input transition key. This input transition is the transition that led to this state of static dissipation.

4.6 Generating the power report

After executing the algorithm detailed in the previous section, we obtain, for each gate in the circuit, a list of objects describing the energy activity within the gate. These objects have information such as the consumed energy, the activity's start and end time points, and whether the activity is associated with dynamic or static power dissipation. This section describes how we can use this output in a power report.

We can extract several metrics to be included in the power report. Two primary metrics are the total energy and the average power of the whole circuit over the simulated period. While both metrics are useful for analyzing the power behavior of a specific input pattern, the average power is particularly interesting when used with statistical techniques, such as the method in Burch et al. (1993), in order to achieve a vector-free power analysis. Additionally, these metrics can also be decomposed into their dynamic and static components.

Instead of calculating energy and average power for the whole circuit, these metrics can also be computed for different levels of granularity. For example, power metrics for each *logic gate* can help identify hotspots in the circuit and allow for targeted optimizations. Similarly, computing these metrics for each *cell type* can assist designers in deciding which cells to optimize.

5 RESULTS AND DISCUSSION

We used the Julia programming language to implement scripts to apply the power estimation methodology detailed in Chapter 4 across multiple circuits. We initially used them to estimate the power consumption of basic characterization circuits as a preliminary way of assessing the implementation and the methodology for any potential major anomalies. Then, we benchmarked the methodology's accuracy by applying it to a realistic state-of-the-art circuit: the sign-parity computation module in an LDPC decoder. The evaluation of these tests and benchmarks is conducted based on the error metrics outlined in the subsequent section.

5.1 Error metrics

We define two major error metrics, each of them serving a different purpose: total error (TE) and mean absolute error (MAE). They are defined as follows:

$$\mathrm{TE}(\widehat{E}) = \sum_{i \in P} \widehat{E}_i - E_i \tag{5.1}$$

$$\mathrm{MAE}(\widehat{E}) = \frac{\sum_{i \in P} |\widehat{E}_i - E_i|}{\#P} \tag{5.2}$$

Here, $\widehat{E}_i$ is the model's energy estimator for a period $i \in P$, where P is the set of estimated clock periods. E_i is the SPICE energy for the period $i \in P$, which is considered the ground truth in relation to the model. The symbol # is used to denote the cardinality of a set.

Using energy per period to calculate error may seem unusual since the power model's main output is the average power of a circuit given some stimuli. However, the average power is simply derived from energy per period, which is a more finely-grained output that better represents the core of the model. Therefore, evaluating energy with respect to periods enables more detailed metrics such as MAE.

We also define their respective normalized forms (given in %) so that different datasets can be compared:

$$\mathrm{NTE}(\widehat{\mathsf{E}}) = \frac{\mathrm{TE}(\widehat{\mathsf{E}})}{\sum_{i \in P} \mathsf{E}_i} \tag{5.3}$$

$$\mathrm{NMAE}(\widehat{\mathsf{E}}) = \frac{\#P \cdot \mathrm{MAE}(\widehat{\mathsf{E}})}{\sum_{i \in P} \mathsf{E}_i} \tag{5.4}$$

TE represents the error that will be seen by the final user, as the total energy error is directly comparable to the average power error. One trait of this metric is that the energy per error is naively accumulated, so one period's underestimation may cancel out another period's overestimation and mask part of the total error of the internal model. From TE, we can infer the direction of the overall error, i.e. either under- or overestimation.

MAE is an error metric that accumulates the *absolute value* of each period error and then takes the average, therefore not facing the same problem as TE. As an analogy, while TE may serve to indicate the overall error's direction, MAE expresses its magnitude. We chose to evaluate the model using MAE instead of the more common root-mean-square error because the latter is disproportionately affected by outliers due to the squaring of the error. Since the main objective of the methodology is to compute the *average* power over a bulk of periods, outliers are weighted linearly, thus making MAE a more natural metric of error.

5.2 Cell characterization circuits

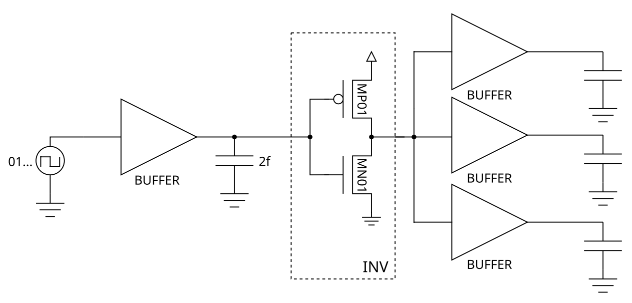
Our initial objective is to test our implementation of the methodology for major anomalies and validate the methodology in the context of simple power analysis scenarios. One way to achieve this is to apply the implementation to the cell characterization circuits that were simulated in Section 4.2. In the characterization phase, we extracted several parameters from these SPICE simulations so that the LUTs could be constructed. When estimating these circuits' power using these LUTs, information extracted from simulations would be used to estimate these same simulations, in a circular manner. If there are no significant flaws in our implementation, the estimation should be highly accurate, thus serving as a preliminary test for the overall correctness of our methodology and implementation.

Furthermore, due to their simplicity, it should be straightforward to model these circuit's power with relatively high precision. And by adjusting some of their parameters, it is possible to assess different parts of the methodology. For instance, estimating a basic circuit whose driving and loading conditions are not covered in the characterization LUT allows us to judge the interpolation's error. Similarly, by estimating a PTL cell with level degradation, we can analyze the methodology's accuracy in the presence of degraded signals. Such tests may reinforce our understanding of the methodology's applicability across different contexts.

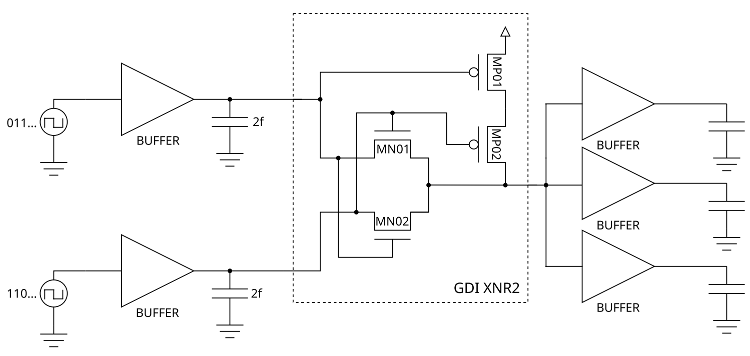
The inverter serves as the most fundamental block for digital design, from which the electrical behavior of more complex CMOS circuits can be derived (Rabaey; Chandrakasan; Nikolić, 2003). Given its significance, as a first test we estimate the power of a minimal complementary CMOS inverter driven by minimal buffers with an ideal capacitance of 2 fF connected to their outputs and having a load of three minimal buffers, as portrayed by Figure 5.1a. Since this circuit's SPICE simulation was used to generate the LUT for this cell, there will be near-exact matches for the events in the LUT, and consequently, interpolation will not play a significant role in the estimation. The matches will not be perfectly exact because the LUT keys are composed of average values, as explained in Section 4.2. The estimation results for this setup are available in Table 5.1 under the name INV 1.

Figure 5.1 – Simulation setups for the characterization of two different cells under specific driving and loading conditions.

(a) INV 1 setup



(b) XNOR setup



Source: Author's own elaboration.

As expected, the errors for setup INV 1 are low, indicating that our methodology and implementation are effective for this basic scenario. Moreover, the results for INV 1 also give us the confidence that our method of building the LUTs, averaging values to use as keys, is not a significant source of error in the estimation of complementary CMOS cells.

As a second test, we estimate the energy of the same setup, INV 1, but with a slightly different driver. Instead of connecting an ideal capacitance of 2.0 fF to the output of the driving buffer, we connect a capacitance of 2.5 fF. As the load, we also include an additional ideal capacitance of 0.3 fF in parallel with the three buffers. The characterization LUT includes data from simulations with drivers with a capacitance of 2.0 fF and 3.0 fF, but not 2.5 fF. In a similar manner, the LUT also does not contain data for this specific loading condition, which implies that energy values are interpolated by the power model. This test aims to evaluate the error introduced by interpolating pre-characterized data points. The results for this setup are presented in Table 5.1 under the label INV 2.

Table 5.1 – Comparison of estimated and actual SPICE energy values and error metrics for the power estimation of different cell characterization circuits.

Setup	Est. energy [aJ]	SPICE energy [aJ]	TE [aJ] (%)	MAE [aJ] (%)
INV 1	10 178.38	10 266.65	−88.26 (−0.86 %)	8.83 (1.98 %)
INV 2	7 860.07	8 158.00	−297.93 (−3.65 %)	15.44 (4.35 %)
XNOR	98 063.59	94 646.95	3 416.64 (3.61 %)	16.74 (5.64 %)

Source: The author.

As observed, both TE and MAE for setup INV 2 are larger than those for setup INV 1, but they still remain relatively low when normalized. This indicates that although interpolating the data points may have introduced some error to the estimation, the error is not substantial in the context of complementary CMOS cells. It is relevant to note that estimating setup INV 2 using LUTs generated with more sparse driving conditions yields different results, as the LUT keys will be calculated differently and energy values will be interpolated using possibly more distant data points.

Our goal in this final test is to assess the error associated with the electrical behavior of PTL cells, especially the influence of degraded transitions on the estimation. To achieve this, we apply the methodology to the characterization circuit of a PTL XNOR cell, as shown in Figure 5.1b, using the same drivers and load as setup INV 1. The results for this setup are displayed in Table 5.1 under the name XNOR.

The larger MAE for setup XNOR suggests that the model's accuracy decreased in comparison to the modeling of the inverter, which was to be expected considering the higher complexity of such a cell. Nevertheless, in absolute terms the total error still remains on par with the other setup's total errors.

5.3 Sign-parity computation module in an LDPC decoder

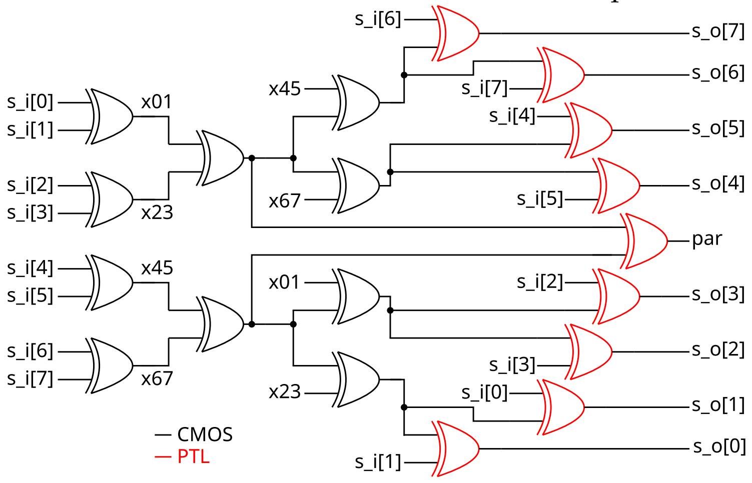
LDPC codes are a family of forward error correction codes widely used in unreliable communication channels, such as in IEEE 802.11 compliant networks (IEEE..., 2021). When optimally designed, these codes have been shown to achieve channel capacity close to the Shannon limit. LDPC codes are defined by a sparse parity check matrix $\mathbf{H}$, with columns representing code bits and rows representing parity checks. For our purposes, they are more effectively represented as a factor graph, where bits correspond to variable nodes and parity checks to check nodes, with edges connecting a variable node j to a check node i if and only if $\mathbf{H}_{ij} = 1$.

A prominent algorithm used in LDPC decoders is the message passing algorithm, which iteratively refines the codeword decision by exchanging messages between variable and check nodes until convergence or a maximum number of iterations is reached. This process, known as soft decoding, makes a decision not only based on hard information, i.e. whether a bit is one or zero, but also on soft information, which can be interpreted as the probability that the bit is actually correct. Check nodes often represent them as a sign and magnitude.

The implementation of a check node can be divided into processing the sign and magnitude of the check-to-variable messages. The sign-parity module handles the sign part, calculating the signs of the extrinsic messages through XOR operations. It also computes the parity bit of the variable-to-check messages as a byproduct, which is used in approximate parity checking to implement an early termination criteria for the message passing algorithm. This last technique can reduce the power consumption of some decoders by more than 60 % (Darabiha; Carusone; Kschischang, 2008).

The sign-parity module is structured as an XOR tree. Figure 5.2 illustrates an implementation with a degree $d_c = 8$, where the final stage utilizes PTL cells. In contrast, the module simulated in this work has a higher degree $d_c = 20$, featuring a deeper XOR tree. In total, the synthesized module is mapped to 67 logic gates, including complementary CMOS cells, such as inverters and AOIs, and PTL cells, like multiplexers and XNORs. Among these, 21 are PTL gates driving the module's

Figure 5.2 – Design of a sign-parity computation module for a check node with $d_c = 8$. Red instances indicate PTL cells, while black instances represent CMOS cells.



Source: Author's own elaboration.

outputs (`s_o[0:19]` and `par`). Note that these gate counts do not include the minimal buffers as drivers and loads used for simulation purposes.

The module was simulated on Cadence's Spectre with 800 random input transitions under the same technology node and PVT conditions as the characterization setups in Section 4.2. Then, the event-driven trace is extracted from the simulation to estimate the module's energy consumption based on our methodology. The results are presented in Table 5.2.

Table 5.2 – Comparison of estimated and actual SPICE energy values, along with error metrics, of the sign-parity computation module in an LDPC decoder.

Est. energy [fJ]	SPICE energy [fJ]	TE [fJ] (%)	MAE [aJ] (%)
35 367.54	36 971.85	−1 604.31 (−4.34 %)	2 590.46 (5.61 %)

Source: The author.

As part of a state-of-the-art design, the sign-parity module is remarkably more intricate than the characterization circuits and displays numerous characteristics that may be challenging to model. It comprises interacting logic gates spanning several stages, generating and propagating hazards throughout the circuit. Among these gates, many are PTL cells exhibiting coupling and level degradation. Nevertheless, the module's total estimation error, TE, remains well within our initial goal. The additional complexity of such a circuit also did not significantly impact the MAE.

A breakdown of the results for each cell type is presented in Table 5.3. Although the estimated energy of most cell types is within 10 % of SPICE, the

Table 5.3 – Comparison of estimated and actual SPICE energy values, along with error metrics, for the cell types of the sign-parity computation module in an LDPC decoder.

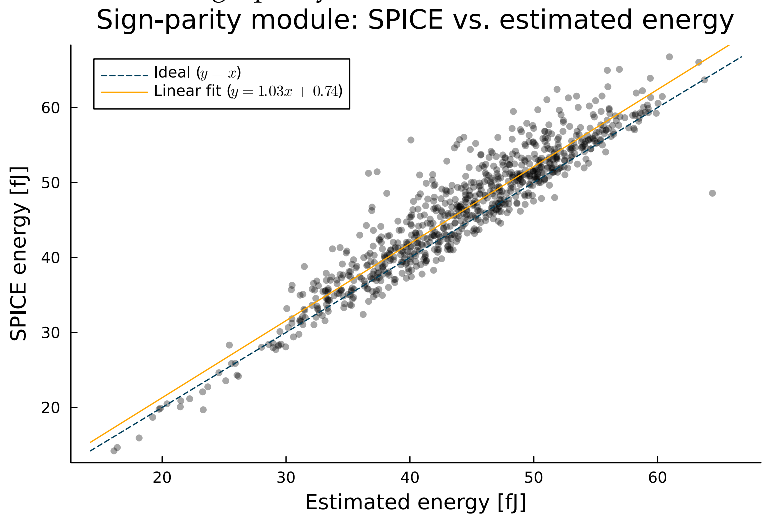
Cell type	Est. energy [fJ]	SPICE energy [fJ]	TE [fJ] (%)	MAE [aJ] (%)
AO22IA1A2	20 534.50	21 454.15	−919.65 (−4.29 %)	1 769.25 (6.61 %)
PTL XNOR	5 446.68	5 380.44	66.23 (1.23 %)	317.22 (4.72 %)
OAI22	2 377.07	2 709.08	−332.01 (−12.26 %)	484.76 (14.33 %)
AOI22	2 348.36	2 651.05	−302.69 (−11.42 %)	427.17 (12.91 %)
INV	2 177.80	2 377.91	−200.11 (−8.42 %)	255.44 (8.60 %)
OA22IA1A2	1 303.99	1 237.54	66.45 (5.37 %)	158.10 (10.23 %)
PTL MUX2	1 179.15	1 161.67	17.48 (1.50 %)	134.91 (9.30 %)

Source: The author.

AOI22 and OAI22 gates have slightly larger absolute errors. PTL cells exhibit below-average errors, indicating accurate modeling of PTL characteristics. It is also possible to observe that the individual MAEs are usually larger than the entire circuit's MAE. From this, we can infer that gate estimations balance out each other when aggregated for the whole circuit, thus reducing the final error.

As exemplified by the estimations in Figure 5.3, this balancing out also happens over clock cycle intervals, where one period's underestimation may cancel out another period's overestimation and so on. In general, Figure 5.3 indicates a fairly accurate energy estimation on a per-period basis, with very few outliers. Nevertheless, the model still displays a small underestimation bias, as indicated by the slight deviation of the linear fit from the ideal line.

Figure 5.3 – Estimated versus SPICE energy consumption for each clock period of the sign-parity module simulation.



Source: Author's own elaboration.

6 CONCLUSIONS AND FUTURE WORK

This work investigated a perceived gap in power estimation methodologies for PTL circuits. We reviewed several works that propose estimation methodologies and discussed how they solve common problems. We observed that the estimation of dynamic power at the gate level can be done by looking up and interpolating cell characterization values under similar driving and loading conditions. Furthermore, we also assessed ways of estimating leakage power using pre-characterized values and correcting the power of glitches through different heuristics, such as the normalized output voltage swing.

Building on existing literature, we devised a power estimation methodology of our own, specifically targeting circuits with PTL cells. One of its cornerstone pieces is an event-driven simulator with timing and capacitance capabilities that provide the means of accurately estimating the power of PTL circuits. The information provided by the simulator is used to access dynamic and static energy values in LUTs generated in a custom cell characterization process, taking PTL's unique electrical characteristics into account. As a way of adjusting to hazards, the interpolated dynamic energy values are corrected based on their voltage swing.

We developed scripts to apply the methodology to basic characterization circuits as well as the sign-parity module of a state-of-the-art LDPC decoder. The results from the characterization circuits validated specific components of the implementation and the methodology, confirming that their errors were within an acceptable margin. Meanwhile, the total estimation error of $-4.34\,\%$ for the sign-parity module ensured that our initial objective was achieved, with an error within $10\,\%$ of SPICE. In conclusion, this work explored a promising direction to be pursued for enabling the power estimation of PTL circuits.

As future work, the next logical step would be to integrate the power model into an event-driven simulator with the described capabilities. Although using a mock-up of such a simulator allowed us to assess the power model's error independently of errors related to the simulator, exploring how these components interact in a realistic setting would be a valuable contribution to the topic. This integration would also enable the comparison between an implementation of the methodology and SPICE with respect to runtime and memory complexity.

Additionally, it may also be worthwhile to validate the methodology by applying it to reference combinational circuits, such as the ISCAS'85 benchmark suite. These results would serve as a basis for further comparative analysis between other power estimation tools.

REFERENCES

BOLIOLO, A. et al. Gate-level power and current simulation of CMOS integrated circuits. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 5, n. 4, p. 473–488, 1997.

BURCH, R. et al. A Monte Carlo approach for power estimation. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 1, n. 1, p. 63–71, 1993.

CHINAZZO, A. L. et al. Investigation of pass transistor logic in a 12nm FinFET CMOS technology. In: **2022 29th IEEE International Conference on Electronics, Circuits and Systems (ICECS)**. [S.l.: s.n.], 2022. p. 1–4.

CONCEIÇÃO, C. M. d. O.; REIS, R. A. d. L. Transistor count reduction by gate merging. **IEEE Transactions on Circuits and Systems I: Regular Papers**, v. 66, n. 6, p. 2175–2187, 2019.

DARABIHA, A.; CARUSONE, A. C.; KSCHISCHANG, F. R. Power reduction techniques for LDPC decoders. **IEEE Journal of Solid-State Circuits**, v. 43, n. 8, p. 1835–1845, 2008.

FAVALLI, M.; BENINI, L. Analysis of glitch power dissipation in CMOS ICs. In: **Proceedings of the 1995 International Symposium on Low Power Design**. New York, NY, USA: Association for Computing Machinery, 1995. (ISLPED '95), p. 123–128. ISBN 0897917448.

GEORGE, B. et al. Power analysis for semi-custom design. In: **Proceedings of IEEE Custom Integrated Circuits Conference - CICC '94**. [S.l.: s.n.], 1994. p. 249–252.

IEEE Standard for Information Technology–Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks–Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Enhancements for High-Efficiency WLAN. **IEEE Std 802.11ax-2021 (Amendment to IEEE Std 802.11-2020)**, p. 1–767, 2021.

KRODEL, T. Power play-fast dynamic power estimation based on logic simulation. In: **[1991 Proceedings] IEEE International Conference on Computer Design: VLSI in Computers and Processors**. [S.l.: s.n.], 1991. p. 96–100.

KRUSE, L.; RABE, D.; NEBEL, W. VHDL power simulator: Power analysis at gate-level. In: ____. **Hardware Description Languages and their Applications: Specification, modelling, verification and synthesis of microelectronic systems IFIP TC10 WG10.5 International Conference on Computer Hardware Description Languages and their Applications, 20–25 April 1997, Toledo, Spain**. Boston, MA: Springer US, 1997. p. 317–333. ISBN 978-0-387-35064-6.

KURODA, T.; SAKURAI, T. Overview of low-power ULSI circuit techniques. **IEICE Transactions on Electronics**, v. 78, p. 334–344, 1999.

LAPPAS, J. et al. Revisiting pass-transistor logic styles in a 12nm FinFET technology node. In: **2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)**. [S.l.: s.n.], 2022. p. 1083–1084.

LAPPAS, J. et al. Timing analysis beyond complementary CMOS logic styles. In: **2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)**. [S.l.: s.n.], 2024. p. 189–194.

MEIXNER, M.; NOLL, T. G. Accurate estimation of CMOS power consumption considering glitches by using waveform lookup. **IEEE Transactions on Circuits and Systems II: Express Briefs**, v. 64, n. 7, p. 787–791, 2017.

MORGENSHTEIN, A.; FISH, A.; WAGNER, I. Gate-diffusion input (GDI): a power-efficient method for digital combinatorial circuits. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 10, n. 5, p. 566–581, 2002.

NASSER, Y. et al. RTL to transistor level power modeling and estimation techniques for FPGA and ASIC: A survey. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, v. 40, n. 3, p. 479–493, 2021.

PATTERSON, D.; HENNESSY, J. **Computer Organization and Design: The Hardware/Software Interface**. 5. ed. [S.l.]: Morgan Kaufmann, 2014. ISBN 9780124077263.

RABAEY, J.; CHANDRAKASAN, A.; NIKOLIĆ, B. **Digital Integrated Circuits: A Design Perspective**. [S.l.]: Pearson Education, 2003. (Prentice Hall electronics and VLSI series). ISBN 9780131207646.

RABE, D. et al. Power-simulation of cell based ASICs: accuracy- and performance trade-offs. In: **Proceedings Design, Automation and Test in Europe**. [S.l.: s.n.], 1998. p. 356–361.

REWIEŃSKI, M. A perspective on Fast-SPICE simulation technology. In: ____. **Simulation and Verification of Electronic and Biological Systems**. Dordrecht: Springer Netherlands, 2011. p. 23–42. ISBN 978-94-007-0149-6.

SHAFIQUE, M. et al. The EDA challenges in the dark silicon era: Temperature, reliability, and variability perspectives. In: **Proceedings of the 51st Annual Design Automation Conference**. New York, NY, USA: Association for Computing Machinery, 2014. (DAC '14), p. 1–6. ISBN 9781450327305.

SHELAR, R.; SAPATNEKAR, S. BDD decomposition for delay oriented pass transistor logic synthesis. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 13, n. 8, p. 957–970, 2005.

SYNOPSYS. **PrimePower RTL to Signoff Power Analysis**. 2020. <https://www.synopsys.com/content/dam/synopsys/implementation&signoff/datasheets/primepower-ds.pdf>. Accessed: 18/06/2024.

WESTE, N.; HARRIS, D. **CMOS VLSI Design: A Circuits and Systems Perspective**. [S.l.]: Addison Wesley, 2011. ISBN 9780321547743.

YANO, K. et al. Top-down pass-transistor logic design. **IEEE Journal of Solid-State Circuits**, v. 31, n. 6, p. 792–803, 1996.

ZIMMERMANN, R.; FICHTNER, W. Low-power logic styles: CMOS versus pass-transistor logic. **IEEE Journal of Solid-State Circuits**, v. 32, n. 7, p. 1079–1090, 1997.