Diss. ETH No. 14395

Data Statistics and Low-Power Digital VLSI

A dissertation submitted to the
SWISS FEDERAL INSTITUTE OF TECHNOLOGY
ZURICH

for the degree of
Doctor of Technical Sciences

presented by

JÜRGEN WASSNER
Dipl.-Ing. Elektrotechnik
born April 9th, 1967
citizen of Germany

accepted on the recommendation of

Prof. Dr. W. Fichtner, examiner
Prof. Dr. L. Thiele, co-examiner

2001

Acknowledgments

First of all, I wish to thank my supervisor, Prof. Wolfgang Fichtner. His faith in me and my work has been an essential source of motivation, and the excellent professional environment he provides at the Integrated Systems Laboratory (IIS) is invaluable. I would also like to thank Prof. Lothar Thiele for reading and co-examining the thesis.

I am greatly indebted in Dr. Hubert Kaeslin and Dr. Norbert Felber for their continuous support and encouragement during the work, myriad consultations, and their overall kindness and generosity. Without them, this thesis would not have been possible. I also want to express my gratitude to all colleagues at the IIS who contributed to the perfect working environment.

Part of this work has been carried out in collaboration with Bernafon Inc., Switzerland. In this regard I am thankful to Dr. Paul Zbinden and Arthur Schaub for their kind support. Furthermore, I acknowledge the financial sponsorship of KTI, the Commission for Technology and Innovation of the Swiss Government.

Last but not least, very special thanks go to all of my family. In particular, I gratefully think of my parents for all they have done for me. And, I would like to express my heartfelt gratitude to my wife, Dana, for her loving support and generously making many sacrifices during the past four years I was working on this project. Special thanks also go to our two daughters, Clara and Olivia, for keeping alive in me the child's perspective.

Contents

Acknowledgments vii

Abstract xv

Zusammenfassung xvii

1 Introduction 1

1.1 Motivation 1
1.2 Overview 2
1.2.1 Goal of the Thesis 2
1.2.2 Outline of the Thesis 2
1.2.3 Contribution of the Thesis 3

2 Sources of Power Dissipation 5

2.1 Dynamic Power 5
2.1.1 Capacitor Charging 6
2.1.2 Crossover Current 8
2.2 Static Power 8
2.2.1 Leakage Current 9
2.2.2 Resistive Loads 9

2.3 Implications 9

3 Probabilistic Circuit Analysis for Power Estimation 11

3.1 Introduction 11
3.1.1 Motivation 11
3.1.2 Previous Work 12
3.1.3 Outline 12
3.2 Gate-level Power Estimation 13
3.2.1 Circuit and Power Model 13
3.2.2 Power Measures 14
3.2.3 Switching Activity Estimation 15
3.3 Probabilistic Measures for Binary Signals 19
3.3.1 One-dimensional Binary Signals 19
3.3.2 n-dimensional Binary Signals 23
3.3.3 Complexity Considerations 25
3.4 Probabilistic Analysis of Combinational Circuits . . . 27
3.4.1 Spectral Transform of Boolean Functions . . . 27
3.4.2 Exact Calculation of Switching Activities . . . 31
3.4.3 Approximation of Signal Correlation 35
3.4.4 Experimental Results and Discussion 38
3.4.5 Extension to Multi-Delay Model 45
3.5 Probabilistic Analysis of Sequential Circuits 50
3.5.1 Feedback Circuits 50
3.5.2 Analysis via Spectral Transformation 52
3.5.3 Markovian Analysis 53
3.5.4 Implications 55
3.6 Summary 56

4 Data Statistics and Minimum Power Dissipation 59

- 4.1 Introduction 59
  - 4.1.1 Motivation 59
  - 4.1.2 Previous Work 60
  - 4.1.3 Outline 61
- 4.2 The Limits of VLSI Technology 62
  - 4.2.1 The Ideal MOSFET 62
  - 4.2.2 Technology Trends 63
  - 4.2.3 VLSI and Thermodynamics 64
  - 4.2.4 Implications 66
- 4.3 Taxonomy for Lower Bound Problem 67
  - 4.3.1 Problem Formulation 67
  - 4.3.2 Representation of Digital Operations 69
  - 4.3.3 Representation of Data Statistics 73
  - 4.3.4 Problem Hierarchy 74
- 4.4 The General Lower Bound Problem 77
  - 4.4.1 Tightness of the Lower Bound 77
  - 4.4.2 Energy Efficiency Rating 78
  - 4.4.3 Axiomatic Requirements 79
  - 4.4.4 Implications 81
- 4.5 Lower Bound for Data Transmission 81
  - 4.5.1 Power Dissipation Bound 82
  - 4.5.2 Switching Activity Bounds 86
  - 4.5.3 Variable-Length Coding 89
  - 4.5.4 Fixed-Length Coding 90
  - 4.5.5 Implications 94
- 4.6 Lower Bound in Boolean Optimization 96

4.6.1 Problem Description 96
4.6.2 Related Efforts 97
4.6.3 Implications 98
4.7 Lower Bound on Supply Voltage 99
4.7.1 Information-theoretic Results 99
4.7.2 Application to VLSI Processing 101
4.7.3 Limitations 103
4.8 Information Theory vs. Low Power VLSI 106
4.8.1 Digest of Information-theoretic Results 106
4.8.2 Problem Comparison 107
4.8.3 Implications 108
4.9 Summary 110

5 Energy-Efficient Processing of Speech **113**
5.1 Introduction 113
5.1.1 Motivation 113
5.1.2 Previous Work 114
5.1.3 Outline 115
5.2 Speech Features and Coding 115
5.2.1 Statistical Properties of Speech Data 115
5.2.2 Speech Coding Techniques 117
5.3 Activity Analysis for Speech Data 118
5.3.1 Fixed-length Lossless Encodings 118
5.3.2 Signal Parameters 119
5.3.3 Switching Activity vs. Signal Parameters . . . 120
5.4 FIR Filtering of Speech Data 125
5.4.1 Target Application 125

5.4.2 Reference Architecture with 2sC Data 125
5.4.3 Sign-Magnitude Representation 129
5.4.4 Differential Encoding 132
5.4.5 Adaptive Encoding 135
5.4.6 Logarithmic Encoding 137
5.5 Evaluation of Coding Schemes 141
5.5.1 Application Parameters 141
5.5.2 Power Estimation Model 141
5.5.3 Experimental Results and Discussion 143
5.5.4 Derivation of General Coding Guidelines 147
5.6 Summary 150

6 Concluding Remarks **151**
6.1 Prospects for Practical Application 151
6.2 Future Work 153

A Basics of Probability Theory **155**

B Basics of Information Theory **159**

C FSM Analysis Example **163**

D Memory Structures for FIR Filters **167**

Bibliography **173**

Abstract

Ongoing advances of semiconductor technology enable the integration of more and more complex circuits, thus making feasible devices of ever increasing functionality for ever new applications. Naturally, power dissipation has evolved into a key constraint for the design of VLSI circuits, particularly for battery-powered applications.

This thesis deals specifically with power dissipation of digital VLSI circuits subject to the statistical properties of the data being processed. The dependence of power consumption on data statistics is explored with regard to three major aspects: power estimation, power minimization, and practical low-power design.

1. In the context of gate-level power estimation, a probabilistic approach to determine the switching activity in logic circuits from the given input statistics is presented. This approach can take into account any kind of signal correlation and calculates the correct switching activity for every node in the circuit. A novel approximation technique for the efficient control of the estimation accuracy is proposed.

2. The question of fundamental lower bounds on power dissipation in consideration of data statistics is pursued. A systematic view of minimum power dissipation is developed, which leads to a classification of the lower bound problem. The resulting subproblems are set in context to known information-theoretic bounds on data coding and transmission.

3. By means of speech filtering it is exemplified how designers of application-specific circuits may avail themselves of data statis-

tics to cut down the energy use. Various number representations and data encodings are examined, and potential power savings are quantified subject to statistical signal properties and operating conditions. This permits the derivation of general coding guidelines for application-specific processing of speech data.

Zusammenfassung

Die anhaltende Weiterentwicklung der Halbleitertechnologie gestattet die Integration von immer komplexeren Schaltkreisen. Dies ermöglicht die Realisierung hochfunktioneller Geräte und eröffnet immer neue Einsatzfelder. Es überrascht daher nicht, dass die Leistungsaufnahme von VLSI Schaltkreisen mittlerweile eine Schlüsselposition im Entwurfsprozess einnimmt, insbesondere bei batteriebetriebenen Anwendungen.

In der vorliegenden Arbeit wird speziell die Leistungsaufnahme von digitalen VLSI Schaltkreisen im Zusammenhang mit den statistischen Eigenschaften der zu verarbeitenden Daten behandelt. Die Abhängigkeit der Leistungsaufnahme von der Datenstatistik wird im Hinblick auf drei wesentliche Aspekte untersucht: Leistungsabschätzung, Leistungsminimierung und praktischer Entwurf stromsparender Schaltungen.

1. Für die Leistungsabschätzung auf Gatterniveau wird ein wahrscheinlichkeitstheoretischer Zugang zur Bestimmung der Schaltaktivität in logischen Netzwerken vorgestellt. Dieser Zugang ermöglicht die Berücksichtigung jeglicher Art von Signalkorrelation, so dass eine genaue Berechnung der Knotenaktivitäten möglich ist. Ein neuartiges Näherungsverfahren erlaubt die elegante Steuerung der Schätzgenauigkeit.

2. Es wird der Frage nach einer grundsätzlichen unteren Schranke für die Leistungsaufnahme nachgegangen. Eine systematische Darstellung der Fragestellung führt zu einer Klassifizierung des Problems. Die daraus resultierenden Teilprobleme werden im

Zusammenhang mit bekannten informationstheoretischen Aussagen über Codier- und Übertragungsgrenzen diskutiert.

3. Anhand der Filterung von Sprachsignalen wird beispielhaft demonstriert, wie beim Entwurf anwendungsspezifischer Schaltungen die Statistik der Daten zur Senkung des Stromverbrauchs ausgenutzt werden kann. Das Sparpotential verschiedener Zahlendarstellungen und Codierungen wird in Abhängigkeit von statistischen Signalparametern und Betriebsbedingungen gemessen. Das erlaubt, Codierrichtlinien für die anwendungsspezifische Verarbeitung von Sprachsignalen aufzustellen.

Chapter 1

Introduction

1.1 Motivation

There is one formula that governs low-power digital VLSI design. It appears in virtually any text related to the subject matter and, presumably, is known to any VLSI designer who faces power dissipation issues. This formula goes

$$\begin{aligned}\textsc{Power} &= \textsc{Voltage}^2 \times \textsc{Capacitance} \\ &\quad \times \textsc{Clock frequency} \times \textsc{Switching activity}\,.\end{aligned}$$

The present thesis will be about this enchanting formula. More specifically, our investigations will revolve mainly around the rearmost term in this equation - switching activity. Generally speaking, switching activity designates the time behavior, or liveliness, of logic signals. This time behavior, in one form or the other, must reflect the characteristics of the application data at hand. The resulting dependence of power dissipation on data statistics is the leitmotif for this thesis, as it combines challenges at the conceptual level with questions of practical significance.

With ongoing advances of semiconductor technology and the consequential boost in the complexity of integrated circuits (IC), power dissipation in general has been trending higher on the list of VLSI design constraints. Today it is at or near the top of this list, notably for ICs in portable devices where battery lifetime is a major concern. A considerable portion of these devices processes data with statistical properties that are a priori known. In this case, data statistics may be seen as an additional parameter for the design of application-specific ICs (ASIC) and instruction set processors (ASIP). Fields of application include wireless communication, portable audio and video devices, and digital hearing instruments.

1.2 Overview

1.2.1 Goal of the Thesis

In this thesis we focus on three major aspects of low-power digital VLSI design in relation to the statistical properties of the application data. The corresponding goals can be formulated as follows:

1. *Power estimation:* Develop an analytical model suitable for explicit calculation of switching activity in logic circuits with arbitrary input statistics. Find means such that this model can deal with the state-explosion problem associated with bit-level analysis.

2. *Minimum power dissipation:* Explore the question of fundamental lower bounds on power dissipation by appending data statistics to the list of low-power VLSI design parameters. Illuminate the relation between the power dissipation limit and known information-theoretic bounds on data coding and transmission.

3. *Practical low-power design:* Show in the occurrence of speech processing that data statistics can be employed for the design of application-specific circuits with reduced power consumption.

1.2.2 Outline of the Thesis

The thesis is organized as follows: In chapter 2 the basic mechanisms of power dissipation in CMOS circuits are reviewed. The three subsequent chapters then are devoted to the three major aspects described above: Chapter 3 deals with probabilistic methods for switching activity estimation in logic circuits. Chapter 4 pursues the question of minimum power dissipation in consideration of data statistics. Chapter 5 shows by means of speech filtering that data statistics can serve as vehicle for reduction of switching activity and energy use. Each of these three chapters is largely self-contained, but they all share one common theme - power dissipation of VLSI circuits in the context of data statistics. Finally, the thesis closes in chapter 6 with some general remarks and suggestions for future work.

1.2.3 Contribution of the Thesis

The main contributions of the thesis to the three aspects of low-power digital VLSI mentioned above can be summarized as follows.

Probabilistic circuit analysis: In chapter 3, a theoretical basis for exact calculation of switching activity in logic circuits is developed by bridging the concept of moments of a random variable known from probability theory and the concept of spectral transformation known from the theory of switching functions. Based on this theoretical foundation, a novel heuristic for the approximation of signal correlation in combinational circuits is suggested. This heuristic improves the state of the art in that it allows to control the degree of approximation for any kind of signal correlation by means of a single parameter [WKFF99].

Lower bound on power dissipation: In chapter 4, a systematic view of minimum power dissipation in consideration of data statistics is developed, which culminates in a classification of the problem of lower bounds on power dissipation. Solutions are provided for two of the resulting subproblems, i.e. “minimum supply voltage” and “minimum switching activity”. An existing information-theoretic approach to minimum power dissipation [Sha97] is put into perspective of the new taxonomy. It is shown that the most general form of the lower bound problem is different in nature from the transmission bound problem of information theory.

Energy-efficient processing of speech: In chapter 5, waveform coding techniques known from low bit-rate communication are examined for energy-efficient filtering of speech signals. Arithmetic datapath units corresponding to the various coding schemes are presented, including a new circuit structure for the approximation of binary logarithm. Potential power savings are quantified subject to statistical signal properties and operating conditions. This permits the derivation of general coding guidelines for application-specific processing of speech data [WKFF00, WKFF01].

Chapter 2

Sources of Power Dissipation

This chapter reviews the basic mechanisms of power dissipation in digital VLSI circuits in order to reveal its relation to data statistics. A static CMOS logic style is presumed as this is the predominant choice for the implementation of VLSI circuits.

There are two fundamentally different sources of power dissipation in VLSI systems: *dynamic* and *static* power. Dynamic power is associated with the variation of signal voltages over time. This variation represents the progression of the information processing task at hand. Therefore, dynamic power dissipation is an inherent property of information representation and processing in VLSI circuits. Static power, on the other hand, gets dissipated even if all signals remain stable. In fully complementary-MOS circuits, static power is neither essential to information representation nor information processing.

2.1 Dynamic Power

Dynamic power dissipation has two sources: Current flowing for the charging and discharging of parasitic capacitors, and current resulting from conductive paths crossing over PMOS and NMOS devices. In figure 2.1 these currents are indicated for a CMOS inverter circuit.

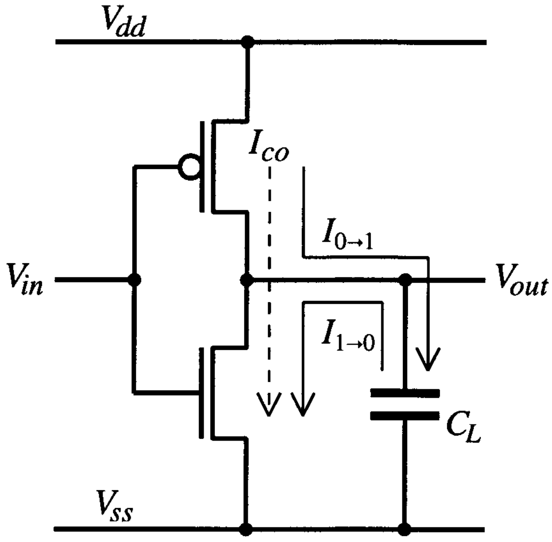


Figure 2.1: CMOS inverter circuit with crossover current I_{co} and capacitor charge ($I_{0\to1}$) and discharge ($I_{1\to0}$) current.

2.1.1 Capacitor Charging

Capacitor components

For the inverter circuit in figure 2.1 the voltage across the capacitor C_L represents the logical value of the output signal. Assuming that the inverter drives another identical stage, C_L represents the following physical capacitance components [CB95, Kae00]:

- The output-to-ground capacitance of the driving inverter circuit, made up of the junction capacitance between the drain regions and the local substrate.

- The capacitance between the drain regions and the gate electrode of the driving inverter, which must get mapped to C_L with almost its fourfold value due to the *Miller effect* [ALES98].

- The input-to-ground capacitance of the driven circuit dominated by the thinoxide below the gate electrodes, and also containing the source-to-gate capacitances there.

- The total wiring capacitance of the interconnect between output of driving and input of driven inverter. This capacitance is made up of the wire-to-ground capacitance, which itself has a parallel-plate and fringing-field component, and the wire-to-wire capacitance, which is significant for modern technologies with their reduced line spacing.

Furthermore, if the driven circuit has dependent inner nodes due to series-connected NMOS or PMOS transistors, as is the case for any multi-input logic gate, extra capacitance appears due to the drain/source regions at these inner nodes (junction and overlap capacitance, see above). Formally, the inner-node capacitance is attributed to the load capacitance C_L of the driving cell, but under a simplified circuit model sometimes also is ascribed to the driven cell.

Charging/Discharging power dissipation

Assuming rail-to-rail logic, i.e. signal voltages that swing back and forth between V_{dd} and V_{ss}, the energy injected from the power supply in order to fully charge the load capacitor C_L during a '0' to '1' transition of V_{out} is

$$E_{supply} = V_{dd}^2 C_L \ . \tag{2.1}$$

Half of this energy is dissipated in the p-channel device for the other half to be delivered to the capacitor C_L. During the subsequent '1' to '0' transition, the other half of the injected energy is wasted by discharging C_L to ground via the n-channel device, see figure 2.1. Thus, the invested energy E_{supply} is used for one charge-discharge cycle of C_L, corresponding to one switching cycle of the signal voltage V_{out}.

The average power dissipated during one charge-discharge cycle is obtained from dividing the invested energy E_{supply} in (2.1) by the time required to complete one such cycle. Let T_{ci} denote the time span of a *computation interval* and f_{ci} be its inverse. For instance, for a single-edge-triggered clocking strategy $f_{clk} = f_{ci}$, and in case of dual-edge-triggered clocking $f_{clk} = f_{ci}/2$. Then, the average power dissipation during one charge-discharge cycle may be written as

$$P_{sw} = V_{dd}^2 C_L \frac{f_{ci}}{2} \alpha \tag{2.2}$$

with α denoting the expected number of times V_{out} switches from one logical state to the other within one computation interval[1].

The expected number of switching events of V_{out} is a statistical quantity, which will be formally defined and investigated in chapter 3. For an isolated logic gate, α depends on the statistical properties of the gate's input signals and on the logic function of that very gate. In an overall circuit, α is a function of the data being processed, circuit structure, and the logic function of the gate at hand.

2.1.2 Crossover Current

The charge/discharge power dissipation in (2.2) is independent of the actual voltage waveform V_{out}. However, finite rise and fall times of input voltage V_{in} will result in a direct conducting path between V_{dd} and V_{ss} for short periods of time during switching, see figure 2.1. Specifically, a crossover current I_{co} will flow as long as $V_{th,n} < V_{in} < V_{dd} - |V_{th,p}|$, where $V_{th,n}$ and $V_{th,p}$ is the threshold voltage of the NMOS and PMOS device, respectively. Besides input rise/fall times, the crossover current is an intricate function of various other parameters, e.g. capacitive load C_L, device geometries and threshold voltages.

Just as the switching power P_{sw}, the crossover component of power also varies approximately proportional with the number of signal transitions α, simply because the conductive path between V_{dd} and V_{ss} exists only during switching events. However, the power consumed due to crossover currents contributes much less to the total dynamic power, typically about 10...20% for well-designed circuits [Yea98].

2.2 Static Power

Static power dissipation basically may have two different sources: Leakage currents flowing due to the employed MOS transistor-based implementation technology, and currents flowing from V_{dd} to V_{ss} over functional resistive loads.

---

[1] From a functional point of view, there exists an upper bound of $\alpha \leq 1$ for all data signals. However, spurious signal transitions may inflate α beyond that limit, see section 3.4.5.

2.2.1 Leakage Current

There are two main components of leakage currents:

- sub-threshold leakage due to carrier diffusion between source and drain regions when the transistor is turned off, and
- diode leakage at reverse-biased p/n junctions (drain/well and well/well).

The magnitude of these currents is set predominantly by process parameters, and the contribution to total power usually is about 1% or less, although with advanced technologies and reduced supply and threshold voltages leakage power becomes more significant.

Strictly speaking, leakage power is also subject to data statistics, since sub-threshold leakage and diode leakage at drain/well junctions depend on the logical state of the cell.

2.2.2 Resistive Loads

In truly complementary-MOS circuits there exists no DC path between V_{dd} and V_{ss} if all signal values remain stable. However, sinks of static power may hide in units that do not adhere to a fully complementary-MOS design style. Such units include passive pull-up/downs, clock generation units, macro-cell memories, LVDS receivers/transmitters, and pseudo-NMOS/PMOS subcircuits [Kae00].

Power dissipation due to resistive loads in general depends on the data statistics, because the voltage level of at least one terminal of the resistor will be controlled by some signal amplitude, thus determining the average proportion of time during which current is conducted.

2.3 Implications

Virtually any current causing CMOS circuits to dissipate power is subject to the statistical properties of the data that stimulate the circuit. However, there is one single source that predominates overall dissipation in static CMOS circuits, i.e. the current flowing for charging parasitic capacitances when signals change their logical state. The dependence of the resulting power dissipation P_{sw} in (2.2) on the temporal switching behavior of data signals motivates the investigation of data statistics in the context of low-power VLSI.

Chapter 3

Probabilistic Circuit Analysis for Power Estimation

3.1 Introduction

3.1.1 Motivation

For the design of low-power circuits, power estimation is indispensable as it supplies feedback on the current value of the power cost function during the optimization process. Power estimation techniques and tools exist for every abstraction level in the design process:

- system/algorithm level
- architectural/register-transfer (RT-) level
- logic/gate level
- transistor/physical level .

Among the levels of abstraction, the influence of data statistics on the produced power estimate is most evident at the logic/gate level. This is because the power estimation model (2.2) that applies at this level is sufficiently specific to incorporate a data-related statistical measure, i.e. switching activity α, but yet is simple enough to be applied even to the largest circuit within reasonable time. Although power models at lower levels of abstraction do, in principle, reflect data dependency, they are infeasible even for medium-sized circuits and have lost relevance in the light of standard-cell based design and high-level synthesis. On the other hand, power models at higher levels of abstraction do not, or only partially, reflect the data dependency involved in the power consuming charge/discharge processes, since physical design information is not available at this level.

3.1.2 Previous Work

Due to its significance for low-power design, a huge number of researchers have worked in the field of power estimation. Overviews of this work are provided for instance in [S$^+$95, MPS98]. Previous work within the scope of this chapter will be cited and classified in section 3.2.3.

3.1.3 Outline

This chapter is devoted to power estimation at the gate level, since the influence of data statistics is most evident on this level of abstraction. More specifically, the primary focus will be on probabilistic techniques for switching activity estimation. Despite the large number of publications on this topic, a comprehensive treatment that includes all types of signal correlation and thus allows an analytical calculation of switching activity can hardly be found.

Section 3.2 introduces basic terms and concepts in gate-level power estimation. In section 3.3 statistical measures for binary signals are defined, which subsequently will be employed for probabilistic analysis of logic circuits. Switching activity estimation in combinational circuits is the subject of section 3.4, where a new technique for the approximation of signal correlation is introduced and experimentally verified. Section 3.5 then deals with probabilistic analysis of sequential circuits.

3.2 Gate-level Power Estimation

3.2.1 Circuit and Power Model

A *combinational logic circuit* is modeled as cycle-free directed graph with two types of nodes: primary inputs (PI) and logic gates, see figure 3.1. PIs have none, logic gates at least one incoming edge. All nodes exactly have one outgoing edge, which then may have arbitrary fan-out[1]. The edges of the graph represent electrical connections between the nodes. The orientation of the edges is given by the inputs and outputs of the logic gates. Nodes without any successor node are called primary outputs (PO).

If there are n PIs and m POs, the circuit implements a mapping $\{0,1\}^n \rightarrow \{0,1\}^m$ by means of the $g \geq m$ logic gates. These gates are memory-free and either implement basic Boolean functions such as (N)AND, (N)OR, X(N)OR and NOT, or are themselves subcircuits composed of those basic functions. For now, it is assumed that the gates have zero propagation delay. A non-zero delay model will then be considered in section 3.4.5.

Each node of the circuit is associated with a binary signal. The PIs are associated with signals $x_1, \ldots, x_n$. The logic gates are associated with binary signals $y_1, \ldots, y_{m-1}, y_m \ldots, y_g$. These signals (logic gates) are topologically sorted such that signal y_j $(j = 1, \ldots, g)$ does not depend on any signal y_l with $l > j$.

For power estimation, the power dissipation model (2.2) is applied to every logic gate in the circuit. There are two parameters in this model that need to be identified:

Load capacitance. The load capacitance C_{L_j} associated with gate j $(j = 1, \ldots, g)$ is estimated from technology or physical layout information. It is assumed that all relevant capacitance components, such as the output-to-ground capacitance of gate j, the total capacitance for wiring gate j to its successor gates as well as the input-to-ground and inner-node capacitance of these successor gates, are lumped into a single capacitor at the output of gate j, see section 2.1.1.

---

[1] Multiple-driven nets are not considered here.

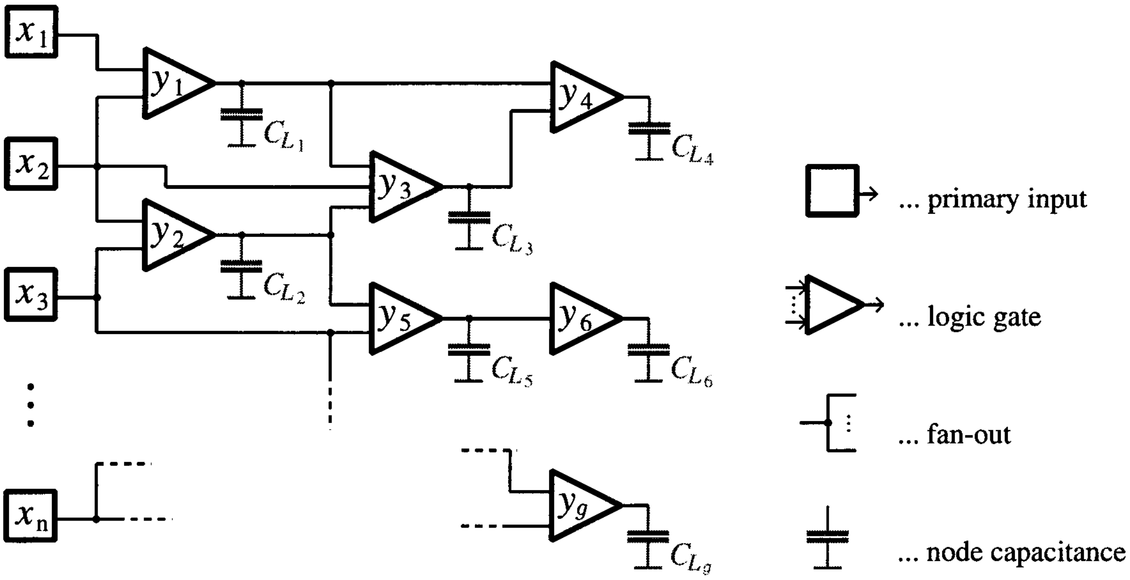


Figure 3.1: Model of logic circuit for gate-level power estimation.

For a given circuit, the load capacitances are independent of data statistics and hence we will not further discuss how to obtain them.

Switching activity. The activity α_{y_j}, $(j = 1, \ldots, g)$ refers to the switching behavior of the output of gate j, i.e. binary signal y_j. The precise implication of the node activities and their derivation depends on the objective of power estimation, as will be discussed next.

3.2.2 Power Measures

The following power measures may be discerned:

Average power. This is the most common objective of gate-level power estimation, where switching activity α_y refers to the average number of times logic gate y changes its binary state within one computation interval. This number strongly depends on the statistical properties of the input data applied to the circuit. Summation over all gates yields the average power dissipation of the circuit:

$$P_{avg} = V_{dd}^2 f_{ci} \sum_{j=1}^{g} C_{L_j} \frac{\alpha_{y_j}}{2} . \tag{3.1}$$

Average power dissipation is of central importance for battery life-time considerations in mobile applications. Furthermore, this measure is of interest in reliability analysis for the overall system, where high power dissipation poses challenges on packaging and heat removal technology.

Instantaneous power. Given the present binary state of each gate, the instantaneous or cycle power of some circuit refers to the dissipation induced by one particular input pattern. Switching activity in this case is not a meaningful concept.

Maximum power. A measure of practical interest results when instantaneous power is maximized over all combinations of present circuit state and possible input pattern. This peak power is used to stay clear of switching noise and electromigration. It is also of interest for electro-magnetic compatibility considerations. Since in practice not all gates change their state at the same time during the computation interval, signal delays must be considered in order to obtain a tight upper bound on peak power.

Since only average power depends on data statistics, all subsequent discussion will be devoted to this measure. The provision of average switching activities α_{y_j} for evaluation of (3.1) will be discussed next.

3.2.3 Switching Activity Estimation

Signal correlation

Average circuit power can be seen as a weighted sum of gate activities. Therefore, a random error afflicted with the estimated α_{y_j} can be tolerated, since such an error tends to average out due to summation over all nodes. However, there are also situations where activity must be estimated accurately for every node, e.g. for gate-level power optimization that performs activity-driven re-structuring of the circuit.

Thereby, the goal is to hide high-activity nodes within complex gates where they drive smaller load capacitances [IP97].

For accurate estimation of gate activities, signal correlation is crucial [SK96]. In combinational logic circuits, two basic types of correlation matter:

Spatio-temporal correlation. This type of correlation refers to statistical dependencies which are present in the application data. *Spatial correlation* refers to dependencies between different PI signals at the same time instance. *Temporal correlation* means dependencies between different time instances of the same or different PI signals. Spatio-temporal correlation of input signals causes correlation of internal signals and thus impacts their switching activity.

Structural correlation. Even if the input data is completely uncorrelated, correlation may be introduced by the circuit structure itself. Structural correlation takes effect through node fanouts which reconverge at subsequent gates in the circuit network. For instance, the inputs to gate y_3 in figure 3.1 are structurally correlated, because they all depend on PI x_2. This shall be taken into account for accurate calculation of α_{y_3}.

Methods to obtain information on the gate switching activities α_{y_j} can be divided into two categories: simulation-based (dynamic) and probabilistic (static) approaches, see figure 3.2. The two approaches fundamentally differ in how they handle signal correlation. There also exist hybrid methods, which use aspects from both categories in order to combine their respective advantages while avoiding their drawbacks [C+98].

Simulation-based activity estimation

In simulation-based approaches activity information is obtained from monitoring the gate toggling during explicit logic simulation of the circuit. In order to obtain representative activity information, the binary stimuli for this simulation shall resemble the statistical properties of the functional input data as close as possible. After a sufficiently large number of stimuli vectors is simulated, the average switching activity at every node will converge towards a narrow interval around the

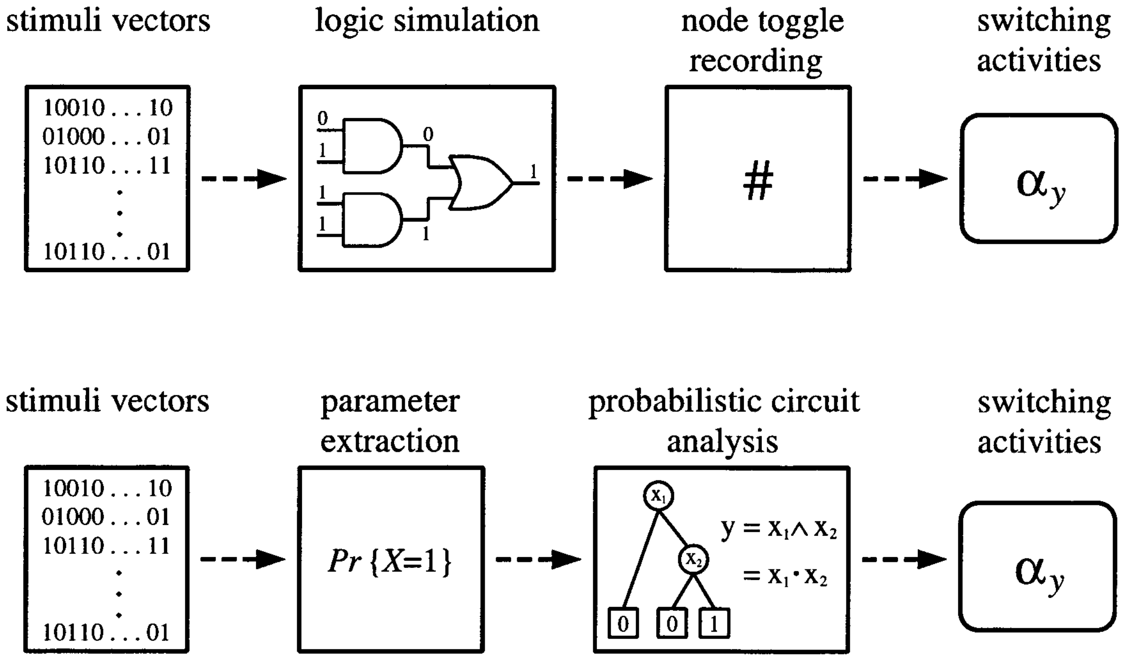


Figure 3.2: Simulation-based (top) and probabilistic (bottom) activity estimation.

true value. The number of vectors necessary for convergence strongly depends on the circuit itself as well as on the desired accuracy. A priori information about the number of vectors required, is in general not available. This issue may be addressed by statistical estimation techniques that run the simulation until a stopping criterion for the overall circuit power [BNYT93, YTK98] or individual node activities [XN94] is met.

A great advantage of simulation-based methods is that they implicitly take into account all kind of signal correlations. From a practical point of view another plus is that logic simulation is a standard task in digital VLSI and a simulator is available virtually anywhere.

Probabilistic activity estimation

As opposed to simulation, probabilistic techniques use statistical parameters to describe the properties of the circuit's input data. These parameters are employed to derive the switching activity of the gates. The advantage of such an approach is that no lengthy binary vectors are required and convergence of activity values is not an issue.

Due to their incremental nature, probabilistic methods are better suited to activity-driven power optimization, where the network structure is being modified as part of the process. Simulation-based methods in this case would require storage of long traces for every gate or complete re-simulation of the circuit.

The largest challenge for probabilistic techniques is to efficiently handle signal correlation effects. While for small circuits the activity of every gate may be calculated accurately, approximation of signal correlation is indispensable for large circuits. Several methods for probabilistic activity estimation have been proposed in the literature, which differ in how and to what extent they consider signal correlations. These methods can be divided in two classes:

Global approach. Methods of this class describe the logic function of any node globally, i.e. with respect to the PIs. To do so, either binary decision diagrams (BDD) [SSW96, MDG$^+$97, DTP98] or symbolic polynomials [CMD97] are employed. In both cases, the effect of reconvergent fanouts is accounted for, but spatial correlation between input signals was ignored. For large circuits, concepts, such as *reconvergence regions* [SSW96] and *super-gates* [CMD97], are introduced in order to cope with complexity.

Incremental approach. These methods describe the logic function of each gate with respect to its immediate input signals. Thus, structural correlation is not captured. In [MMP95, MMP98] correlation coefficients are used to describe signal dependencies. The complexity of this approach has been reduced in [TTSG97]. Methods employing correlation coefficients have the advantage that dependencies among input signals can be modeled by means of these coefficients. In order to handle large circuits efficiently, only pairwise signal correlation is considered. Relevant signal pairs, i.e. signals that converge anywhere in the circuit, must be identified prior to propagation of statistical parameters.

None of the above techniques offers efficient control over the approximation of all types of signal correlation. Section 3.4 describes such a method that governs the *complexity-accuracy tradeoff* by means of a single user-defined parameter. This approach shares features of either of the two classes above: It is global because the circuit is described by polynomials of PI variables. However, these expressions are being constructed incrementally, with approximations being attained by employing immanent information of the Boolean function at hand.

3.3 Probabilistic Measures for Binary Signals

3.3.1 One-dimensional Binary Signals

Signal model

In the framework of probabilistic circuit analysis, a binary discrete-time signal $x(k)$ is modeled as random sequence $\{X_k\}$ with sample space $\Omega = \{0,1\}$ and $k = 0,1,2,\ldots$ being the time index, see appendix A. In general it is assumed that $\{X_k\}$ is stationary in the strict sense[2]. Therefore, time index k is suppressed when appropriate. Furthermore, the properties of the random sequence $\{X_k\}$ are referred to as the properties of signal $x(k)$.

Signal probabilities and temporal correlation

The *static probability* p_x^1 of a binary signal x is defined as the probability of finding x in logic state '1'. Since x is stationary, this probability equals the expected value, i.e. the first moment[3] of the associated binary random variable X:

$$E[X] \;=\; \sum_{x\in\{0,1\}} x\cdot p(x) \;=\; Pr\{x=1\} \;\overset{!}{=}\; p_x^1\,. \qquad (3.2)$$

Thus, in case of a binary random variable the expected value itself represents a probability. The same holds true for the product of any number of binary random variables. This observation is key to the activity estimation algorithm to be presented in section 3.4.

---

[2] For combinational circuits it is sufficient to assume wide-sense stationarity because only first-order statistics are relevant in this case.

[3] From (A.3) follows that for a binary random variable the i-th moment (i=2,3,...) is identical to the first moment.

In particular, $E[X^2] = E[X] = p_x^1$, and thus from (A.5) follows the variance or *signal power* of x as

$$\sigma_x^2 = E[X^2] - (E[X])^2 = p_x^1(1 - p_x^1) \ . \tag{3.3}$$

As expected, for $p_x^1 = 0$ or $p_x^1 = 1$, i.e. for x being a constant, the signal power is zero.

Of specific interest is the expected value of the product of two successive samples in a binary signal x:

$$E[X_k \cdot X_{k+1}] = Pr\{x(k) = 1,\ x(k+1) = 1\} \overset{!}{=} p_{x\text{-}x}^{1\text{-}1} \ . \tag{3.4}$$

Accordingly, all *first-order joint probabilities* will be denoted as[4]

$$p_{x\text{-}x}^{i\text{-}j} = Pr\{x(k) = i,\ x(k+1) = j\} \ , \qquad i,j \in \{0,1\} \tag{3.5}$$

with

$$p_{x\text{-}x}^{0\text{-}0} + p_{x\text{-}x}^{0\text{-}1} + p_{x\text{-}x}^{1\text{-}0} + p_{x\text{-}x}^{1\text{-}1} = 1 \ . \tag{3.6}$$

In general, signal values at time k and $k+1$ will not be independent. The correlation between successive samples in signal x can be measured by the *first-order correlation coefficient* ϱ_x, which follows from (A.8) and the stationarity of x as

$$\begin{aligned} \varrho_x &= \frac{E[X_k \cdot X_{k+1}] - E[X_k] \cdot E[X_{k+1}]}{\sqrt{E\left[(X_k - E[X_k])^2\right] \cdot E\left[(X_{k+1} - E[X_{k+1}])^2\right]}} \\ &= \frac{p_{x\text{-}x}^{1\text{-}1} - (p_x^1)^2}{\sigma_x^2} = \frac{p_{x\text{-}x}^{1\text{-}1} - (p_x^1)^2}{p_x^1(1 - p_x^1)} \ . \end{aligned} \tag{3.7}$$

In general $-1 \leq \varrho_x \leq 1$. If $p_{x\text{-}x}^{1\text{-}1} = (p_x^1)^2$ then $\varrho_x = 0$ and signal x is said to be *temporally uncorrelated.* If $\varrho_x > 0$ ($\varrho_x < 0$) signal x is temporally correlated (anti-correlated), indicating a higher probability of finding two successive samples of x in the same (different) logical state than another signal y with the same static probability $p_y^1 = p_x^1$. Note that ϱ_x is undefined for $\sigma_x^2 = 0$, i.e. for constant signals. However, employing L'Hospital's rule yields $\varrho_x|_{p_x^1=0} = \varrho_x|_{p_x^1=1} = 1$, i.e. maximum correlation for constant signals, which is intuitive.

---

[4] The token "-" is used to indicate the reference to consecutive time points, in contrast to joint probabilities of several random sequences sampled at the same point in time.

$$\xrightarrow{k}$$

$\varrho_x = 0.5$: ...¦0 0 0 0 1 1 1 1¦0 0 0 0 1 1 1 1¦...

$\varrho_y = 0$: ...¦0 1 1 0 0 1 1 0¦0 1 1 0 0 1 1 0¦...

$\varrho_z = -0.5$: ...¦0 1 1 0 1 0 1 0¦0 1 1 0 1 0 1 0¦...

Figure 3.3: Correlated ($\varrho_x = \frac{1}{2}$), uncorrelated ($\varrho_y = 0$), and anti-correlated ($\varrho_z = -\frac{1}{2}$) periodic signal with $p_x^1 = p_y^1 = p_z^1 = \frac{1}{2}$ and $\sigma_x^2 = \sigma_y^2 = \sigma_z^2 = \frac{1}{4}$.

Example:
Figure 3.3 shows three periodic signals[5] x, y, and z with $p_x^1 = p_y^1 = p_z^1 = \frac{1}{2}$ but different temporal correlation. Although the three signals have equal signal power $\sigma_x^2 = \sigma_y^2 = \sigma_z^2 = \frac{1}{4}$, the activity in terms of logical state changes is quite different. □

Switching activity

Formally, the *switching activity* of a binary signal x is defined as

$$\alpha_x = p_{x\text{-}x}^{0\text{-}1} + p_{x\text{-}x}^{1\text{-}0} . \tag{3.8}$$

Thus α_x gives the average number of times, a binary signal changes its logical state in a certain number of computational intervals. α_x is also referred to as activity factor, transition activity or transition probability. The latter name, however, is to be used with precaution, since for non-zero delay models a binary signal may change its logical state more than once per computational interval, resulting in α_x larger than one, see section 3.4.5.

The following theorem provides the basis for the activity estimation algorithm to be presented in section 3.4, by stating that the switching activity of any (stationary) binary signal x can be computed from just two statistical measures of x.

---

[5] Periodic signals are chosen solely for convenience, such that to be able to represent average statistical properties with a finite number of samples.

Theorem 3.1
The switching activity α_x *of a binary signal* x *is given by its static probability* p_x^1 *and the joint probability* $p_{x\text{-}x}^{1\text{-}1}$ *as*

$$\alpha_x \;=\; 2(p_x^1 - p_{x\text{-}x}^{1\text{-}1}) \tag{3.9}$$

Proof: Since signal x is stationary, the probability of finding x at logic state '1' is given by $p_x^1 = p_{x\text{-}x}^{1\text{-}0} + p_{x\text{-}x}^{1\text{-}1} = p_{x\text{-}x}^{0\text{-}1} + p_{x\text{-}x}^{1\text{-}1}$. Hence, the probability of x changing its state from '1' to '0' is the same as the probability of changing from '0' to '1', i.e. $p_{x\text{-}x}^{1\text{-}0} = p_{x\text{-}x}^{0\text{-}1} = p_x^1 - p_{x\text{-}x}^{1\text{-}1}$. Substituting this in (3.8) gives the proposition. ■

Applying (3.9) to the signals in figure 3.3 yields $\alpha_x = \frac{1}{4}$, $\alpha_y = \frac{1}{2}$, and $\alpha_z = \frac{3}{4}$. Thus, for a given static probability p_x^1 the switching activity decreases with increasing correlation coefficient ϱ_x. This relation can explicitly be expressed by substituting (3.3) into (3.7) and combining this with (3.9):

$$\alpha_x \;=\; 2\, p_x^1(1 - p_x^1)(1 - \varrho_x)\;. \tag{3.10}$$

Corollary 3.1
The switching activity α_x *of a temporally uncorrelated signal* x *is twice the signal power:*

$$\alpha_x \;=\; 2\sigma_x^2\;. \tag{3.11}$$

Proof: With $\varrho_x = 0$ this follows from (3.10) and (3.3). ■

Hence, to compute the switching activity α_x of a temporally uncorrelated signal, the knowledge of static probability p_x^1 is sufficient. On the other hand, applying (3.11) to a temporally correlated signal, introduces an error for α_x. The following theorem provides a bound on this approximation error.

Theorem 3.2
The switching activity α_x *of a binary signal* x *is bounded by*

$$\alpha_x \;\leq\; 1 - 2\cdot|p_x^1 - 0.5|\;. \tag{3.12}$$

Proof: There can be at most twice as many signal transitions as there are '1's (or '0's) in x (achieved by the alternating sequence

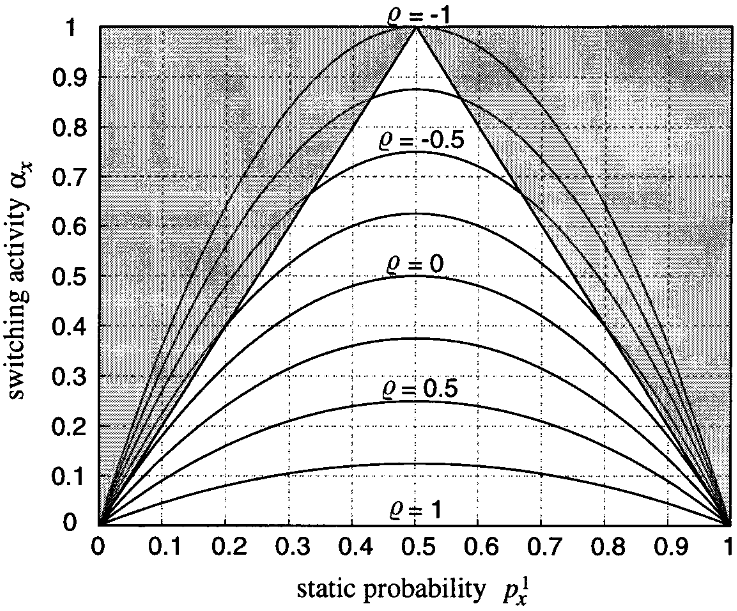


Figure 3.4: Switching activity α_x vs. static probability p_x^1 for a binary signal x. Any valid pair (p_x^1, α_x) falls into the unshaded region.

"01010…"). Thus, $\alpha_x \leq 2p_x^1$ and $\alpha_x \leq 2(1 - p_x^1)$. Together with $\alpha_x \leq 1$ this can be written as (3.12). ■

Figure 3.4 elucidates the relation between switching activity α_x and static probability p_x^1 for a stationary binary signal. Any valid pair (p_x^1, α_x) falls into the unshaded region in figure 3.4. The maximum switching activity of $\alpha_x = 1$ can only be achieved if $p_x^1 = \frac{1}{2}$. For temporally uncorrelated signals ($\varrho_x = 0$) holds $\alpha_x \leq \frac{1}{2}$, with equality if and only if $p_x^1 = \frac{1}{2}$. From figure 3.4 one may also conclude that overestimation of switching activity is more likely than underestimation when erroneously applying (3.11) to a temporally correlated signal. Except for $p_x^1 = \frac{1}{2}$, the maximum absolute error in α_x is always smaller for underestimation than for overestimation.

3.3.2 n-dimensional Binary Signals

The notations introduced previously naturally extend to n-dimensional binary signals $\boldsymbol{x}(k) = (x_1(k), \ldots, x_n(k))$ with $x_i(k)$ $(i = 1, \ldots, n)$ being one-dimensional binary signals as introduced in section 3.3.1. A sample of $\boldsymbol{x}$ at time k is called a *binary word.* Since all $x_i(k)$ are stationary, also the binary words are stationary.

Definition (3.8) may be generalized to the *switching activity* of a n-dimensional binary signal as the sum of the switching activities of its constituting components:

$$\alpha_{\boldsymbol{x}} \;=\; \alpha_{x_1} + \alpha_{x_2} + \ldots + \alpha_{x_n} \; . \tag{3.13}$$

Clearly, it is not reasonable to speak of $\alpha_{\boldsymbol{x}}$ as transition "probability" in this case.

Moments of binary signals

Of peculiar importance for the activity estimation algorithm to be presented in section 3.4 are the moments of $\boldsymbol{x}$. Based on (A.6) and in analogy to (3.2) the i-th *moment of a binary word* is defined as

$$p^{1\ldots 1}_{x_{j_1}\ldots x_{j_i}} \;=\; Pr\{x_{j_1}=1,\ldots,x_{j_i}=1\} \;=\; E[X_{j_1}\cdot\ldots\cdot X_{j_i}] \tag{3.14}$$

with $j_1,\ldots,j_i$ being a combination of i disjoint elements from the set $\{1,\ldots,n\}$. For instance let $\boldsymbol{x}=(x_1,x_2,x_3)$, then the second moment $p^{11}_{x_1x_3}$ gives the probability of finding the first and third component of $\boldsymbol{x}$ simultaneously at logic '1'. The first moments of $\boldsymbol{x}$ are just the static probabilities (3.2) of its component signals.

The i-th moments $p^{1\ldots 1}_{x_{j_1}\ldots x_{j_i}}$ give evidence of spatial correlation between the components of a binary word, just the same way as $p^{1\text{-}1}_{x\text{-}x}$ expresses temporal correlation within a binary signal, see (3.7). In order to handle first-order spatio-temporal correlation within a n-dimensional binary signal $\boldsymbol{x}(k)$, the notation of *lag-one* i-th *moment* is introduced. These moments are joint probabilities of finding i samples of $\boldsymbol{x}$, each of which either drawn at time k or $k+1$, simultaneously at logic '1'. Formally, this can be written as

$$\begin{aligned} p^{1\ldots 1\text{-}1\ldots 1}_{x_{j_1^1}\ldots x_{j_m^1}\text{-}x_{j_1^2}\ldots x_{j_r^2}} \;=\; Pr\{\; & x_{j_1^1}(k)=1,\ldots,x_{j_m^1}(k)=1, \\ & x_{j_1^2}(k{+}1)=1,\ldots,x_{j_r^2}(k{+}1)=1\;\} \end{aligned} \tag{3.15}$$

with $2 \le i = m + r \le 2n$.

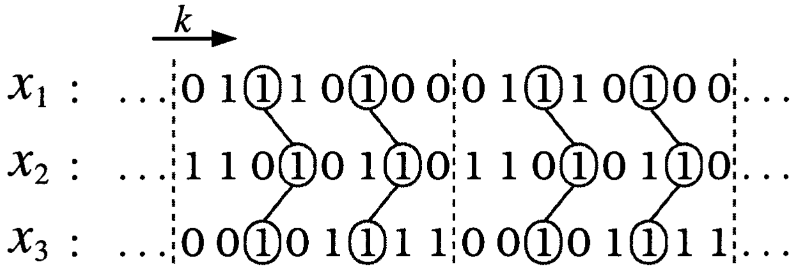

k
x1 : ... 0 1 1 1 0 1 0 0 0 1 1 1 0 1 0 0 ...
x2 : ... 1 1 0 1 0 1 1 0 1 1 0 1 0 1 1 0 ...
x3 : ... 0 0 1 0 1 1 1 1 0 0 1 0 1 1 1 1 ...

Figure 3.5: Example of 3-dimensional periodic binary signal.

Example:
For $n = 3$ the lag-one third moment $p^{11\text{-}1}_{x_1x_3\text{-}x_2}$ gives the probability of simultaneously finding the first and third component of $\boldsymbol{x}$ at time k, and the second component at the following time step, at logic '1'. In case of the periodic 3-dimensional signal shown in figure 3.5 this yields $p^{11\text{-}1}_{x_1x_3\text{-}x_2} = \frac{2}{8}$. ☐

3.3.3 Complexity Considerations

Zero-order statistics

The *zero-order statistics* of a n-dimensional binary signal $\boldsymbol{x}$ is uniquely defined by its probability mass function (pmf), see appendix A. Since $\boldsymbol{x}$ is binary, the pmf $p(\boldsymbol{x}(k))$ comprises 2^n probability values. Only $2^n - 1$ of these values are independent, because the sum over all values must yield one.

It can be shown, that $p(\boldsymbol{x}(k))$ can be constructed given all i-th moments ($i = 1, \ldots, n$) of $\boldsymbol{x}$ as defined in (3.14). To see this, note that there are $\binom{n}{i}$ i-th moments of a n-dimensional binary word, all of which being independent. Furthermore, it is easy to verify that

$$\sum_{i=1}^{n} \binom{n}{i} = 2^n - 1 \; . \tag{3.16}$$

Thus, knowing all i-th moments ($i = 1, \ldots, n$) provides exactly the number of independent statistical measures necessary to completely specify the zero-order statistics of $\boldsymbol{x}$.

Example:
If for $n = 2$ the three moments $p^1_{x_1}$, $p^1_{x_2}$, and $p^{11}_{x_1x_2}$ are given, the remaining joint probabilities follow from

$$\begin{aligned} p^{10}_{x_1x_2} &= p^1_{x_1} - p^{11}_{x_1x_2} \\ p^{01}_{x_1x_2} &= p^1_{x_2} - p^{11}_{x_1x_2} \\ p^{00}_{x_1x_2} &= 1 - p^{11}_{x_1x_2} - p^{10}_{x_1x_2} - p^{01}_{x_1x_2} \end{aligned}$$

which is the pmf $p(x_1, x_2)$. For any $n > 2$ the pmf can be constructed by recursively applying this procedure. □

First-order statistics

The *first-order statistics* of a n-dimensional binary signal $\boldsymbol{x}$ is uniquely defined by the pmf $p(\boldsymbol{x}(k), \boldsymbol{x}(k+1))$, which comprises 2^{2n} probability values. Given all i-th moments as in (3.14) and all lag-one i-th moments as in (3.15), $p(\boldsymbol{x}(k), \boldsymbol{x}(k+1))$ can be constructed in analogy to the example above.

To determine the total number of moments $M(n)$ that fully specify the first-order statistics of $\boldsymbol{x}$, the following is observed: First, according to (3.16), 2^n-1 i-th moments are required to describe $\boldsymbol{x}(k)$. Since the signal is stationary, the same moments also apply to $\boldsymbol{x}(k+1)$. Second, according to (3.16) there are 2^n-1 possibilities of selecting at least one out of n signals at time k. Hence, the total number of lag-one moments is $(2^n-1)^2$, sine at least one signal at time points k and $k+1$ must be selected, see (3.15). Thus, altogether

$$M(n) \;=\; 2^n-1 \;+\; (2^n-1)^2 \;=\; 2^n(2^n-1)\;. \tag{3.17}$$

$M(n)$ also is the minimum number of values required to fully specify the first-order statistics of a binary signal. Although $M(n) = 2^{2n} - 2^n < 2^{2n}$, the complexity of first-order statistical analysis remains exponential.

Example:
For $n = 2$ the pmf $p(\boldsymbol{x}(k), \boldsymbol{x}(k+1))$ is constituted of $2^{2\cdot 2} = 16$ joint probabilities. Given the $2^2 - 1 = 3$ moments $p^1_{x_1}$, $p^1_{x_2}$, and $p^{11}_{x_1x_2}$, as well as the $(2^2 - 1)^2 = 9$ lag-one moments $p^{1\text{-}1}_{x_1\text{-}x_1}$, $p^{1\text{-}1}_{x_1\text{-}x_2}$, $p^{1\text{-}1}_{x_2\text{-}x_1}$,

$p^{\text{1-1}}_{x_2\text{-}x_2}$, $p^{\text{1-11}}_{x_1\text{-}x_1x_2}$, $p^{\text{1-11}}_{x_2\text{-}x_1x_2}$, $p^{\text{11-1}}_{x_1x_2\text{-}x_1}$, $p^{\text{11-1}}_{x_1x_2\text{-}x_2}$, and $p^{\text{11-11}}_{x_1x_2\text{-}x_1x_2}$ the pmf $p(x_1(k), x_2(k), x_1(k+1), x_2(k+1))$ can be derived as follows:

$$\begin{aligned}
p(1111) &= p^{\text{11-11}}_{x_1x_2\text{-}x_1x_2} \\
p(1110) &= p^{\text{11-1}}_{x_1x_2\text{-}x_1} - p(1111) \\
p(1101) &= p^{\text{11-1}}_{x_1x_2\text{-}x_2} - p(1111) \\
&\vdots \\
p(1100) &= p^{\text{11}}_{x_1x_2} - p(1101) - p(1110) - p(1111) \\
&\vdots \\
p(0000) &= 1 - p(0001) - p(0010) - \ldots - p(1111)\ .
\end{aligned}$$

□

3.4 Probabilistic Analysis of Combinational Circuits

This section introduces an exact and an approximate method for switching activity calculation in combinational logic circuits based on spectral transformation of Boolean functions. It is assumed that the logical state switching for all signals in the circuit happens instantaneously, i.e. the logic gates have zero propagation delay. The extension to non-zero delay models is possible and will be explained in section 3.4.5.

3.4.1 Spectral Transform of Boolean Functions

Why use spectral transforms?

Although other approaches to probabilistic activity estimation exist, see section 3.2.3, the spectral transform approach offers three major advantages:

1. The arithmetic notation used in some transforms greatly facilitates the application of the expectation operator $E[.]$ compared

to an analysis based on ordinary logic equations. In particular, the arithmetic notation enables the application of the well-known relation $E[X+Y] = E[X]+E[Y]$ for any two random variables X and Y. This feature is at the heart of our approach.

2. Spectral transformation of Boolean functions results in a canonical representation. This representation comprises the minimum number of statistical measures (moments) that fully specify the relevant statistics of the input variables, see section 3.3.3.

3. As will be seen in section 3.4.3, the spectral transform provides a means for the efficient approximation of all types of signal correlation, where a single parameter controls the degree of approximation.

Basics of spectral transformation

In general, a *discrete function* f denotes a mapping $f : X \to Y$, where X and Y are finite non-empty sets. A *switching function* is a discrete function with $X = \{0,1\}^n$ and $Y = \{0,1\}$. Commonly, the term *Boolean function* is used as synonym for switching function. For some fixed variable ordering any Boolean function $y = f(x_1, \ldots, x_n)$ is specified by a binary $(2^n, 1)$-vector $\tilde{\boldsymbol{f}} = (f(0), \ldots, f(2^n - 1))$, which corresponds to the output column of the truth table associated with f. The *spectral transform* of a Boolean function $y = f(x_1, \ldots, x_n)$ with associated binary vector $\tilde{\boldsymbol{f}}$ is defined as

$$B(n) \cdot \tilde{\boldsymbol{f}} = \boldsymbol{a} \tag{3.18}$$

with the $(2^n, 2^n)$ transformation matrix $B(n)$ being independent of f. The $(2^n, 1)$-vector $\boldsymbol{a} = (a_{0\ldots00}, a_{0\ldots01}, \ldots, a_{1\ldots11})$ is called the *spectrum of f*. Given its spectrum, each Boolean function can be expressed as multivariate linear polynomial:

$$y = \sum_{i_n \ldots i_1} a_{i_n \ldots i_1} \cdot x_n^{i_n} \cdot \ldots \cdot x_1^{i_1} \tag{3.19}$$

where the summation is over all binary vectors $i_n \ldots i_1 \in \{0,1\}^n$. Depending on the definition of $B(n)$ and the interpretation of (3.19), the transforms have different names, such as Reed-Muller transform, Walsh transform, and arithmetic transform [SF96]. For the activity estimation algorithm to be presented, only the latter will be of interest.

Arithmetic transform

In case of arithmetic transform, the transformation matrix $B(n)$ is recursively defined as

$$B(n) = \begin{bmatrix} B(n-1) & 0 \\ -B(n-1) & B(n-1) \end{bmatrix} \tag{3.20}$$

with $B(0) = 1$. Thus, in case of arithmetic transform, $B(n)$ in (3.18) maps the binary vector $\tilde{\boldsymbol{f}}$ to an integer vector $\boldsymbol{a}$. Addition and multiplication in (3.19) then denote ordinary arithmetic operations. For convenience, the polynomial (3.19) obtained as arithmetic transform of $y = f(x_1, \ldots, x_n)$ will be denoted $\mathcal{A}(y)$.

Depending on the domain of f, the arithmetic transform may be further classified: If $x_i \in \mathbb{Z}$ $(i = 1, \ldots, n)$ then $y \in \mathbb{Z}$. If $x_i \in [0,1]$ $(i = 1, \ldots, n)$ then $y \in [0,1]$ and (3.19) is also referred to as *probabilistic transform.* The vector of coefficients $\boldsymbol{a} \in \mathbb{Z}^{2^n}$ is identical in both cases and, depending on the domain, is called *arithmetic spectrum* or *probability spectrum* of f. Naturally, probabilistic transform and probability spectrum are the relevant forms in the context of probabilistic analysis of logic circuits.

To show that the probability spectrum $\boldsymbol{a}$ of any Boolean function f is unique, it is observed that $B(n)$ $(n = 0, 1, 2, \ldots)$ is regular, and hence the inverse matrix $B^{-1}(n)$ exists. Then, the *inverse arithmetic transform*

$$B^{-1}(n) \cdot \boldsymbol{a} = \tilde{\boldsymbol{f}} \tag{3.21}$$

yields a set of 2^n linear equations for the elements of $\boldsymbol{a}$, which can be obtained by substituting the 2^n distinct input assignments of f and the corresponding output value into (3.19). Since with every input assignment a new component of $\boldsymbol{a}$ is selected, the 2^n equations are linearly independent. Consequently, there always exists a unique solution for $\boldsymbol{a} = (a_{0\ldots00}, a_{0\ldots01}, \ldots, a_{1\ldots11})$ in (3.21) and thus $\mathcal{A}(y)$ is a canonical representation of f.

Example:
For $n = 2$, (3.19) can be written as $y = a_{00} + a_{01}x_1 + a_{10}x_2 + a_{11}x_2x_1$. In case of the exclusive-OR function (XOR)

$$y = f(x_1, x_2) = (x_1 \wedge \bar{x}_2) \vee (\bar{x}_1 \wedge x_2) = x_1 \oplus x_2$$

the set of linear equations (3.21) becomes

$$\begin{aligned} a_{00} &= 0 \\ a_{00} + a_{01} &= 1 \\ a_{00} + a_{10} &= 1 \\ a_{00} + a_{01} + a_{10} + a_{11} &= 0 \end{aligned}$$

which has the unique solution $\boldsymbol{a} = (a_{00}, a_{01}, a_{10}, a_{11}) = (0, 1, 1, -2)$. Hence, for the two-input XOR holds $\mathcal{A}(y) = x_1 + x_2 - 2x_2x_1$. □

In general, $\mathcal{A}(y)$ contains 2^n terms. Depending on the specific mapping f, however, some of the components of $\boldsymbol{a}$ are zero. Table 3.1 shows the arithmetic transform for basic Boolean functions.

It should be noted that any arithmetic transform permits two different interpretations:

1. $\mathcal{A}(y)$ represents the logic equation of the Boolean function $y = f(x_1, \ldots, x_n)$ in an arithmetic notation, i.e. substitution of logic values '0' and '1' for the x_i $(i = 1, \ldots, n)$ in $\mathcal{A}(y)$ yields the logic value of y corresponding to f.

Name	$y = f(x_1, x_2)$	$\mathcal{A}(y)$
NOT	$y = \overline{x_1}$	$y = 1 - x_1$
AND	$y = x_1 \wedge x_2$	$y = x_1x_2$
NAND	$y = \overline{x_1 \wedge x_2}$	$y = 1 - x_1x_2$
OR	$y = x_1 \vee x_2$	$y = x_1 + x_2 - x_1x_2$
NOR	$y = \overline{x_1 \vee x_2}$	$y = 1 - x_1 - x_2 + x_1x_2$
XOR	$y = x_1 \oplus x_2$	$y = x_1 + x_2 - 2x_1x_2$
XNOR	$y = \overline{x_1 \oplus x_2}$	$y = 1 - x_1 - x_2 + 2x_1x_2$

Table 3.1: Arithmetic transforms for basic Boolean functions.

2. $\mathcal{A}(y)$ expresses the probability that the output of the corresponding Boolean function assumes the logic value '1' given the probabilities for the individual inputs being '1', i.e. substitution of $p^1_{x_i}$ $(i = 1, \ldots, n)$ in $\mathcal{A}(y)$ yields p^1_y.

3.4.2 Exact Calculation of Switching Activities

Given a topologically sorted logic circuit together with the first-order statistics of its primary inputs, the goal is to compute the switching activity α_{y_j} $(j = 1, \ldots, g)$, see figure 3.1. The following algorithm performs this computation in three major steps, where one of the three kinds of signal correlation (see section 3.2.3) matters in each step:

(I) Construct the arithmetic transforms $\mathcal{A}(y_j)$ $(j = 1, \ldots, g)$ in consideration of structural correlation;

(II) From $\mathcal{A}(y_j)$ compute the static signal probabilities $p^1_{y_j}$ $(j = 1, \ldots, g)$ in consideration of spatial correlation;

(III) From $\mathcal{A}(y_j)$ compute the joint probabilities $p^{1\text{-}1}_{y_j\text{-}y_j}$ in consideration of temporal correlation, and combine $p^1_{y_j}$ and $p^{1\text{-}1}_{y_j\text{-}y_j}$ to obtain the transition activities α_{y_j} $(j = 1, \ldots, g)$.

Example:
The subsequent description of steps (I) through (III) shall be elucidated by means of a small example circuit with periodic input signals as shown in figure 3.6. The goal is to find the switching activity of gate y_3 by means of the above algorithm. From analyzing the binary output signal the expected result is inferred as $\alpha_{y_3} = 1$. □

(I) Construction of $\mathcal{A}(y_j)$

Starting at the PI and proceeding in topological order to the PO, $\mathcal{A}(y_j)$ for each gate in the circuit is incrementally constructed using the basic arithmetic transforms in table 3.1. Compound logic gates

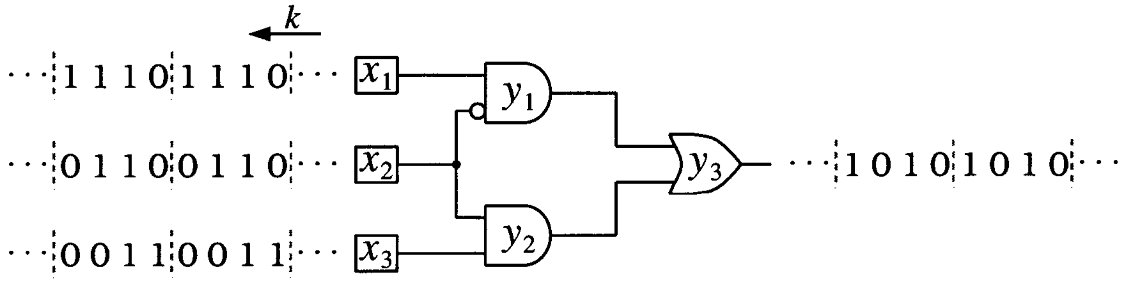


Figure 3.6: Example circuit with periodic input signals.

are decomposed. In the course of this procedure, structural correlation of signals arising from reconvergent fanouts are accounted for by applying the rule of *exponent suppression* [PM75]:

$$(x_i)^\beta \rightarrow x_i \quad \text{for} \quad \beta \geq 1 \tag{3.22}$$

In other words, signal reconvergence is taken into account by the *idempotence* of multiplication in the Boolean algebra.

At the end of step (I) each node of the circuit is described as linear polynomial of the PI variables. This circuit model is by no means specific to activity estimation but has also proven useful in the context of VLSI verification and test [SF96].

Example:
From table 3.1 the arithmetic transforms for gates y_1 and y_2 in the example of figure 3.6 follow as $\mathcal{A}(y_1) = x_1(1 - x_2) = x_1 - x_2x_1$ and $\mathcal{A}(y_2) = x_3x_2$. For construction of $\mathcal{A}(y_3)$ the exponent suppression rule must be applied, since x_2 reconverges in y_3. Thus, $\mathcal{A}(y_3) = \mathcal{A}(y_1) + \mathcal{A}(y_2) - \mathcal{A}(y_1)\mathcal{A}(y_2) = x_1 - x_2x_1 + x_3x_2$. □

(II) Computation of $p^1_{y_j}$

In this step the statistical properties of the input signals get involved. Given the arithmetic transform $\mathcal{A}(y_j)$ the associated static signal probability $p^1_{y_j}$ can be obtained as follows: From the definition of static signal probability (3.2) one has

$$p^1_{y_j} = E[Y_j] = E[\mathcal{A}(Y_j)] \,. \tag{3.23}$$

This enables the application of the well-known relation $E[X_1+X_2] = E[X_1]+E[X_2]$ for any two random variables X_1 and X_2 to decompose $\mathcal{A}(y_j)$ as follows:

$$\begin{aligned}
p^1_{y_j} &= E\left[\sum_{i_n\ldots i_1} a_{i_n\ldots i_1} x_n^{i_n}\ldots x_1^{i_1}\right] \\
&= a_{0\ldots 00} \\
&\quad +a_{0\ldots 01}E[X_1]+\ldots+a_{10\ldots 0}E[X_n] \\
&\quad +a_{0\ldots 011}E[X_2X_1]+\ldots+a_{110\ldots 0}E[X_nX_{n-1}]+\ldots \\
&\quad +a_{1\ldots 11}E[X_n\ldots X_1] \\
&= a_{0\ldots 00} \\
&\quad +a_{0\ldots 01}p^1_{x_1}+\ldots+a_{10\ldots 0}p^1_{x_n} \\
&\quad +a_{0\ldots 011}p^{11}_{x_2x_1}+\ldots+a_{110\ldots 0}p^{11}_{x_nx_{n-1}}+\ldots \\
&\quad +a_{1\ldots 11}p^{1\ldots 1}_{x_n\ldots x_1}\ . \qquad (3.24)
\end{aligned}$$

The probability values $p^1_{x_1}, p^1_{x_2}, \ldots, p^{1\ldots 1}_{x_n\ldots x_1}$ in (3.24) are the m-th moments ($m = 1,\ldots,n$) of the n-dimensional signal at the PI of the circuit. Substituting these moments into terms with non-vanishing spectral components yields the static probability $p^1_{y_j}$ in consideration of spatial correlation.

The decomposition process in (3.24) corresponds to finding a *disjoint cover* for $y_j = f_j(x_1,\ldots,x_n)$ in disjunctive form [MDG+97]. The arithmetic transform provides an elegant solution to this problem by employing the equivalence of expectation and the probability of being '1' of a binary random variable.

At this point, the switching activities α_{y_j} ($j = 1,\ldots,g$) could be computed from $p^1_{y_j}$ by virtue of (3.11) which, however, does not account for temporal correlation of signals.

Example:
Evaluation of $\mathcal{A}(y_3) = x_1 - x_2x_1 + x_3x_2$ with $p^1_{x_1} = \frac{3}{4}$, $p^{11}_{x_2x_1} = \frac{2}{4}$ and $p^{11}_{x_3x_2} = \frac{1}{4}$ yields $p^1_{y_3} = \frac{1}{2}$. The switching activity without consideration of temporal correlation follows as $\hat{\alpha}_{y_3} = 2p^1_{y_3}(1-p^1_{y_3}) = \frac{1}{2}$, which is only half the true value.

Now consider the disjunctive form $y_3 = x_1\bar{x}_2 \vee x_2x_3$. In order to apply the expectation operator as in (3.24), i.e. to obtain $p^1_{y_3}$, the terms of the right-hand side must be independent. Thus, the terms must form a disjoint cover for y_3, e.g. $y_3 = x_1\bar{x}_2x_3 \vee x_1\bar{x}_2x_3 \vee x_1x_2x_3 \vee \bar{x}_1x_2x_3$. This then yields the correct value $p^1_{y_3} = \frac{1}{2}$. However, this procedure would in general require all 2^n pmf values of the PI signals to be known. ☐

(III) Computation of $p^{1\text{-}1}_{y_j\text{-}y_j}$ and α_{y_j}

From theorem 3.1 follows that, in order to account for temporal correlation at node j, the joint probability $p^{1\text{-}1}_{y_j\text{-}y_j}$ must be known. Contrary to [CMD97], where polynomial expressions for all four fist-order joint probabilities (3.5) were incrementally constructed, it suffices to know $p^{1\text{-}1}_{y_j\text{-}y_j}$. Moreover, the polynomial expression associated with $p^{1\text{-}1}_{y_j\text{-}y_j}$ can be obtained directly from $\mathcal{A}(y_j)$ and does not have to be incrementally constructed from PI variables. This greatly reduces the computational burden.

Since $p^{1\text{-}1}_{y_j\text{-}y_j}$ is the probability of signal y_j being '1' at two consecutive time steps, it can be obtained from evaluating $\mathcal{A}(y_j(k) \wedge y_j(k+1))$. This arithmetic transfrom can be written as

$$\begin{aligned}\mathcal{A}(y_j^k y_j^{k+1}) &= \mathcal{A}(y_j^k) \cdot \mathcal{A}(y_j^{k+1}) \\ &= \sum_{i_n \ldots i_1} \sum_{j_n \ldots j_1} a_{i_n \ldots i_1} a_{j_n \ldots j_1} (x_n^k)^{i_n} \ldots (x_1^k)^{i_1} (x_n^{k+1})^{j_n} \ldots (x_1^{k+1})^{j_1} \\ & \qquad i_r, j_r \in \{0,1\}\ (r = 1, \ldots, n) \qquad (3.25)\end{aligned}$$

where for brevity time index k is denoted as superscript. Following the same procedure as in (3.24), one gets

$$\begin{aligned}p^{1\text{-}1}_{y_j\text{-}y_j} &= a_{0\ldots00}(a_{0\ldots00} + a_{0\ldots01}p^1_{x_1} + \ldots + a_{10\ldots0}p^1_{x_n}) + \ldots \\ &\quad + a_{0\ldots01}(a_{0\ldots00} + a_{0\ldots01}p^{1\text{-}1}_{x_1\text{-}x_1} + \ldots + a_{10\ldots0}p^{1\text{-}1}_{x_1\text{-}x_n}) + \ldots \\ &\quad + (a_{1\ldots11})^2 p^{1\ldots1\text{-}1\ldots1}_{x_n\ldots x_1\text{-}x_n\ldots x_1}\ . \qquad (3.26)\end{aligned}$$

Thus, the expression for $p^{1\text{-}1}_{y_j\text{-}y_j}$ is comprised of the probability spectrum corresponding to $y_j = f_j(x_1, \ldots, x_n)$, the m-th moments $(m = 1, \ldots, n)$, and the lag-one t-th moments $(t = 2, \ldots, 2n)$ of the

PI signal. Evaluation of each non-vanishing term with these moments yields the numerical value for $p^{1\text{-}1}_{y_j\text{-}y_j}$.

Finally, the switching activity in consideration of all three types of correlation can be obtained from $p^{1\text{-}1}_{y_j\text{-}y_j}$ and $p^{1}_{y_j}$ computed in step (II) by means of theorem 3.1.

Example:
The arithmetic transform of y_3 at two consecutive time steps follows from $\mathcal{A}(y_3) = x_1 - x_2x_1 + x_3x_2$ as

$$\begin{aligned}\mathcal{A}(y_3^k y_3^{k+1}) \quad = \quad & x_1^k x_1^{k+1} - x_1^k x_2^{k+1} x_1^{k+1} + x_1^k x_3^{k+1} x_2^{k+1} - x_2^k x_1^k x_1^{k+1} \\ & + x_2^k x_1^k x_2^{k+1} x_1^{k+1} - x_2^k x_1^k x_3^{k+1} x_2^{k+1} + x_3^k x_2^k x_1^{k+1} \\ & - x_3^k x_2^k x_2^{k+1} x_1^{k+1} + x_3^k x_2^k x_3^{k+1} x_2^{k+1} \ .\end{aligned}$$

Evaluation with the corresponding moments yields[6]

$$p^{1\text{-}1}_{y_3\text{-}y_3} = \frac{1}{4}(2 - 1 + 0 - 2 + 1 - 0 + 1 - 1 + 0) = 0 \ .$$

Together with $p^1_{y_3} = \frac{1}{2}$ from step (II), this gives the correct switching activity $\alpha_{y_3} = 2(p^1_{y_3} - p^{1\text{-}1}_{y_3\text{-}y_3}) = 1$. ☐

3.4.3 Approximation of Signal Correlation

Complexity considerations

The above procedure exactly models all relevant signal correlation. Its computational complexity with respect to the number of gates in the circuit is $O(g)$. However, the complexity with respect to the number of PI is $O(2^{2n})$. This is due to a maximum of 2^n non-vanishing terms in $\mathcal{A}(y_j)$ and the operation in (3.25). Also, as discussed in section 3.3.3, the number of moments for the PI signals to be stored is $M(n) = 2^n(2^n - 1)$. In order to handle large circuits, the following heuristic for approximation of signal correlation is proposed by the author [WKFF99]. This heuristic trades accuracy for computational cost.

---

[6] Note the direction of time index k in figure 3.6 when evaluating the moments.

Essential variables

The approximation is applied in step (I) of the exact algorithm. Assume that the inputs to gate j are named j' and j''. Then, prior to construction of $\mathcal{A}(y_j)$ the number of PI variables in $\mathcal{A}(y_{j''})$ and $\mathcal{A}(y_{j'})$ is limited to some value d. These *d essential variables* are kept symbolically, while all other variables are evaluated to their respective static probability $p^1_{x_i}$. Such partial evaluation yields real-valued probability spectra $\boldsymbol{a}$.

The evaluation of some variable x_i potentially introduces an error at subsequent nodes because

a) the exponent suppression rule is violated if paths from PI x_i reconverge,

b) spatial correlation of x_i can not be considered when evaluating $\mathcal{A}(y_j)$ or $\mathcal{A}(y^k_j y^{k+1}_j)$, and

b) temporal correlation of x_i can not be considered when evaluating $\mathcal{A}(y^k_j y^{k+1}_j)$.

Thus, the choice of d represents a tradeoff between computation/storage complexity and accuracy. With $d = 0$, the approximation is equivalent to propagating static probability values, because all PI variables x_i are evaluated to $p^1_{x_i}$ $(i = 1, \ldots, n)$ at the start of the algorithm. In this case any signal correlation is ignored. The other extremum, $d = \hat{n} \leq n$, produces exact switching activities, where $\hat{n}$ denotes the maximum number of PIs that any gate function depends on.

Selecting essential variables

The key question is how to select the d essential variables in the two polynomials $\mathcal{A}(y_{j''})$ and $\mathcal{A}(y_{j'})$ at the input to gate j, without giving up the incremental nature of the algorithm. First, variables common to $\mathcal{A}(y_{j''})$ and $\mathcal{A}(y_{j'})$ are chosen such that exponent suppression can take place correctly. However, reconvergence occurs only at a fraction of all nodes. In particular, variables that are actually due to exponent suppression at the current node, might have been evaluated already at earlier stages of the algorithm. Thus, a secondary selection criteria must be found. Experiments showed that random selection is not a good choice, because estimation accuracy can widely vary between runs and monotonic behavior with respect to d is not justified.

The generic structure of (3.19) however provides a means for guiding the selection in consideration of monotonic behavior as follows:
The contribution of term $i_n \dots i_1$ to $\mathcal{A}(y_j)$ depends on $a_{i_n \dots i_1}$ and the order of that term. By assuming $x_i = \frac{1}{2}$ $(i = 1, \dots, n)$, a *normalized probability spectrum* $\tilde{\boldsymbol{a}}$ may be defined:

$$\tilde{a}_{i_n \dots i_1} \;=\; |a_{i_n \dots i_1}| \cdot 2^{-\sum_{r=1}^{n} i_r} \tag{3.27}$$

$i_r \in \{0, 1\}$ $(r = 1, \dots, n)$. Thus, $\tilde{a}_{i_n \dots i_1}$ measures the relevance of term $i_n \dots i_1$ in $\mathcal{A}(y_j)$. Evaluation of variables that appear only in terms of low relevance will introduce a small error if the exponent suppression rule for those variables is violated at subsequent nodes. Similarly, neglecting correlation by evaluating variables with their static probabilities instead of the required moment is less critical for terms of low relevance.

Based on the above observations, the essential variables in the two polynomials $\mathcal{A}(y_{j''})$ and $\mathcal{A}(y_{j'})$ are chosen with the following priorities:

1. variables that appear in both polynomials;

2. variables from terms with largest $\tilde{a}_{i_n \dots i_1}$.

Note that the selection process still contains an element of choice if terms of equal relevance exist. This non-determinism asks for experimental verification of the average performance of the proposed heuristic.

Example:
Consider the arithmetic transform $\mathcal{A}(y_3) = x_1 - x_2 x_1 + x_3 x_2$ of gate y_3 in figure 3.6. The non-zero terms of the normalized probability spectrum follow from (3.27): $\tilde{a}_{001} = \frac{1}{2}$, $\tilde{a}_{011} = \frac{1}{4}$, $\tilde{a}_{110} = \frac{1}{4}$. If $d = 1$ essential variables are to be identified, the choice will be x_1 which

yields the approximated transform $\mathcal{A}(y_3) \approx x_1 - p^1_{x_2}x_1 + p^1_{x_3}p^1_{x_2} = \frac{1}{4} + \frac{1}{2}x_1$.

For $d = 2$, however, either x_2 or x_3 can be selected as additional essential variable, since $\tilde{a}_{011} = \tilde{a}_{110}$. The corresponding approximation would be either $\mathcal{A}(y_3) \approx x_1 + \frac{1}{2}x_2 - x_2x_1$ or $\mathcal{A}(y_3) \approx \frac{1}{2}x_1 + \frac{1}{2}x_3$, yielding different results for subsequent gates. □

3.4.4 Experimental Results and Discussion

The proposed activity calculation algorithm and correlation approximation scheme have been implemented using a standard symbolic computation package, in order to validate the following two requirements:

- estimation accuracy increases monotonically with increasing number of essential variables d, and
- significant accuracy improvements are achieved for feasible values of d.

The first requirement shall ensure consistency between runs with different parameters. The second requirement is necessary because the number of terms in $\mathcal{A}(y_j)$ is 2^{2d} in the worst case, if d essential variables are kept symbolically in $\mathcal{A}(y_{j''})$ and $\mathcal{A}(y_{j'})$.

Experimental results

For the experiments the complete ISCAS'85 benchmark set of combinational circuits and a 32-bit carry-lookahead adder were used. To express estimation accuracy for individual nodes by means of a single quantity, the following *RMS switching activity error* is being used:

$$\mathrm{RMS}_\alpha = \sqrt{\frac{1}{g}\sum_{j=1}^{g}(\hat{\alpha}_{y_j} - \alpha_{y_j})^2} \tag{3.28}$$

where $\hat{\alpha}_{y_j}$ is the estimated switching activity of node j, and α_{y_j} is the true value which was obtained from logic simulation of 10^5 input vectors.

Two sets of experiments with different input stimuli were conducted. The first set assumes random, uncorrelated PI signals with $p^1_{x_i} = \frac{1}{2}$ $(i = 1, \ldots, n)$. In this case the approximation involves only structural correlation. For the second set of experiments correlated input vectors were used. These vectors were generated by passing the output of binary counters (temporal correlation) through a logic network with high fanout of PI signals (spatial correlation).

Table 3.2 shows the RMS error for different values of d. It can be seen, that with two exceptions (c1908 correlated, cla32 random) the RMS error decreases monotonically for increasing d. On average, the RMS error decreases by around 50% if $d = 4$ essential variables are identified. Also, substantial accuracy improvements are achieved even for small values of d. The highest relative accuracy improvement is observed between $d = 0$ and $d = 1$.

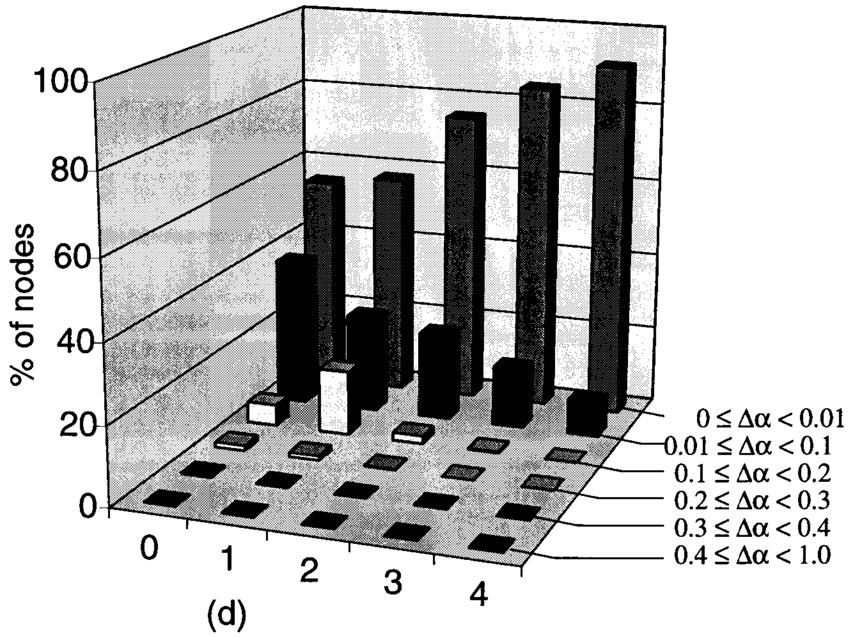
For three selected circuits, figures 3.7 - 3.9 show the percentage of nodes with absolute error of switching activity $\Delta\alpha_{y_j} = |\hat{\alpha}_{y_j} - \alpha_{y_j}|$ within certain bounds. Each of these plots combines five histograms over six error intervals. The trend of these histograms again indicates nearly monotonic behavior with respect to d. Even for circuit c6288 in figure 3.9, which has been qualified as notoriously difficult by other researchers due to a immense number of reconvergent fanouts [HYH99], the proposed approximation scheme does not break down. The monotonic behavior is retained for this circuit, although at a much higher level of error.

	number of			random inputs					correlated inputs				
circuit	gates	PI	PO	d=0	d=1	d=2	d=3	d=4	d=0	d=1	d=2	d=3	d=4
c432	160	36	7	75	44	42	42	40	211	157	104	101	98
c499	202	41	32	4	4	4	4	4	130	92	83	83	81
c880	383	60	26	20	9	6	4	4	179	151	67	64	46
c1355	546	41	32	56	40	29	29	12	160	100	74	71	45
c1908	880	33	25	19	6	5	3	3	136	76	64	44	*49*
c2670	1193	157	64	64	47	42	40	39	201	143	117	87	68
c3540	1669	50	22	56	44	43	40	34	176	163	134	97	91
c5315	2307	178	123	38	37	26	15	10	207	165	118	93	76
c6288	2416	32	32	130	117	109	103	97	213	188	170	154	147
c7552	3512	206	107	55	34	24	17	12	189	169	149	116	104
cla32	578	64	32	16	14	9	6	*8*	222	152	116	109	87
avg.	-	-	-	52	38	33	30	26	182	142	109	92	81

Table 3.2: Statistics of benchmark circuits (column 2 through 4), and RMS switching activity error $[10^{-3}]$ for random (column 5 through 9) and correlated (column 10 through 14) inputs and different numbers d of essential variables.

For comparison, if the absolute error in switching activity is uniformly distributed between +/−0.25 (+/−0.1, +/−0.05) then $\mathrm{RMS}_\alpha \approx 145 \cdot 10^{-3}$ ($60 \cdot 10^{-3}$, $30 \cdot 10^{-3}$).

random inputs



correlated inputs

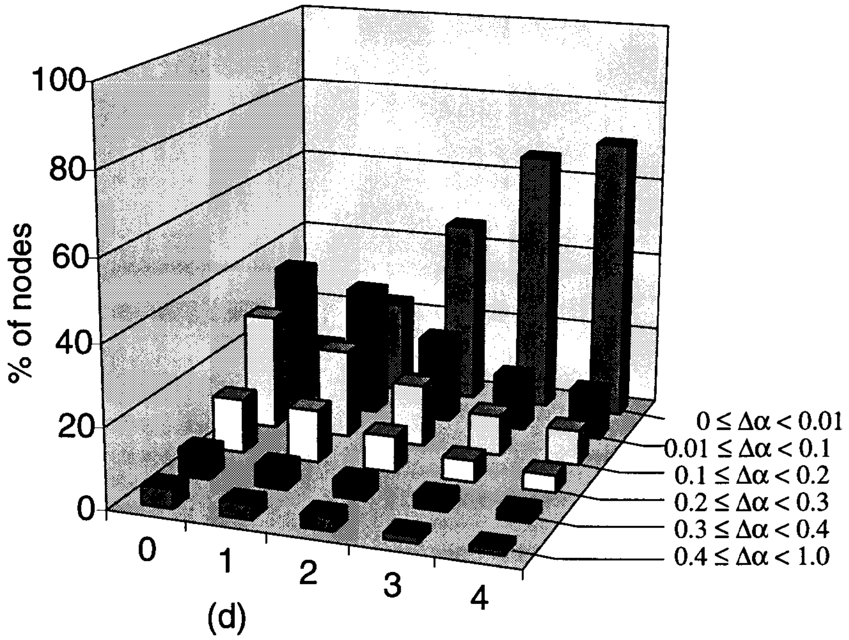
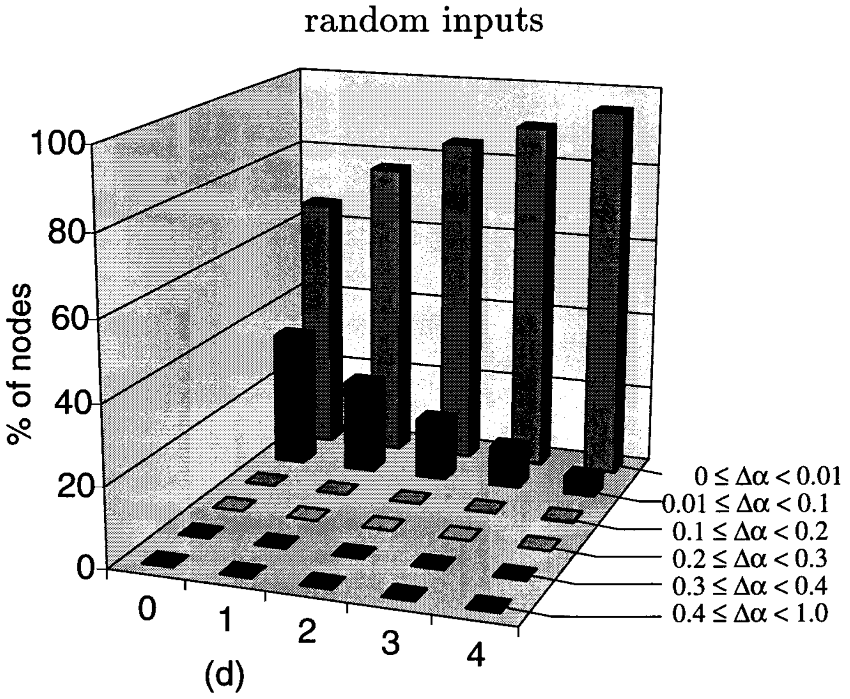


Figure 3.7: Percentage of nodes vs. switching activity error interval and number of essential variables (d) for circuit c7552 with random (top) and correlated (bottom) inputs.



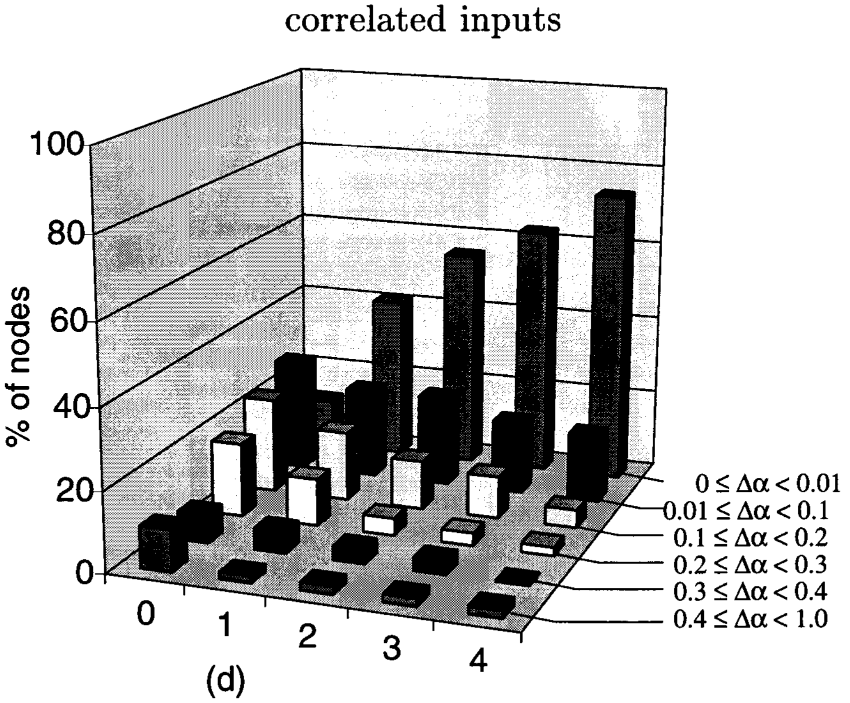
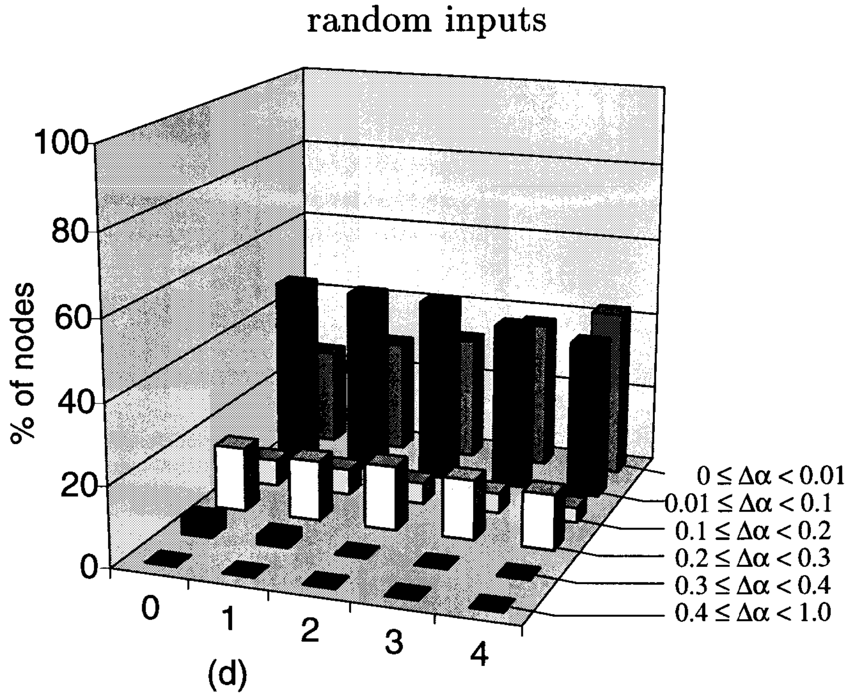


Figure 3.8: Percentage of nodes vs. switching activity error interval and number of essential variables (*d*) for 32-bit carry-lookahead adder with random (top) and correlated (bottom) inputs.



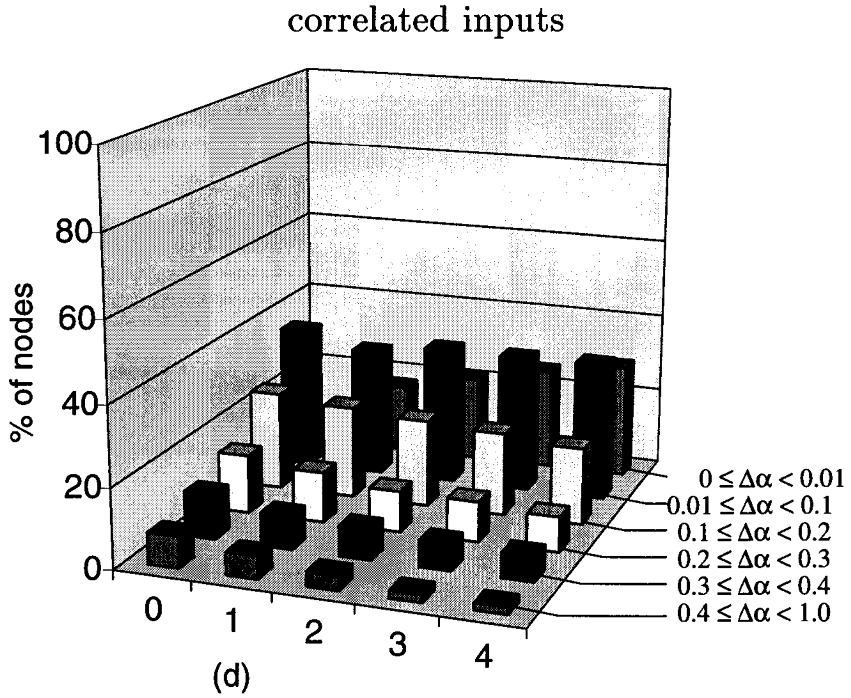


Figure 3.9: Percentage of nodes vs. switching activity error interval and number of essential variables (d) for high-reconvergent fanout circuit c6288 with random (top) and correlated (bottom) inputs.

Discussion

Compared to existing methods the presented approach has the following advantages:

- + Potentially, all three types of signal correlation, i.e. spatial, temporal, and structural correlation are accurately modeled. Moreover, accurate switching activity can be calculated for every node from the theoretical minimum number of statistical measures (moments).
- + For large circuits, the approximation of all types of signal correlation can be controlled by means of a single parameter. Approximation accuracy with respect to this control parameter behaves nearly monotonically.
- + When signal correlations are approximated, incremental processing of the network is still possible and no pre-processing is required. Thus, the complexity with respect to the number of nodes g remains $O(g)$.

These advantages come at the expense of the following critical points:

- – The complexity of the algorithm with respect to the approximation parameter d is $O(2^{2d})$. It has been shown, however, that even for small values of d the estimation accuracy improves significantly compared to mere propagation of signal probabilities $p^1_{x_i}$ and neglecting all correlations ($d = 0$).
- – The number of statistical measures (moments) to be extracted from the application data and stored, grows exponentially with the approximation parameter d. A possible workaround is to extract fewer moments than the maximum allowed value of d would require. In this case, different degrees of approximation would apply to structural correlation and spatio-temporal correlation of input data.

3.4.5 Extension to Multi-Delay Model

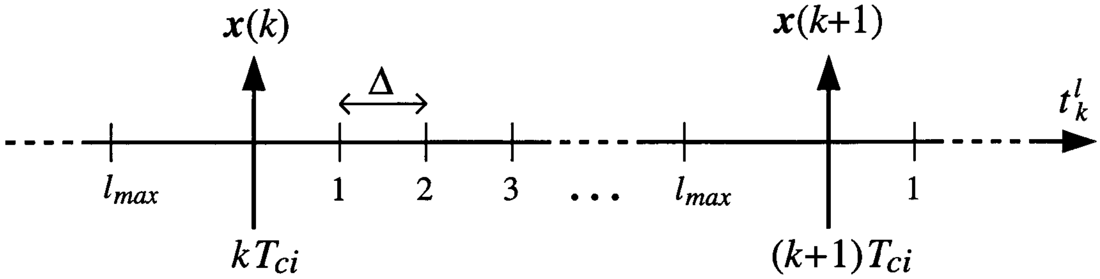
Discrete-time switching

In practice, signals do not propagate instantaneously through the circuit but are afflicted with time lag due to gate and interconnect delays. This time lag provokes spurious signal transitions (glitches) in the circuit that are not necessary from a purely functional point of view. Typically, activity due to glitches is about 10-25% of the total switching activity [GDKW92]. However, for particular circuits such as multipliers, glitching might contribute considerably more.

For consideration of glitches in gate-level activity estimation, a *multi-delay model* is commonly assumed [DTP98, MDG$^+$97]. Under this model, the delay time associated with each gate is a multiple integer of some fixed time unit Δ. This implies that signal switching is modeled to occur only at discrete points t_k^l during the computation interval T_{ci}:

$$t_k^l \;=\; kT_{ci} + l \cdot \Delta \qquad (l = 0, 1, \ldots, l_{max}) \tag{3.29}$$

At time points kT_{ci} $(k = 0, 1 \ldots)$ corresponding to the active clock-edge, new data samples arrive at the PIs. $l_{max} \cdot \Delta$ is the maximum delay. A condition to ensure correct functionality of the circuit is: $l_{max}\Delta < T_{ci}$.



Let $\alpha_{y_j}^l$ be the switching activity of signal y_j at time points t_k^l. The total switching activity of gate j during one computation interval is then given by

$$\alpha_{y_j} \;=\; \sum_{l=0}^{l_{max}} \alpha_{y_j}^l \qquad (j = 1, \ldots, g)\ . \tag{3.30}$$

The switching activities $\alpha_{y_j}^l$ $(l = 0, 1, \ldots, l_{max};\ j = 1, \ldots, g)$ can be calculated in a straightforward manner by extending the zero-delay procedure described in section 3.4.2. Moreover, provided that the PIs x_i $(i = 1, \ldots, n)$ change their logical state no more than once during any computation interval, the switching activities $\alpha_{y_j}^l$ are explicitly defined by the same $M(n) = 2^n(2^n - 1)$ moments of the PI signals that were required in the zero-delay case.

The necessary adjustments for the activity calculation procedure from section 3.4.2 shall be described next. Without loss of generality it is assumed that PI signal switching occurs at times kT_{ci} $(k = 0, 1, \ldots)$.

Extension of zero-delay procedure

Step (I)

Gate j potentially switches at times t_k^l if the circuit contains a delay path of length l from any of the PIs to the output of j. This set of potential switching points $L_j \subset \{0, 1, \ldots, l_{max}\}$ is given as union of the set of switching points of the precursor nodes of gate j, offset by the delay time of gate j. For $l \notin L_j$ holds $\alpha_{y_j}^l \equiv 0$.

For each $l \in L_j$, an arithmetic transform $\mathcal{A}(y_j^l)$ is constructed from those precursor node transforms that are relevant for the transition of gate j at time l. Unlike under the zero-delay model, the PI variables now must be time-labeled, since PI samples from the previous $(k - 1)$ and current (k) computation intervals may appear in the same arithmetic transform.

When constructing $\mathcal{A}(y_j^l)$, variable x_i in the polynomial at some input of gate j has label $k - 1$, if the delay path from x_i via this gate input to the output of gate j is longer than l. Otherwise, x_i has label k. The exponent suppression rule (3.22) applies only to variables that refer to the same time point. The arithmetic transform that refers to the last switching point of gate j is identical to the one under the zero delay model, i.e. all variables in this transform are consistently labeled with either $k - 1$ or k depending on whether y_j refers to the final value of the previous or current computation interval.

Step (II)
For each $l \in L_j$, the static probability $p^1_{y_j^l}$ of signal y_j at time point l is obtained from evaluation of $\mathcal{A}(y_j^l)$. As opposed to the zero-delay model, the static probabilities $p^1_{y_j^l}$ this time may depend on lag-one moments of the PI signals, since time labels $k-1$ and k had to be assigned to PI variables in step (I).

Step (III)
For each $l \in L_j$ the arithmetic transform $\mathcal{A}(y_j^{l-1} y_j^l) = \mathcal{A}(y_j^{l-1})\mathcal{A}(y_j^l)$ is constructed such as to reflect the correlation between adjacent switching time points of gate j. For the first switching time point l, $\mathcal{A}(y_j^{l-1})$ corresponds to the last switching time point of the previous computation interval, i.e. in $\mathcal{A}(y_j^{l-1})$ all variables are labeled with $k-1$.

Evaluation of $\mathcal{A}(y_j^{l-1} y_j^l)$ with the corresponding PI moments yields $p^{1\text{-}1}_{y_j^{l-1}\text{-}y_j^l}$. From this, and the static probabilities $p^1_{y_j^{l-1}}$ and $p^1_{y_j^l}$ corresponding to time points $l-1$ and l, the switching activity of gate j at time point $l \in L_j$ follows as[7]

$$\begin{aligned}
\alpha^l_{y_j} &= p^{1\text{-}0}_{y_j^{l-1}\text{-}y_j^l} + p^{0\text{-}1}_{y_j^{l-1}\text{-}y_j^l} \\
&= p^1_{y_j^{l-1}} - p^{1\text{-}1}_{y_j^{l-1}\text{-}y_j^l} + p^1_{y_j^l} - p^{1\text{-}1}_{y_j^{l-1}\text{-}y_j^l} \\
&= p^1_{y_j^{l-1}} + p^1_{y_j^l} - 2 \cdot p^{1\text{-}1}_{y_j^{l-1}\text{-}y_j^l} \; . \qquad (3.31)
\end{aligned}$$

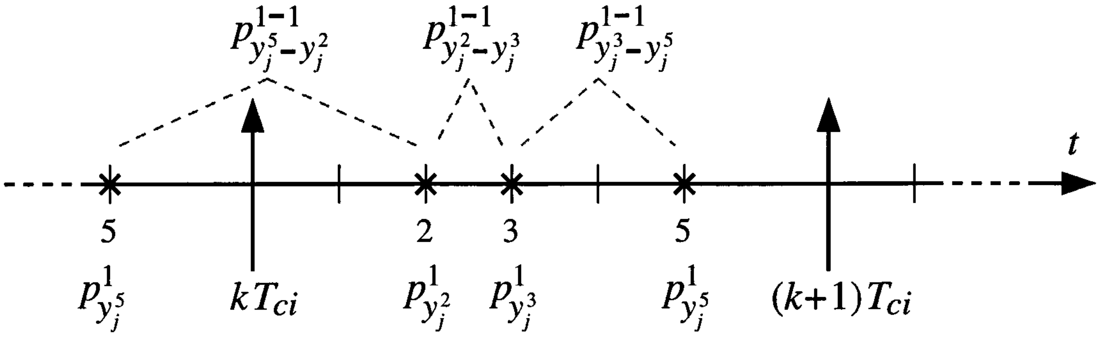
---

[7] Under the same timing model (3.29), in [SP00] a different formula for the probability of switching from '0' at time point $l-1$ to '1' at time point l has been derived in analogy to (3.10). In our notation this formula goes

$$p^{0\text{-}1}_{y_j^{l-1}\text{-}y_j^l} = p^1_{y_j^{l-1}}(1 - p^1_{y_j^l})(1 - \varrho_{y_j^{l-1}\text{-}y_j^l})$$

where $\varrho_{y_j^{l-1}\text{-}y_j^l}$ is the correlation coefficient of the random variables associated with time points $l-1$ and l, see (A.8). The above formula bases on the incorrect assumption that $p^{0\text{-}1}_{y_j^{l-1}\text{-}y_j^l} = p^{1\text{-}0}_{y_j^{l-1}\text{-}y_j^l}$. Although this holds if y_j^{l-1} and y_j^l refer to consecutive samples of a *stationary* binary signal, see proof of theorem 3.1, this assumption is fallacious under a multi-delay model. In this case, y_j^{l-1} and y_j^l do not refer to a stationary signal, because in general $p^1_{y_j^{l-1}} \neq p^1_{y_j^l}$.

The diagram below elucidates (3.31) for $L_j = \{2, 3, 5\}$.


$p^{1\text{-}1}_{y_j^5\text{-}y_j^2}$
$p^{1\text{-}1}_{y_j^2\text{-}y_j^3}$
$p^{1\text{-}1}_{y_j^3\text{-}y_j^5}$
t
5
2
3
5
$p^1_{y_j^5}$
kT_{ci}
$p^1_{y_j^2}$
$p^1_{y_j^3}$
$p^1_{y_j^5}$
$(k+1)T_{ci}$

If gate j can switch only at a single time point, $p^1_{y_j^{l-1}} = p^1_{y_j^l} = p^1_{y_j}$ and $p^{1\text{-}1}_{y_j^{l-1}\text{-}y_j^l} = p^{1\text{-}1}_{y_j\text{-}y_j}$. In this case, (3.31) is identical to the switching activity (3.9) under the zero-delay model.

Example:
Consider again the example circuit in figure 3.10 and assume gate y_1 has a propagation delay of 2Δ, and gates y_2 and y_3 a delay Δ. Thus, the set of potential switching time points of gate y_3 is $L_3 = \{2, 3\}$. The arithmetic transform for y_3 at time point $l = 2$ follows as

$$\begin{aligned}\mathcal{A}(y_3^2) &= \mathcal{A}(y_1^{k-1}) + \mathcal{A}(y_2^k) - \mathcal{A}(y_1^{k-1})\mathcal{A}(y_2^k) \\ &= x_1^{k-1} - x_2^{k-1}x_1^{k-1} + x_3^k x_2^k - (x_1^{k-1} - x_2^{k-1}x_1^{k-1})x_3^k x_2^k\end{aligned}$$

and the corresponding static signal probability is

$$p^1_{y_3^2} = p^1_{x_1} - p^{11}_{x_2x_1} + p^{11}_{x_3x_2} - p^{1\text{-}11}_{x_1\text{-}x_3x_2} + p^{11\text{-}11}_{x_2x_1\text{-}x_3x_2} \,.$$

Note how the static probability depends on the first-order statistics of the PIs.

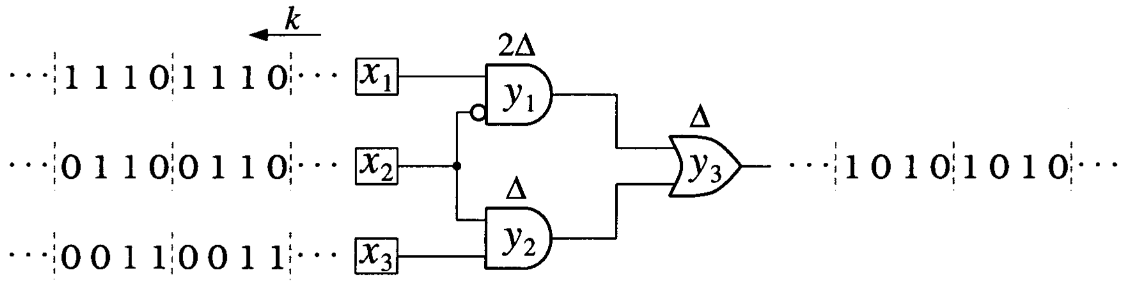


Figure 3.10: Example circuit with gate delays.

The arithmetic transform at the last switching point $l = 3$ is equivalent to the zero-delay transform for y_3:

$$\mathcal{A}(y_3^3) \quad = \quad x_1 - x_2 x_1 + x_3 x_2 \; .$$

In step (III), depending on whether y_3^3 refers to the previous or current computation interval, the variables in $\mathcal{A}(y_3^3)$ are labeled with $k-1$ or k as follows: $p^{1\text{-}1}_{y_3^3\text{-}y_3^2}$ is obtained from evaluation of

$$\begin{aligned} \mathcal{A}(y_3^3 y_3^2) &= \mathcal{A}(y_3^3)|_{k-1} \cdot \mathcal{A}(y_3^2) \\ &= x_1^{k-1} - x_2^{k-1} x_1^{k-1} + x_3^{k-1} x_2^{k-1} x_3^k x_2^k \; . \end{aligned}$$

$p^{1\text{-}1}_{y_3^2\text{-}y_3^3}$ is obtained from evaluation of

$$\begin{aligned} \mathcal{A}(y_3^2 y_3^3) &= \mathcal{A}(y_3^2) \cdot \mathcal{A}(y_3^3)|_k \\ &= x_1^{k-1} x_1^k - x_2^{k-1} x_1^{k-1} x_1^k - x_1^{k-1} x_2^k x_1^k \\ &\quad + x_2^{k-1} x_1^{k-1} x_2^k x_1^k + x_3^k x_2^k \; . \end{aligned}$$

Note that in general $p^{1\text{-}1}_{y_3^3\text{-}y_3^2} \neq p^{1\text{-}1}_{y_3^2\text{-}y_3^3}$ as well as $p^1_{y_3^2} \neq p^1_{y_3^3}$.

Finally, the switching activity of y_3 under a multi-delay model follows from (3.30) and (3.31) as

$$\begin{aligned} \alpha_{y_3} &= \alpha^2_{y_3} + \alpha^3_{y_3} \\ &= 2(p^1_{y_3^2} + p^1_{y_3^3} - p^{1\text{-}1}_{y_3^3\text{-}y_3^2} - p^{1\text{-}1}_{y_3^2\text{-}y_3^3}) \; . \end{aligned}$$

For the periodic input sequence shown in figure 3.10 this evaluates to $\alpha_{y_3} = 1$, which is easily verified with logic simulation. Interestingly, in the present case zero- and multi-delay model result in the same switching activity of gate y_3 although there is one additional switching point under the multi-delay model. ☐

The correlation approximation scheme proposed in section 3.4.3 may be applied equivalently under the multi-delay model. However, for the selection of essential variables in $\mathcal{A}(y_j^l)$ time labels shall be ignored, i.e. x_i^{k-1} and x_i^k represent the same essential variable.

Glitch filtering

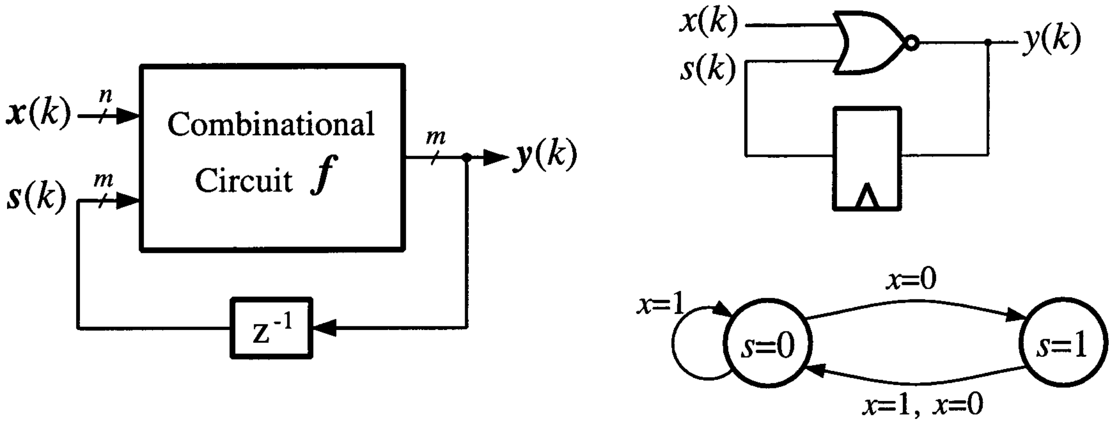
For the given timing model, the above procedure exactly calculates the switching activity for every gate in a combinational circuit. However, discrepancies compared to gate- or transistor-level simulation with detailed timing capabilities can occur due to the abstract model employed. In particular, the multi-delay model ignores the effect of *inertial delay* on the switching behavior of logic gates, resulting in overestimation of switching activity [MDG+97, DTP98]. Due to nonzero charging/discharging time of internal capacitances, a signal transition at the gate input will only cause a transition at the gate output, if the new input values remain stable for a sufficiently long period of time. Short-pulse glitches will not pass through the gate.

In order to account for this effect, a glitch filtering scheme has been proposed in [DTP98], which suppresses output pulses that are shorter than the delay time of the gate. A similar approximation could be incorporated into the above procedure by weighting a switching time point l_1 of gate j that is subject to suppression through the next time point l_2 with some probability of occurrence. This occurrence probability is determined by the Boolean function of gate j and $p^{1}_{y_j^{l_1}}$, $p^{1}_{y_j^{l_2}}$, $p^{1\text{-}1}_{y_j^{l_1}\text{-}y_j^{l_2}}$.

3.5 Probabilistic Analysis of Sequential Circuits

3.5.1 Feedback Circuits

Sequential circuits contain memory elements and either implement recursive or nonrecursive computations depending on whether the memory elements are located in feed-back or feed-forward signal paths. For sequential circuits without feedback, switching activity information for the combinational blocks can be obtained as explained in section 3.4. However, activity calculation for circuits with feedback requires the statistical properties of the feedback signal to be known, see figure 3.11(a). Such feedback circuits are usually viewed as finite-state machines (FSM), by regarding the current feedback value $s(k)$ as the state of the system.



(a) (b)

Figure 3.11: General logic circuit with sequential feedback (a), and example schematic with associated state-transition graph (b).

In case of feedback, the combinational circuit has an extended set of input signals $\tilde{\boldsymbol{x}}$ formed of primary inputs $\boldsymbol{x}$ and state signals $\boldsymbol{s}$, i.e.[8]

$$\tilde{\boldsymbol{x}} = (x_1, \ldots, x_n, s_1, \ldots, s_m) \ .$$

If there are m state signals, the system has a total of 2^m states.
The sequential feedback

$$\boldsymbol{s}(k+1) \ = \ \boldsymbol{y}(k) \ = \boldsymbol{f}\left(\boldsymbol{x}(k), \boldsymbol{s}(k)\right) \tag{3.32}$$

introduces a new form of signal dependency, which manifests itself in spatio-temporal correlation of the extended input signal $\tilde{\boldsymbol{x}}$.
In analogy to (3.14) and (3.15), the i-th moments

$$p^{1\ldots1}_{\tilde{x}_{j_1}\ldots\tilde{x}_{j_i}} \quad (i = 1, \ldots, n+m) \tag{3.33}$$

and the lag-one i-th moments

$$p^{1\ldots1\text{-}1\ldots1}_{\tilde{x}_{j_1^1}\ldots\tilde{x}_{j_q^1}\text{-}\tilde{x}_{j_1^2}\ldots\tilde{x}_{j_r^2}} \quad (i = q + r = 2, \ldots, 2(n+m)) \tag{3.34}$$

---

[8] Without loss of generality it may be assumed that all m primary outputs of the combinational circuit $y_{g-m+1}, \ldots, y_g$ are state signals, see figure 3.1.

for the extended input $\tilde{\mathbf{x}}$ are required for exact calculation of gate activities. Moments that exclusively comprise primary inputs x_i $(i = 1, \dots, n)$ are assumed to be known a priori. All other moments, which refer to any of the state signals s_i $(i = 1, \dots, m)$ need to be determined prior to activity computation.

3.5.2 Analysis via Spectral Transformation

Explicit calculation of the unknown moments in (3.33) and (3.34), and hence switching activity, is a non-trivial task. The problem can be attacked by using the same technique that was utilized for switching activity calculation for combinational circuits in section 3.4, i.e. spectral transformation of Boolean functions. Since the general mathematical notation for the derivation of the unknown moments in (3.33) and (3.34) becomes very bulky, the basic concept shall be indicated by means of the example shown in figure 3.11(b).

First, the arithmetic transform of the next-state function $s(k+1) = \overline{x(k) \vee s(k)}$ is constructed. For brevity, time index k will be denoted as superscript. This yields (3.35), see table 3.1, which involves a second unknown term $x^k s^k$. Equation (3.36) corresponding to this term is constructed as arithmetic transform of $x(k) \wedge s(k)$. This in turn introduces two other unknown terms $s^k x^{k+1}$ and $x^k s^k x^{k+1}$, for which equations are formed in (3.37) and (3.38). This process would need to be continued indefinitely.

$$
\begin{aligned}
s^{k+1} &= 1 - x^k - s^k + x^k s^k && (3.35)\\
x^{k+1} s^{k+1} &= x^{k+1} - x^k x^{k+1} - s^k x^{k+1} + x^k s^k x^{k+1} && (3.36)\\
s^{k+1} x^{k+2} &= x^{k+2} - x^k x^{k+2} - s^k x^{k+2} + x^k s^k x^{k+2} && (3.37)\\
x^{k+1} s^{k+1} x^{k+2} &= x^{k+1} x^{k+2} - x^k x^{k+1} x^{k+2} \\
&\quad - s^k x^{k+1} x^{k+2} + x^k s^k x^{k+1} x^{k+2} && (3.38)\\
&\ \vdots
\end{aligned}
$$

Applying the expectation operator in the same manner as in section 3.4.2 and assuming stationary primary inputs and hence stationary state signals, the above set of arithmetic transform equations corresponds to the following equations for moments:

$$\begin{aligned}
p^{1}_{s} &= 1 - p^{1}_{x} - p^{1}_{s} + p^{11}_{xs} && (3.39)\\
p^{11}_{xs} &= p^{1}_{x} - p^{1\text{-}1}_{x\text{-}x} - p^{1\text{-}1}_{s\text{-}x} + p^{11\text{-}1}_{xs\text{-}x} && (3.40)\\
p^{1\text{-}1}_{s\text{-}x} &= p^{1}_{x} - p^{1\text{-}1}_{x\text{-}x} - p^{1\text{-}.\text{-}1}_{s\text{-}.\text{-}x} + p^{11\text{-}.\text{-}1}_{xs\text{-}.\text{-}x} && (3.41)\\
p^{11\text{-}1}_{xs\text{-}x} &= p^{1\text{-}1}_{x\text{-}x} - p^{1\text{-}1\text{-}1}_{x\text{-}x\text{-}x} - p^{1\text{-}1\text{-}1}_{s\text{-}x\text{-}x} + p^{11\text{-}1\text{-}1}_{xs\text{-}x\text{-}x} && (3.42)\\
&\vdots
\end{aligned}$$

As can be seen, moments referring to more than two consecutive time points get involved, e.g. $p^{1\text{-}1\text{-}1}_{x\text{-}x\text{-}x}$. In general, the above set of equations may be solved, if temporal correlation of the primary input x is confined to a given number of time steps. In appendix C, this solution is worked out for first-order correlation, i.e. only p^{1}_{x} and $p^{1\text{-}1}_{x\text{-}x}$ are given.

To obtain information on the temporal correlation of s, e.g. the first-order moment $p^{1\text{-}1}_{s\text{-}s}$, a set of equations similar to the one above can be constructed, see the example in appendix C. Note that even if the primary input x is temporally uncorrelated, state signal s in general is not, i.e. $p^{1\text{-}1}_{s\text{-}s} \neq p^{1}_{s} p^{1}_{s}$.

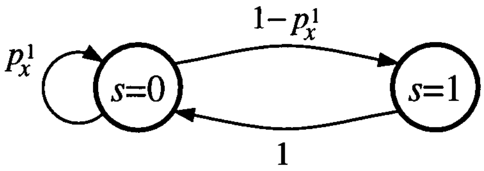
In the above example, the circuit has only one primary input and one state signal. The extension to circuits with multiple inputs and state signals is possible in a straightforward manner, by replacing x and s in the above analysis by vectors $\boldsymbol{x}$ and $\boldsymbol{s}$. In this case, spatial correlation between primary inputs can be acknowledged with the corresponding exponential increase in complexity, see section 3.3.3.

3.5.3 Markovian Analysis

Traditionally, a feedback circuit, or FSM, has been modeled as *Markov chain* [HMPS96, T^{+}95, CSY96, GDKW92] , such that

$$Pr\{\boldsymbol{s}^{k+1} | \boldsymbol{s}^{k}, \ldots, \boldsymbol{s}^{0}\} = Pr\{\boldsymbol{s}^{k+1} | \boldsymbol{s}^{k}\} \tag{3.43}$$

for all possible state values $\boldsymbol{s}$ of the system. In this case, the circuit's behavior is determined by specification of all $(2^{m})^{2}$ state transition probabilities $Pr\{\boldsymbol{s}^{k+1} | \boldsymbol{s}^{k}\}$, see appendix A. These transition probabilities result from the static probabilities of the primary input signals $p^{1}_{x_i}$ $(i = 1, \ldots, n)$ (or their joint probabilities $p(x_1, \ldots, x_n)$ if spatial correlation is acknowledged), and are associated with the edges of the state-transition graph of the system. For the example circuit in figure 3.11(b) this model is shown below.

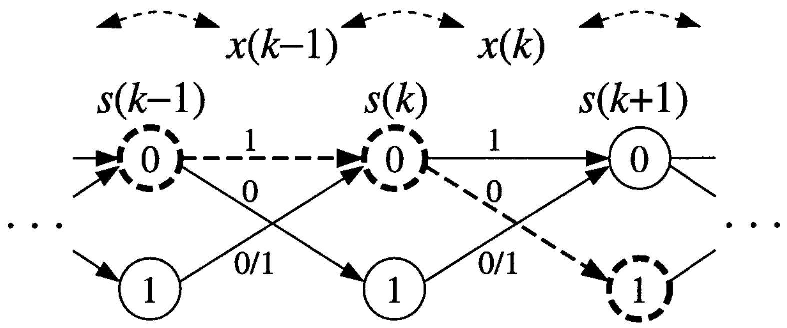


Since with the above model the memory of the system reaches back only one time step, (3.43) also is called lag-one Markov chain. Such a model, however, totally neglects temporal correlation in the primary input $\boldsymbol{x}$. Thus, in the context of data statistics (3.43) only represents an approximation, although this is often ignored [T$^+$95, GDKW92]. To correctly model feedback circuits for arbitrary input data statistics, the concept of higher-order Markov chains must be employed [MMP99]. Formally, the following result is presented.

Theorem 3.3
Assume a circuit with sequential feedback as in (3.32). If temporal correlation in the primary input $\boldsymbol{x}$ extends over τ time steps, then state signal $\boldsymbol{s}$ forms a lag-$(\tau+1)$ Markov chain, i.e.

$$Pr\{\boldsymbol{s}^{k+1}|\boldsymbol{s}^{k},\boldsymbol{s}^{k-1},\ldots,\boldsymbol{s}^{0}\} = Pr\{\boldsymbol{s}^{k+1}|\boldsymbol{s}^{k},\ldots,\boldsymbol{s}^{k-\tau}\}\ . \qquad (3.44)$$

Proof: The proof is outlined for $\tau = 1$ (first-order temporal correlation of primary input) and the example circuit in figure 3.11(b), since spatial correlation is immaterial here. Assume that state s advances as indicated by dashed lines in the trellis diagram below.



Under a lag-one Markov chain model the probability for the next state only depends on the static signal probability of x if the present state is fixed, e.g. $Pr\{s^{k+1} = 1 | s^k = 0\} = Pr\{x^k = 0\}$. Under a lag-two Markov chain model the probability for the next state depends on the one-step conditional probability of x if the present and previous states are fixed, e.g. $Pr\{s^{k+1} = 1 | s^k = 0, s^{k-1} = 0\} = Pr\{x^k = 0 | x^{k-1} = 1\}$. Since $\tau = 1$, in general $Pr\{x^k = 0\} \neq Pr\{x^k = 0 | x^{k-1} = 1\}$ and hence $Pr\{s^{k+1} | s^k\} \neq Pr\{s^{k+1} | s^k, s^{k-1}\}$, which contradicts the lag-one Markov chain model (3.43).

On the other hand, modeling s as lag-three Markov chain is not necessary, because for $\tau = 1$ follows that $Pr\{x^k | x^{k-1}\} = Pr\{x^k | x^{k-1}, x^{k-2}\}$ and hence $Pr\{s^{k+1} | s^k, s^{k-1}\} = Pr\{s^{k+1} | s^k, s^{k-1}, s^{k-2}\}$. ■

Thus, in the context of switching activity calculation, modeling a circuit as lag-one Markov chain will result in erroneous moments $p_s^1, p_{xs}^{11}, p_{s\text{-}s}^{1\text{-}1}, \ldots$ if the primary inputs are temporally correlated. This in turn induces an error on the switching activity of the gates in the combinational part of the circuit. It has been found that this error can exceed 100% depending on the specific circuit structure and input correlation [SK96]. In appendix C the discrepancy between a lag-one and a lag-two Markov chain model is exemplified for the circuit in figure 3.11(b).

3.5.4 Implications

At this point the following implications can be formulated:

- The conventional Markov chain model for feedback circuits is applicable only in case of uncorrelated primary input signals. For correct modeling of correlated input sequences, higher-order Markov models are required.

- The arithmetic transform approach provides a means for explicit calculation of switching activities in feedback circuits under a higher-order Markov model. Unlike in combinational circuits, where moments of the primary inputs up to lag-one are sufficient for exact analysis, feedback circuits require the knowledge of moments up to lag-τ if temporal correlation in the input signal stretches over τ time steps.

- An approximation technique similar to the one examined for combinational circuits in section 3.4.3 is conceivable for the estimation of state signal statistics. This would require an appropriate selection procedure for relevant terms in sets of equations such as (3.35)-(3.38). In practice, one resorts to logic simulation in order to infer the statistical properties of the state signals as well as gate switching activities. In this case, compact input stimuli are desired that closely resemble the relevant statistical properties of the target input signal [MMP97].

3.6 Summary

This chapter was concerned with the probabilistic analysis of switching activity for average power estimation. We have provided the theoretical basis for exact calculation of switching activity in logic circuits by bridging known concepts from two different disciplines: The notation of moments of a random variable known from probability theory, and the concept of spectral transformation known from the theory of switching functions. Based on this theoretical foundation, a novel heuristic for the approximation of signal correlation in combinational circuits has been proposed.

Our work improves the state of the art for probabilistic activity estimation in combinational circuits [MDG$^+$97, CMD97, MMP98] as follows :

- The proposed exact technique accurately models structural, spatial, and temporal correlation, where each step of the procedure is dedicated to one of these three types of correlation. Such exact technique is not only of theoretical interest but is also useful for calibrating approximation techniques.

- Contrary to [CMD97], where polynomial expressions for all four fist-order joint probabilities were incrementally constructed, we have shown that it suffices to incrementally construct *one* polynomial for each node. From this polynomial the correct joint probabilities, and hence switching activity for every node can be derived.

- Under a zero- and multi-delay model, the gate switching activities in a combinational circuit can be calculated exactly from the minimum number of statistical measures that fully specify the first-order statistics of the primary inputs.
- The proposed estimation scheme provides control over the approximation of all three types of correlation by means of a single parameter. The new estimation scheme requires no preprocessing of the network and retains incremental processing. The monotonic behavior and effectiveness of this approximation method have been verified experimentally.

Furthermore, it was demonstrated how the arithmetic transform technique also can be used for explicit analysis of circuits with sequential feedback. In this case, higher-order correlation of inputs can be taken into account.

Seite Leer /
Blank leaf

Chapter 4

Data Statistics and Minimum Power Dissipation

4.1 Introduction

4.1.1 Motivation

As has been seen in the previous chapter, power consumption in digital VLSI is subject to the statistical properties of the application data at hand. Given a gate-level circuit and sufficient information on the data characteristics, switching activity and power dissipation can be predicted. Another question arising in this context is the existence of an absolute lower bound on switching activity and power dissipation subject to statistical data properties. Although related to power estimation, this lower bound problem differs in that it asks for the amount of energy indispensable for the completion of some given computational task. Unlike in power estimation, the lower bound shall account for data statistics without assuming any specific network structure.

Besides its theoretical significance such a lower bound would also provide useful guidance for practical low-power design. Knowing the minimum dissipation achievable for certain operations would allow to rate the energy efficiency of state-of-the-art implementations for these operations. Implementations with poor energy efficiency should then be prime candidates for future optimization efforts.

4.1.2 Previous Work

Thermodynamics of computation

The relation between entropy and the second law of thermodynamics on one hand, and the physical limits of computation on the other hand, has challenged researchers for many decades [LR90, Hey99]. Despite its long record in the history of science, the topic has remained a controversial subject among physicists. Its pervasive consequences even fuel philosophical debates and speculations [KS99]. Section 4.2.3 shall discuss the relation between the thermodynamics of computation and minimum power consumption in VLSI.

Information-theoretic approaches

Next to thermodynamic entropy, the concept of information-theoretic entropy [CT91] has also been employed to investigate the limits of computing. One such approach is the notion of *computational work*, which equals the entropy of the computation's output sequence, multiplied by the number of possible input symbols [Hel72]. However, computational work is invariant with respect to data statistics, since it assumes uniformly distributed input data.

Given the information-theoretic entropy of a sequence of symbols, lower and upper bounds on the switching activity of a binary representation of this sequence have been derived in [RSH99b]. In section 4.5, this result will be employed for the analysis of minimum power consumption in the context of data transmission.

In [Sha97], the problem of minimum power consumption in digital VLSI has been investigated in analogy to Shannon's joint source-channel coding theorem [Sha48]. This approach will be discussed in section 4.7, and classified according to the problem hierarchy to be developed.

Technology limits

Despite ongoing research for alternatives, e.g. quantum information processing [SR00] or low-temperature superconducting ICs [BTR00]), to date MOS-transistor based VLSI, and here predominantly CMOS[1], has remained the implementation technology of choice for virtually any application manipulating time-discrete digital data. For volume production, this is not expected to change for at least the next 10 to 20 or so years.

Opportunities for future gigascale CMOS integration were thoroughly analyzed in [Mei95] by considering theoretical and practical limits at different levels of hierarchy: *Fundamental limits* are inferred from the basic laws of thermodynamics, quantum mechanics, and electromagnetics. *Material limits* are dictated by microscopic properties of Si, such as carrier mobility and saturation velocity. *Device limits* for MOS-transistors are given by the minimal channel length. Apart from technological feasibility, channel length is confined by transistor-internal short channel effects. *Circuit limits* refer to requirements that must be met by logic gates and interconnects in CMOS designs, and are linked to the power-delay product of a single switching event. On the *system level*, switching energy limits are imposed by heat removal and cycle time requirements.

For a given delay, the limits on power consumption were found to be more restrictive the higher one moves up in the hierarchy [Mei95]. However, statistical data properties were not taken into account in this analysis. This shall be done in this chapter.

4.1.3 Outline

Since the choice of implementation medium determines the role of data statistics for power minimization, this issue is discussed first in section 4.2 for CMOS. The problem of lower bounds on power dissipation in the context of data statistics will be formulated and classified in section 4.3. Subsequent sections then deal with the four subproblems resulting from this classification. Section 4.8 compares

---

[1] Different logic styles are being used in conjunction with CMOS technology. In what follows, the term CMOS will imply static CMOS logic.

the lower bound problem with the information-theoretic transmission bound problem.

4.2 The Limits of VLSI Technology

4.2.1 The Ideal MOSFET

For minimum power consumption in the context of data statistics, it is of interest to explore the minimum dissipation associated with a binary switching event. This shall be done at the device level, since MOS transistors are the basic devices in CMOS technology.

The energy stored on the gate capacitance of a single MOSFET in a CMOS inverter is

$$E_g = Q_g V_{dd}/2 .$$

This amount of energy is dissipated for every switching of logic state, represented by the gate charge. The minimum possible gate charge is one electron: $Q_g = q$. Approximating the minimum allowable supply voltage for a simple inverter circuit as [MD00]

$$V_{dd} = 2(\ln 2)kT/q \approx 36\text{mV} ,$$

the ideal MOSFET dissipates a minimum energy of

$$E_0 = (\ln 2)kT \tag{4.1}$$

for every binary switching transition[2]. At room temperature this amounts to

$$E_0 \approx 3 \cdot 10^{-21}\ \text{J} \quad @\ 300\ \text{K} . \tag{4.2}$$

Such a device that only stores a single electron in its gate capacitance when being charged has been named SETE (single electron, thermal excitation) MOSFET [MD00]. It should be noted that this is a hypothetical device that ignores many practical aspects such as layout parasitics and technology imperfections.

---

[2] Here, $k \approx 1.38 \cdot 10^{-23}$J/K is Boltzmann's constant and T is the absolute temperature.

4.2.2 Technology Trends

Table 4.1 compares several CMOS standard-cell libraries available today to the SETE MOSFET. L_{min} denotes the minimum feature size (= MOS transistor channel length), and V_{dd} is the supply voltage. The gate capacitance C_g is derived from the input capacitance of a standard inverter with 1x-drive, assuming a ratio of two for the p/n-channel widths. Gate charge $Q_g = V_{dd}C_g$ is given as number of electrons. The energy dissipated per switching event $E_{sw} = C_g V_{dd}^2/2$ is given in units of fundamental energy limits E_0 in (4.2).

As can be seen, even the most advanced technology dissipates four to five orders of magnitude more energy than the theoretical limit would imply. This is in compliance with experimental data provided in [Key01]. On the other hand, the number of electrons stored in a transistor gate is surprisingly small already. However, extrapolation of figures from table 4.1 to speculate about the future trend and when the theoretical limit will be reached is notoriously difficult.

Library		L_{min}	V_{dd}	C_g	Q_g	E_{sw}
Name	Vendor	[nm]	[V]	[aF]	[q_e]	[E_0]
CXB	AMS	800	5.0	12’000	375’000	$5.2 \cdot 10^7$
CUB	AMS	600	5.0	10’000	312’500	$4.3 \cdot 10^7$
C075	Philips	350	3.3	3’000	62’000	$5.7 \cdot 10^6$
C050	Philips	250	2.5	2’000	31’250	$2.2 \cdot 10^6$
CMOS18	Philips	180	1.8	1’500	17’000	$8.5 \cdot 10^5$
SA-27[a]	IBM	150	1.3	1’075	8’750	$3.2 \cdot 10^5$
Cu-11[b]	IBM	110	1.0	330	2’000	$5.8 \cdot 10^4$
SETE	[MD00]	14	0.04	4.47	1	1

[a] nominal V_{dd}=1.8V, [b] nominal V_{dd}=1.2V

Table 4.1: CMOS technology trend in terms of number of elementary charges stored on a single gate capacitance, and energy dissipation per switching event.

4.2.3 VLSI and Thermodynamics

Thermodynamic results

In his 1929 paper [Szi29] Szilard introduced his famous one-molecule model of a thermodynamic system. The so-called *Szilard engine* for the first time linked the concepts of entropy, information and memory and led to the notion of a "bit" of information. In this way, Szilard's work provided the foundation for Shannon's theory of information [Sha48], which will be the starting point for the discussion in section 4.7.

Another important milestone in the development of today's understanding of the physics of computing is the work of Landauer and Bennett. Before their work it was believed that any computer operating at temperature T must dissipate at least $(\ln 2)kT$ Joule for every "elementary act of information". This is exactly the energy limit (4.1) previously derived for CMOS.

In [Lan61] Landauer introduced the concept of *logical reversibility* and argued that logical irreversibility implies physical irreversibility, which is accompanied by heat dissipation. This follows from the fact that a logically irreversible operation would otherwise be able to decrease the thermodynamic entropy of the computer's memory without a compensating entropy increase elsewhere in the universe, thereby violating the second law of thermodynamics. Subsequently, any logically irreversible operation, such as the mapping of an unknown, randomly chosen state to a known successor state, must dissipate a minimum amount of energy. On the other hand, computation steps that do not discard information can be done reversible in principle. Today, this fact is known as *Landauer's principle* [BGL$^+$98].

Bennett [Ben73] expanded Landauer's idea, arguing that *every* computation step can be made in a logically reversible manner. Thus, erasure of information is not essential to computing. Adopting this conclusion, many thought experiments to construct computers with arbitrary little dissipation have since been performed, including reversible logic (Fredkin) gates, the frictionless billiard-ball computer, Brownian and enzymatic Turing machines [BL85]. These models give evidence to the fact, that the laws of physics do not preclude the invention of a technology that allows dissipationless computing. However, they do not provide a practical design procedure for such a machine.

Adiabatic logic

One approach to "dissipationless" computing based on MOS transistors, called *adiabatic logic*, emerged in the early 1990's [CB95]. This approach dispenses with constant-voltage charging of capacitances as done in ordinary CMOS. In analogy to thermodynamics, the defining property of "adiabatic" logic is the dissipation decrease with increased time allowed for the completion of the charge transfer process. This requires time-varying supply voltages as well as special circuit styles.

However, *fully-adiabatic* processing can only be achieved by using gates that implement invertible functions exclusively, i.e. by restriction to reversible logic in the sense of Bennet [Ben73]. The overhead induced by reversible logic can be avoided by resorting to *partially-adiabatic* designs. In this case, dissipation involves a non-adiabatic component, which can not be decreased by slowing down the charge transfer process.

Although feasible in principle, the complicated circuitry and the resonant power supplies required, have largely prevented adiabatic logic from being commercially used to date.

Hierarchy of dissipation limits

As minimum feature size of VLSI technology continues to shrink, energy dissipation per binary switching event diminishes. If todays CMOS technology could be advanced to the SETE level, *every* switching event, whether associated to a logically irreversible operation or not, would dissipate E_0 Joule independently of the time allowed to complete the switching. Further reduction is impossible because of the basic concept of information representation and manipulation, i.e. constant-voltage charging and draining of capacitances.

If one is to break the E_0-per-switching barrier at room temperature, one will be forced to use an implementation technology that dissipates no energy for logically reversible operations (Landauer's

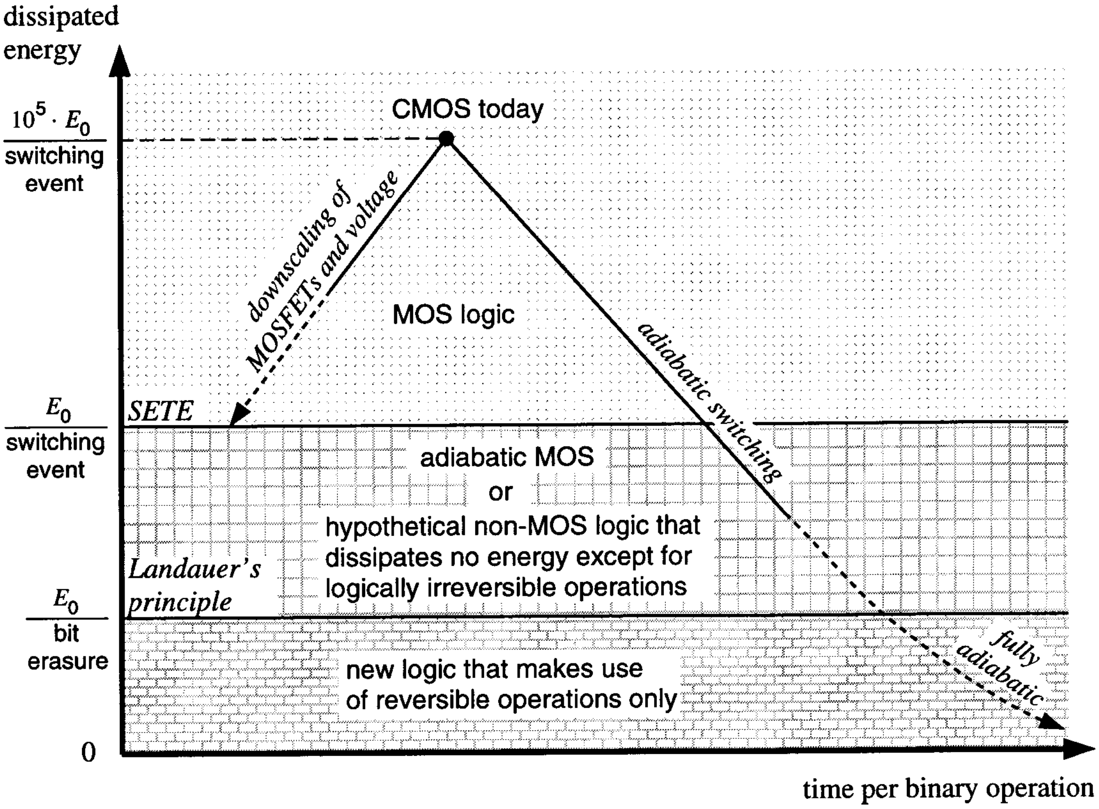

dissipated energy
$10^5 \cdot E_0$ / switching event
E_0 / switching event
E_0 / bit erasure
0
time per binary operation
CMOS today
downscaling of MOSFETs and voltage
MOS logic
adiabatic switching
SETE
adiabatic MOS or hypothetical non-MOS logic that dissipates no energy except for logically irreversible operations
Landauer's principle
new logic that makes use of reversible operations only
fully adiabatic

Figure 4.1: Evolution of energy dissipation limits.

Principle)[3]. If one then is to reduce energy dissipation beyond E_0 per bit of information discarded, one will be forced to compute in a logically reversible manner using an appropriate implementation medium. However, zero energy dissipation can only be reached asymptotically if the time for completing a binary switching event is allowed to go to infinity. Figure 4.1 illustrates the evolution of minimum energy dissipation.

4.2.4 Implications

At this point, the following statements can be made:

- CMOS technology in conjunction with constant-voltage charging and draining of capacitances imposes a E_0-per-switching barrier on minimum energy dissipation.

---

[3] Decreasing T below room temperature is another potential which, however, is not feasible in most practical situations.

- Consequently, from the energy efficiency point of view, CMOS with constant supply voltage is a suboptimal implementation technology since it does not obey Landauer's principle.

- Since with CMOS technology every switching event is accompanied by energy dissipation, the number of those events shall be reduced to the absolute minimum necessary. This motivates the question of a lower bound on switching activity for a given processing task. In section 4.3 this question will be formalized in the context of data statistics.

4.3 Taxonomy for Lower Bound Problem

In this section, the power dissipation bound problem will be formulated as abstract mathematical optimization problem in order to introduce a consistent classification scheme. This taxonomy shall help to comprehend the issue, and allow for categorization of existing approaches to solve the problem.

4.3.1 Problem Formulation

Let the implementation technology be CMOS with fixed process parameters. Then, in the light of data statistics, the lower bound problem (LBP) for low-power digital VLSI is stated as follows:

> *"A digital operation is to be performed on a set of input data items with specified statistical properties in a given amount of time. What is the minimum energy any implementation of this processing task must dissipate?"*

Since the processing time is fixed, minimum energy dissipation is equivalent to minimum average power. Therefore, the terms energy and power may be used interchangeably.

In order to arrive at a mathematical formulation of LBP all parameters that affect power dissipation need to be identified. These parameters are:

1. the digital operation[4], denoted Γ
2. input data statistics, denoted $Pr\{.\}$
3. the processing time[5], denoted T_P
4. the RT-level architecture[6], denoted $A_{|\oplus|}$
5. the binary data representation, denoted $\mathcal{B}\{.\}$
6. the gate-level architecture, denoted $A_{\text{-}\rhd\!\circ\text{-}}$
7. the supply voltage level V_{dd}.

Hence, the *power dissipation function* P may be written as:

$$P \;=\; P(\Gamma, Pr\{.\}, T_P, A_{|\oplus|}, \mathcal{B}\{.\}, A_{\text{-}\rhd\!\circ\text{-}}, V_{dd}) \;. \tag{4.3}$$

The seven variables in this function are mutually dependent and construct an entangled relation for power dissipation.

For the lower bound problem the variables in (4.3) may be divided into two groups:

given input parameters : $\Gamma,\ Pr\{.\},\ T_P$
free optimization variables : $A_{|\oplus|},\ \mathcal{B}\{.\},\ A_{\text{-}\rhd\!\circ\text{-}},\ V_{dd}$.

The three input parameters jointly define a *processing task*, which shall be performed on a digital circuit. Note that input data statistics are an integral part of the design specification for such a circuit.

For a specific problem instance of LBP the input parameters are given. The goal is to find the minimum power dissipation for any combination of the optimization variables:

$$P(A_{|\oplus|}, \mathcal{B}\{.\}, A_{\text{-}\rhd\!\circ\text{-}}, V_{dd}) \longrightarrow \text{Min} \;. \tag{4.4}$$

---

[4] In the present context, the term operation will always refer to a time-discrete digital-valued algorithm.

[5] Without loss of generality it is assumed that T_P is the processing time for *one* input data item

[6] It is assumed that with the RT-level architecture all other high-level implementation options are defined. These options include the data acquisition scheme (synchronous/asynchronous), the data processing scheme (serial/parallel), and the operation frequency which is chosen such as to comply with the required processing time T_P.

Solving this optimization problem provides the lower bound on power dissipation for specific Γ, $Pr\{.\}$, and T_P, and also the settings of the optimization variables that achieve this bound.

The problem is, that in order to solve (4.4), an analytical expression for P is required. For voltage level V_{dd} and processing time T_P the relation to power consumption may be stated explicitly. Γ, $Pr\{.\}$, $A_{|\oplus|}$, $\mathcal{B}\{.\}$, and $A_{-\triangleright\!\circ-}$ are, however, difficult to cast in mathematical formulas. Such formulas not only would need to describe the relation to power dissipation, but also must reflect the interdependence of variables. This makes the analytical treatment of LBP a rather unpromising undertaking and simplification of LBP seems indispensable.

Such simplifications can be attained by regarding some of the optimization variables as input parameters. Fixing different combinations of optimization variables to given values results in a classification of LBP. As will be seen, this classification naturally follows from making certain assumptions on the operation Γ and data statistics $Pr\{.\}$.

4.3.2 Representation of Digital Operations

Any digital operation Γ may be written as

$$o(k) = \Gamma\left(i(k), i(k-1), \ldots, i(k-\infty), o(k-1), \ldots, o(k-\infty)\right) \quad (4.5)$$

where $i(k)$ and $o(k)$ denote the present input and output value, respectively. In order to tackle LBP, a more precise specification of the operation Γ is required. This specification will bear the first simplification of the problem.

FSM representation

The processing functionality of any digital VLSI system, including those that implement DSP algorithms, may be modeled as a finite-state machine (FSM). The most general form of an FSM, i.e. a Mealy automaton, is depicted in figure 4.2(a). I, O, and S denote the set of input, output, and state symbols, respectively. The current state of the system $s(k)$ is stored in some memory element, which in practice may be a set of register cells or a RAM macro-cell.

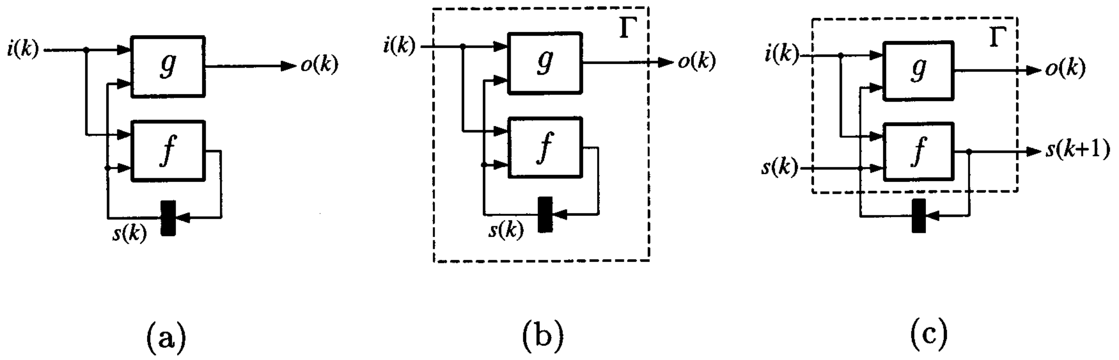


Figure 4.2: General finite-state machine (a), digital operation Γ modeled as FSM (b), and memoryless mapping (c).

Let operation Γ in (4.5) be associated with the general FSM representation as shown in figure 4.2(b). Then, the mathematical specification of Γ involves:

1. a special element $s(0) \in S$, referred to as initial state,
2. a next-state function $f : I \times S \to S$, and
3. an output function $g : I \times S \to O$.

There are two serious difficulties with this representation of Γ that point to the problem of finding an appropriate generic description for a general digital operation:

- The specification of Γ involves the initial state $s(0)$ of the system. This is highly undesirable, since it implies that two otherwise identical processing tasks will correspond to different instances of LBP and in general will have different lower bounds due to deviating initial states. To avoid this peculiarity, $s(0)$ would need to be appended to the list of optimization variables, which complicates the power dissipation function P, and thus the optimization problem (4.4) even further.
- Even for a given initial state $s(0)$, $(f,\, g)$ defines no canonic representation of Γ because different pairs of next-state and output function can yield the same I/O-behavior. Every pair $(f,\, g)$ describing the same operation Γ will represent different instances

of LBP. This is because f and g imply a certain state encoding, and vice versa. In other words, specifying next-state and output function constrains the optimization variable RT-level architecture $A_{|\oplus|}$.

Consequently, a generic model for Γ needs to detach from FSM related concepts. This suggests to separate the memoryless mapping from the memory.

Memoryless mapping

Let the RT-level architecture $A_{|\oplus|}$ for the implementation of Γ be defined. By discarding the feedback of information as shown in figure 4.2(c), Γ can then be modeled as memoryless mapping

$$\Gamma : X \to Y \tag{4.6}$$

where $X = I \times S$, $Y = O \times S$. This implies that the input (output) of the data memory is considered part of the output (input) of Γ. Hence, (4.6) is a generic model for any digital operation Γ excluding all sequential actions involved. Utilizing (4.6) for the solution of LBP, the lower bound on power dissipation will refer to a particular RT-level architecture rather than to the operation itself. In other words, describing Γ as memoryless mapping moves $A_{|\oplus|}$ from the list of optimization variables to the list of input parameters. Obviously, this also circumvents the problem of initial state as described above.

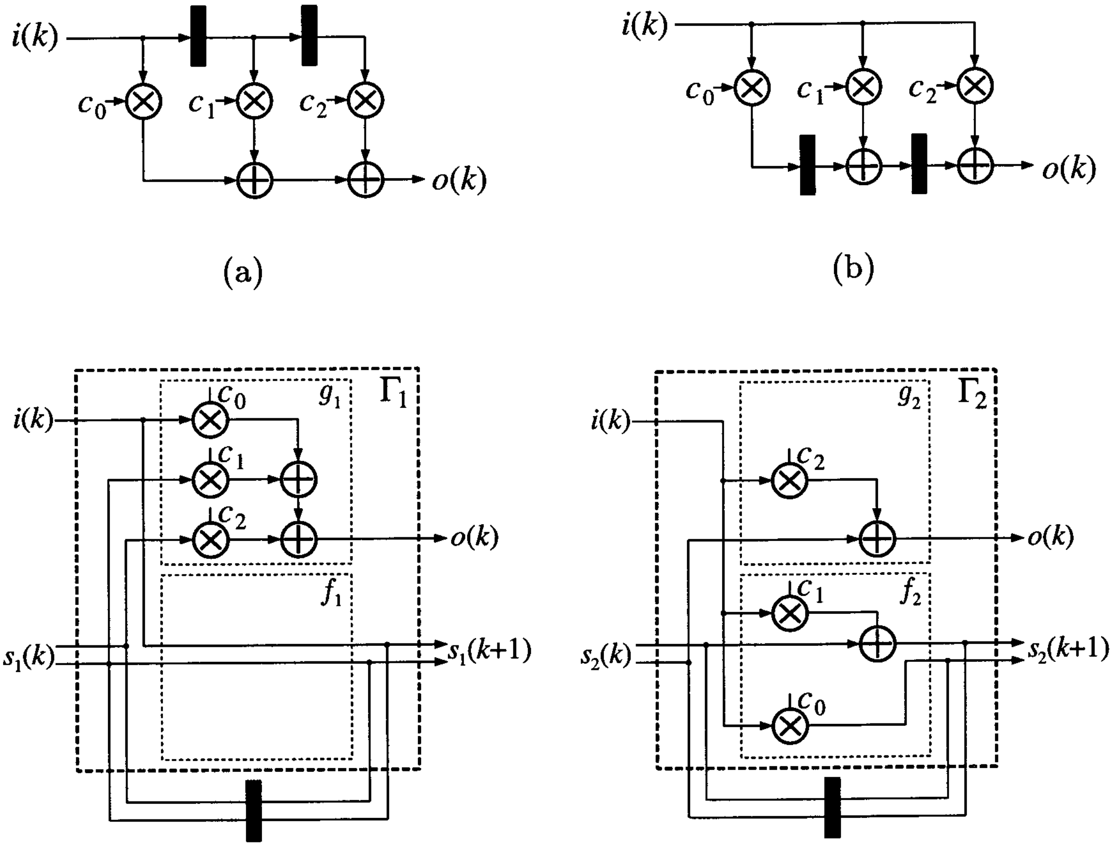
Γ in figure 4.2(c) comprises two memoryless mappings g and f, that share the same domain $I \times S$ but differ in their ranges. The lower bounds of g and f are independent only, if the two functions are to be implemented separately. In this case, the lower bound for Γ can be obtained from a summation over the lower bounds for g and f. Information regarding the statistical properties of the input data, i.e. $Pr\{I \times S\}$, may be obtained from simulating a RT-level description of Γ or from theoretical analysis of the associated FSM representation, see section 3.5.

Example:

Assume the operation Γ to be the linear time-invariant difference equation

$$o(k) \; = \; c_0 \cdot i(k) + c_1 \cdot i(k-1) + c_2 \cdot i(k-2) \tag{4.7}$$

with integer operands. In order to arrive at a memoryless mapping as in (4.6), the RT-level architecture $A_{|\oplus|}$ needs to be defined. Figure 4.3(a) and (b) show two alternative architectures that implement (4.7). The two architectures can be translated to their corresponding FSM representation, see figure 4.3(c) and (d). The resulting sets of state symbols S_1 and S_2, and thus the memoryless mappings Γ_1 and Γ_2, are quite different. This in turn will yield different lower bounds for the two architectures. □



(a) (b)

(c) (d)

Figure 4.3: Two different RTL architectures implementing the same DSP operation (a,b) and the corresponding FSM representations (c,d).

4.3.3 Representation of Data Statistics

Next to operation Γ, data statistics $Pr\{.\}$ is the other important input parameter to LBP. Data statistics may be given at two different levels of abstraction. Each of these levels implies a specific interpretation of the memoryless mapping (4.6), thus giving rise to further classification of the lower bound problem.

Word-level data statistics

One way to specify the statistical properties of the application data is at the word level. In this case, X and Y in $\Gamma : X \rightarrow Y$ must be seen as abstract sets of words, or symbols. Without loss of generality it can be assumed that the symbols are integer numbers, i.e. $X, Y \in \mathbb{Z}$. However, nothing is said about the binary representation of those symbols. Hence, binary representation $\mathcal{B}\{X\}$ remains an optimization variable. The basic question then is, up to which extent can the statistical properties of the application data $Pr\{X\}$ be taken advantage of by means of appropriate coding.

A particularly compact representation of the statistical properties of the input data is enabled via the information-theoretic notion of *entropy*, see appendix B. In section 4.5 this will be elaborated further.

Bit-level data statistics

The alternative way to specify data statistics is at the bit level by means of probability measures as introduced in section 3.3. This implies, that the binary representation for each input symbol is known. In this case, $\mathcal{B}\{X\}$ is an input parameter rather than an optimization variable.

Assuming that the binary representation of input symbols implies the binary representation of output symbols, and that input and output symbols are represented with a fixed number of L and M bits, the memoryless mapping (4.6) becomes

$$\Gamma : \mathcal{B}^L \rightarrow \mathcal{B}^M \; . \tag{4.8}$$

The lower bound problem then is equivalent to finding the minimum power dissipation of any combinational circuit implementing the

Boolean function (4.8). At this point, two optimization variables remain, namely gate-level architecture $A_{\triangleright\!\circ}$ and supply voltage V_{dd}. If either of these is defined as well, the respective other variable is the only degree of freedom left for minimizing power consumption and is constraint by processing time T_P.

4.3.4 Problem Hierarchy

Based on the discussion above a categorization of LBP into four main problem classes is suggested. Each of these problem classes is associated with a special optimization problem as in (4.4). Figure 4.4 shows this classification of the lower bound problem. The four subproblems are denoted according to the number of variables in the optimization problem associated:

LBP-4: This is the most general form of the lower bound problem, where all four optimization variables are available for minimizing power dissipation. No answer to this problem is known. Section 4.4 discusses the requirements a potential solution should comply with.

LBP-3: Here, Γ is given as a memoryless mapping, which defines the RT-level architecture. Since data statistics is specified at the word-level, binary representation of data $\mathcal{B}\{.\}$ remains subject to optimization. LBP-3 corresponds to the problem of minimum power dissipation for data transmission, which can be solved by an information-theoretic approach, see section 4.5.

LBP-2: In this case, the binary data representation $\mathcal{B}\{.\}$ is defined, and the lower bound is subject to gate-level architecture $A_{\triangleright\!\circ}$ and supply voltage V_{dd}. No solution to LBP-2 is known. Section 4.6 reviews related results and discusses their qualification for LBP-2.

LBP-1: This is the most basic version of the lower bound problem, where power dissipation is to be minimized solely due to the supply voltage level. As shown in figure 4.4 there are two paths to LBP-1:

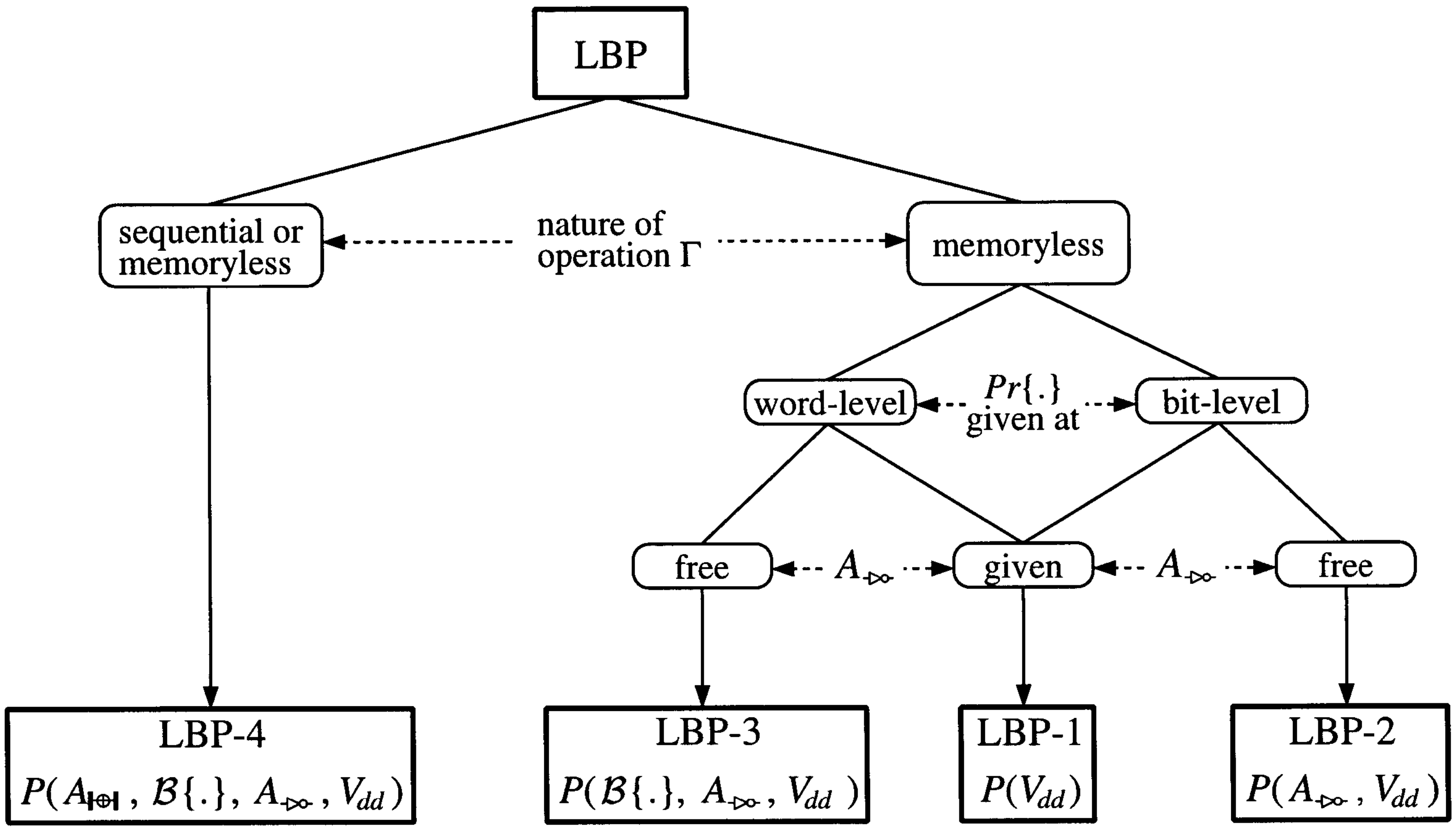

LBP
sequential or memoryless
nature of operation Γ
memoryless
word-level
$Pr\{.\}$ given at
bit-level
free
$A_{\triangleright\circ}$
given
$A_{\triangleright\circ}$
free
LBP-4
$P(A_{|\oplus|}, \mathcal{B}\{.\}, A_{\triangleright\circ}, V_{dd})$
LBP-3
$P(\mathcal{B}\{.\}, A_{\triangleright\circ}, V_{dd})$
LBP-1
$P(V_{dd})$
LBP-2
$P(A_{\triangleright\circ}, V_{dd})$

Figure 4.4: Classification of the lower bound problem LBP (input data statistics $Pr\{.\}$, RT-level architecture $A_{|\oplus|}$, binary data representation $\mathcal{B}\{.\}$, gate-level architecture $A_{\triangleright\circ}$, supply voltage V_{dd}; see pp. 68).

1. The path via bit-level statistics can be thought of as the traditional voltage-scaling problem with given RT- and gate-level architecture. If the delay of the critical path in the system is known for some nominal supply voltage, then the minimum V_{dd} can be calculated such as to meet the processing time requirement [CB95].
2. The path via word-level statistics is mostly of academic interest, since for given binary representation $\mathcal{B}\{.\}$ and gate-level architecture $A_{\triangleright\!\circ}$ the minimum supply voltage does not depend on data statistics and can be calculated as above. However, an information-theoretic approach which applies to LBP-1 has been suggested in [Sha97], and will be discussed in section 4.7.

Two general remarks regarding the proposed classification of LBP are due:

- The four optimization variables $A_{|\oplus|}$, $\mathcal{B}\{.\}$, $A_{\triangleright\!\circ}$, V_{dd}, in that order, correspond to decisions at different stages in the design flow. Violating this order when determining these variables is expected to overly limit the design space, and often is even impractical. For instance, fixing the gate-level architecture $A_{\triangleright\!\circ}$ first, and then optimizing with respect to binary data representation $\mathcal{B}\{.\}$, makes no sense. Therefore, not every combination of optimization variables corresponds to a useful subproblem of LBP. The four subproblems LBP-1, ..., LBP-4 are regarded as the essential ones. In particular, further subclassification within LBP-4 seems not fruitful because the RT-level architecture $A_{|\oplus|}$ is not defined.

- The hardness of the lower bound problem does not vary monotonically with the number of optimization variables. The reason is that the four optimization variables are related in an intricate fashion. For instance, LBP-3 is "easier" to solve than LBP-2. The additional optimization variable $\mathcal{B}\{.\}$ in this case simplifies the problem, see section 4.5. On the other hand, LBP-2 is harder than LBP-1, obviously. In this case the additional degree of freedom provided by $A_{\triangleright\!\circ}$ can only be employed in a subspace of the LBP-3 solution space, and the minimum of the power dissipation function P may in general be more difficult to find in this subspace.

4.4 The General Lower Bound Problem

As pointed out above, LBP-4 is the most general and, presumably, most difficult lower bound problem, with all four optimization variables being subject to variation. The difficulty consists in finding an appropriate power dissipation function that comprises all optimization variables and input parameters, see (4.3). No such power function is currently known. This section provides general guidelines which may be helpful for directing future attempts to tackle the problem.

4.4.1 Tightness of the Lower Bound

Let the processing task be written as vector of input parameters

$$t = (\Gamma, Pr\{.\}, T_P) \ ,$$

and the optimization variables be denoted

$$z = (A_{|\oplus|}, \mathcal{B}\{.\}, A_{\triangleright\!\circ}, V_{dd}) \ .$$

Knowing the analytical expression for the power dissipation function $P(t, z)$, the optimization problem (4.4) can be solved. This not only provides the lower bound on power dissipation for a specific processing task t, but also yields the optimum z, denoted as z^*. With this information, the most energy-efficient implementation of t can be designed.

Forgoing the information on optimum values, a different approach to LBP may be taken. In this case it suffices to find some power dissipation function $\hat{P} = \hat{P}(t)$ that only depends on the three input parameters. Since $\hat{P}$ shall provide a lower bound, there must hold

$$\hat{P}(t) \ \leq \ P(t, z) \ , \quad \forall \ t, z \ . \tag{4.9}$$

Intuitively, it seems less difficult to construct $\hat{P}$ than P. However, in order to be a practical lower bound, there are certain requirements that $\hat{P}$ has to meet. For instance, the naive lower bound $\hat{P} \equiv 0$ complies with (4.9) but is useless because it is not tight for most combinations of input parameters, i.e. not close to the true lower bound provided by minimizing P. In general, a lower bound on power dissipation for some processing task t is of limited practical use, if nothing is known about its quality, i.e. its tightness.

The requirement of tightness can be made more specific in the context of energy efficiency, as discussed next.

4.4.2 Energy Efficiency Rating

Let the power dissipation of some implementation of processing task t be $P_{impl}(t)$. Then, a natural definition for the *energy efficiency* of this implementation of t is

$$\eta(t) \;=\; \frac{P(t, z^*)}{P_{impl}(t)} \; . \tag{4.10}$$

$\eta \leq 1$ with equality only if the implementation at hand is optimum.

In order to rate the energy efficiency according to (4.10) the optimum $A^*_{|\oplus|}$, $\mathcal{B}\{.\}^*$, $A^*_{-\triangleright\!\circ-}$, and V^*_{dd} must be known. This can be circumvented by employing the lower bound $\hat{P}(t)$ instead of the true power dissipation function $P(z, t)$, in order to obtain an estimated efficiency:

$$\hat{\eta}(t) \;=\; \frac{\hat{P}(t)}{P_{impl}(t)} \; . \tag{4.11}$$

As pointed out above, the usefulness of the efficiency estimated in this way hinges on the quality of the lower bound $\hat{P}(t)$. In particular, the call for reliable comparison of efficiency between two implementations of different processing tasks t_1 and t_2, poses the following requirement[7]:

$$\eta(t_1) - \eta(t_2) \;=\; \hat{\eta}(t_1) - \hat{\eta}(t_2) \; . \tag{4.12}$$

---

[7] Requirement (4.12) results from the question: How much closer does $P_{impl}(t_1)$ come to its lower bound than $P_{impl}(t_2)$? A relaxed requirement would be:
$\hat{\eta}(t_1) > \hat{\eta}(t_2)$ if $\eta(t_1) > \eta(t_2)$, and $\hat{\eta}(t_1) < \hat{\eta}(t_2)$ if $\eta(t_1) < \eta(t_2)$.
This can be satisfied by $\hat{P}(t) \;=\; a \cdot P(t, z^*)$ with $a \leq 1$. In fact, however, this is not easier to ensure than (4.14), since it shall hold true for any processing task t.

Substitution of (4.10) and (4.11) yields

$$P_{impl}(t_2)\big[P(t_1, z^*) - \hat{P}(t_1)\big] \;=\; P_{impl}(t_1)\big[P(t_2, z^*) - \hat{P}(t_2)\big] \;. \qquad (4.13)$$

For arbitrary $P_{impl}(t_1)$ and $P_{impl}(t_2)$ this can only be satisfied if

$$\hat{P}(t) \;\equiv\; P(t, z^*) \;, \quad \forall\, t \;. \qquad (4.14)$$

Hence, for a lower bound function $\hat{P}(t)$ to be beneficial for general rating of energy efficiency, it must evaluate to the true minimum power dissipation, i.e. it has to be tight for any processing task t. In this light it is questionable whether a lower bound $\hat{P}(t)$ that satisfies (4.14) is easier to find than the true power dissipation function $P(t, z)$. Since $\hat{P}(t)$ evaluates to the true minimum power, it is expected that knowing such a lower bound also hints at how to design the most energy-efficient implementation of t, even though there is no explicit dependence on the optimization variables $A_{|\oplus|}$, $\mathcal{B}\{.\}$, $A_{\rhd\!\circ}$, and V_{dd}.

4.4.3 Axiomatic Requirements

Besides the tightness demand discussed above, there are other, common-sense requirements a model $\hat{P}(t)$ for minimum power dissipation shall comply with. Following, some of these axiomatic requirements are stated without claim for completeness. Each of these postulates refers to one of the three input parameters Γ, $Pr\{.\}$, or T_P. For the sake of clarity, only the relevant input parameter is included in the mathematical formulation of the axioms.

A-1 If the operation to be performed is the constant function, the minimum power dissipation is zero:

$$\hat{P}(\Gamma(x) = \mathrm{c}) \;\equiv\; 0 \;. \qquad (4.15)$$

This postulate is in contradiction with the thermodynamic standpoint of Landauer's principle, see section 4.2.3, because an operation $\Gamma(x) = \mathrm{c}$ discards information and should therefore be dissipative. However, from a CMOS implementation point of view, the realization of a known constant value implies no switching activity. This, by virtue of (2.2), suggests zero power consumption. The discrepancy between the two viewpoints stems from the fact, that for a practical realization of $\Gamma(x) = \mathrm{c}$ the circumstances of generating the input are disregarded, while in thermodynamics they are part of the system under consideration.

A-2 The minimum power dissipation for the identity mapping is smaller than for any other operation on the same input:

$$\hat{P}(\Gamma(x) = x) \ \leq\ \hat{P}(\Gamma)\ . \tag{4.16}$$

This is because implementing the identity mapping requires only wires for the transmission of data, see section 4.5. Any other operation will need these wires plus appropriate logic to discern between different input symbols.

A-3 The minimum power dissipation for some processing task is not larger than that of its sequential decomposition (chain rule):

$$\hat{P}(\Gamma) \ \leq\ \hat{P}(\Gamma_1) + \hat{P}(\Gamma_2) \quad \text{if} \quad \Gamma(x) = \Gamma_2\,(\Gamma_1(x))\ . \tag{4.17}$$

If the most energy-efficient way to perform operation Γ is by means of two sequential operations, then equality will hold in (4.17). Otherwise, $\hat{P}(\Gamma)$ must be smaller than the sum, because it shall provide a lower bound. Stated differently, the lower bound model should account for sequential decomposition. Similarly, the lower bound model must be conscious of any other architectural transformation that could be employed for power reduction.

A-4 Uncorrelated input symbols with uniform distribution $\mathcal{U}$ imply the largest minimum power consumption for any data statistic:

$$\hat{P}(Pr\{.\}) \ \leq\ \hat{P}(\mathcal{U})\ . \tag{4.18}$$

This intuitively follows from the fact, that random data feature no redundancy that could be employed by clever encoding, i.e. optimization of $\mathcal{B}\{.\}$, to reduce switching activity or to simplify circuitry, see section 4.5.

A-5 Increasing the time allowed to process a certain number of input symbols decreases minimum power dissipation:

$$\hat{P}(T_{P_1}) \;<\; \hat{P}(T_{P_2}) \quad \text{if} \quad T_{P_1} > T_{P_2} \; . \tag{4.19}$$

This is justified by the basic relation for energy dissipation $E = C_L V_{dd}/2$, because the relaxed processing speed can be utilized to reduce the supply voltage V_{dd} or to design slower circuits with smaller capacitive load C_L.

The above axioms shall hold true for the general lower bound problem LBP-4. For the most part, they also apply to the other subproblems, with the exception of A-4. This axiom does not apply if the binary data encoding is fixed, as in LBP-2 and LBP-1, because then redundancy can not be utilized for power reduction. In this case, unfavorable bit-level correlation can imply higher switching activity than a random bit stream, see figure 3.4.

4.4.4 Implications

At this point the following can be concluded:

- For the reliable rating of energy efficiency of known implementations, an estimated lower bound must be identical to the true lower bound on power dissipation, see (4.14). This requirement is valid for the most general lower bound problem LBP-4 and for any of the other subproblems alike.

- It is unclear how requirement (4.14) can be satisfied without knowledge of the optimum implementation. Hence, we conjecture that there is no implementation-independent lower bound on power dissipation.

- Since no explicit power dissipation function is known, the solution of the general lower bound problem LBP-4 is unknown.

4.5 Lower Bound for Data Transmission

This section investigates the lower bound problem LBP-3, where minimum power dissipation is subject to optimization of binary data representation $\mathcal{B}\{.\}$, gate-level architecture $A_{-\triangleright\!\circ-}$, and supply voltage V_{dd}, see section 4.3.4. Because data statistics $Pr\{.\}$ are specified at the word-level and binary data representation $\mathcal{B}\{.\}$ can be chosen arbitrarily, LBP-3 may be viewed as the problem of finding the minimum power dissipation for transmitting data over an information channel. This will be elucidated in section 4.5.1. Subsequent sections then will deal with minimum switching activity in the context of data transmission.

4.5.1 Power Dissipation Bound

Deterministic information channel

As mentioned in section 4.3.3, X and Y in $\Gamma : X \rightarrow Y$ are interpreted as abstract sets of symbols, because data statistics $Pr\{.\}$ are specified at the word-level. This suggests an information-theoretic model for LBP-3 as indicated in figure 4.5(a).

An information source sends out symbols at a rate equal to the inverse of the processing time T_P. The sequence of symbols $\{X_i\}$ has a certain information content, which is specified as *entropy* $H(X)$, see appendix B. Since Γ is ultimately to be implemented with a two-level sensing technology, the abstract input symbols have to be given a binary representation $\mathcal{B}\{X\}$ by means of encoding. The memoryless mapping Γ partitions the domain of input symbols in M disjoint subsets, where M is the number of possible output symbols, see figure 4.5(b). Γ may therefore be seen as memoryless information channel, because every input symbol is mapped to a specific output symbol with probability one [Abr63].

Intuitively, for a given number of input symbols, the implementation of Γ seems more costly the more domains M must be discerned. For instance, if $M = 1$ the output is constant, and according to axiom (4.15) no power needs to be dissipated. However, if $M > 1$, the output diversity will also depend on the probability of input symbols. Obviously, the problem is to find an analytical measure that describes the complexity of Γ subject to input and output statistics. On one hand, this measure shall relate the three input parameters Γ, $Pr\{.\}$, and T_P to the lower bound on power dissipation $\hat{P}$. On the other hand, this measure must be implementation independent, because $\mathcal{B}\{.\}$, $A_{\triangleright\!\circ}$, and V_{dd} are subject to optimization.

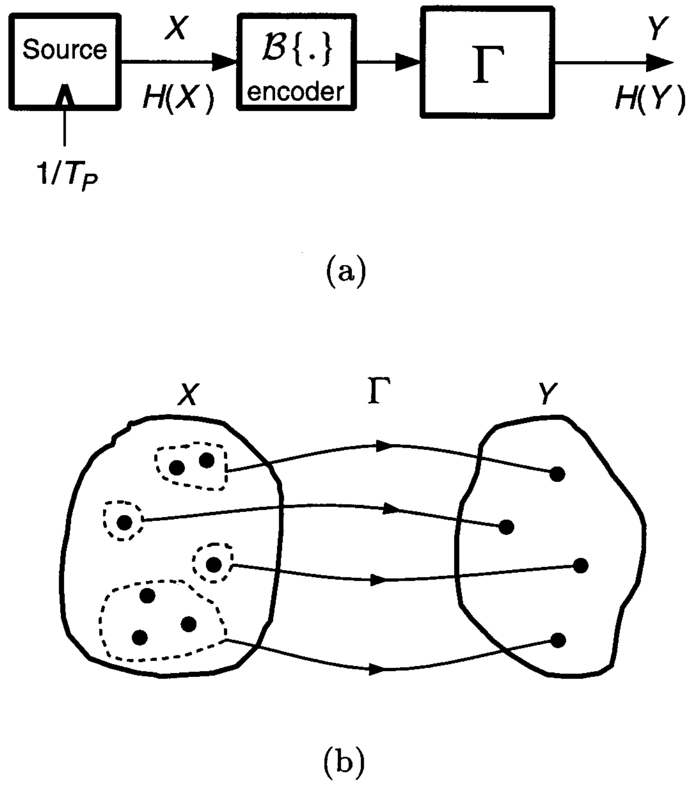


Figure 4.5: Operation Γ as deterministic communication channel: (a) block diagram, (b) partition of input domain by memoryless mapping.

Specification of processing task

One such measure that incorporates the statistics of two sequences is the *mutual information* $I(X;Y)$, see appendix B. Mutual information $I(X;Y)$ can be viewed as the reduction in uncertainty in X due to the knowledge of Y. In the context of operation Γ, this reduction in uncertainty is due to the information transfered from the input X to the output Y. The *information transfer rate* R associated with Γ may therefore be defined as in [Sha97]:

$$R = \frac{I(X;Y)}{T_P} \,. \tag{4.20}$$

Since Γ is a memoryless mapping, it follows that the conditional entropy $H(Y|X) = 0$ and subsequently $I(X;Y) = H(Y)$, see (B.8).

Hence, any transformation Γ requires an information transfer rate of

$$R = \frac{H(Y)}{T_P} \tag{4.21}$$

bits per second [bps]. Since $H(Y)$ depends on the input data statistics and on the operation Γ, information transfer rate R is a function of all three input parameters and thus specifies a processing task $t = (\Gamma, Pr\{.\}, T_P)$. However, R is independent of the implementation of the processing task.

Lower bound function $\hat{P}$

Next, in order to be useful for LBP-3, information transfer rate R has to be associated with power dissipation. One way to do so is to specify the lower bound function $\hat{P}(t)$ depending on R. Intuitively,

$$\hat{P}(t) \propto R = \frac{H(Y)}{T_P} . \tag{4.22}$$

For a specific problem instance, $\hat{P}(t)$ shall be smaller than, or ideally equal to the minimum power dissipation achievable with optimum $\mathcal{B}\{.\}^*$, $A^*_{\rhd\!\circ}$, and V^*_{dd}. However, the problem is how to justify this condition without knowledge of the optimum implementation and the corresponding minimum power consumption. In case of LBP-3 this perplexity may be resolved, because data statistics is specified at the word-level, and the binary data representation $\mathcal{B}\{.\}$ can be chosen arbitrarily for any processing task t.

Optimal encoding

By virtue of (4.21), optimal binary encoding $\mathcal{B}\{.\}^*$ of input symbols X shall imply the lowest information transfer rate R such that the output symbols Y can be uniquely identified by the decoder (not shown in figure 4.5(a)). Therefore, $\mathcal{B}\{.\}^*$ should assign equal codewords to input symbols within the same partition of the input domain, see figure 4.5(b). In this case, no logic to assign appropriate output symbols to different input symbols is required, and the gate-level architecture will be least dissipative. Thus, $A^*_{\rhd\!\circ}$ is a set of wires, used to transmit the required R bps. The optimum supply voltage V_{dd}^* in this case depends on the transmission scheme employed.

From these manifestations of the optimal implementation of processing task t, the true minimum power dissipation can be deduced.

Minimum power dissipation

Assume that every input domain is encoded with a fixed number of $r \geq H(Y)$ bits. This limit on the number of bits is imposed by Shannon's first theorem, i.e. the source coding theorem [Abr63]. Then, the minimum power dissipation of a processing task t follows from (2.2) as:

$$P(t, z^*) \;=\; \frac{1}{2} C_L \frac{\alpha_S^*}{T_P} (V_{dd}^*)^2 \,. \tag{4.23}$$

C_L denotes the load capacitance of a transmission wire, and is assumed to be determined by the CMOS technology. α_S^* is the minimum number of signal transitions per symbol transmitted. The following sections will explore α_S^* as function of the entropy of the sequence of symbols and the number of bits per symbol r. In general, the number of transitions per symbol is related to the switching activity of a binary signal α_x as defined in (3.8) by means of

$$\alpha_x \;=\; \frac{\alpha_S}{r} \,. \tag{4.24}$$

If more than one signal (wire) is used for transmission, (4.24) gives the average switching activity for these signals.

The optimum supply voltage V_{dd}^* in (4.23) depends on the time schedule used for the transmission of the r bits. For serial transmission by means of a single binary signal (wire), the optimum operating voltage V_{dd}^* corresponds to a propagation delay of the transmission wire of T_P/r seconds. For parallel transmission over r wires, V_{dd}^* corresponds to a propagation delay of T_P seconds. Assuming that minimum admissible supply voltage varies proportionally with the inverse of the propagation delay, parallel transmission ideally permits a diminution of minimum power consumption by a factor of $1/r^2$ compared to serial transmission, simply because power varies quadratically with V_{dd}. This, however, is only true within certain limits, because the propagation delay of a transmission line drastically increases as V_{dd} approaches the threshold voltage of the MOS devices involved [CB95].

Practical applicability

Minimum power dissipation $P(t, z^*)$ as given in (4.23) may be seen as the general solution for LBP-3. However, this solution is mainly of academic interest. Its practical implications are limited, because the power consumption inferred does not include the cost of binary encoding. This is due to the abstract specification of the operation Γ, which, together with word-level data statistics, give rise to the information-theoretic model (4.21) for the processing task. Such a model in turn permits to move the burden of computation from the processing unit under consideration to the preceding unit. In order to pinpoint the computational burden, means must be provided to bridge the information-theoretic viewpoint and the implementation that ultimately must use two-level sensing. Abstract specification of data statistics at the word-level, without any relation to the binary implementation, is inappropriate for this matter.

Basically there are two ways to go round the problem above: One is to specify data statistics at the bit-level as in LBP-2, see section 4.6. The other way is to abide word-level data statistics and close the gap to binary implementation by assuming a given gate-level architecture. This is the basic concept of the information-theoretic approach to LBP-1, see section 4.7.

4.5.2 Switching Activity Bounds

This section provides a general result for the minimum number of transitions per symbol α_S^* as it appears in the minimum power dissipation function (4.23). Subsequent sections then will deal with two special encoding schemes.

A general encoding scheme not only will consider symbol probabilities, but may also employ temporal correlation between symbols. Therefore, the information content of some sequence $\{X_i\}$ to be coded must be given as *entropy rate* $\mathcal{H}\left(\{X_i\}\right)$, rather than as entropy $H(X)$, see (B.9). Given the entropy rate, there exist an upper and lower bound on the switching activity of the binary signal that represents the encoded symbols, see theorem 4.1. For brevity, $\mathcal{H}\left(\{X_i\}\right)$ will be written as $\mathcal{H}$. If $\{X_i\}$ is an i.i.d. process, entropy rate $\mathcal{H}$ is equal to the entropy $H(X)$, see (B.10).

Theorem 4.1 ([RSH99b])
If a discrete random process of entropy rate $\mathcal{H}$ is encoded with $r \geq \mathcal{H}$ bits per symbol on average, the expected number of signal transitions per symbol α_S is bounded by

$$h^{-1}\left(\frac{\mathcal{H}}{r}\right) r \;\leq\; \alpha_S \;\leq\; \left(1 - h^{-1}\left(\frac{\mathcal{H}}{r}\right)\right) r \tag{4.25}$$

where $h^{-1}(\cdot) \in [0,1]$ is the inverse of the binary entropy function defined in (B.4) with $h^{-1}(y) = x$ and $x \in [0, 0.5]$.

The above result is universal in a sense that it applies to any encoding scheme, parallel and serial transmission of bits, and to any stationary signal source with arbitrary statistics. Figure 4.6 elucidates (4.25) by showing the lower and upper bound on α_S for different values of r. Both α_S and r are depicted in units of $\mathcal{H}$. If the source is coded with $r = \mathcal{H}$ bits, which corresponds to optimal entropy coding, the binary signal representing the codeword will be completely random and uncorrelated, i.e. $\alpha_x = \alpha_S/r = \alpha_S/\mathcal{H} = 0.5$. By increasing r one obtains more freedom to choose the code, which can be employed to code the source using fewer and fewer, or, if desired, using more and more signal transitions. However, any valid coding scheme must reside within the unshaded region in figure 4.6.

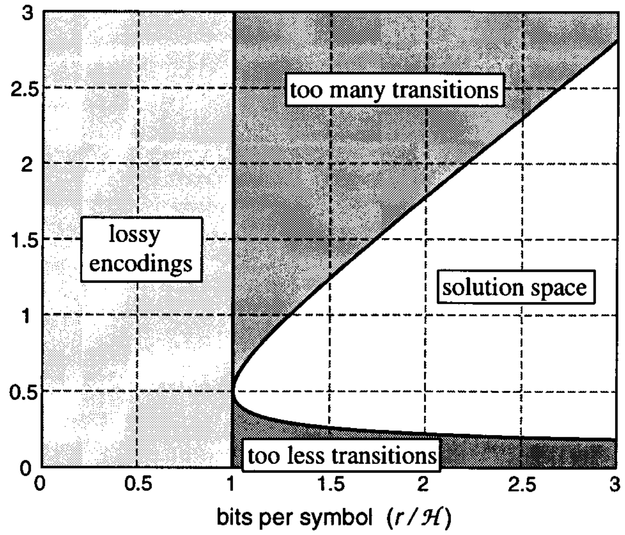


Figure 4.6: Lower and upper bound on the number of transitions per symbol α_S (adapted from [RSH99b]).

Transition signaling

The proof of theorem 4.1 is given in [RSH99b] and runs along the following line: Given the entropy rate of a binary sequence, the number of '1's in this sequence is bounded by means of the binary entropy function (B.4). Then, interpreting logic '1' as binary transition and logic '0' as no transition, the number of transitions is bounded. Such an interpretation is justified because entropy only depends on the probability distribution, but is independent of the actual symbol value.

The explanation of logic values '1' and '0' in the codeword as 'transition' and 'no transition' on the corresponding data bus line is called *transition signaling.* It is a convenient approach to the problem of minimizing the number of transitions in a binary sequence, which then is identical to minimizing the number of '1's in that sequence.

In case of transition signaling, the encoder must generate codewords that indicate a transition by logic '1', and no transition by logic '0'. This requires extra circuitry at the encoder and decoder side in order to convert transition domain codewords to the level domain of the physical data bus, and vice versa. Figure 4.7 compares the principle of ordinary transmission (level signaling) with transition signaling. In both cases, the physical data lines are identical.

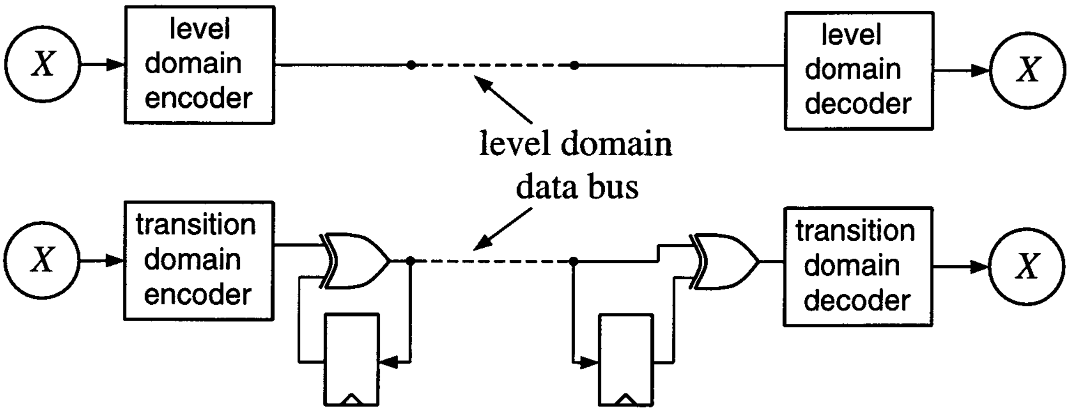


Figure 4.7: Principle of ordinary level signaling (top) and transition signaling (bottom).

4.5.3 Variable-Length Coding

One way to approach the bounds in (4.25) is to use variable-length codes by assigning shorter codewords to more probable symbols and using longer ones for less frequent symbols. An example of such a coding approach is the well-known Huffman code [CT91].

Example:
Assume an i.i.d. process X with five symbols and probability distribution as shown in table 4.2. The entropy of the source is computed as $H(X) = 2.122$ bits. When coding the five symbols with the conventional binary code, the number of transitions per symbol follows from (3.10) for uncorrelated signals as $\alpha_S(bin) = \sum_{i=0}^{2} 2\mu_{x_i}(1 - \mu_{x_i}) = 0.48 + 0.48 + 0.18 = 1.14$. This is within the theoretical bounds of $0.578 \leq \alpha_S \leq 2.422$ computed from (4.25).

Using transition signaling with $r = 3$ bits per symbol reduces the activity to $\alpha_S(trans) = 0.7$. This is within 18% of the theoretical lower bound. The corresponding code is shown in table 4.2 (column four), where ’|’ stands for ’transition’ and ’−’ for ’no transition’.

Constructing a variable length code and using transition signaling as shown in column five (vlc), yields $\alpha_S(vlc) = 0.9 > \alpha_S(trans)$. However, since the variable length code on average only uses $r = 2.2 < 3$ bits per symbol, it only misses the theoretical lower bound by 5%, and hence must be considered more efficient. □

x	p_x	binary	trans.	vlc
0	1/10	000	$\vert\,\vert\,-$	$-\,-\,\vert\,-$
1	2/10	001	$-\,-\,\vert$	$-\,\vert$
2	4/10	010	$-\,-\,-$	$\vert$
3	2/10	011	$-\,\vert\,-$	$-\,-\,-$
4	1/10	100	$\vert\,-\,-$	$-\,-\,\vert\,\vert$

Table 4.2: Probability distribution, conventional binary code, optimal ($K = 1$) transition domain code, and variable-length code for example i.i.d. source with 5 symbols.

Variable codeword lengths as used in the preceding example are opposed to today's predominant synchronous, word-parallel design style. It is always possible to convert a variable-length code to a fixed-length code without increasing switching activity by padding with '$-$' (no transition). The codewords can then be transmitted over a parallel data bus that has as many wires as the longest codeword requires. However, the effectiveness of this new code in terms of reaching the lower activity bound is much worse, since the many redundant bits are not used efficiently. Therefore, it is of interest to investigate the theoretical bounds on switching activity when fixed-length codes are to be used exclusively.

4.5.4 Fixed-Length Coding

In a fixed-length code every codeword consists of r bits when coding each symbol individually. Grouping K symbols into one block and then coding these blocks instead of individual symbols, yields a codeword length of rK bits. As K increases, the bounds in (4.25) may be asymptotically achieved. However, the number of codewords to be stored in the codebook is $|X|^K$. Therefore, the required coding hardware grows exponentially with block length K. Furthermore, real-time encoding requires extra storage capacity for joint evaluation of K symbols.

For fixed-length coding the activity bounds in (4.25) are related to three parameters: the entropy rate $\mathcal{H}$ of the symbol source, the number of bits per symbol r, equivalently to the data bus width, and the coding block length K. These relations are discussed next.

Activity bounds depending on entropy rate

Assume a fixed-length code of r bits to be transmitted over a given physical data link, e.g. an 8-bit bus. Also, let the block length be fixed, e.g. $K = 1$. Then, as $\mathcal{H}$ increases, the difference between lower and upper bound decreases. If $\mathcal{H} = r$, then $\alpha_S = r/2$, and from (4.24) follows the switching activity for each line of the data bus as $\alpha_{x_i} = 1/2$ $(i = 1, \ldots, r)$. Conversely, as $\mathcal{H}$ decreases, the gap between lower and upper bound in (4.25) grows, meaning that there exist binary representations that are highly inefficient from a power consumption point of view.

Consequently, for sources with high redundancy, i.e. highly unequal symbol probabilities and/or memory over many time steps, appropriate encoding promises larger power savings than for low-redundancy sources. This is in accordance with the results of chapter 5, where the standard 2's-complement encoding is shown to waste power in case of speech data, which indeed features a high degree of redundancy.

Activity bounds depending on the number of bits

In practice, the coding block length K will be limited due to system-level constraints such as area or power consumption. Since the coding hardware grows exponentially with block length, K in general must be rather small. In fact, for most practical systems only $K \leq 2$ is feasible.

It is possible to compute the number of bits per symbol necessary to minimize α_S for a given block length. However, for arbitrary sources and distributions the notation for explicit expressions on this number becomes very cumbersome. The following result for uniformly distributed symbols demonstrates the general reasoning.

Corollary 4.1
Let X be an i.i.d. source with uniform distribution. The minimum number of bits per symbol r^* that minimizes the switching activity for block length K is given by

$$r^*(K) = \left\lceil \frac{|X|^K - 1}{K} \right\rceil . \tag{4.26}$$

Coding with $r > r^*$ bits per symbol can not further reduce α_S.

Proof: The minimum number of transitions for fixed-length coding is achieved if there is exactly one codeword with zero transition, and all other codewords have one transition. If the codewords are n bits long, there are at most n codewords with one transition. Hence, at most $n + 1$ symbols can be coded with the minimum number of transitions.

On the other hand, for block length K, there are $|X|^K$ symbols to be coded with $n = rK$ bits. Since each of the corresponding $|X|^K$ codewords shall cause at most one transition, we have

$$|X|^K \leq n+1 = rK+1 .$$

Since for fixed-length encoding r must be an integer number, (4.26) follows. Employing $r > r*$ bits does not reduce α_S because the codewords must differ in at least one bit position. ■

Example:
Assume an i.i.d. source X with five symbols, all being equally likely. From (4.26) one has $r^*(K = 1) = 4$, $r^*(K = 2) = 12$, and $r^*(K = 3) = 42$. Figure 4.8 shows the minimum switching activity as function of r for block lengths $K = 1, 2, 3$, together with the absolute lower bound any coding scheme could achieve. As can be seen, for a given K it makes no sense to increase r beyond r^*, since the number of transitions per symbol does not further decrease. For $r = r^*$, the deviation from the absolute lower bound is minimum, indicating that for a given block length K coding with $r^*(K)$ bits per symbol is most efficient in terms of switching activity. □

Activity bounds depending on block length

In figure 4.8 it can be seen that for a given number r of bits the switching activity α_S does not monotonically decrease with increasing block length K. However, for $K \to \infty$ the absolute lower bound is approached. Therefore, the following question is naturally to ask: For some given r and K, what is the optimal block length K^* such that $K^* \leq K$ and α_S is minimum.

The derivation of a general explicit expression for K^* is impracticable. However, determination of K^* via a computer program is rather straightforward. Such a program, that calculates the minimum switching activity achievable for an i.i.d. source with arbitrary statistics for some given r and K, has been implemented. The following result provides the general reasoning on which this program is based.

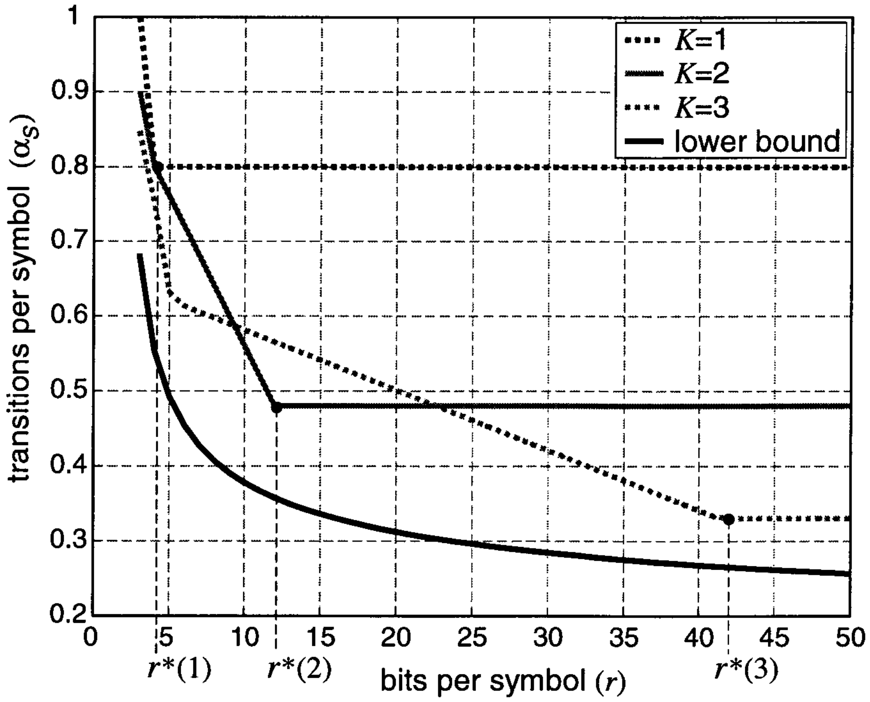


Figure 4.8: Minimum number of transitions per symbol vs. number of bits per symbol for different block length K ($|X| = 5$, uniform distribution).

Corollary 4.2

Let X be an i.i.d. source with uniform distribution. For a given number of r bits per symbol, the minimum number of transitions per symbol α_S^* as function of block length K satisfies

$$\frac{\sum_{t=1}^{t_K^*-1} t\binom{rK}{t}}{K \cdot |X|^K} < \alpha_S^*(K) \le \frac{\sum_{t=1}^{t_K^*} t\binom{rK}{t}}{K \cdot |X|^K} . \tag{4.27}$$

t_K^* is the maximum number of transitions that must be used to code any block of K symbols, and is uniquely given by

$$\sum_{t=0}^{t_K^*-1} \binom{rK}{t} < |X|^K \le \sum_{t=0}^{t_K^*} \binom{rK}{t} . \tag{4.28}$$

Proof: For an n-bit fixed-length code there are exactly $\binom{n}{t}$ codewords with t '1's ($t = 1, \ldots, n$), corresponding to t transitions when using transition signaling. For block length K the codewords associated to blocks of symbols are $n = rK$ bits long. For minimum switching activity codewords are assigned in increasing order with respect to the number of transitions t. Since $|X|^K$ codewords are required, (4.28) follows. Furthermore, the $\binom{n}{t}$ codewords with t transitions each, cause together $t\binom{n}{t}$ transitions. Since all codewords are equally likely by proposition, and α_S^* is the number of transitions per individual symbol, (4.27) follows. ■

The bounds provided in (4.27) are rather loose. However, the expression for tighter bounds, and the relation of minimum switching activity to block length K for sources with arbitrary statistics become even more entangled. The legal ranges of $\alpha_S^*(K)$ and $\alpha_S^*(K+1)$ imposed by (4.27) in general overlap significantly. If $\alpha_S^*(K)$ comes very close to its lower bound and $\alpha_S^*(K+1)$ does not, then non-monotonic behavior of minimum switching activity with respect to block length K is observed.

Example:
Assume again an i.i.d. source X with five symbols, all being equally likely. For this source, figure 4.9 shows the minimum switching activity as function of block length K for $r = 3, \ldots, 7$ bits per symbol. As can be seen, for a given r the minimum activity achievable does not always decrease with increasing block length K. For instance, let the data bus width be restricted to $r = 5$ bits (two redundant bit lines are used). If the current optimal coding employs a block length of $K = 3$, then it is useless to increase the block length by one only, since with $K = 4$ switching activity can not be reduced further. □

4.5.5 Implications

As pointed out previously, the solution of LBP-3 provided in section 4.5.1 is of limited practical utility, because it applies to data transmission only. Moreover, the cost of encoding, which is applied in order to achieve minimum power consumption on the transmitting wires, is not included in the analysis. The application of complex coding schemes is therefore confined to high-capacitance data links, e.g. off-chip buses or highly loaded memory interfaces. In this case switching activity savings attained by appropriate encoding can

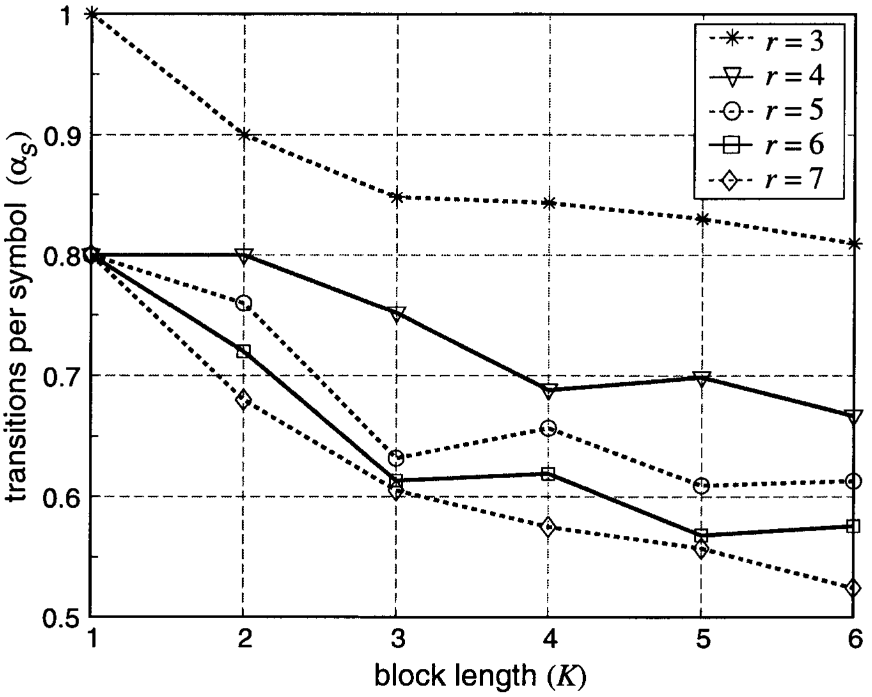


Figure 4.9: Minimum number of transitions per symbol vs. block length K for different number of bits per symbol r ($|X| = 5$, uniform distribution).

justify the coding overhead [BMM+00].

From the examination of the theoretical activity bounds (4.25) the following general conclusions regarding data encoding for low power dissipation may be drawn:

- For sources with high redundancy, i.e. highly unequal symbol probabilities and/or memory over many time steps, appropriate encoding promises larger power savings than for low-redundancy sources.
- For a fixed number of bits per symbol r, minimum switching activity does not monotonically decrease with ascending block length K.
- In practice, one will often be restricted to fixed-length codes and a coding block length of $K \lesssim 2$. In this case there exists a limit on the number of bits per symbol r, beyond which no activity reduction can be achieved.

4.6 Lower Bound in Boolean Optimization

4.6.1 Problem Description

According to figure 4.4, in LBP-2 gate-level architecture $A_{-\triangleright\circ-}$ and supply voltage V_{dd} are optimization variables, while RT-level architecture $A_{|\oplus|}$ and bit-level data statistics $\mathcal{B}\{.\}$ are known input parameters. No general solution for this problem is known. However, some comments on the problem's disposition can be formulated.

Assuming fixed-length encoded inputs and outputs, LBP-2 corresponds to finding the most efficient combinational circuit that implements a specific Boolean function $\Gamma : \mathcal{B}^L \to \mathcal{B}^M$. The condition for maximum delay of the critical path in this circuit is given by the processing time T_P. In this sense, LBP-2 is reminiscent of the ordinary synthesis problem, where a gate-level netlist is to be constructed from an abstract functional description. The peculiarities of the lower bound problem may be elucidated by opposing LBP-2 and the synthesis problem.

- In ordinary synthesis, the supply voltage V_{dd} is fixed and every gate has a known delay according to this nominal voltage. For LBP-2 V_{dd} is an optimization variable and can not be chosen independently of the gate-level architecture $A_{-\triangleright\circ-}$, because the two are linked by the delay of the critical path. However, even if V_{dd} is treated as given input parameter, the lower bound problem differs from the synthesis problem in the following points.
- Unconstrained synthesis may be performed in two consecutive steps. First, the Boolean function is minimized with respect to some generic representation. Subsequently, this representation is mapped to a set of technology-dependent logic gates. Such successive execution simplifies the synthesis problem, but is inadequate for matters of optimum implementation. In LBP-2 Boolean minimization and technology mapping must be considered jointly with respect to power dissipation, as is done in timing-driven synthesis where power cost is replaced by the delay of the critical path. However, for LBP-2 there is an additional parameter, namely data statistics.

The problem may be simplified if mapping is confined to a small subset of generic gates that form a complete basis for the realization of Boolean functions, e.g. inverter and two-input AND and OR. Even so, LBP-2 differs from the synthesis problem in the following respect.

- For synthesis, the gate-equivalent count typically is a fair measure for circuit area. Minimizing this number in general is assumed to also reduce power consumption. For minimum power dissipation in regard of data statistics this conclusion is not justified. If very few input patterns have a very high probability of occurrence, a larger circuit with a high degree of redundancy could be more efficient by means of *precomputation* [AMD$^+$94]. In this case the redundant logic is used to identify and evaluate the most frequent input patterns by a small, energy-efficient subcircuit. The larger, more dissipative main circuit then would only be enabled upon a miss of the subcircuit.

The lower bound problem LBP-2 becomes even more intricate if variable-length codes are permitted for the binary representation $\mathcal{B}\{.\}$. Thus, a universal power dissipation model should include this potential as well, even if it is rarely used in practice because it impedes synchronous word-parallel processing. From a theoretical point of view such coding may indeed seem reasonable as it naturally commences circuit activity only as needed. This has also been noted in [Ger96], where a computational coding theory in generalization of Boolean minimization algorithms was postulated.

4.6.2 Related Efforts

Although LBP-2 has, to all appearances, formally not been attacked by the VLSI research community, there is previous work on RT-level power estimation that relates to the lower bound problem. Information-theoretic concepts have been employed for RT-level power estimation in two ways.

Complexity analysis of Boolean functions. The entropy of the output of some Boolean function, assuming that all input combinations are equally likely, is considered a measure for the area-complexity of this function. Under this hypothesis an area model which is exponential in the number of inputs was derived in [CA90]. For most practical circuits this model is not very accurate and even for reasonable number of inputs dramatically overestimates the gate count. This model was improved for single-output Boolean functions [NN96a] and multi-output functions [NN99] based on the notion of *cube complexity* .

Estimation of switching activity. The other use of entropy for power estimation is to predict the average switching activity in a circuit that implements the Boolean function at hand. In [MMP96, NN96b] formulas for the average entropy per net $\overline{H}$ as a function of average input and output entropy were developed. This requires simplifying assumptions on the logic structure of the circuit, as well as on the correlation of signals. Average switching activity in the circuit may then be approximated as

$$\alpha_x \approx \overline{H}/2 \ . \tag{4.29}$$

This assumes temporally uncorrelated signals, see figure B.1.

Combining the above results, relative power dissipation can be estimated as the product of circuit area and average switching activity.

Direct application of the above approach to LBP-2 is not possible for two reasons. First, for both complexity analysis and activity estimation strong assumptions on input data statistics have to be made. However, the specific input statistics $Pr\{.\}$ are a crucial input parameter, subject to which minimum dissipation should vary. Second, isolated analysis of Boolean function complexity and switching activity might be acceptable for relative power estimation, but it is not when searching for the absolute lower limit of dissipation. In this case the gate-level structure must account for the given signal statistics.

4.6.3 Implications

In the absence of an analytical power dissipation model $P(\Gamma, Pr\{.\}, T_P, A_{|\oplus|}, \mathcal{B}\{.\}, A_{\rhd\!\circ}, V_{dd})$ it is not clear how to tackle LBP-2 nor the simplified version with fixed V_{dd}. Furthermore, the lower bound function $\hat{P}(\Gamma, Pr\{.\}, T_P, A_{|\oplus|}, \mathcal{B}\{.\})$ again should obey the tightness requirement (4.14). It is not apparent how such $\hat{P}$ could be formulated without reference to the optimal solution $A^*_{\rhd\!\circ}$, V^*_{dd}.

4.7 Lower Bound on Supply Voltage

In section 4.5 operation Γ has been associated with a deterministic information channel, such as to infer the minimum power dissipation for data transmission. The actual cost of data processing was not included, however. There has also been an attempt to associate operation Γ with a noisy information channel [Sha97], in order to include data processing power. The lower bound on dissipation so established refers to one particular implementation of the processing task at hand. That is, $A_{|\oplus|}$, $A_{\rhd\!\circ}$, and $\mathcal{B}\{.\}$ are given input parameters and V_{dd} remains the sole optimization variable, as in LBP-1.

This section first will present the information-theoretic approach from [Sha97] to LBP-1, and then discuss its limitations.

4.7.1 Information-theoretic Results

Noisy communication channel

Let $\Gamma : X \rightarrow Y$ denote a deterministic channel as in section 4.5.1, but this time with noise N imposed on the channel output such that

$$Y' \;=\; Y + N \;=\; \Gamma(X) + N \quad . \tag{4.30}$$

In this case $H(Y'|X) > 0$ and is a function of the probability distributions of X and N. The noise imposed on the channel manifests itself in the fact, that for a specific channel input symbol different output symbols can appear, each of which has its own probability.

The *channel capacity* C is defined as the maximum mutual information $I(X;Y')$ or, equivalently, the minimum conditional entropy $H(Y'|Y)$ over all possible distributions of X [Abr63]. For a given noise power spectrum, capacity C characterizes the channel's ability of transferring information and is independent of the input statistics. In [Sha48] it was shown that the capacity of a channel bandlimited to frequency W is

$$C \;=\; \int_0^W \log[1 + \mathrm{SNR}(f)]df \tag{4.31}$$

where C is in bps. $\mathrm{SNR}(f)$ denotes the signal-to-noise ratio which, for flat signal and noise power spectra, is given by the ratio of signal and noise variance:

$$\mathrm{SNR}(f) = \frac{\sigma_X^2}{\sigma_N^2} . \tag{4.32}$$

Power dissipation bound

The joint source-channel coding theorem [Sha48] states, that an information transfer rate R over a noisy channel is achievable, with the probability of error approaching zero, as long as

$$R < C . \tag{4.33}$$

Substitution of (4.31) and (4.32) into (4.33) yields the following constraint on signal power for reliable transmission:

$$\sigma_X^2 > (2^{R/W} - 1)\sigma_N^2 . \tag{4.34}$$

From (4.34) immediately follows a lower bound on the power dissipation of the channel.

Theorem 4.2 ([Sha97])

For a channel bandlimited to W Hz, with flat signal and noise power spectrum, and a desired information transfer rate of R bps, the lower bound on power dissipation is given by

$$P_{min} > P\left((2^{R/W} - 1)\sigma_N^2\right) \tag{4.35}$$

where $P(\sigma_X^2)$ is a linear monotonically increasing function relating the signal power σ_X^2 to power dissipation, and σ_N^2 is the noise power.

At first, the lower bound (4.35) is independent of the implementation technology and does not refer to the problem of minimum power consumption for digital VLSI processing. In [Sha97] an attempt was made to apply theorem 4.2 to CMOS VLSI by making a number of simplifying assumptions, which will be discussed next.

4.7.2 Application to VLSI Processing

Figure 4.10(a) shows the information-theoretic model of a processing task realized in a noisy implementation medium. The architecture implementing operation Γ is viewed as noisy bandlimited channel for which theorem 4.2 must apply. By identifying all parameters in theorem 4.2 with characteristics from LBP, a relation between (4.35) and minimum dissipation of some processing task can be established based on the assumptions made in [Sha97].

Information transfer rate. Let R be defined as in section 4.5.1, $R = H(Y)/T_P$. Since operation Γ, input data statistics $Pr\{.\}$, and processing time T_P determine R, the processing task under consideration is implicitly included in the power dissipation bound (4.35).

Noise power. A specific implementation of Γ is associated with a certain noise power σ_N^2. It is assumed that a noise model is known, from which σ_N^2 can be deduced for a given gate-level architecture $A_{-\!\triangleright\!\circ\!-}$ [Sha97]. This model shall lump all noise components of the circuit into a single noise source at its output, see figure 4.10(a).

Bandwidth. According to the sampling theorem, a channel bandlimited to W Hz can be used a maximum of $2W$ times per second without interference of the transmit pulses. In [Sha97] it was shown, that this maximum possible transmission rate indeed achieves the lowest power dissipation for a given information transfer rate R. Thus, for the delay T_d of the critical path of the given implementation $A_{-\!\triangleright\!\circ\!-}$ must hold

$$T_d = \frac{1}{2W} . \tag{4.36}$$

This is reminiscent of a dual-edge-triggered clocking strategy for the implementation with $W = f_{clk} = f_{ci}/2$, see section 2.1.

Power function. Let the transmit pulse have ideal square-wave shape of amplitude equal to the supply voltage V_{dd} such that

$$\sigma_X^2 = V_{dd}^2/4 . \tag{4.37}$$

Assuming that every channel use induces a signal transition, the power dissipation function $P(\sigma_X^2)$ for CMOS follows from (2.2) as

$$P(\sigma_X^2) \;=\; 4C_L W \sigma_X^2 \;. \qquad (4.38)$$

Here C_L denotes the average load capacitance associated with the gate-level architecture $A_{\rhd\!\circ}$ at hand. (2.2) obeys the requirement of linearity and monotony of $P(\sigma_X^2)$ posed in theorem 4.2.

Relation to LBP-1

The assumed knowledge of $A_{\rhd\!\circ}$ for derivation of σ_N^2 implies that the binary encoding of input data $\mathcal{B}\{.\}$ is known. Furthermore, the

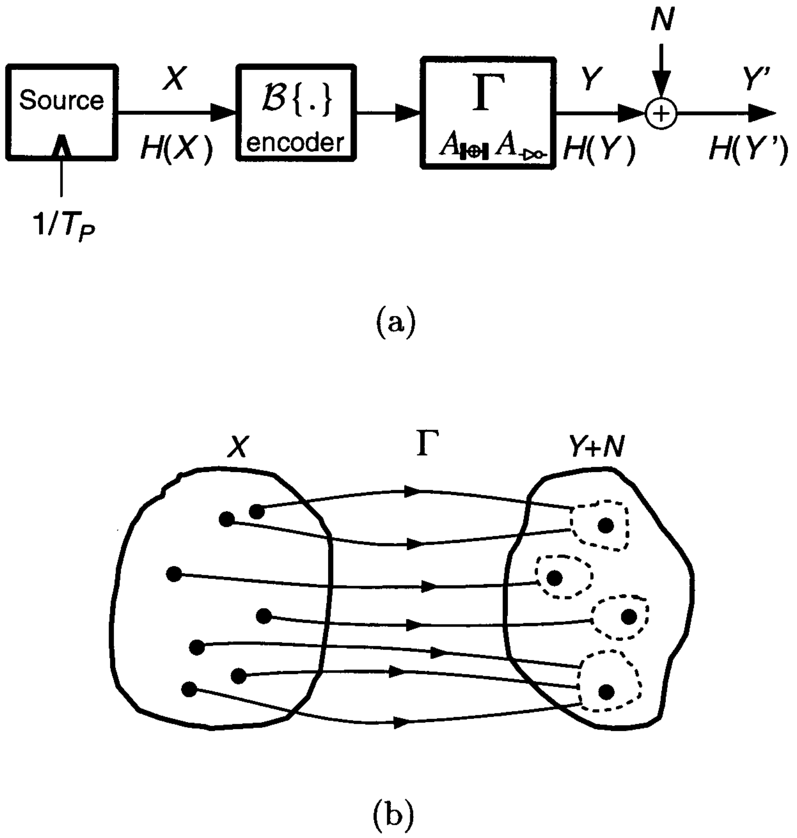


Figure 4.10: Processing task in a noisy implementation media: (a) block diagram, (b) memoryless mapping Γ blurred by noise (adapted from [Sha97]).

RT-level architecture $A_{|\oplus|}$ is given, since Γ is a memoryless mapping. Hence, the only optimization variable left for achieving minimum power consumption is the supply voltage V_{dd}, which complies with LBP-1. The lower bound function $\hat{P}(t)$, see section 4.4.1, follows from application of (4.38) to theorem 4.2 as

$$\hat{P}(t) \;>\; 4C_L W(2^{R/W} - 1)\sigma_N^2 \tag{4.39}$$

with $t = (\Gamma, Pr\{.\}, T_P, A_{|\oplus|}, \mathcal{B}\{.\}, A_{\rhd\!\circ})$. The derivation of $\hat{P}(t)$ also provides the true minimum power dissipation $P(t, z^*)$, because the optimal supply voltage V_{dd}^* follows directly from substituting (4.37) into (4.34). This fact reinforces the conjecture made in the context of energy efficiency rating, that there is no implementation-independent lower bound $\hat{P}(t)$ which is tight to the true lower bound $P(t, z^*)$.

Figure 4.11 summarizes the information-theoretic approach in the context of LBP-1. In the theory by [Sha97], V_{dd}^* corresponds to the minimum signal power that permits unique decoding of symbols from the noisy output Y'. This is symbolized in figure 4.10(b), where the dashed fields around the true output symbols may not overlap to allow unambiguous decoding. Thus, operation with optimal V_{dd}^* changes the channel characteristic to noiseless transmission.

4.7.3 Limitations

In order to arrive at the lower bound (4.39) a number of simplifications were introduced by [Sha97]. Critical examination of these simplifications yields the following limitations of the theory.

- The channel capacity C, associated with an implementation, depends on the noise power σ_N^2. However, there exists no consistent approach to derive σ_N^2 for a given gate-level netlist. Such an approach would need to refer any noise to a single source at the output of the circuit, see figure 4.10(a).

- In CMOS circuits, noise is mainly due to signal switching that induces voltage fluctuations (ground and power bounce) by means of series impedance in common power supply lines. Waveform and amplitude of this noise voltage depend on the supply voltage V_{dd} [KAK99]. One deficiency of the information-theoretic approach is the assumed independence of noise power

σ_N^2 and signal power σ_X^2, which is equivalent to V_{dd}. Furthermore, switching noise is highly periodical, whereas a flat signal power spectrum (white noise) was assumed.

Similarly, the theory neglects the dependence of bandwidth W on supply voltage V_{dd}, which in practice are linked via the delay of the critical path, see (4.36).

Taking the dependencies of σ_N^2 and W on V_{dd} together yields an entangled relation between channel capacity C and supply voltage V_{dd}. Thus, (4.34) is an implicit relation for signal power σ_X^2 and the inference of theorem 4.2 is not justified for CMOS implementation.

- With the assumption that every channel use induces a signal transition, i.e. $\alpha = 1$, (4.38) actually sets an upper limit on power dissipation as function of signal power, and thus, does not qualify for the application to theorem 4.2. Since no useful lower bound on the number of transitions per channel use exists, average switching activity in the circuit under consideration could act as a makeshift.

- Even if all of the above issues could be resolved, the information-theoretic approach proposed has limited practical significance because it predicts minimum power dissipation and supply voltage for a given gate-level implementation of some processing task. However, the same information can readily be obtained by means of gate-level power estimation and an ordinary voltage-scaling approach.

- The cost of coding is not included in the analysis. Therefore, by means of optimal encoding as discussed in section 4.5.1, the implementation architecture also could be a set of mere wires. In this case LBP-1 is equivalent to LBP-3.

This rather long list of limitations indicates that the solution for LBP-1 can not be derived from straightforward application of Shannon's theory of information, as was attempted in [Sha97]. This fact alone suggests that the general lower bound problem LBP-4 can not be solved this way either. The following section provides a formal reasoning for this conclusion.

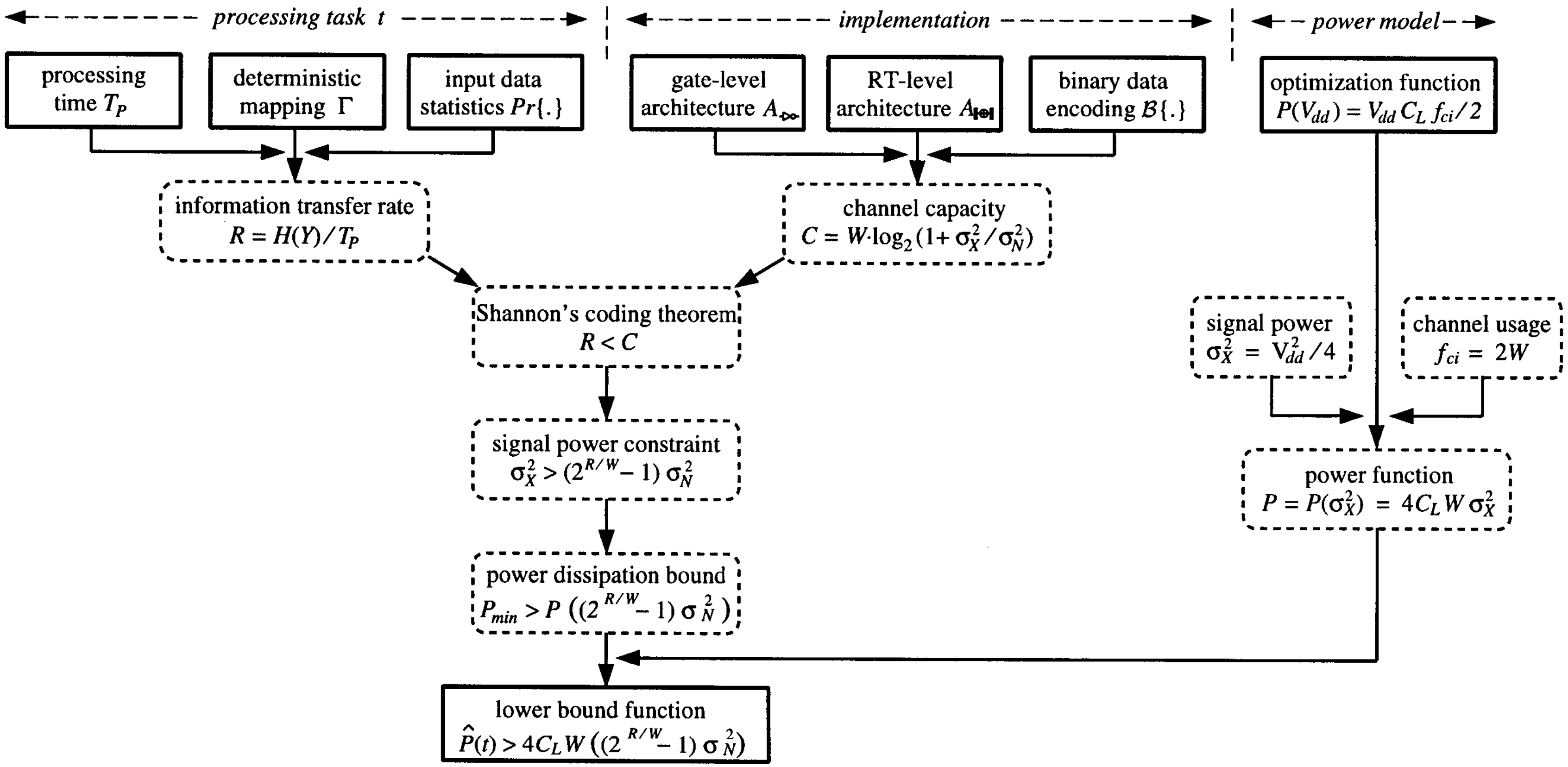

processing task t
implementation
power model
processing time T_P
deterministic mapping Γ
input data statistics $Pr\{.\}$
gate-level architecture A
RT-level architecture A
binary data encoding $\mathcal{B}\{.\}$
optimization function $P(V_{dd}) = V_{dd} C_L f_{ci}/2$
information transfer rate $R = H(Y)/T_P$
channel capacity $C = W \cdot \log_2(1 + \sigma_X^2/\sigma_N^2)$
Shannon's coding theorem $R < C$
signal power $\sigma_X^2 = V_{dd}^2/4$
channel usage $f_{ci} = 2W$
signal power constraint $\sigma_X^2 > (2^{R/W} - 1)\,\sigma_N^2$
power function $P = P(\sigma_X^2) = 4 C_L W \sigma_X^2$
power dissipation bound $P_{min} > P\,((2^{R/W} - 1)\,\sigma_N^2)$
lower bound function $\hat{P}(t) > 4 C_L W\,((2^{R/W} - 1)\,\sigma_N^2)$

Figure 4.11: Information-theoretic approach [Sha97] to minimum power dissipation (dashed boxes) in the framework of LBP-1 (solid boxes). For a list of debatable assumptions made in [Sha97] see section 4.7.3.

4.8 Information Theory vs. Low Power VLSI

Similarly to [Sha97], also the work presented in this chapter initially was inspired of an alleged analogy between the information-theoretic transmission bound problem and LBP as formulated in section 4.3.1. However, our point of view has changed in the course of the research performed. This section provides the rationale behind this change in our viewpoint.

4.8.1 Digest of Information-theoretic Results

Source coding

Let the information rate R of some code be defined as the number of bits of information per code symbol, where n code symbols form one codeword. Shannon's first theorem, i.e. the source coding theorem states that a source emitting H bits of information per source symbol can be represented by a code of information rate $R > H$.

Channel coding

Shannon's second theorem, i.e. the channel coding theorem states that reliable transmission over a channel with capacity C is possible, as long as the information rate R of the code is below the channel capacity: $R < C$. Reliable transmission means that the codewords can be decoded at the receiver side of the channel with an arbitrary small probability of error.

Joint source-channel coding

The joint source-channel coding theorem then combines the above two theorems and shows that reliable transmission is possible as long as the entropy of the source is below the channel capacity: $H < C$. This means, that a two-stage coding process for separately designing most efficient source and channel codes, is as efficient as considering the two problems together [CT91].

4.8.2 Problem Comparison

The joint source-channel coding theorem may also be viewed as answer to the question of the minimum channel capacity C required for reliable transmission of source symbols with entropy H. This in fact is exactly the approach of [Sha97], where it was noted that *"the design of a digital system is akin to the selection or development of an appropriate communication network topology with sufficient capacity C".*

In the following, we want to make this comparison more explicit. This requires an approach that does not comply with the usual interpretation of information theory [Sha48, Abr63, CT91]. Table 4.3 summarizes the comparison of the transmission bound problem of information theory (IT) and the dissipation bound problem of low-power VLSI (LP).

Given problem instance. In either case, a specific task is to be performed for a given set of parameters t.

IT: Symbols emitted by a source with information content H are to be transmitted over a noisy communication channel.

LP: A digital operation Γ is to be performed on a set of data items with statistics $Pr\{.\}$ in a given amount of time T_P.

Minimum cost. Both the transmission and processing task are to be performed with regard to a specific cost measure, which shall be minimized.

IT: The mathematical model for data transmission includes no explicit cost measure. Here, cost is assumed to be the channel capacity which must be provided for reliable transmission[8].

LP: The processing task shall be performed using the least amount of energy. Since processing time T_P is fixed, this is synonymous to minimum power dissipation P.

---

[8] In practice, the cost of transmission usually is regarded as the number of channel uses per symbol necessary for reliable communication, since this determines the bandwidth needed. The number of channel uses per symbol corresponds to the codeword length n, which in the present interpretation is part of the optimization variable.

Optimization variables. The minimum cost requirement shall be met by means of a set of optimization variables.

IT: The required channel capacity can be minimized through appropriate encoding of blocks of source symbols. This includes the choice of codeword length n.

LP: Power dissipation is minimized by choosing an appropriate RT-level architecture $A_{|\oplus|}$, binary data representation $\mathcal{B}\{.\}$, gate-level architecture $A_{-\triangleright\!\circ-}$, and supply voltage V_{dd}.

Cost function. A certain relation between the cost measure and the other terms involved is assumed in either theory.

IT: Channel capacity C by definition is independent on how the channel is used, i.e. is independent of the data encoding. With optimum encoding, the required capacity C solely depends on the entropy H of the source.

LP: The indispensable power dissipation depends both on the processing task to be performed as well as on the actual implementation of this task.

Lower Bound. The aim is to derive a lower bound on the respective cost measure.

IT: The joint source-channel theorem gives a lower bound on channel capacity C necessary for reliable transmission. The proof employs a random-coding argument and does not provide the minimizing encoding.

LP: The general lower bound on power dissipation, i.e. the solution to LBP-4 is unknown.

4.8.3 Implications

In information theory the cost measure (channel capacity) is independent of the optimization variable (encoding) by definition. This is justified because in practice the channel and the encoder/decoder units can be implemented separately. Hence, the lower bound for

information theory	low-power VLSI
`problem instance`	
$t = (H)$	$t = (\Gamma, Pr\{.\}, T_P)$
`minimum cost`	
$C \rightarrow$ Min	$P \rightarrow$ Min
`optimization variables`	
$z = (\mathcal{B}\{.\})$	$z = (A_{\vert\oplus\vert}, \mathcal{B}\{.\}, A_{-\triangleright\!\circ-}, V_{dd})$
`cost function`	
$C = C(t)$	$P = P(t, z)$
`lower bound`	
$C > H$	?

Table 4.3: Comparison of information-theoretic transmission and VLSI power dissipation bound problem.

the cost measure could be derived without knowledge of the optimum encoding.

In low-power VLSI, on the other hand, the cost measure (power) depends on the optimization variable (hardware implementation). This dependency of power dissipation on $A_{|\oplus|}$, $\mathcal{B}\{.\}$, $A_{-\triangleright\!\circ-}$, and V_{dd} certainly agrees with the physical reality that we wish to model. Using a simplified cost function $\hat{P}(t)$ for stating the lower bound does of course not decouple power dissipation and implementation in practice. Nevertheless, we want this simplified model to agree with the physical world, and hence are faced with requirement (4.14) for useful rating of energy efficiency:

$$\hat{P}(t) \equiv P(t, z^*) , \quad \forall \, t .$$

Consequently, we conclude that the power dissipation bound problem is different in nature from the information-theoretic transmission bound problem. Only the special case LBP-3 can be solved in analogy because power dissipation associated with optimum encoding $\mathcal{B}\{.\}^*$ is not included in the cost function, see section 4.5.1.

4.9 Summary

First, the energy dissipation limits of todays MOS transistor based VLSI technology have been discussed in the light of existing thermodynamic bounds. It was concluded that, from the energy efficiency point of view, CMOS technology in conjunction with constant-voltage charging and draining of capacitances is a suboptimal implementation medium.

Then, a new view of minimum power dissipation in the context of data statistics has been established by systematically combining the various parameters involved in the low-power VLSI design process. This approach for the first time enables a classification of the lower bound problem. Two of the resulting subproblems, i.e. minimum power dissipation for data transmission (LBP-3) and lower bound on supply voltage (LBP-1), can be solved in principle. No solution for the most general lower bound problem LBP-4 and the Boolean optimization problem LBP-2 exist due to the lack of appropriate analytical models for power dissipation.

However, the finding of an analytical power dissipation function $P = P(\Gamma, Pr\{.\}, T_P, A_{|\oplus|}, \mathcal{B}\{.\}, A_{\rhd\!\circ}, V_{dd})$ that universally applies to any processing task, indeed would revolutionize low-power digital design. Depending on which subproblem of LBP this model applies to, low-power design at the corresponding level of abstraction would become an automated process. On the other hand, we have shown that an implementation-independent lower bound must be tight to the true minimum dissipation, because a reliable rating of energy efficiency for known implementations is otherwise impossible. Therefore, we conclude that there is no useful implementation-independent lower bound on power dissipation.

An existing information-theoretic approach to minimum power dissipation [Sha97] has been investigated in detail, and several limitations of this theory have been uncovered. Furthermore, the approach in [Sha97] has been identified to target only the most simple subproblem within the proposed classification of the lower bound problem, i.e. the lower bound on supply voltage (LBP-1).

By means of a formal comparison it has been shown in this thesis that the general lower bound problem on power dissipation is different in nature from the information-theoretic transmission bound problem. While information theory neglects the cost of optimizing the transmission, this is not appropriate for low-power VLSI processing. Only in case of plain data links (LBP-3), the power dissipation bound arises in direct analogy to information theory.

Seite Leer /
Blank leaf

Chapter 5

Energy-Efficient Processing of Speech

5.1 Introduction

5.1.1 Motivation

Traditionally, hardware designers have tried to find economic arithmetic units by minimizing the total gate count. Low power dissipation was assumed to follow. However, from chapter 3 it is clear that minimum power dissipation hinges on switching activity which, in turn, can be minimized by appropriate encoding as was shown in chapter 4. The present chapter now shall combine the hardware designer's perspective with the idea of minimizing power dissipation in consideration of data statistics and switching activity.

Unlike [BMM$^+$00], where data statistics has been employed to reduce the switching activity on data links, this chapter exemplifies data statistics as vehicle for low-power implementation of digital signal processing (DSP) algorithms by means of application-specific design [WKFF00, WKFF01]. Finite impulse response (FIR) filtering of speech data serves as an example. FIR filtering is, for instance, a standard task for the realization of digital hearing instruments [KB98]. Currently, results of this chapter are being employed for the energy-efficient implementation of a noise-reduction algorithm in a commercial hearing instrument [Sch99]. In general, the results to be presented shall support the designer of application-specific digital signal processors in making decisions on data representation and arithmetic units.

5.1.2 Previous Work

A large number of approaches have been suggested to minimize the power consumption of digital filters. Only those related to data statistics will be cited here.

A technique, called approximate processing, has been suggested in [LNC96] and was generalized in [NOC$^+$97]. This technique dynamically adapts the filter order to changing input signal characteristics such as to keep the stopband energy of the filtered signal below a predefined threshold. Recently, an audio chip using this technique in decimation and interpolation filters has been reported to achieve power saving between 20% and 70% depending on signal statistics [Pan00].

In [SRB97] the differential coefficients method (DCM) has been proposed. The fundamental idea is to trade a long-coefficient multiplication for a short one at the cost of some overhead due to additional storage requirements. The authors examined this tradeoff by means of an analytical energy cost model. The bottom line is that DCM is effective only if the coefficient bit-width is appreciably reduced by differentiation. The above idea has been generalized to the decorrelating transformation (DECOR) in [RSH99a], such as to extend the principle of differential encoding from the filter coefficients to the filter input signal. However, the power savings that resulted from applying DECOR to the filter input were actually due to data word width reduction. In practice, such a reduction often will be impossible since clipping of data samples cannot be tolerated. Moreover, in [RSH99a] the authors employed the analytical energy cost model from [SRB97] to evaluate DECOR, thereby disregarding the effect of differential encoding on data statistics and thus, switching activity. This chapter shall provide a more realistic appraisal of differential encoding for speech filtering.

In [NS99] the authors employed data statistics to optimize the design of a low-power FIR filter bank for digital hearing instruments by means of asynchronous control and datapath logic. The present work rests on the same idea of exploiting the statistical properties of speech for low-power data processing at the example of FIR filtering. However, the present work differs as it provides a comparative study of alternative data encodings under various application parameters.

5.1.3 Outline

In section 5.2 the statistical features of speech are reviewed and coding schemes suited for data processing are identified. Section 5.3 presents an experimental study of the relation between word-level statistical properties of speech and bit-level switching activity. In section 5.4 the target application and the associated reference architecture are introduced. Subsequently, implementation details for alternative data encodings are discussed. Finally, the encodings are experimentally evaluated with respect to their energy efficiency in section 5.5.

5.2 Speech Features and Coding

5.2.1 Statistical Properties of Speech Data

Speech signals are inherently non-stationary with distinct short-term and long-term statistical properties. Short-term properties refer to speech segments of about 20 . . . 200 ms duration, while long-term characteristics refer to speech segments that are several orders of magnitude longer [JN84].

Short-term characteristics of speech

Speech in general comprises two types of signals: quasi-periodic signal components resulting from voiced speech (mostly vowels), and noise-like components resulting from unvoiced speech (fricatives and plosives). Quasi-periodic signals contain energy from about 100 Hz up to 4.5 kHz. Noise-like signals contains frequency components up to about 8 kHz but at a much lower energy level.

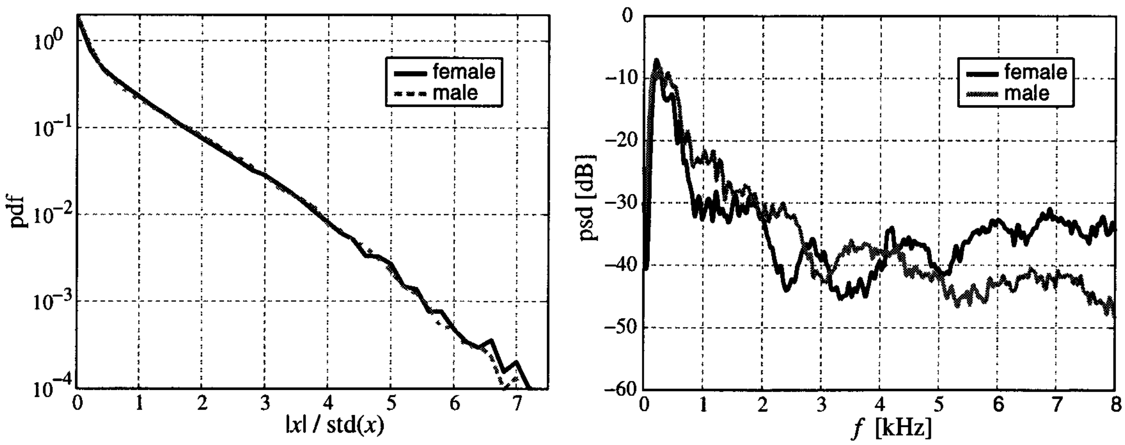


Figure 5.1: Long-term averaged probability density function (left) and power spectral density (right) for benchmark speech sequences.

Long-term characteristics of speech

In the present context, the long-term behavior of speech is of interest as this determines average switching activity and hence average power. This means that sufficiently long speech samples must be used such as to have a typical share of quasi-periodic and noise-like signal components. By averaging all statistical measures over a sufficiently long period of time, speech is approximated as stationary signal.

Typical for the long-term behavior of speech is the so-called *Gamma* probability density function [JN84], because small signal values around zero are orders of magnitude more likely than values close to the maximum amplitude. Figure 5.1 shows the probability density function for two benchmark speech samples of twenty seconds duration each [SQA]. The sequences represent a female and a male speaker, articulating several sentences at natural speed in English. Despite the similarity of their probability distributions, the sequences have different spectral properties, as can be seen in their power spectrum in figure 5.1. The male voice has higher signal power for frequencies below 1/10 the sampling frequency $f_S = 16$ kHz, while the female voice contains considerably more energy at higher frequencies. Nevertheless, correlation of adjacent samples, which is of primary importance for switching activity, differs only marginally for female and male voice ($\varrho_x = 0.99$ for the male vs. $\varrho_x = 0.95$ for the female speaker).

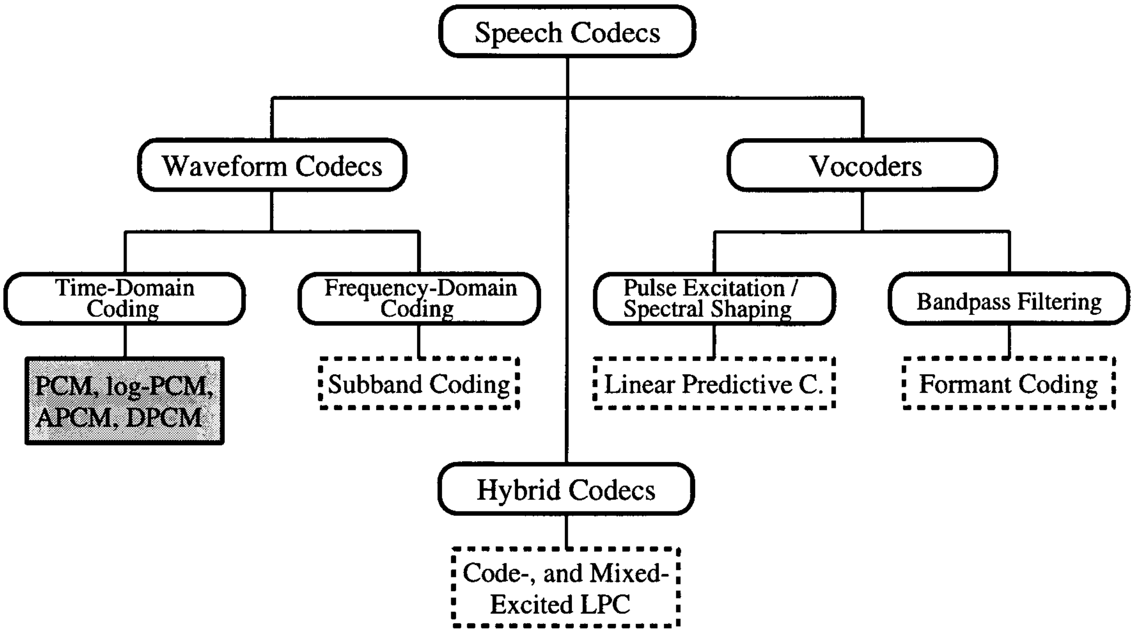


Figure 5.2: Classification of low bit-rate speech coding techniques. Only time-domain waveform coding is applicable to data processing.

5.2.2 Speech Coding Techniques

The term speech coding traditionally refers to employing the statistical properties of speech signals to reduce the required transmission bit rate in communication systems by means of data compression. Such coding techniques may be broadly divided into waveform coding and source coding (vocoders), see figure 5.2. Hybrid codecs combine methods from waveform and source coding to produce the best speech quality at low bit rates [Kon94].

As will be seen, proper encoding of speech can also be useful for low-power implementation of signal processing algorithms. In this case, however, the choice of coding is limited to time-domain waveform coding, because for the implementation of all codecs some time-domain processing is required. Therefore, all codecs potentially benefit from efficient implementation of PCM systems. The following coding schemes will be considered for low-power FIR filtering: linearly quantized PCM, PCM with adaptive (APCM) and logarithmic (log-PCM) quantization, and differential PCM (DPCM).

5.3 Activity Analysis for Speech Data

In section 4.5.2 theoretical lower bounds for the average number of transitions per symbol for given word-level data characteristics were discussed. However, minimum switching activity in general implies complex and highly dissipative coding hardware. Furthermore, minimum activity codes are not appropriate for efficient implementation of arithmetic operations. For the word-level activity analysis in this section, only fixed-length lossless encodings suitable for implementation of processing algorithms will be considered. For analysis and evaluation of lossy encodings, a meaningful signal quality measure is indispensable. Such a quality measure will be inferred from the target application in section 5.4.2.

5.3.1 Fixed-length Lossless Encodings

Assume a digital discrete-time speech signal

$$X(k) \;=\; \left[X(kT_{ci})\right]_{\langle x\rangle}\,, \qquad k = 0, 1, \ldots \tag{5.1}$$

with $-1 < X(k) < 1$, where $[.]_{\langle x\rangle}$ denotes quantization to $\langle x\rangle$ bits by rounding to the nearest binary value. The codeword for $X(k)$ is the binary vector $\boldsymbol{x}^k = (x_0^k, x_1^k, \ldots, x_{\langle x\rangle-1}^k)$. The following four encodings are considered:

PCM with 2's-complement representation (2sC). For this code, the codeword $\boldsymbol{x}^k$ is formed such that

$$X(k) \;=\; (-1)\cdot x_0^k + \sum_{i=1}^{\langle x\rangle-1} x_i^k \cdot 2^{-i}\,. \tag{5.2}$$

2sC is the most widely used representation for signed numbers because it facilitates efficient implementation of addition/subtraction. However, for speech signals where small values around zero are orders of magnitude more likely than very large values, the sign-extension of 2sC greatly inflates switching activity.

PCM with sign-magnitude representation (S&M). In this case, the codeword $\boldsymbol{x}^k$ is formed such that

$$X(k) = (-1)^{x_0^k} \cdot \sum_{i=1}^{\langle x \rangle - 1} x_i^k \cdot 2^{-i} . \tag{5.3}$$

S&M representation conforms better to the characteristics of speech, but implies more costly adders/subtracters.

Differential PCM with 2sC representation (D-2sC). The codeword $\boldsymbol{x}^k$ is recursively constructed such that

$$X(k) - X(k-1) = (-1) \cdot x_0^k + \sum_{i=1}^{\langle x \rangle - 1} x_i^k \cdot 2^{-i} . \tag{5.4}$$

Since the amplitude of speech signals changes slowly over time, the differentiation in (5.4) potentially reduces the dynamic range. For the target application, differential encoding is associated with the aforementioned DECOR transform, to which we will return in section 5.4.4.

Differential PCM with S&M representation (D-S&M). In this case, for the codeword $\boldsymbol{x}^k$ holds

$$X(k) - X(k-1) = (-1)^{x_0^k} \cdot \sum_{i=1}^{\langle x \rangle - 1} x_i^k \cdot 2^{-i} . \tag{5.5}$$

5.3.2 Signal Parameters

Switching activities α_{x_i} $(i = 0, \ldots, \langle x \rangle - 1)$ and the total number of transitions per symbol

$$\alpha_x = \sum_{i=0}^{\langle x \rangle - 1} \alpha_{x_i} \tag{5.6}$$

shall be investigated experimentally subject to four signal parameters:

Signal variance σ_X^2 [dB]. This parameter corresponds to perceived loudness and is defined as

$$\sigma_X^2 \text{ [dB]} = 10 \cdot \log_{10} \left(\frac{1}{n} \sum_{k=1}^{n} (X(k) - \mu_X)^2 \right)$$

where μ_X is the expected value.

Signal-to-noise ratio SNR [dB]. If $\tilde{X}(k) = X(k) + N(k)$ with $N(k)$ being noise samples, this parameter is defined as

$$\text{SNR[dB]} = 10 \cdot \log_{10} \frac{\sigma_{\tilde{X}}^2}{\sigma_N^2} .$$

Sampling frequency f_S [kHz]. According to (5.1) this parameter is defined as

$$f_S = 1/T_{ci} .$$

Quantization resolution $\langle x \rangle$ [bit]. For fixed-length codes, number of bits per symbol, data word width, and quantization resolution $\langle x \rangle$ are synonyms.

The two speech sequences investigated in section 5.2.1 served as benchmark signals for the experiments. For sake of brevity, only the main results for the male speaker are reproduced here. Complete results are given in [WKF00]. It was found that female voice in general induces slightly higher switching activity due to lower temporal correlation resulting from increased spectral power at higher frequencies. However, the qualitative behavior with respect to the above parameters is the same for male and female speakers.

5.3.3 Switching Activity vs. Signal Parameters

Signal variance

The influence of signal power on switching activity is investigated by fixing the sampling frequency to 16 kHz and the quantization resolution to 16 bits. A sinusoidal signal sampled at its minima and maxima defines the reference of $\sigma_X^2 = 0$ dB.

Figure 5.3 shows the switching activity profiles for two grossly different loudness levels. In these plots, the left-most bit position denotes the sign bit x_0. The right-most position is the LSB. The table at the bottom of figure 5.3 reports the absolute and relative number of transitions per symbol, see (5.6). From the results in figure 5.3 the following may be concluded:

- Word-level switching activity α_x grows with increasing signal power for all encodings considered.
- Absolute and relative activity savings[1] of S&M over 2sC increase with decreasing signal power, meaning 2sC being particularly inefficient for low voice.
- Differential PCM with 2sC considerably amplifies switching activity not only for redundant but also for information bearing bits.
- Differential PCM with S&M considerably reduces switching activity compared to PCM with S&M for all information bearing bits. Activity savings of D-S&M over S&M therefore grow with signal variance.

Signal-to-noise ratio

In order to explore different SNR, noise has been added to the speech sequence. σ_X^2 of the target signal was kept constant such as to exclude the influence of signal variance on switching activity for this set of experiments.

Figure 5.4 shows the activity profiles for colored noise, which was obtained by passing white noise through a filter with magnitude response similar to the power spectrum of the original speech signal. Results for other noise types including multiple-speaker modeling are given in [WKF00]. From figure 5.4 the following implications follow:

- As signal quality deteriorates, switching activity grows for all considered encodings.

---

[1] Increased absolute savings correspond to a larger area enclosed by the activity profiles under consideration. Relative savings may be inferred from the position of this area: The higher up in the diagram the larger the relative savings.

- Absolute activity savings of S&M over 2sC are largely unaffected by the SNR. Relative savings marginally decline.
- Differential encoding becomes less effective with decreasing SNR.

Sampling frequency

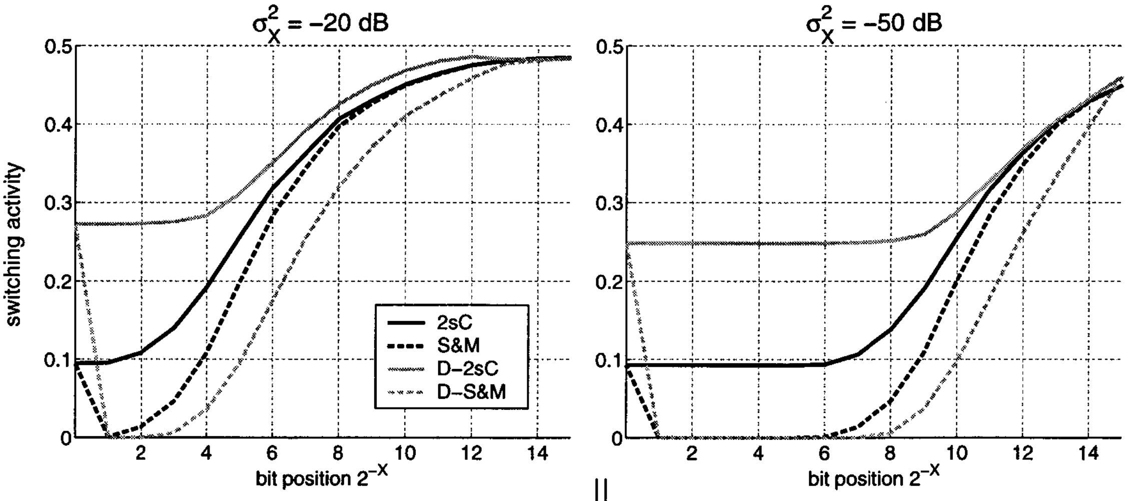
For some completely uncorrelated data source, the number of transitions per symbol α_x is constant for any sampling frequency f_S. Hence, by virtue of (2.2), power dissipation linearly grows with f_S. However, this does not hold true for speech signals. Figure 5.5 typifies the relation between switching activity and sampling frequency for speech:

- As a consequence of increasing correlation, switching activity α_x reduces for higher sampling frequencies and hence, power dissipation, as a function of f_S, grows with smaller rate than for uncorrelated signals.
- The advantage of S&M over 2sC diminishes for increasing sampling frequency. This can be explained by a smaller proportion of sign changes taking place, which lessens the adverse effect of sign-extension in 2sC representation.
- Because of increasing correlation between samples, differential PCM becomes more efficient with raised sampling frequency.

Quantization resolution

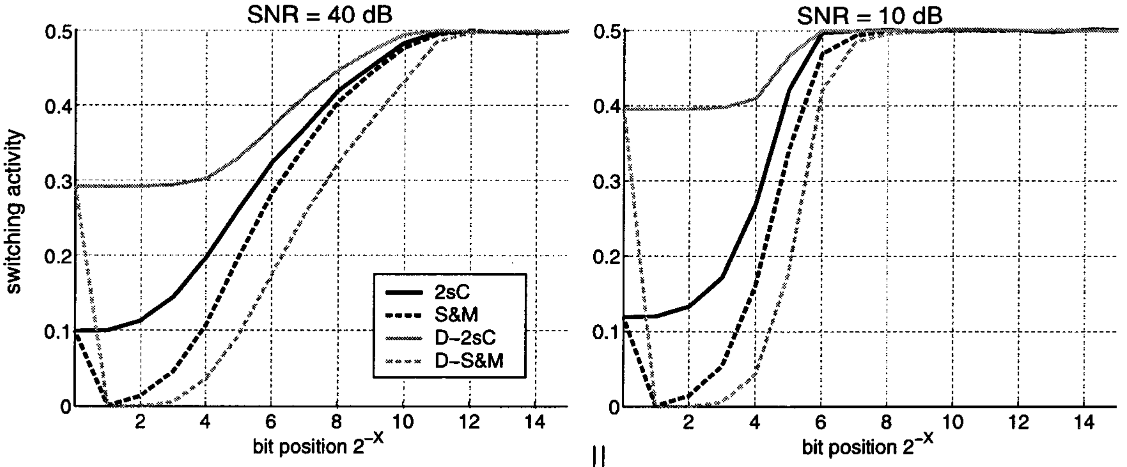
Finally, by definition (5.6), switching activity α_x will also be affected by the number of bits per symbol used for encoding. Figure 5.6 depicts the activity profiles and total activities for quantization resolutions of 12 and 24 bits. The experimental data reveal the following:

- Resolutions in excess of about sixteen bits will add random-like high-activity bits to the data word and hence disproportionately increase overall activity.
- Relative activity savings of S&M and D-S&M over 2sC decline as the quantization resolution enhances.



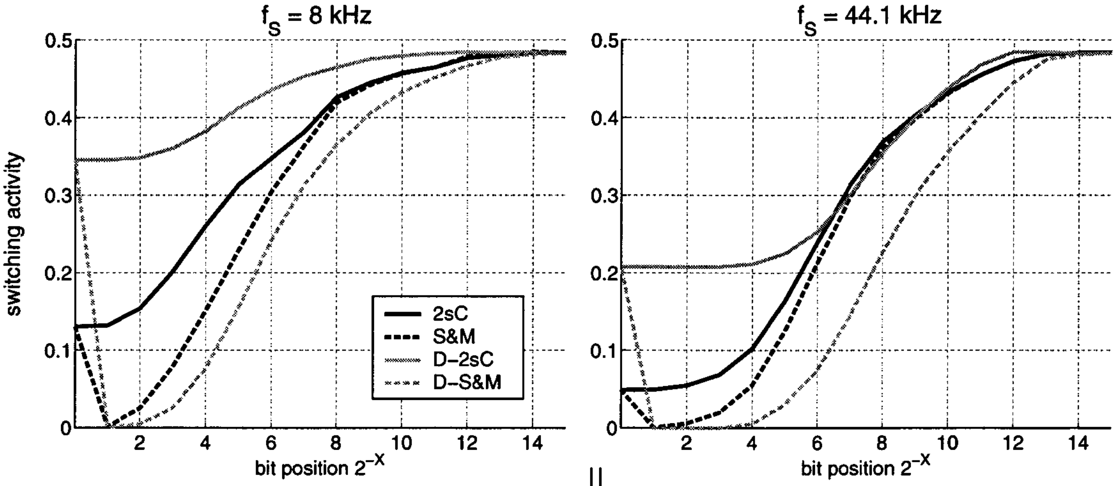
	α_x	[%]	α_x	[%]
— 2sC	5.3	100	3.3	100
---- S&M	4.8	91	2.4	73
— D-2sC	6.2	117	4.8	145
---- D-S&M	4.3	81	2.0	61

Figure 5.3: Switching activity profiles and total switching activity α_x for two different signal power levels ($\langle x \rangle$ = 16 bit, f_S = 16 kHz).



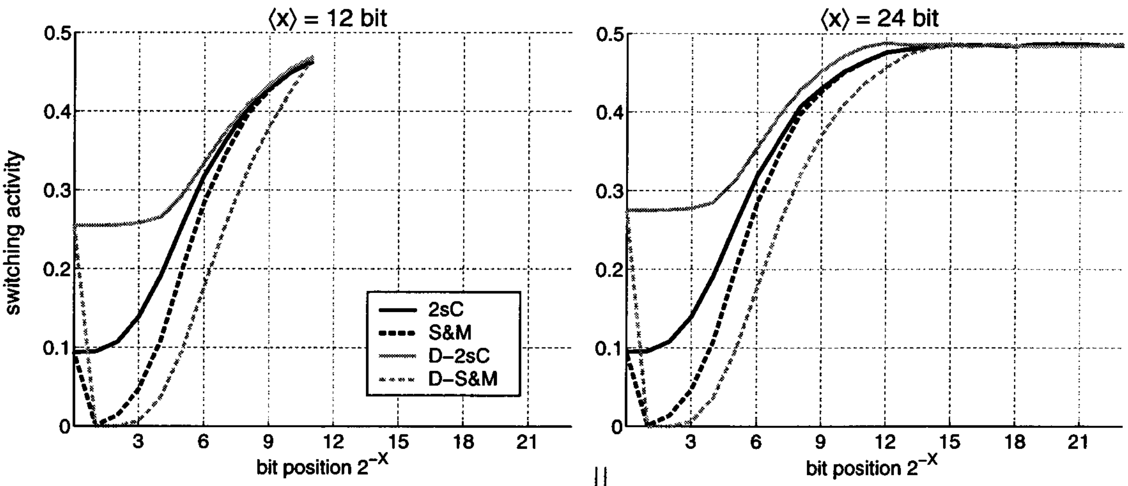
	α_x	[%]	α_x	[%]
— 2sC	5.5	100	6.2	100
---- S&M	4.9	89	5.6	90
— D-2sC	6.5	118	7.5	121
---- D-S&M	4.5	82	5.5	89

Figure 5.4: Switching activity profiles and total switching activity α_x for two different signal-to-noise ratios ($\langle x \rangle$ = 16 bit, σ_X^2 = -20 dB, f_S = 16 kHz).



	α_x	[%]	α_x	[%]
— 2sC	5.6	100	4.6	100
---- S&M	5.0	89	4.3	93
— D-2sC	6.9	123	5.4	117
---- D-S&M	4.7	84	3.6	78

Figure 5.5: Switching activity profiles and total switching activity α_x for two different sampling frequencies ($\langle x \rangle$ = 16 bit, σ_X^2 = -20 dB).



	α_x	[%]	α_x	[%]
— 2sC	3.3	100	9.1	100
---- S&M	2.8	85	8.6	94
— D-2sC	4.1	124	10.1	111
---- D-S&M	2.4	73	8.1	89

Figure 5.6: Switching activity profiles and total switching activity α_x for two different quantization resolutions (σ_x^2 = -20 dB, f_S = 16 kHz).

5.4 FIR Filtering of Speech Data

5.4.1 Target Application

The FIR filter algorithm is a prime candidate for data encoding, because every data item is used multiple times in a generic operation, i.e. multiply-accumulate (MAC). It is assumed, that an N-tap FIR filter operation

$$Y(k) = \sum_{n=0}^{N-1} c_n X(k-n) \tag{5.7}$$

is to be performed on an application-specific signal processor with fixed-point arithmetic. Again, $\langle x \rangle$ is used to denote the number of bits in the binary representation of signal X. The filter coefficients c_n are freely programmable but invariant during the period of operation.

Figure 5.7 shows the model employed for evaluating different data codes. Power analysis is carried out for encoder/decoder, functional data memory, and MAC unit. Since the focus is on exploiting data statistics, the coefficient memory is not included in the power analysis, although different data codes may infer specific coefficient codings. For the same reason any potential symmetry in the coefficient vector is not exploited.

5.4.2 Reference Architecture with 2sC Data

The reference architecture operates with data in linear PCM and 2sC representation. Neither encoding nor decoding is necessary in this case. Thus, in the evaluation model in figure 5.7, power is dissipated only by the data memory and the MAC unit.

In [BG99] it was found that for a fully time-shared architecture the direct form (DF) filter is more energy-efficient than the transposed form due to increased state-storage requirements in the latter. Furthermore, for a time-shared architecture the transposed form poses problems with the coefficient update in adaptive filtering. Hence, the DF filter structure has been selected for the present investigations.

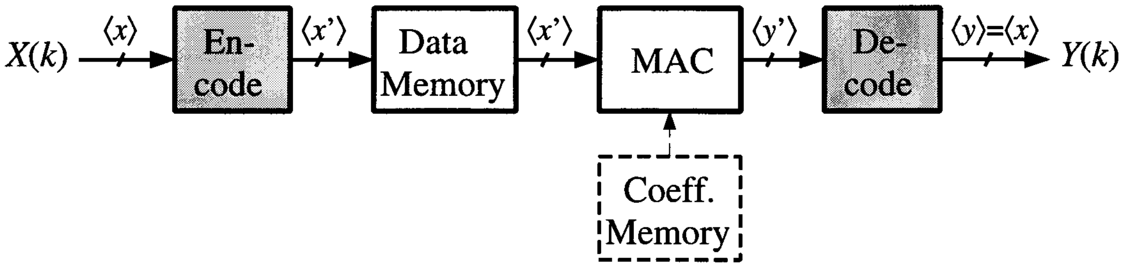


Figure 5.7: Block diagram of evaluation model for data coding.

Data memory

Assume the data memory is to be built with standard-cell flip-flops. There are basically three different schemes to realize the necessary data storage and access for DF FIR filtering in a fully time-shared architecture:

(1) circular shift register with fixed read/write positions,

(2) linear shift register with fixed write position and random read access,

(3) register file with random read/write access.

Obviously, architecture (1) is the most area-efficient configuration since it works without address generation overhead. However, with regard to power consumption, it is the least desirable version since the shift register must be clocked at N times the sampling frequency. We experimentally found, see appendix D, that configurations (2) and (3) are comparable in terms of area and power consumption for a wide range of filter orders. However, architecture (3) with direct read/write addressing permits to gate the clock signal. With this feature, architecture (3) outperformed all other structures in view of power dissipation. Hence, a structure as shown in figure 5.8 has been selected for the data memory of the reference architecture. All alternative architectures shall use the same memory structure but possibly differ in data word width $\langle x \rangle$.

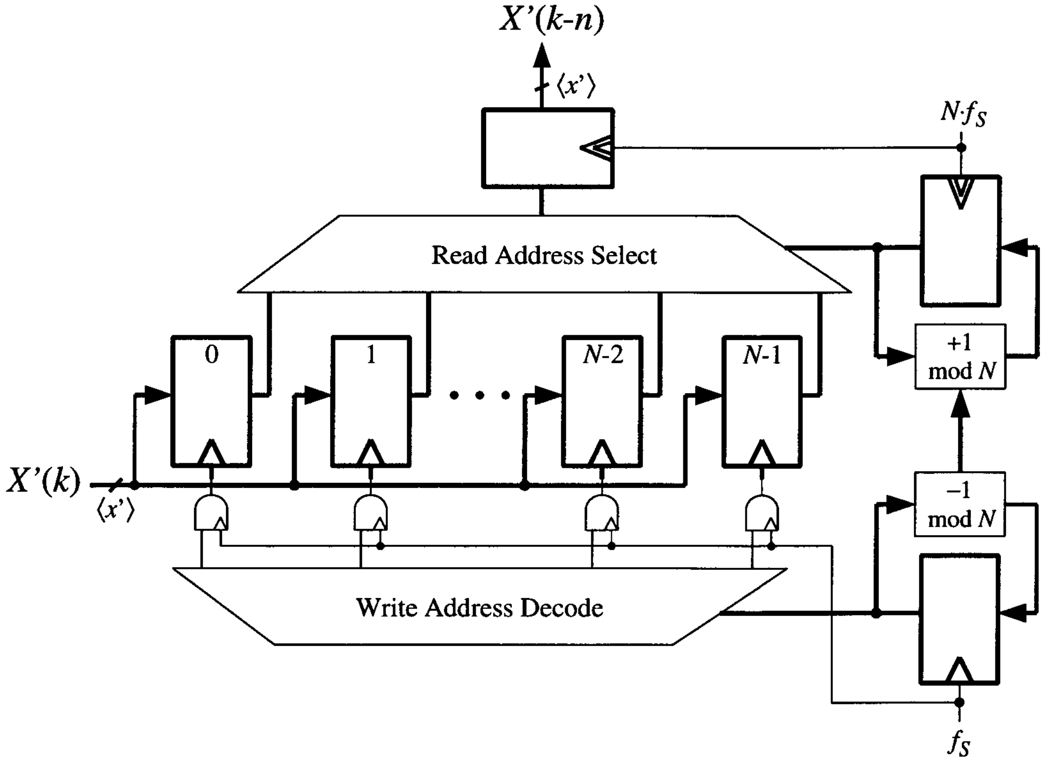


Figure 5.8: Reference architecture of the data memory.

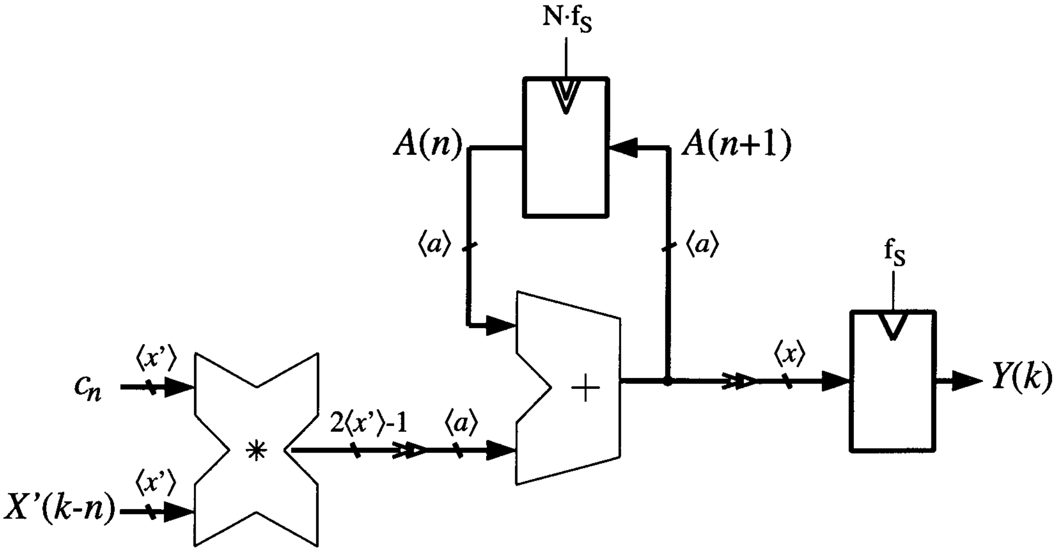


Figure 5.9: Reference architecture of the MAC unit.

MAC unit

The MAC unit performs standard 2sC multiplication by means of a parallel multiplier. As shown in figure 5.9 the coefficient word width is presumed equal to the data word width, resulting in a product width of $2\langle x\rangle - 1$ bits[2]. The product is truncated such as to retain $\langle a\rangle$ most-significant bits for the accumulator A with $2\langle x\rangle - 1 \geq \langle a\rangle \geq \langle x\rangle$. Finally, the filter output signal $Y(k)$ is truncated to the input data word width of $\langle x\rangle$ bits.

Theoretically, for an arbitrary N-tap filter $\lceil \log_2(N)\rceil$ guard bits have to be provided for the accumulator in order to avoid arithmetic overflow. However, given filter coefficients which correspond to practical impulse responses and the particular input data characteristic at hand, it is extremely unlikely that the full theoretical range will be exercised. In fact, for all benchmark signals from section 5.3 and over a wide range of filter specifications the accumulator can do without any guard bit. Therefore, zero guard bits were used for the reference as well as for any alternative architecture.

Formally, the filter operation performed by the reference implementation (PCM-2sC) can be summarized as

$$\begin{aligned} X'(k) &= X(k) \\ A(n+1) &= \lfloor c_n \cdot X'(k-n) \rfloor_{\langle a\rangle} + A(n) \\ Y'(k) &= \lfloor A(N) \rfloor_{\langle x\rangle} \\ Y(k) &= Y'(k) \end{aligned} \tag{5.8}$$

($n = 0, \ldots, N-1$; $a(0) = 0$), where $\lfloor x \rfloor_w$ denotes the truncation of signal x in 2sC representation to w bits (rounding towards $-\infty$).

Quality measure

The accumulator width $\langle a\rangle$ of the reference architecture determines the target quality of the filtered signal, which is assessed as the signal-to-quantization-noise ratio

---

[2] Since both operands are confined to $-1 < X(k), c_n < 1$ the product will fall in the same range. Therefore, $2(\langle x\rangle - 1)$ bits are required for the product magnitude, plus one bit for the product sign.

$$\mathrm{SQNR[dB]} = 10 \cdot \log_{10} \frac{\frac{1}{n}\sum_{k=1}^{n}\left(\overline{Y}_r[k]\right)^2}{\frac{1}{n}\sum_{k=1}^{n}\left(\overline{Y}_r[k] - \overline{Y}[k]\right)^2} \tag{5.9}$$

where $\overline{Y}$ and $\overline{Y}_r$ denote the mean-compensated output signal associated with full and reduced precision accumulation, respectively. SQNR is the previously mentioned quality measure that permits meaningful evaluation of lossy data encodings. For $\langle a \rangle < 2\langle x \rangle - 1$ such codings in principle allow to shift part of the approximation from within the MAC unit to the preceding encoder unit, which potentially simplifies multiplication.

5.4.3 Sign-Magnitude Representation

As was seen in section 5.3, S&M representation substantially reduces the average switching activity of binary encoded speech data streams compared to 2sC. S&M offers the advantage of easy implementation of many arithmetic operations. Particularly multiplication of S&M numbers is straightforward, since it simply requires an unsigned multiplication of magnitudes and XOR-ing the input sign bits. Unfortunately, the most basic arithmetic operation, i.e. addition of two numbers, is considerably more complicated with S&M than with 2sC. It is this fact that has prevented the common use of S&M representation for DSP implementations.

However, since multiplication is more costly than addition, S&M is an attractive alternative to 2sC for DSP applications, even when the statistical properties of the application data are left aside. Two alternative S&M MAC units based on S&M and 2sC addition have been investigated. In either case the same unsigned multiplier is used.

S&M accumulation

Addition of two S&M numbers can be implemented by a 1's-complement adder with end-around carry signal [Hwa79]. This requires conditional inversion of one input operand as well as of the sum output, see figure 5.11. For the application at hand this has the advantage of operating the accumulator with the preferred S&M representation. Furthermore, when the S&M product is truncated prior to accumulation, the error introduced will in general be lower than in the reference implementation. This is because truncation of 2sC numbers corresponds to rounding towards minus infinity, see figure 5.10, which will cause the error to build up during accumulation of individual filter tap results. On the other hand, pruning S&M numbers complies with rounding towards zero and the error tends to average out over several filter taps.

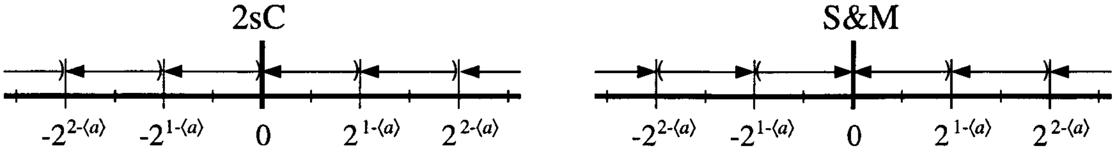


Figure 5.10: Truncation for 2sC and S&M numbers.

The above benefits of S&M accumulation come at the expense of increased activity in the adder circuitry due to the end-around carry signal and pre- and post-conversion overhead. Besides, although a 1's-complement adder with end-around carry is algorithmically stable, oscillation may occur in the hardware implementation due to skewed input signals. To eliminate oscillation, the end-around carry signal has been gated with a clock-derived enable signal as shown in figure 5.11.

The filter operation with S&M accumulation can be formally described as follows:

$$\begin{aligned} X'(k) &= \{X(k)\}_{S\&M} \\ A(n+1) &= [c_n \cdot X'(k-n)]_{\langle a\rangle} + A(n) \\ Y'(k) &= [A(N)]_{\langle x\rangle} \\ Y(k) &= \{Y'(k)\}_{2sC} \end{aligned} \tag{5.10}$$

where $[X]_w$ denotes the truncation to w bits of signal X in S&M representation.

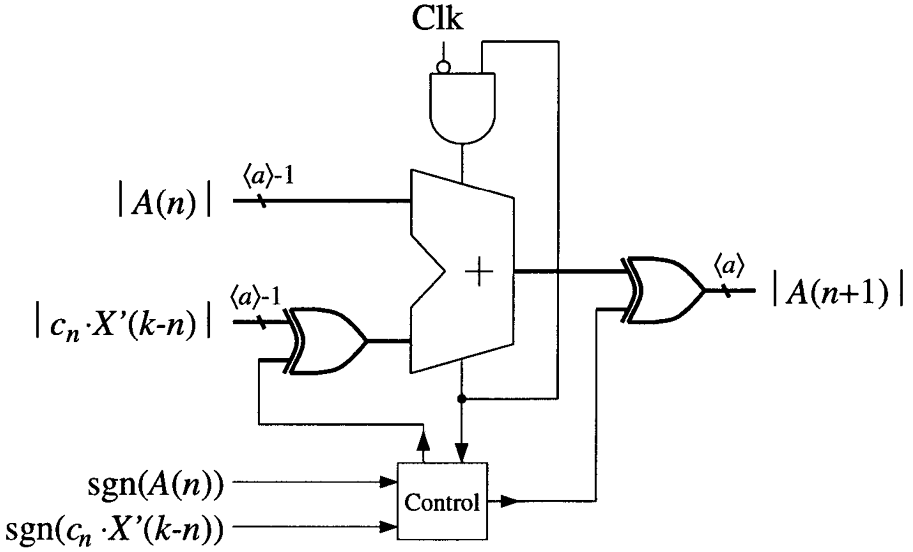


Figure 5.11: S&M addition with gated end-around carry signal.

2sC accumulation

To facilitate a more simple addition, the S&M product may be converted to 2sC. Precise conversion would imply the circuitry shown in figure 5.12(a), where the extra row of $2\langle x\rangle - 2$ half-adder cells is undesirable for energy efficiency reasons. Since truncation is performed post conversion, the biased 2sC truncation error is associated with this architecture. Approximate conversion from S&M to 2sC as shown in figure 5.12(b) is not only less costly, but at the same time accomplishes the unbiased rounding of S&M truncation. In this case, the sign-bit of the product is fed as carry input to the accumulating adder, which implies almost no overhead.

With the conversion as shown in figure 5.12(b), the formal filter description for 2sC accumulation is

$$\begin{array}{rcl} X'(k) & = & \{X(k)\}_{S\&M} \\ A(n+1) & = & \{[c_n \cdot X'(k-n)]_{\langle a\rangle}\}_{2sC} + A(n) \\ Y'(k) & = & \lfloor A(N) \rfloor_{\langle x\rangle} \\ Y(k) & = & Y'(k)\ . \end{array} \tag{5.11}$$

Note that in the occurrence of 2sC accumulation, no explicit decoder unit is required.

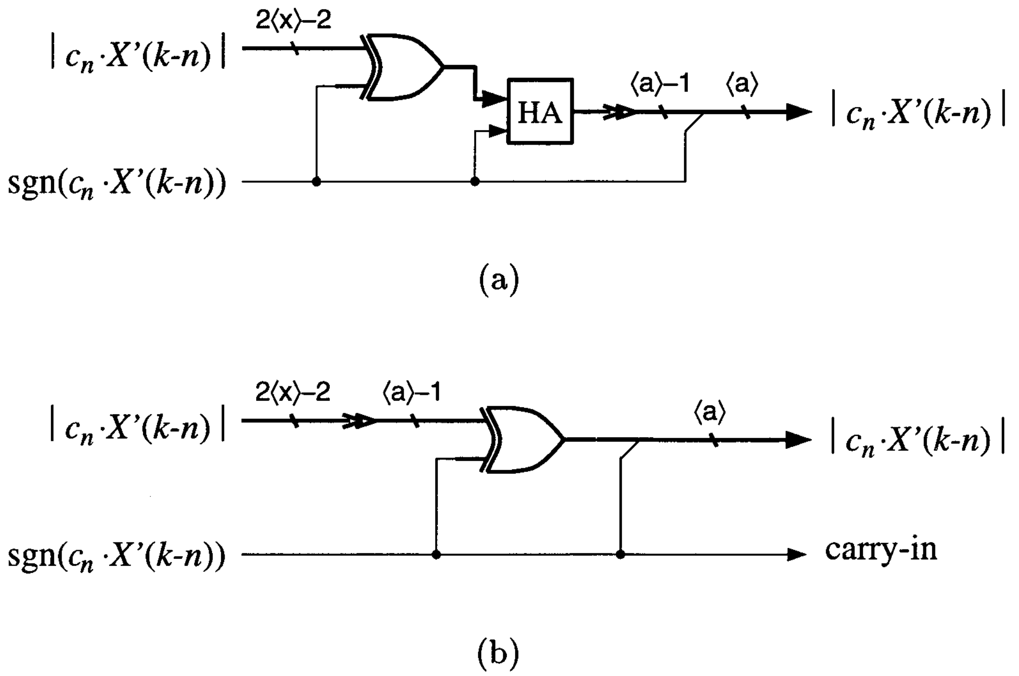


Figure 5.12: Conversion of the S&M product for 2sC accumulation with (a) truncation post-conversion and (b) truncation pre-conversion.

5.4.4 Differential Encoding

Differential encoding employs the high correlation between consecutive samples to remove short-term redundancy in the speech signal. DPCM corresponds to the decorrelating transform *DECOR* [RSH99a], when applied to the input data stream. In DECOR the filter transfer function is multiplied and divided by the same polynomial.

$$Y(z) \;=\; X(z)\cdot(1+\alpha z^{-\beta})^{m}\cdot H(z)\cdot\frac{1}{(1+\alpha z^{-\beta})^{m}} \qquad (5.12)$$

Since correlation is highest between adjacent speech samples the obvious choice for α and β in (5.12) is $\alpha=-1$ and $\beta=1$. Parameter m determines the number of recursive difference operations that are applied to the input data stream. Figure 5.13 depicts the system architecture for $\alpha=-1$, $\beta=1$, $m=1$.

With this choice of parameters, en- and decoder units are particularly simple in that they require no multiplication. However, a

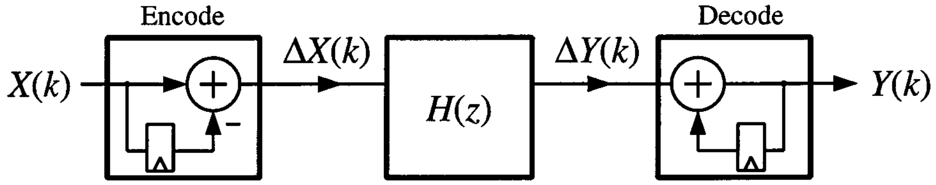


Figure 5.13: DECOR transform for $\alpha = -1$, $\beta = m = 1$.

key obstacle in a practical realization of this transform is the infinite-impulse response (IIR) operation $1/(1 - z^{-1})$ to be performed in the decoder. The pole on the unit circle is only canceled by the preceding feed-forward section if finite-precision effects are avoided. This has two important consequences which will be discussed next.

Error propagation

Most of the time the difference between adjacent speech samples will have a smaller dynamic range than the original signal. Thus, in [RSH99a], power savings were based on word width reduction by allowing for deterioration of the SQNR. However, the decoding operation $1/(1 - z^{-1})$ will propagate the smallest error in any of the filter output samples to all subsequent samples. Thus, there is a risk of not achieving the target SQNR of the reference implementation. To exclude error propagation, the differential input is represented with $\langle x\rangle$+1 bits (for $m = 1$). Although this additional bit was found to be redundant for all speech signals investigated in section 5.3, it seems to be inevitable from a practical point of view, whenever the input sequence is not a priori known.

Hence, due to error propagation, the IIR operation of the decoder restricts the choice of word width for the differential input .

Error accumulation

Assume a certain number of bits being truncated from the multiplier output prior to accumulation and let

$$\epsilon_\Delta(k) \;=\; \Delta Y_r(k) - \Delta Y(k) \tag{5.13}$$

be the corresponding error associated with the differential filter output. Then, by recursively applying $Y(k) = Y(k-1) + \Delta Y(k)$ and using (5.13), the error in the decoded output signal becomes

$$\begin{aligned}\epsilon(k) &= Y_r(k) - Y(k)\\ &= Y_r(0) + \sum_{i=1}^{k} \Delta Y_r(i) - Y(0) - \sum_{i=1}^{k} \Delta Y(i)\\ &= \epsilon(0) + \sum_{i=1}^{k} \epsilon_{\Delta}(i)\ . \qquad (5.14)\end{aligned}$$

From (5.14) it is clear, that any non-zero mean error induced during the filter operation will accumulate in the IIR decoder and consequently spoil the SQNR. To avoid this, the authors in [RSH99a] suggested to perform true rounding instead of truncation for reducing the product word width. However, rounding only accomplishes a zero mean error if the original signal itself has zero mean, see figure 5.10. This, in general, can not be justified. And even if assumed for the filter input signal, a DC bias may result from multiplication with finite-precision filter coefficients.

Thus, the only way to exclude error accumulation in practice, is to refrain from word size reduction in the MAC unit.

Practical implementation of DECOR

The preceding discussion suggests that the statistical properties of speech cannot be utilized for low-power computation by differential encoding when targeting synchronous bit-parallel processing. However, as shown in section 5.3, differential encoding reduces average switching activity if used in conjunction with S&M representation even for the nominal word width $\langle x' \rangle = \langle x \rangle + 1$. Therefore, the following differential filter implementation has been chosen:

$$\begin{aligned}X'(k) &= \{X(k) - X(k-1)\}_{S\&M}\\ A(n+1) &= c_n \cdot X'(k-n) + A(n)\\ Y'(k) &= \{A(N)\}_{2sC} + Y'(k-1)\\ Y(k) &= \lfloor Y'(k) \rfloor_{\langle x \rangle}\ .\end{aligned} \qquad (5.15)$$

To avoid error accumulation, truncation is only performed at the decoder output following the IIR operation.

With this implementation, the experimental evaluation in section 5.5 exclusively marks the low-power potential of DPCM in view of switching activity reduction, permitting a more realistic assessment than in [RSH99a], where power savings were bound to word size diminution.

5.4.5 Adaptive Encoding

Adaptive PCM (APCM) for low-rate communication models speech as non-stationary signal to accommodate the dynamic range with a fewer number of bits. For the target application, this translates into a reduction in the data path width.

Adaptation scheme

Of the various adaptation schemes proposed in the communication framework, feed-forward instantaneous gain adaption best suites the application at hand. Instantaneous adaption means that each individual data sample is approximated as

$$X(k) \;\simeq\; 2^{-G(k)} \cdot M(k)\ . \tag{5.16}$$

Feed-forward adaption is akin to sending the gain exponent $G(k)$ together with the mantissa $M(k)$ through the data path in order to control the computation. The data path thus must be capable of floating point arithmetic.

The mantissa word width $\langle m \rangle$ is determined by SQNR requirements and the word width of the gain exponent is given as

$$\langle g \rangle \;=\; \lceil \log_2(\langle x \rangle - \langle m \rangle + 1) \rceil \tag{5.17}$$

if original data samples were represented using $\langle x \rangle$ bits. Thus, approximation (5.16) provides a reduction in the overall data word width, i.e.

$$\langle m \rangle + \langle g \rangle \;<\; \langle x \rangle \tag{5.18}$$

as long as $0 < \langle m \rangle < \langle x \rangle - 2$.

Figure 5.14 illustrates the formation of the mantissa codeword $\boldsymbol{m}^k$ and gain exponent codeword $\boldsymbol{g}^k$ from the original 2sC codeword for

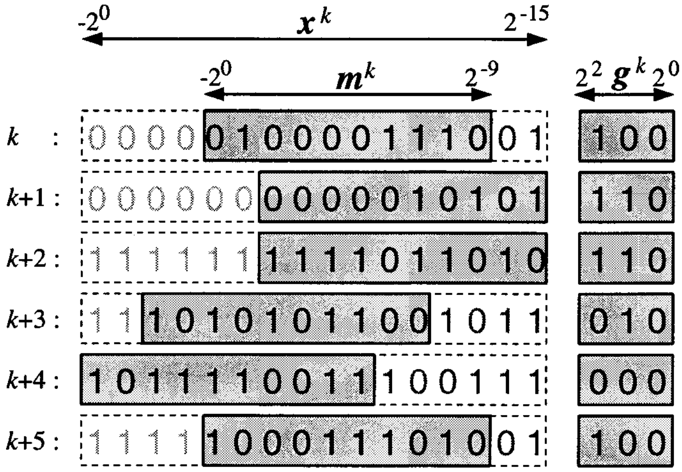


Figure 5.14: Formation of mantissa codeword $\boldsymbol{m}^k$ and gain exponent codeword $\boldsymbol{g}^k$ from the 2sC codeword $\boldsymbol{x}^k$ in APCM.

$\langle x \rangle = 16$ and $\langle m \rangle = 10$. The neglected bits at the left-hand side of the mantissa window correspond to redundant information, while truncating bits at the right-hand side of the window will introduce noise. Adaptation of S&M data can be performed accordingly.

Implementation of target application

Based on (5.16) the following implementation has been chosen for evaluating the energy efficiency of APCM:

$$\begin{array}{rcl} X'(k) & = & (\{M(k)\}_{S\&M}\ ,\ G(k)) \\ A(n+1) & = & \{[c_n \cdot M(k-n) \ll G(k-n)]_{\langle a \rangle}\}_{2sC} + A(n) \\ Y'(k) & = & \lfloor A(N) \rfloor_{\langle x \rangle} \\ Y(k) & = & Y'(k) \end{array} \tag{5.19}$$

The MAC unit utilizes floating point arithmetic rather than fixed point arithmetic as in the reference implementation. Using gain factor $G(k)$, the product is rescaled to the accumulator word width, which is governed by the target SQNR. With this scheme, potential power savings of APCM are based on appropriate rounding of individual data samples prior to multiplication.

Alternative implementations of APCM are possible. First, a unique scale factor may be used for N consecutive input samples. This scale factor is updated at the beginning of each filter cycle. This scheme corresponds to block floating point arithmetic [Opp70, RB97] and requires storage of the original data samples in the functional memory, but has the advantage that rescaling has to be performed only for filter output values. The effectiveness of this method strongly depends on the filter order and is especially appropriate when non-overlapping blocks of input data are to be processed.

Second, in addition to data samples, filter coefficients may be adaptively scaled if so permitted by the impulse response of the filter at hand. Since the focus here is on employing the data statistics for low-power computation independently of specific filter characteristics the above two alternatives are not further considered.

5.4.6 Logarithmic Encoding

As a standard technique for speech compression, logarithmic quantization employs the long-term statistical properties of speech to reduce the transmission bit rate. For data processing, the main attraction of logarithmic encoding is the substitution of costly multiplication in the linear domain by simple addition in the logarithmic domain. On the other hand, addition in the logarithmic domain calls for lookup tables (LUT) which grow exponentially with increasing precision [SA75, Lew90].

With the approximation technique explained below, LUTs for the conversion between logarithmic and linear domains are smaller than those required for addition. Since in the target application the final filter output must be in linear PCM format, anti-logarithm is performed within the MAC unit prior to accumulation.

Approximation of logarithm

To convert between logarithmic and linear domains one could either use direct table-lookup or some approximation technique. Depending on the target SQNR, the data word width may be significantly reduced if direct table-lookup is employed, i.e. $\langle x' \rangle < \langle x \rangle$. However, the conversion overhead quickly becomes unacceptable even for medium-scale $\langle x \rangle$.

On the other hand, the linear approximation $\log_2(1+x) \approx x$ $(0 \leq x \leq 1)$ proposed in [Mit62] permits efficient implementation, but is too imprecise to attain any practical SQNR for the target application. The piecewise linear approximation strategy from [CZV65] improves conversion accuracy but involves multiplications. The method proposed in [SBG99] enhances precision by means of a corrective addition. This approach will be modified, such that the conversion accuracy can be adjusted to meet the SQNR constraint.

Let $X \geq 1$, then the logarithm can be written as[3]

$$\begin{aligned} \log(X) &= \log(2^I + F2^I) \\ &= I + \log(1+F) \\ &= I + F + C(F) \;. \end{aligned} \tag{5.20}$$

Integer part I of the logarithm is formed by means of a simple leading-one detector and a small LUT. The fractional part $\log(1+F)$ of the logarithm is formed by adding a correction term $C(F)$ to F, the linear approximation of $\log(1+F)$. There holds

$$C(F) \;=\; \log(1+F) - F \;>\; 0, \qquad (0 < F < 1) \tag{5.21}$$

and

$$\frac{\mathrm{d}\,C(F)}{\mathrm{d}F} = \begin{cases} > 0 & 0 \leq F < F^* \\ = 0 \quad \text{for} & F = F^* \\ < 0 & F^* < F \leq 1 \;. \end{cases} \tag{5.22}$$

$F^* \approx 0.4427$, and $C(F^*) \approx 0.0861$ being the maximum of the correction term, see figure 5.15.

In practice, the correction term $C(F)$ must be realized with finite precision, and (5.20) becomes an approximation. The monotonic behavior of $C(F)$ can be employed for implementing $C(F)$ as LUT whose number of entries may be minimized in regard of the target SQNR by appropriate nonuniform quantization of F. Next to the number of entries, the number of bits to represent $C(F)$ is the second parameter for the optimization of this LUT. Figure 5.16 shows the principle and a numerical example for conversion from the linear to

---

[3] In the present context, speech samples $X(k)$ are interpreted as integer numbers, and logarithm always is to the base two.

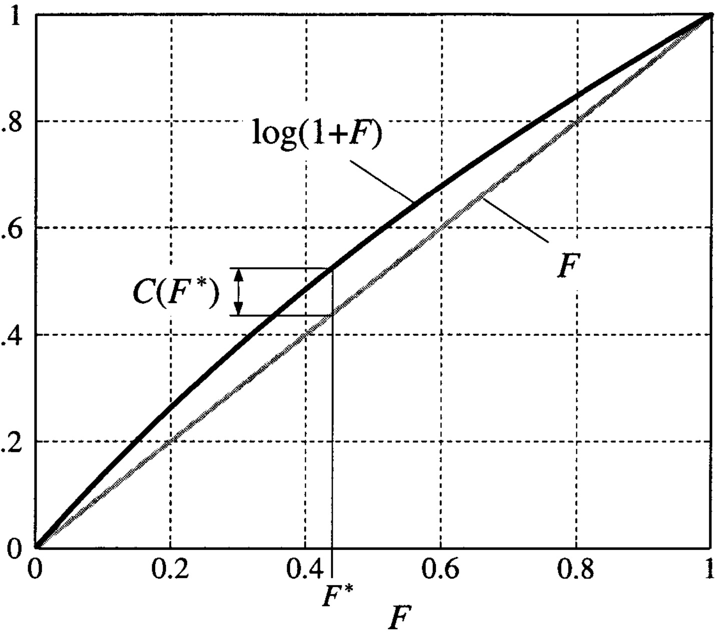


Figure 5.15: Linear approximation $\log(1+F) \approx F$ for $0 \leq F \leq 1$ with maximum error $C(F^*)$.

the logarithmic domain.

A similar approach can be used for the implementation of anti-logarithm by employing the approximation

$$F \approx \log(1+F-C(1-F)) \qquad (0 \leq F \leq 1) . \tag{5.23}$$

With this, the anti-logarithm for some given integer part I and fractional part F becomes

$$\begin{aligned} X &= 2^{I+F} \\ &\approx 2^I 2^{\log(1+F-C(1-F))} \\ &\approx 2^I(1+F-C(1-F)) . \end{aligned} \tag{5.24}$$

This is easily implemented by putting a '1' in the appropriate position according to I and appending the corrected fractional part. The symmetry in the correction terms for logarithm and anti-logarithm is an interesting feature, which could be employed such as to use only one common LUT. In the present case this was not done because of the required modification of the overall filter architecture compared to the reference implementation.

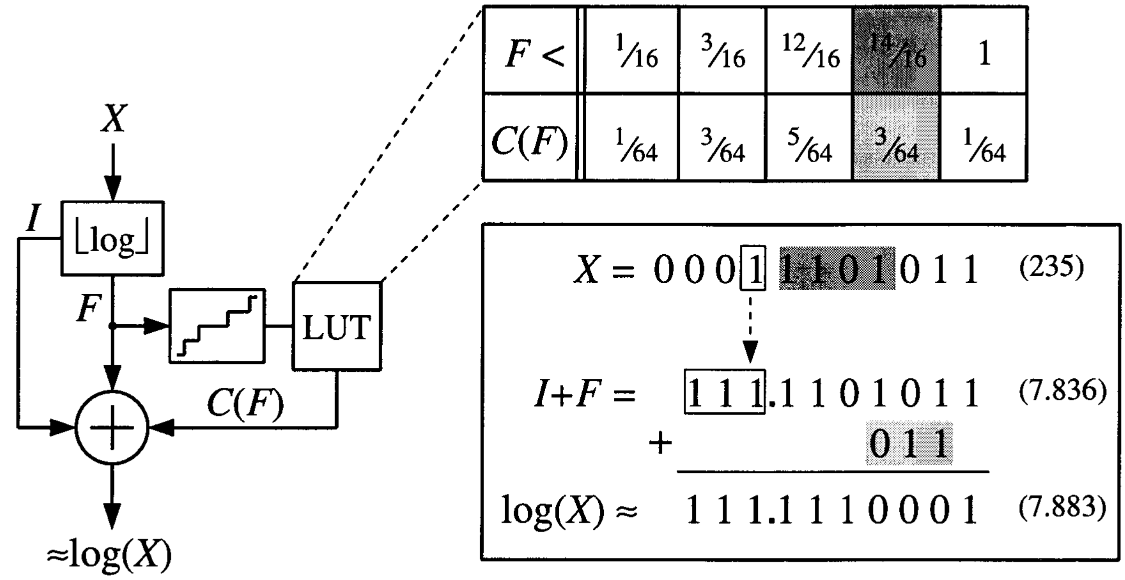

X
⌊log⌋
I
F
LUT
C(F)
≈log(X)
F <
1/16
3/16
12/16
14/16
1
C(F)
1/64
3/64
5/64
3/64
1/64
X = 0 0 0 1 1 1 0 1 0 1 1 (235)
I+F = 1 1 1.1 1 0 1 0 1 1 (7.836)
+ 0 1 1
log(X) ≈ 1 1 1.1 1 1 0 0 0 1 (7.883)

Figure 5.16: Architecture for logarithmic encoding with linear approximation and additive correction (left). Numerical example $\log_2(235) = 0.7877$ for LUT with five entries and three bits resolution for the correction terms (right).

Implementation of target application

With (5.20) and (5.24) the FIR filter based on logarithmic encoding (log-PCM) can hence be given as

$$\begin{aligned}
X'(k) &= I_X(k) + F_X(k) + C(F_X(k)) \\
P(n) &= \log(|c_n|) + X'(k-n) \\
A(n+1) &= \big\{S_X(k-n)S_{c_n}\big[2^{I_P(n)}(1+F_P(n)- \\
&\quad C(1-F_P(n)))\big]_{\langle a\rangle -1}\big\}_{2sC} + A(n) \\
Y'(k) &= \lfloor A(N)\rfloor_{\langle x\rangle} \\
Y(k) &= Y'(k)
\end{aligned} \tag{5.25}$$

with I_X and F_X being integer and fractional part of X in the logarithmic domain as above, and S_X denoting the sign of X.

5.5 Evaluation of Coding Schemes

5.5.1 Application Parameters

In order to cover a wide range of typical situations in speech filtering, the different codings have been evaluated at extreme points of the following parameters:

A/D-conversion bit rate I [bit/s]. This parameter comprises sampling frequency f_S and quantization resolution $\langle x \rangle$, that were dealt with separately in section 5.3 in order to reveal their influence on switching activity. The two considered extreme points are $I_L = 96$ kbit/s and $I_H = 705.6$ kbit/s, corresponding to telephony (toll) and CD-quality (see table 5.1).

Signal quality. This embraces the remaining free parameters from section 5.3, i.e. signal power σ_x^2 and signal-to-noise ratio SNR. Based on the results from section 5.3, the following signals were chosen: male speaker with high SNR to represent high signal quality [SQA], and faint colored noise to emulate low signal quality, see table 5.1. The colored noise was formed by passing Gaussian white noise through a filter with a magnitude response that matches the long-term power spectral density of speech.

Processing accuracy. For the target application processing accuracy is associated with the internal accumulator word width and rounding scheme used for summation over all product terms, and is assessed by the SQNR as defined in (5.9). The two extreme points to be investigated coincide with full accumulator width and accumulator width equal to input data width in the reference architecture. The corresponding SQNR of the high quality signal sets the target on processing accuracy for any alternative coding scheme.

5.5.2 Power Estimation Model

Since encoding for data processing has a joint impact on switching activity, circuit structure, and type of arithmetic units, analytical power cost models seem inappropriate in this case. Particularly

Parameter	Logo	Specification
Bit rate	I_L	$\langle x \rangle = 12$ bit, $f_S = 8$ kHz
	I_H	$\langle x \rangle = 16$ bit, $f_S = 44.1$ kHz
Signal quality	Q_L	$\sigma_x^2 = -50$ dB, SNR $= 0$ dB
	Q_H	$\sigma_x^2 = -20$ dB, SNR $= 40$ dB
Processing accuracy	A_L	$\langle a \rangle = \langle x \rangle$
	A_H	$\langle a \rangle = 2\langle x \rangle - 1$

Table 5.1: Specification of application parameters.

simple counting of basic operations as being performed in [SRB97] and [RSH99a] is hard to justify. Instead, a simulation approach to power estimation has been taken here. Although it does not guarantee exact results in absolute terms, it is considered sufficient for relative comparisons between alternatives.

For each encoding scheme corresponding gate-level circuits were designed for encoder/decoder, MAC unit and data memory. All units were described in VHDL and then synthesized and mapped to a 0.25μm CMOS standard cell library. Synthesis and technology mapping were performed using `Synopsys DesignCompiler` with optimization directed towards minimum area implementation. Consistent low-level architectures were selected for adders (ripple-carry) and multipliers (signed or unsigned array multiplication with carry-save addition of partial products) to reduce the influence of gate-level implementation details.

Subsequently, the gate-level netlist of each unit has been exercised with corresponding stimuli. Activities resulting from spurious transitions have been captured by utilizing gate-delay models provided by the technology library. The functionality of each circuit has been verified by comparing gate-level responses with expected responses as obtained from bit-true behavioral models written in `Matlab`.

The node toggle information so acquired was annotated on the netlist employing `Synopsys DesignPower`. Finally, average power consumption has been computed using estimated capacitive loads.

Note that the terms *power* and *energy* dissipation are synonyms in the present case of real-time operation and fixed data rate.

5.5.3 Experimental Results and Discussion

Experimental results

A $N = 17$-tap low-pass filter with cutoff frequency of $0.65 \cdot f_S/2$ was applied to the benchmark signals with statistics as given in table 5.1. The high-amplitude signals Q_H had a duration of eight seconds, i.e. 64'000 (352'800) samples at 8 (44.1) kHz, and were used for SQNR computations. For the noise signals Q_L 10'000 samples were simulated. The above choice of N and number of data samples is a compromise between sufficient insensitivity of the results on the particular filter/data characteristic and simulation run time.

Table 5.2 shows the estimated energy consumption per processed data sample across implementation alternatives and application parameters. For each filter the total energy use is split into encoder/decoder (Cod), data memory (Mem) and MAC unit.

Discussion

Three general conclusions can be drawn from the results in table 5.2:

- The total dissipation in the reference implementation (5.8) is clearly dominated by the MAC unit, which implies that provisions to reduce total power must attack here.
- Energy use of the reference implementation is relatively insensitive to signal amplitude, i.e. processing of Q_L takes the same (or even more) energy as processing Q_H.
- Any of the alternatives investigated outperforms PCM-2sC. Although there is no single optimal implementation for all parameter sets, differences between the alternative encodings are relatively minor. Therefore, other criteria such as area cost may be considered for rating the coding schemes.

		I_L				I_H			
		A_L		A_H		A_L		A_H	
Coding	Unit	Q_L	Q_H	Q_L	Q_H	Q_L	Q_H	Q_L	Q_H
PCM-2sC as in (5.8)	Cod	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Mem	15.9	15.1	15.9	15.1	20.0	18.3	20.0	18.3
	Mac	118.7	119.5	147.5	149.8	256.8	242.6	281.3	266.3
	$\sum$	134.6	134.6	163.4	164.9	276.8	260.9	301.3	284.6
PCM-S&M as in (5.10)	Cod	0.2	0.1	0.3	0.1	0.3	0.2	0.4	0.2
	Mem	12.0	13.8	12.0	13.8	15.5	17.1	15.5	17.1
	Mac	17.2	46.9	41.4	74.5	71.9	135.2	116.6	178.7
	$\sum$	29.4	60.8	53.7	88.4	87.7	152.5	132.5	196.0
PCM-S&M as in (5.11)	Cod	0.1	0.1	0.1	0.1	0.2	0.1	0.2	0.1
	Mem	12.0	13.8	12.0	13.8	15.5	17.1	15.5	17.1
	Mac	17.5	44.4	36.3	67.0	68.7	124.8	104.9	161.4
	$\sum$	29.6	58.3	48.4	80.9	84.4	142.0	120.6	178.6
DPCM-S&M as in (5.15)	Cod	n.a.	n.a.	2.2	2.3	n.a.	n.a.	3.5	3.3
	Mem	n.a.	n.a.	12.6	13.8	n.a.	n.a.	16.2	16.8
	Mac	n.a.	n.a.	39.2	58.6	n.a.	n.a.	111.4	131.8
	$\sum$	n.a.	n.a.	54.0	74.7	n.a.	n.a.	131.1	151.9
APCM-S&M as in (5.19)	Cod	0.5	0.3	0.3	0.2	0.8	0.3	0.6	0.3
	Mem	11.6	13.3	12.0	13.8	15.1	16.6	15.5	17.1
	Mac	18.7	47.5	37.5	69.2	78.2	132.7	107.9	164.3
	$\sum$	30.8	61.1	49.8	83.2	94.1	149.6	124.0	181.7
Log-PCM as in (5.25)	Cod	1.0	0.7	n.a.	n.a.	4.5*	2.4*	n.a.	n.a.
	Mem	12.5	14.7	n.a.	n.a.	16.6*	17.9*	n.a.	n.a.
	Mac	25.0	40.1	n.a.	n.a.	131.4*	168.5*	n.a.	n.a.
	$\sum$	38.5	55.5	n.a.	n.a.	152.5*	188.8*	n.a.	n.a.

*SQNR 10 dB below target

Table 5.2: Average energy use [10^{-7} J] per input data sample for 17-tap low-pass filter.

		I_L		I_H	
Coding	Unit	A_L	A_H	A_L	A_H
PCM-2sC as in (5.8)	Cod	0.0	0.0	0.0	0.0
	Mem	4.9	4.9	6.3	6.3
	Mac	3.4	4.0	6.0	6.6
	$\sum$	8.3	8.9	12.3	12.9
PCM-S&M as in (5.10)	Cod	0.2	0.2	0.3	0.3
	Mem	4.9	4.9	6.3	6.3
	Mac	2.9	3.5	5.4	6.1
	$\sum$	8.0	8.6	12.0	12.7
PCM-S&M as in (5.11)	Cod	0.1	0.1	0.2	0.2
	Mem	4.9	4.9	6.3	6.3
	Mac	2.8	3.3	5.3	6.0
	$\sum$	7.8	8.3	11.8	12.5
DPCM-S&M as in (5.15)	Cod	n.a.	1.3	n.a.	1.8
	Mem	n.a.	5.2	n.a.	6.6
	Mac	n.a.	3.6	n.a.	6.3
	$\sum$	n.a.	10.1	n.a.	14.7
APCM-S&M as in (5.19)	Cod	0.3	0.2	0.5	0.4
	Mem	4.5	4.8	5.9	6.2
	Mac	2.3	3.2	4.3	5.4
	$\sum$	7.1	8.2	10.7	12.0
Log-PCM as in (5.25)	Cod	0.6	n.a.	2.2*	n.a.
	Mem	4.9	n.a.	6.3*	n.a.
	Mac	1.4	n.a.	2.9*	n.a.
	$\sum$	6.9	n.a.	11.4*	n.a.

*SQNR 10 dB below target

Table 5.3: Cell area [10^{-2} mm^2] of FIR filter processor with different encodings.

Table 5.3 shows cell area figures corresponding to the energy evaluations from table 5.2. Since the same hardware units are used for processing of high- and low-quality signals, parameters Q_L and Q_H do not appear in table 5.3.

The results in table 5.2 and 5.3 permit the following statements for the alternative data encodings:

PCM-S&M. The S&M implementations (5.10) and (5.11) differ only marginally in their energy and area usage, with (5.11), which is based on 2sC accumulation, being slightly more efficient. Compared to PCM-2sC, substantial energy savings are achieved. Reduction is especially high for the low-amplitude signal Q_L which is in accordance with observations from section 5.3. Furthermore, energy savings of PCM-S&M are accompanied by a slightly enhanced area economy of about 5%.

DPCM-S&M. Because of error accumulation and propagation, filtering of differentially encoded data is always to be performed with accurate multiply-accumulate (see section 5.4.4). Therefore, energy and area results are identical for A_L and A_H. Compared with PCM-S&M, DPCM further decreases energy use in case of high signal amplitude Q_H and high processing accuracy A_H. This decrease comes at the expense of an area penalty of about 15%, however.

APCM-S&M. Adaptive quantization (5.19) does not reduce overall dissipation relative to PCM-S&M. Even in the data memory, energy consumption is only slightly better because no significant reduction of the overall data word width was achieved. For example, for low processing accuracy A_L, the data word width could be reduced by four bits in case of both bit rates I_L and I_H. However, since three bits are required to encode the corresponding gain exponent, the net input word width reduction is only one bit. On the other hand, area cost in case of low processing accuracy A_L is about 10% below that of PCM-S&M.

Log-PCM. Logarithmic quantization is only applicable to low- accuracy processing, because unrealistic conversion precision between linear and logarithmic domain is required otherwise. For low bit rate I_L and high signal amplitude Q_H, Log-PCM does with less energy than PCM-S&M. In this case $C(F)$ in (5.20) was manually optimized such that the required SQNR could be attained with a small (4×3)-bit LUT containing five distinct correction terms, see figure 5.16. Besides, the area cost was reduced by about 12% compared to PCM-S&M.

To meet the target processing accuracy for the high-amplitude signal Q_H, a considerably larger LUT is required. Optimizing a 7×7 bit LUT for eighty distinct correction terms, processing accuracy of 10 dB below that of the reference implementation was achieved. Since in this case energy consumption and area cost already exceed that of other encodings, further expansion of the LUT's complexity is not beneficial.

5.5.4 Derivation of General Coding Guidelines

General guidelines for the choice of coding in FIR filtering of speech data shall be derived from the results in table 5.2. Since the respective coding overhead scales with the number of filter taps$N$, the energy efficiency of the encoding schemes depends on N.

In order to extrapolate from filter order, the following assumption is made with respect to data memory and MAC unit: *Energy use per data sample grows linearly with the number of filter taps N.* For the data memory this fact has been experimentally verified in the range $8 \leq N \leq 128$, see figure D.2. For the MAC unit this assumption is justified because no particular filter characteristic is assumed here.

The overall energy consumption per data sample $E_{sample}^{\sum}$ can thus be written as function of the number of filter taps:

$$\begin{aligned} E_{sample}^{\sum}(N) &= E_{sample}^{Cod} + E_{sample}^{Mem} + E_{sample}^{Mac} \\ &= E_{sample}^{Cod} + N \cdot (\, E_{tap}^{Mem} + E_{tap}^{Mac} \,) \; . \end{aligned} \tag{5.26}$$

Coding energy per sample, E_{sample}^{Cod}, is directly given by table 5.2. Energy use per filter tap for data memory and Mac unit E_{tap}^{Mem} and E_{tap}^{Mac}, respectively, are obtained by dividing the corresponding values in table 5.2 by $N = 17$, i.e. the number of taps the results were obtained for.

Figure 5.17 visualizes (5.26) by showing relative energy dissipation when using alternative encodings at the chosen points of the parameter field. From these curves it is clear that the relative merits of the various coding schemes do not change beyond eight filter taps or so. Furthermore, it can be concluded that suitable data recoding is beneficial in terms of energy dissipation even for a single MAC operation.

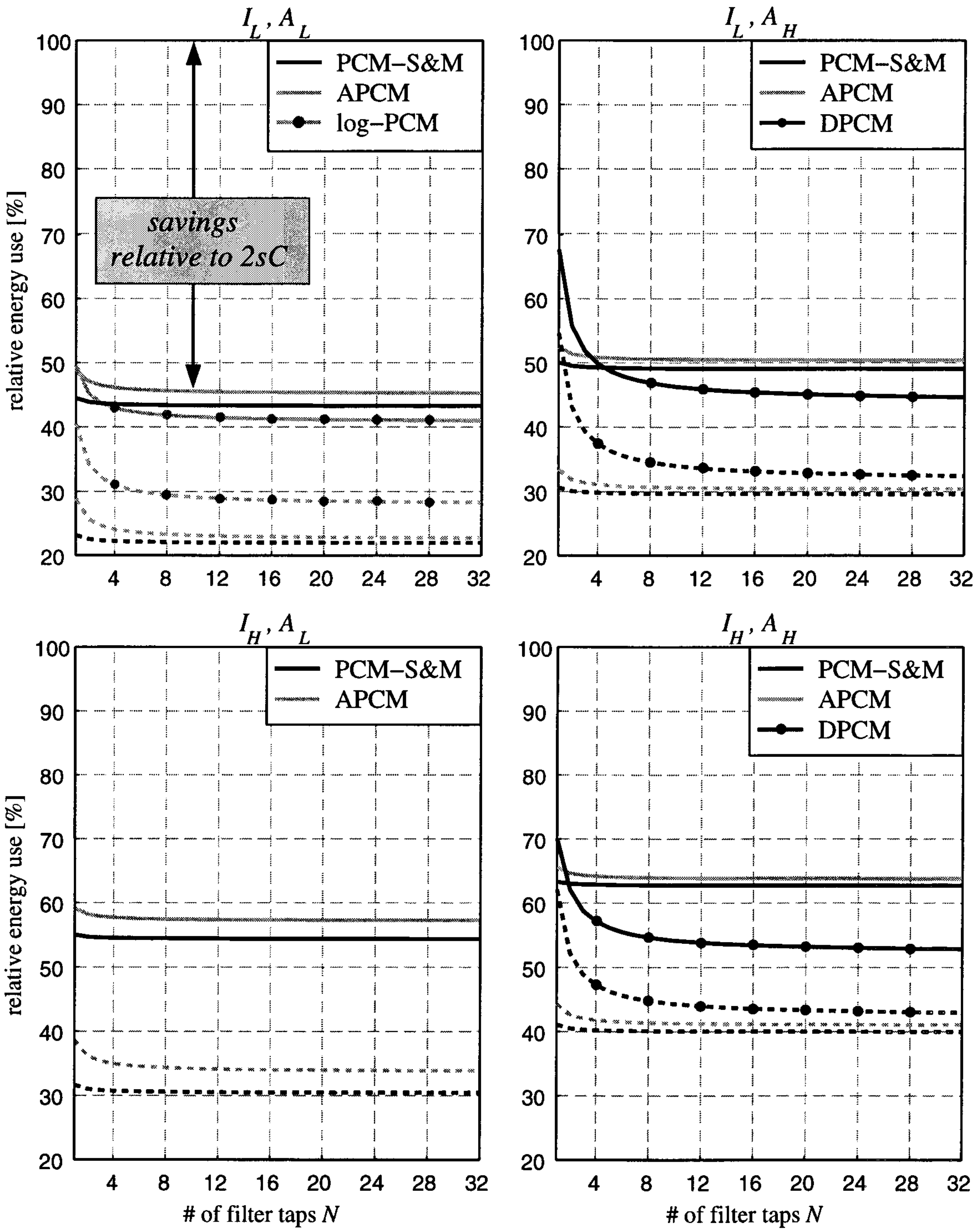


Figure 5.17: Overall energy dissipation relative to PCM-2sC [%] as function of number of filter taps N for low (high) bit rate I_L (I_H) and processing accuracy A_L (A_H). Solid (dashed) lines correspond to high (low) signal amplitude Q_H (Q_L). Encoding schemes that are not available for certain parameter settings have been dropped.

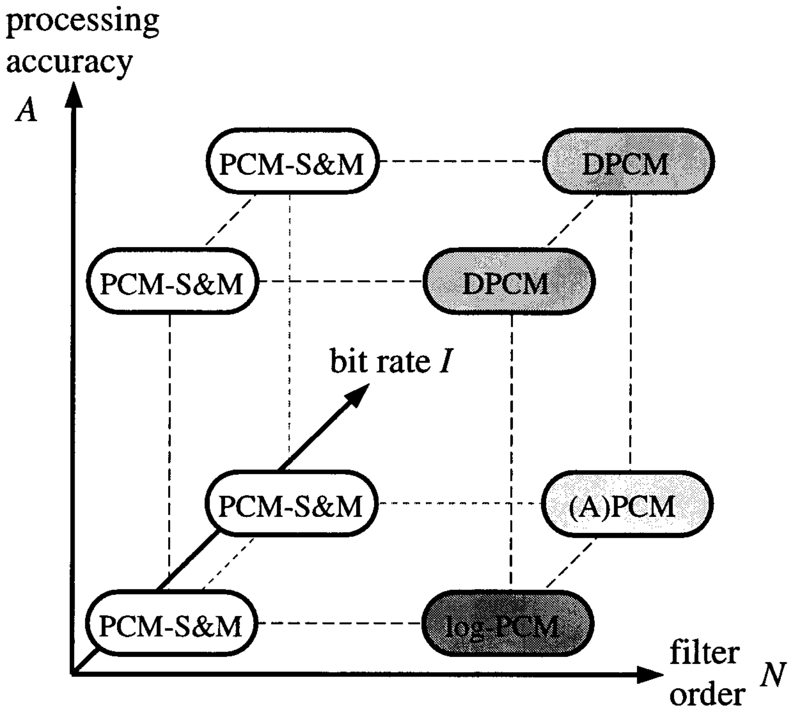


Figure 5.18: Guideline for the choice of energy-efficient encoding in the three-dimensional application space of speech filtering.

Since in most applications signal amplitude will vary, the decision as to which encoding is preferable in a particular situation shall be based on appropriate weighting of power savings associated to Q_L and Q_H. Because high-amplitude signals imply higher absolute energy dissipation than noise-like signals, Q_H should in general be weighted stronger.

According to this policy, figure 5.18 gives a coarse guideline for the choice of energy-efficient encoding in the three-dimensional application space of speech filtering, spanned by parameters bit rate, processing accuracy, and number of MAC operations. In this figure, DPCM and log-PCM relate to (5.15) and (5.25) as before. (A)PCM indicates that, depending on actual word size reduction that can be realized, it might be beneficial to combine S&M with adaptation as in (5.19).

5.6 Summary

Waveform coding techniques such as PCM with sign-magnitude representation, differential PCM, adaptive and logarithmic quantization have been investigated for low-power digital filtering of speech data. Depending on bit rate, processing accuracy, and number of filter taps, guidelines for the choice of coding in application-specific digital signal processors have been established.

Experimental results confirmed linear PCM with sign-magnitude representation as universal coding for energy-efficient implementation of algorithms dominated by multiply-accumulate operations. Depending on signal characteristics and processing accuracy, S&M can save 25-75% of the energy in a single-tap filter with 2's-complement arithmetic. With increasing filter order, other encodings become even more efficient. Differential encoding in conjunction with S&M is particularly useful in case of high processing accuracy and high bit rate. With the proposed circuit structure for linear approximation of logarithms, log-PCM proved very energy-efficient for low processing accuracy and low bit rates.

Furthermore, experimental results imply that the coding overhead for the encoding techniques considered is small enough to justify recoding of speech data even for a single multiply-accumulate operation. Although the results were obtained for FIR filters, the presented encoding techniques and corresponding arithmetic units may as well be adapted to IIR filters and other processing algorithms dominated by multiply-accumulate operations.

Chapter 6

Concluding Remarks

This thesis has explored three major aspects of low-power digital VLSI design in the context of data statistics: estimation of switching activity in logic circuits using probabilistic techniques, lower bounds on power dissipation for a given processing task, and filtering of speech data as example for practical low-power design. The essence of the achieved results and conclusions has been given in the introductory section 1.2.3. More detailed accounts have been provided in the summary sections of chapter 3 (pp. 56), chapter 4 (pp. 110), and chapter 5 (p. 150).

This concluding chapter shall reflect on the prospects for practical application of the work presented in this thesis, and provide suggestions for future work.

6.1 Prospects for Practical Application

Switching activity estimation. Probabilistic activity estimation is a fascinating topic that has stimulated an enormous amount of work within the VLSI research community. On the other hand, it is barely used in practice. This paradox is easily explained by the existence of a more simple alternative, i.e. logic simulation. Even if one could show any probabilistic approach to handle the runtime-accuracy tradeoff more efficiently, which on a fair basis is notoriously difficult to do, we would not expect this situation to change drastically.

There is, however, one application which might represent an exception: Activity-driven gate-level power optimization. The need to cope with a circuit structure that changes during the optimization process could render the shift from a well-tried to an unfamiliar technique necessary. A shift which otherwise seems to go against the force of habit.

Lower bound on power dissipation. Compared to the overall amount of references on low-power digital VLSI available, relatively little work has been reported on the conceptual level, which almost inevitably would lead one to the question of a lower bound on power dissipation. Since practical implications are profound, this fact must be ascribed to the difficulty of the problem and the restricted prospect of success to solve it.

One existing attempt to tackle the lower bound problem is [Sha97]. However, in section 4.7 this approach was shown to suffer from a number of limitations although, according to our taxonomy, it only attacks the most simple subproblem of the lower bound issue. Rightfully, we have to admit not to have found the general solution either, even though at the very outset we fancied this possibility. Now, with the conjecture that there can be no useful implementation-independent lower bound, we are much less euphoric concerning a general solution. As there is little hope that an analytical power dissipation model which comprises all design parameters can be found, a general low-power design theory is out of sight. Therefore, ingenuity and experience will remain desirable features for low-power digital VLSI designers.

Low-power design in consideration of data statistics. In the absence of a general low-power design theory, the practical utility of data statistics for power reduction has been demonstrated by means of an example, i.e. FIR filtering of speech data. To be able to compare the energy efficiency of various data coding schemes, several coding and arithmetic units were designed and optimized manually at the gate level. Since much of this laborious work is specific to the application at hand, the corresponding optimization process is hard to automate, even though a hardware description language and synthesis environment which supports more than the ordinary 2's-complement number representation would be helpful.

However, any time-consuming search in the design space is in deep contrast to today's shorter product development cycles imposed by competitive pressure. In the light of system-level design languages such as SystemC, it appears that this pressure sometimes leads to ranking power dissipation below time-to-market, even for typical low-power applications.

6.2 Future Work

The following suggestions for future work can be given:

Switching activity estimation The extension of the proposed correlation approximation technique to circuits with sequential feedback would be valuable. This approximation then could be incorporated into a tool for the analysis of feedback circuits.

Of course, the integration of the proposed activity estimation scheme in a gate-level power estimation/optimization tool would be a desirable continuation of this thesis.

Lower bound on power dissipation Potential future work on the lower bound problem seems most realistic for the subproblem "Boolean Optimization" (LBP-2) with the simplifying assumptions of a fixed supply voltage and a restricted set of logic gates.

Practical low-power design The waveform coding techniques investigated for FIR filtering of speech could be evaluated for other DSP operations that are frequently encountered in audio applications, e.g. discrete Fourier or cosine transform. Furthermore, a similar investigation for typical DSP data other than speech, such as music, video or multimedia data, would be interesting.

Seite Leer /
Blank leaf

Appendix A

Basics of Probability Theory

This appendix reviews basic concepts from probability theory. In the context of digital systems the discussion is limited to probability measures for discrete events. The treatment is neither complete nor rigorous, but adapted to the needs in the context of probabilistic activity calculation in chapter 3. More information on the subject can be found for instance in [SW86, BS91].

Random Variables

Let sample space Ω be a finite set of events ω. Then, a *discrete random variable* X formally is a mapping $X : \Omega \to \mathbb{R}$. For simplicity we assume $\Omega \subset \mathbb{Z}$ and $X(\omega) = \omega \ , \ \omega \in \Omega$. The *probability mass function* (pmf) $p(x)$ of X is defined as

$$p(x) \ = \ Pr\{X = x\} \tag{A.1}$$

for all $x \in \Omega$. Thus, the pmf fully specifies the statistical properties of X.

The above notation naturally extends to *random vectors* $\boldsymbol{X} = (X_1, \ldots, X_n)$ with X_i $(i = 1, \ldots, n)$ being random variables. The *joint pmf* is defined as

$$p(x_1,\ldots,x_n) \;=\; Pr\{X_1 = x_1,\ldots,X_n = x_n\}\ . \tag{A.2}$$

The *marginal pmf* $p(x_i)$ is obtained by summing over all evaluations of $p(x_1,\ldots,x_n)$ at $x_j = \omega$ for all $j \neq i$ and $\omega \in \Omega$.

The i-th *moment* of a discrete random variable X is defined as

$$\mu_i \;=\; \sum_{x\in\Omega} x^i p(x)\ . \tag{A.3}$$

μ_1 is the *expected value* of X, also denoted as $E[X]$. The i-th *central moment* of X is

$$\lambda_i \;=\; \sum_{x\in\Omega} (x - E[X])^i \cdot p(x)\ . \tag{A.4}$$

The second central moment λ_2 is called the *variance* of X and is denoted σ_X^2:

$$\begin{aligned} \sigma_X^2 &= \sum_{x\in\Omega} (x - E[X])^2 \cdot p(x) \\ &= E[(X - E[X])^2] \\ &= E[X^2] - (E[X])^2\ . \end{aligned} \tag{A.5}$$

$\sqrt{\sigma_X^2} = \sigma_X$ is called the *standard deviation* of X.

The above definitions extend to random vectors. In chapter 3 the following i-th moment of a random vector $\boldsymbol{X} = (X_1,\ldots,X_n)$ is of interest:

$$\mu_{j_1\ldots j_i} \;=\; \sum_{(x_1,\ldots,x_n)\in\Omega^n} x_{j_1}\ldots x_{j_i} \cdot p(x_1,\ldots,x_n)\ . \tag{A.6}$$

The first moments μ_j $(j = 1,\ldots,n)$ are exactly the expected values $E[X_j]$ of the components of $\boldsymbol{X}$.
The second central moments of $\boldsymbol{X}$ follow in analogy to (A.5) as

$$\begin{aligned} \lambda_{j_1 j_2} &= E[(X_{j_1} - E[X_{j_1}])(X_{j_2} - E[X_{j_2}])] \\ &= E[X_{j_1}X_{j_2}] - E[X_{j_1}] \cdot E[X_{j_2}]\ . \end{aligned} \tag{A.7}$$

$\lambda_{j_1 j_2}$ $(j_1, j_2 \in \{1, \ldots, n\})$ is called the *covariance* of the random variables X_{j_1} and X_{j_2} and is a measure for their correlation. If the *correlation coefficient* $\varrho_{X_{j_1} X_{j_2}}$, being defined as

$$\varrho_{X_{j_1} X_{j_2}} = \frac{\lambda_{j_1 j_2}}{\sqrt{\sigma^2_{X_{j_1}} \sigma^2_{X_{j_2}}}} , \tag{A.8}$$

is equal to zero, X_{j_1} and X_{j_2} are said to be uncorrelated. For $j_1 = j_2 = j$ the covariance is identical to the variance and $\varrho_{X_j X_j} = 1$.

Random Sequences

A *random process* $\{X_k\}$ is a collection of random variables indexed by k. If the randam variables have a discrete sample space Ω as above and k assumes values from the set of positive integers $0, 1, 2, \ldots$, then $\{X_k\}$ is called a *random sequence.* Index k can be thought of as the discrete time. As in the context of probabilistic power estimation in chapter 3, the random variables X_k may itself be multi-dimensional, i.e. the X_k may be random vectors.

A random sequence is said to be *strict-sense stationary* if the pmf's associated to any two sequences $X_k, X_{k+1}, \ldots, X_{k+n}$ and $X_{k+\tau}, X_{k+\tau+1}, \ldots, X_{k+\tau+n}$ are identical. This requires that all moments of $X_k, X_{k+1}, \ldots, X_{k+n}$ are independent of time k. As opposed to this, *wide-sense stationarity* only requires $E[X_k]$ and $E[X_k X_{k+\tau}]$, i.e. the first two moments to be independent of k. In general, the term *stationary* is used to indicate time-independence for all statistical properties.

A random sequence $\{X_k\}$ is said to be an *i.i.d. sequence* if all its random variables are independent and identically distributed, i.e. have the same probability mass function $p(x)$.

A random sequence $\{X_k\}$ forms a *Markov chain* if

$$Pr\{X_{k+1} = x_{k+1} | X_k = x_k, \ldots, X_0 = x_0\} = Pr\{X_{k+1} = x_{k+1} | X_k = x_k\} \tag{A.9}$$

for all *states* $x_i \in \Omega$, where $Pr\{X = x|Y = y\}$ is the *conditional probability* of X assuming the value x given that Y has assumed y. Thus, in a Markov chain the next state $k+1$ depends only on the current state k but is conditionally independent of the past. Therefore, a Markov chain is completely specified by the *transition probability matrix* Q with elements q_{ij} $(i, j \in 1, \ldots, |\Omega|)$, denoting the probability of changing from state i to state j in the next time step, i.e. $q_{ij} = Pr\{X_{k+1} = j|X_k = i\}$.

If the probabilities $\pi_i = Pr\{X_k = i\}$, $(i = 1, \ldots, |\Omega|)$ converge for $k \to \infty$, the row vector $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_{|\Omega|}]$ is called *stationary distribution* of the Markov chain and can be obtained from solving the set of linear equations (Chapman-Kolmogorov equation)

$$\begin{aligned} \boldsymbol{\pi} Q &= \boldsymbol{\pi} \\ \pi_1 + \ldots + \pi_{|\Omega|} &= 1 \,. \end{aligned} \tag{A.10}$$

Appendix B

Basics of Information Theory

This section provides an overview of basic quantities and their properties as being used in digital information theory. It builds upon elementary notations from probability theory as given in appendix A. The material widely follows [CT91] in style and naming convention.

Entropy

Let X be a discrete random variable X with alphabet $\mathcal{X}$ and probability mass function (pmf) $p(x)$ as defined in appendix A. The *entropy* or information content, of X, or uncertainty in X, is defined as

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x) \tag{B.1}$$

with the convention $0 \log 0 = 0$. If the log in (B.1) is to the base 2, then entropy is measured in *bits*, which is assumed throughout this text. Alternatively for $H(X)$ one may write $H(p_1, \ldots, p_{|\mathcal{X}|})$ for $p_i = p(x_i)$, $i = 1, \ldots, |\mathcal{X}|$. With the above definition, entropy solely depends on the probability distribution of X, but not on the actual values it can take. Furthermore (B.1) implies

$$0 \leq H(X) \leq \log |\mathcal{X}| \, . \tag{B.2}$$

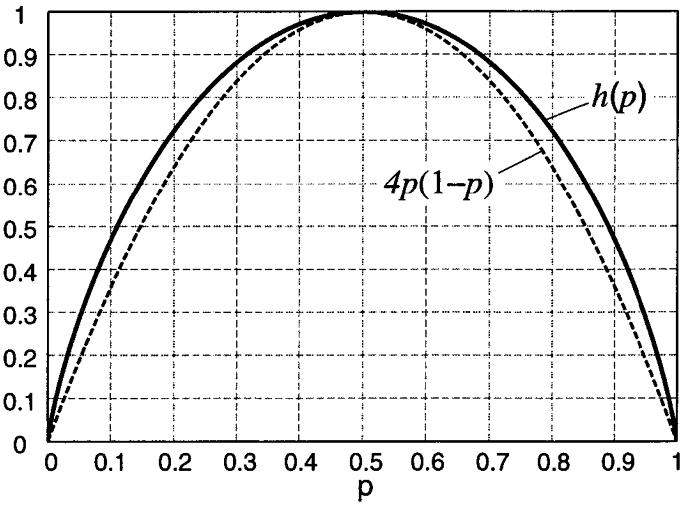


Figure B.1: Binary entropy function $h(p)$ and switching activity function $4p(1-p)$.

The right-side equality holds if and only if $p(x)$ is the uniform distribution, i.e. $p(x) \sim \mathcal{U}$. The *redundancy* in X is defined as the difference between the maximum and the actual entropy:

$$\mathcal{R}(X) \ = \ \log|\mathcal{X}| \ - \ H(X) \ . \tag{B.3}$$

Of particular interest in digital VLSI is the *binary entropy function* $h(p)$ for a binary random variable X with $\mathcal{X} = \{0,1\}$, $p(1) = p$, and $p(0) = 1 - p$:

$$h(p) \ = \ H(X) \ = \ -p\log(p) - (1-p)\log(1-p) \ . \tag{B.4}$$

The graph of $h(p)$ depicted in figure B.1 illustrates that the entropy of a binary random variable is maximized for $p = 0.5$. Furthermore, $h(p)$ is monotonically increasing in the interval (0, 1/2] and decreasing in [1/2, 1). The redundancy of a binary random variable X follows from (B.3) as $\mathcal{R}(X) = 1 - h(p)$. Also note the similarity of the graph for $h(p)$ with the switching activity graph for $\rho = -1$ in figure 3.4. As can be seen

$$h(p) \ \geq \ 4p(1-p) \ , \quad (0 \leq p \leq 1) \ . \tag{B.5}$$

The concept of entropy of a random variable X naturally extends to the *joint entropy* of a collection of random variables $(X_1, X_2, \ldots, X_n)$ with joint probability mass function $p(x_1, x_2, \ldots, x_n)$:

$$H(X_1, \ldots, X_n) = -\sum_{x_1 \in \mathcal{X}_1} \cdots \sum_{x_n \in \mathcal{X}_n} p(x_1, \ldots, x_n) \log p(x_1, \ldots, x_n) .$$

Mutual Information

The *conditional entropy* of a pair of random variables (X, Y) with probability mass function $p(x, y)$ is defined as

$$H(X|Y) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) \log p(x|y) . \tag{B.6}$$

Conditional entropy $H(X|Y)$ can be interpreted as the remaining uncertainty in X given the knowledge of Y. Definition (B.6) ensures correspondence between conditional entropy and conditional probability by virtue of the following *chain rule*:

$$H(X,Y) = H(Y) + H(X|Y) . \tag{B.7}$$

Based on conditional entropy, the *mutual information* between two random variables is defined as

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) . \tag{B.8}$$

Thus, $I(X;Y)$ can be viewed as the reduction in uncertainty in X due to the knowledge of Y. Note that mutual information is symmetric, i.e. X contains as much information about Y as Y contains about X. Mutual information can also be thought of as the "distance" between the joint distribution and the product of the marginal distributions of X and Y: The higher the "distance" between $p(x, y)$ and $p(x)p(y)$, the higher is the correlation between X and Y, and the higher is the mutual information $I(X;Y)$. Figure B.2 explains the relation between entropy and mutual information by means of a Venn diagram.

Entropy Rate

Let $\{X_k\}$ be a random sequence as defined in appendix A. The *entropy rate* $\mathcal{H}$ of $\{X_k\}$ is defined as the average entropy per symbol of a

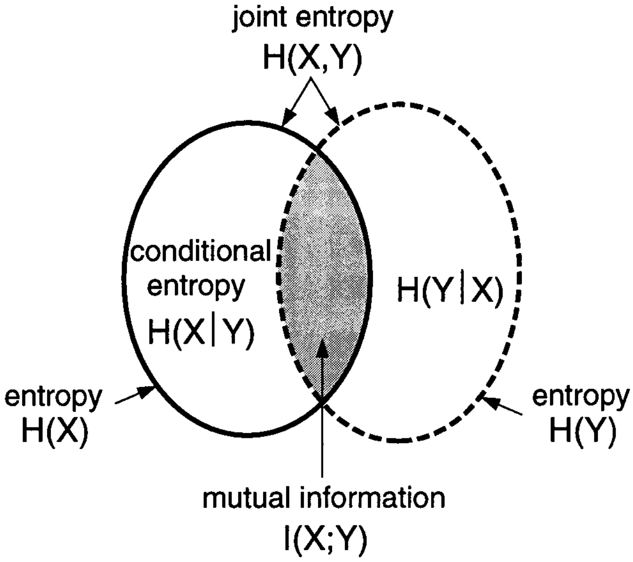


Figure B.2: Relation between entropy and mutual information.

sequence of n random variables as n goes to infinity:

$$\mathcal{H}(\{X_k\}) = \lim_{n\to\infty} \frac{1}{n} H(X_1, X_2, \ldots, X_n) . \tag{B.9}$$

For stationary processes the limit in (B.9) always exists and equals the conditional entropy of the last random variable given the past, i.e. $\mathcal{H}(\{X_k\}) = \lim_{n\to\infty} H(X_n|X_{n-1}, \ldots, X_1)$.

For two types of random processes the entropy rate becomes particularly simple. First, for an independent and identically distributed (i.i.d.) sequence the entropy rate equals the entropy of its random variables:

$$\mathcal{H}(\{X_k\}) = H(X) \quad \text{when} \quad \{X_k\} \text{ i.i.d.} \tag{B.10}$$

Second, if $\{X_k\}$ forms a stationary Markov chain the entropy rate is given by

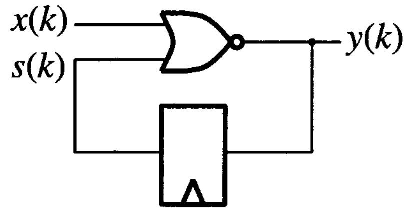
$$\begin{aligned} \mathcal{H}(\{X_k\}) &= H(X_n|X_{n-1}) && \text{(B.11)} \\ &= -\sum_{i,j} \mu_i Q_{ij} \log Q_{ij} && \text{(B.12)} \end{aligned}$$

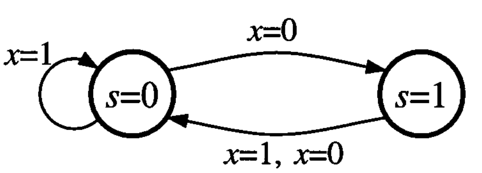
where μ is the stationary distribution and Q is the transition matrix as defined in appendix A.

Appendix C

FSM Analysis Example

By means of the example shown below, this appendix demonstrates how arithmetic transformation[1] can be utilized for the exact analysis of feedback circuits. Assume input x is highly (first-order) anti-correlated ($\varrho_x = -0.6$). Then, state signal s must be modeled as lag-two Markov chain, see theorem 3.3. This is equivalent to using the given moments p_x^1 and $p_{x\text{-}x}^{1\text{-}1}$ for the analysis. On the other hand, a lag-one Markov chain disregards the temporal correlation of x by specifying the state transition probabilities in terms of p_x^1 only. As will be seen, this can result in large errors for the switching activity α_y.





given:

$p_x^1 = \frac{1}{2}$, $p_{x\text{-}x}^{1\text{-}1} = \frac{1}{10}$

to be calculated:

p_s^1, p_{xs}^{11}, $p_{s\text{-}x}^{1\text{-}1}$, $p_{xs\text{-}x}^{11\text{-}1}$

$p_{s\text{-}s}^{1\text{-}1}$, $p_{x\text{-}s}^{1\text{-}1}$, $p_{x\text{-}xs}^{1\text{-}11}$, $p_{s\text{-}xs}^{1\text{-}11}$, $p_{xs\text{-}s}^{11\text{-}1}$, $p_{xs\text{-}xs}^{11\text{-}11}$

α_y

---

[1] The arithmetic transform technique for probabilistic activity calculation in combinational circuits is described in section 3.4. For the meaning of mathematical notations used in the present example, see section 3.3.

Lag-two Markov chain model

Construction of the arithmetic transform for the next-state function[2] $s^{k+1} = \overline{x^k \vee s^k}$ and the extended next-state function obtained from AND-ing both sides with x^{k+1}, yields

$$\begin{aligned} s^{k+1} &= 1 - x^k - s^k + x^k s^k \\ x^{k+1} s^{k+1} &= x^{k+1} - x^k x^{k+1} - s^k x^{k+1} + x^k s^k x^{k+1} \end{aligned}$$

which translates into equations for two of the unknown moments:

$$\begin{aligned} p^{1}_{s} &= 1 - p^{1}_{x} - p^{1}_{s} + p^{11}_{xs} && \text{(C.1)} \\ p^{11}_{xs} &= p^{1}_{x} - p^{1\text{-}1}_{x\text{-}x} - p^{1\text{-}1}_{s\text{-}x} + p^{11\text{-}1}_{xs\text{-}x} \; . && \text{(C.2)} \end{aligned}$$

This brings two other unknowns $p^{1\text{-}1}_{s\text{-}x}$ and $p^{11\text{-}1}_{xs\text{-}x}$ into operation. Under a lag-two Markov chain model $p^{11\text{-}1}_{xs\text{-}x}$ can be expressed in terms of p^{1}_{x}, $p^{1\text{-}1}_{x\text{-}x}$ and p^{11}_{xs} as follows:

$$\begin{aligned} p^{11\text{-}1}_{xs\text{-}x} &= Pr\{x^k = 1, s^k = 1, x^{k+1} = 1\} \\ &= Pr\{s^k = 1, x^{k+1} = 1 | x^k = 1\} Pr\{x^k = 1\} \\ &= Pr\{s^k = 1 | x^k = 1\} Pr\{x^{k+1} = 1 | x^k = 1\} Pr\{x^k = 1\} \\ &= p^{11}_{xs} p^{1\text{-}1}_{x\text{-}x} / p^{1}_{x} \; . && \text{(C.3)} \end{aligned}$$

In (C.3) the third equality holds, because x^{k+1} only depends on x^k but not on previous input values (first-order temporal correlation). On the other hand, s^k depends on x^{k-1} and s^{k-1}. Thus, if x^k is fixed, x^{k+1} and s^k are independent. The derivation of $p^{1\text{-}1}_{s\text{-}x}$ follows a similar reasoning:

$$\begin{aligned} p^{1\text{-}1}_{s\text{-}x} &= p^{11\text{-}1}_{xs\text{-}x} + p^{01\text{-}1}_{xs\text{-}x} \\ &= p^{11\text{-}1}_{xs\text{-}x} + p^{01}_{xs} p^{0\text{-}1}_{x\text{-}x} / p^{0}_{x} \\ &= p^{11\text{-}1}_{xs\text{-}x} + p^{01}_{xs} (p^{1}_{x} - p^{1\text{-}1}_{x\text{-}x}) / (1 - p^{1}_{x}) \; . && \text{(C.4)} \end{aligned}$$

In (C.4) the second term of the right-hand side is derived in analogy to $p^{11\text{-}1}_{xs\text{-}x}$ in (C.3), with $p^{0\text{-}1}_{x\text{-}x} = p^{1}_{x} - p^{1\text{-}1}_{x\text{-}x}$, see (3.9). Finally, p^{01}_{xs} can

---

[2] Time index k is denoted a superscript.

be expressed in terms of moments by constructing the corresponding arithmetic transform $\overline{x^{k+1}}s^{k+1}$:

$$(1-x^{k+1})s^{k+1} = 1-x^{k+1}-x^k(1-x^{k+1})-s^k(1-x^{k+1})+x^k s^k(1-x^{k+1})$$

which translates into

$$p^{01}_{xs} \;=\; 1 - 2p^{1}_{x} + p^{1\text{-}1}_{x\text{-}x} - p^{1}_{s} + p^{1\text{-}1}_{s\text{-}x} + p^{11}_{xs} - p^{11\text{-}1}_{xs\text{-}x} \;. \tag{C.5}$$

With this expression, (C.1)-(C.4) yield a set of linear equations for the unknown moments that refer to the static behavior of state signal s:

$$\begin{pmatrix} 2 & -1 & 0 & 0 \\ 0 & 1 & 1 & -1 \\ \beta & -\beta & 1-\beta & \beta-1 \\ 0 & -p^{1\text{-}1}_{x\text{-}x}/p^{1}_{x} & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} p^{1}_{s} \\ p^{11}_{xs} \\ p^{1\text{-}1}_{s\text{-}x} \\ p^{11\text{-}1}_{xs\text{-}x} \end{pmatrix} = \begin{pmatrix} 1-p^{1}_{x} \\ p^{1}_{x} - p^{1\text{-}1}_{x\text{-}x} \\ \beta \cdot \delta \\ 0 \end{pmatrix}$$

with $\beta = (p^{1}_{x} - p^{1\text{-}1}_{x\text{-}x})/(1-p^{1}_{x})$ and $\delta = 1 - 2p^{1}_{x} + p^{1\text{-}1}_{x\text{-}x}$. Solving this set of equations[3] with the given p^{1}_{x} and $p^{1\text{-}1}_{x\text{-}x}$ gives

$$\underline{p^{1}_{s} = 0.417, \quad p^{11}_{xs} = 0.333, \quad p^{1\text{-}1}_{s\text{-}x} = 0.133, \quad p^{11\text{-}1}_{xs\text{-}x} = 0.067} \;. \tag{C.6}$$

Information on the temporal correlation of the state signal s can be inferred from AND-ing the next-state function with s^k in consideration of the exponent suppression rule, see (3.22). In general, this would eventually result in another set of equations. However, for the present example we have

$$s^k s^{k+1} \;=\; s^k - x^k s^k - s^k + x^k s^k \;\equiv\; 0 \;.$$

From this, the following moments are determined:

$$\underline{p^{1\text{-}1}_{s\text{-}s} \;=\; p^{11\text{-}1}_{xs\text{-}s} \;=\; p^{1\text{-}11}_{s\text{-}xs} \;=\; p^{11\text{-}11}_{xs\text{-}xs} \;=\; 0} \;. \tag{C.7}$$

This result is in accordance with the state transition graph of the example circuit, where there is no self-edge on state $s = 1$. Similarly, from AND-ing the next-state function with x^k, one obtains

$$\underline{p^{1\text{-}1}_{x\text{-}s} \;=\; p^{1\text{-}11}_{x\text{-}xs} \;=\; 0} \;. \tag{C.8}$$

---

[3] Note that there always exists a unique solution because the matrix of coefficients has full rank for any valid combination of values for p^{1}_{x} and $p^{1\text{-}1}_{x\text{-}x}$.

With (C.6)-(C.8) all necessary moments of the circuit input are determined and the switching activity of all gates in the combinational part of the circuit may be calculated by means of the procedure introduced in section 3.4.2. In the present case, the true[4] switching activity α_y is simply given as

$$\underline{\underline{\alpha_y \;=\; 2(p_s^1 - p_{s\text{-}s}^{1\text{-}1}) \;=\; 0.833}}\;. \tag{C.9}$$

Lag-one Markov chain model

Next, the above result shall be compared with the ones available from applying the conventional lag-one Markov chain model to the state signal s, see appendix A. In case of the present example circuit, the one-step state transition probability matrix Q is given as

$$Q \;=\; \begin{bmatrix} p_x^1 & 1-p_x^1 \\ 1 & 0 \end{bmatrix}. \tag{C.10}$$

Solving the corresponding Chapman-Kolmogorov equation, see (A.10), yields the static probability of the state signal:

$$\hat{p}_s^1 \;=\; (1-p_x^1)/(2-p_x^1) \;=\; \frac{1}{3}\,. \tag{C.11}$$

This obviously deviates from the true result in (C.6).

Using this incorrect result and furthermore erroneously assuming that s is temporally uncorrelated, the switching activity of y amounts to

$$\hat{\alpha}_y \;=\; 2\hat{p}_s^1(1-\hat{p}_s^1) \;=\; 0.444 \tag{C.12}$$

which is only half the true value in (C.9). Even exploiting the knowledge of the state-transition probabilities, which is helpful in the present example only because $y = s$, yields

$$\hat{\alpha}_y \;=\; 2\hat{p}_s^1 \;=\; 0.667 \tag{C.13}$$

which is still 20% short of the true value.

Thus, the conventional (lag-one) Markov chain model for finite-state machines only represents a raw approximation for feedback circuits with temporally correlated primary inputs.

---

[4] The result has been verified by logic simulation.

Appendix D

Memory Structures for FIR Filters

This appendix provides an overview on possible realizations of the functional data memory for finite impulse response (FIR) filters, and presents experimental results on power dissipation and area usage for such memory structures. In general, a similar data memory organization is required for the realization of any inner product operation, e.g. in correlator units.

Memory Structures

In an N-tap FIR filter

$$y(k) = \sum_{n=0}^{N-1} c_n x(k-n)$$

each input data sample contributes to N consecutive output samples. Therefore, storage capacity for N input samples is required.

It is assumed, that the FIR filter algorithm is to be performed on a single multiply-accumulate unit in a fully time-shared manner as in section 5.4. The filter coefficients c_n $(n = 0, \ldots, N-1)$ are not known a priori and could be time-varying as in adaptive filters.

Furthermore, the functional data memory shall be built from bistable devices (flip-flops). Then, there are basically three different schemes to realize data storage and access for FIR filtering:

FSR : feedback shift register with fixed read/write position,

SRR : shift register with fixed write and random read access,

RRW : register file with random read/write access.

The register file with random read/write access (RRW) corresponds to a solution with a macro-cell RAM which, however, is not further considered here. Instead, two alternative implementations for RRW are examined:

$\mathrm{RRW_{MX}}$: register file with multiplexed inputs,

$\mathrm{RRW_{CG}}$: register file with gated clock signals.

Figure D.1 shows the the alternative memory structures for any of which the following shall hold:

- New input data samples arrive at a frequency f_S (data sampling frequency).
- Data samples at the output of the memory (input to data path) are registered at frequency $N \cdot f_S$.
- The data path processes the data samples in the following order: $x(k), x(k-1), \ldots, x(k-N+1)$.
- Two in-phase clock signals of frequencies f_S and $N \cdot f_S$ are available at no cost.

Note that the four memory structures in figure D.1 not only differ in their hardware complexity, but also map the same input signal statistics to node activities in distinct ways.

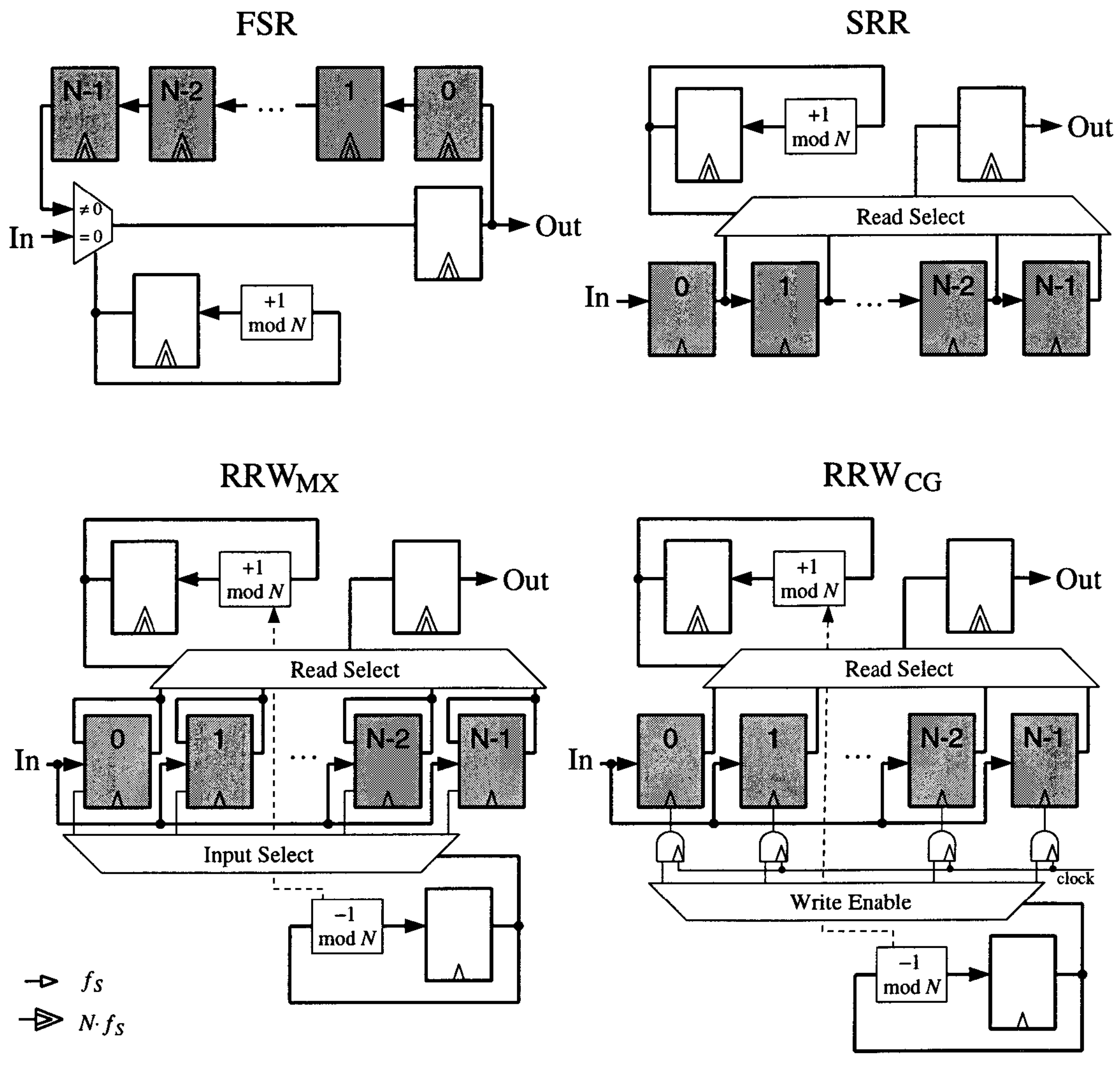


Figure D.1: Alternative memory structures for FIR filtering.

Experimental Evaluation

For each memory structure a VHDL model has been generated. Using `Synopsys DesignCompiler`, netlists were synthesized (0.25μm CMOS standard cell library) for $N = 8, 16, 32, 64, 128$ and a data word width of 12 bits. Each netlist was optimized for minimum area. Then, the gate-level netlists have been exercised with speech stimuli ($f_S = 8$ kHz) in 2's-complement representation. Activities including glitches have been captured by utilizing gate-delay models provided by the technology library. Finally, average power consumption

# of taps N	Power				Area			
	FSR	SRR	$\mathrm{RRW_{MX}}$	$\mathrm{RRW_{CG}}$	FSR	SRR	$\mathrm{RRW_{MX}}$	$\mathrm{RRW_{CG}}$
8	3.9	1.3	1.4	1.0	19	22	31	24
16	12.7	2.8	2.9	2.4	33	39	56	43
32	46.5	5.8	6.2	5.2	61	74	106	80
64	178.8	13.0	14.0	11.7	117	142	204	151
128	704.8	31.4	29.5	26.3	227	279	399	294

Table D.1: Power dissipation [μW] and area usage [$10^3\mu m^2$] for different memory structures of FIR filters (data word width = 12 bits).

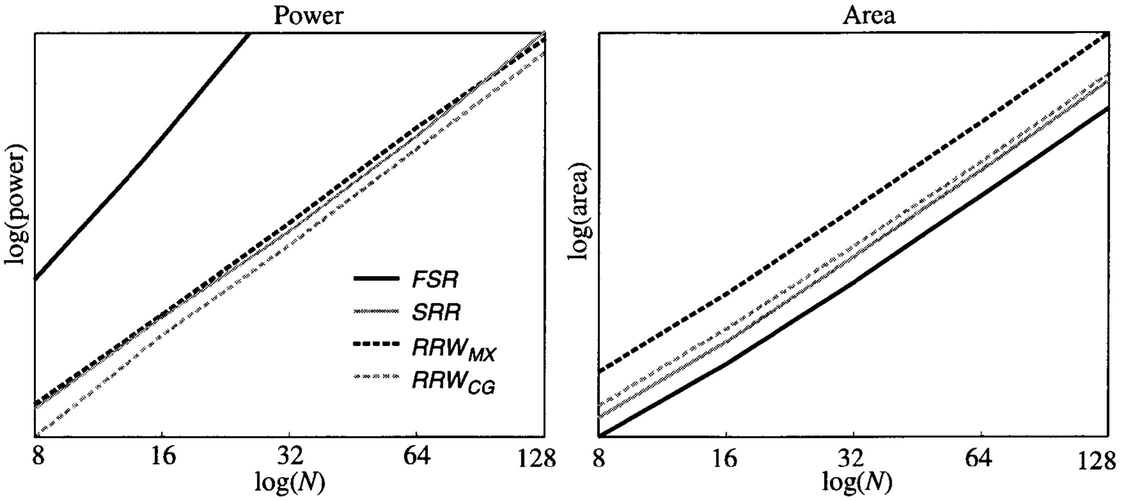


Figure D.2: Interpolated data from table D.1 on double-logarithmic scale.

has been computed for estimated capacitive loads using `Synopsys DesignPower`. Table D.1 summarizes the numerical results. In figure D.2 these results have been interpolated and plotted on a double-logarithmic scale.

The experimental results permit the following conclusions:

- While memory structures SRR, $\mathrm{RRW_{MX}}$, and $\mathrm{RRW_{CG}}$ have comparable power dissipation, the minimum hardware solution FSR dissipates exorbitantly more power. This is due to the clocking of the registers at a frequency of $N \cdot f_S$ for FSR.
- Memory structure $\mathrm{RRW_{CG}}$, which uses clock-gating, achieves the minimum power dissipation of all alternatives. With in-

creasing data word width, the advantage of $\mathrm{RRW_{CG}}$ is expected to grow further because the relative overhead due to the clock-gating circuitry will diminish.

- $\mathrm{RRW_{MX}}$ with multiplexed flip-flop inputs is the least desirable memory structure, since it implies the largest area cost while being less energy-efficient than $\mathrm{RRW_{CG}}$.
- Power dissipation and area cost grow nearly linear with respect to the number of filter taps N for all considered memory structures.

Seite Leer /
Blank leaf

Bibliography

[Abr63] N. Abramson. *Information Theory and Coding.* Electronic Science Series. McGraw-Hill, 1963.

[ALES98] A. Alvandpour, P. Larsson-Edefors, and C. Svensson. Impact of Miller capacitances on power consumption. In *Proc. 8th Int. Workshop Power and Timing Modeling, Optimization and Simulation (PATMOS'98)*, Lyngby DK, Oct. 1998.

[AMD+94] M. Alidina, J. Monteiro, S. Devadas, A. Ghosh, and M. Papaefthymiou. Precomputation-based sequential logic optimization for low power. *IEEE Trans. VLSI Syst.*, 2(4):426–436, Dec. 1994.

[Ben73] C.H. Bennett. Logical reversibility of computation. *IBM J. Res. Develop.*, 17:525–532, 1973.

[BG99] V.A. Bartlett and E. Grass. A low-power FIR filter exploiting data-dependent operation. In *Proc. 9th Int. Workshop Power and Timing Modeling, Optimization and Simulation (PATMOS'99)*, 1999.

[BGL+98] C.H. Bennett, P. Gács, M. Li, P.M.B. Vitányi, and W.H. Zurek. Information distance. *IEEE Trans. Inform. Theory*, 44(4):1407–1423, 1998.

[BL85] C.H. Bennett and R. Landauer. The fundamental physical limits of computation. *Scientific American*, pages 38–46, 1985.

[BMM$^+$00] L Benini, A. Macii, E. Macii, M. Poncino, and R. Scarsi. Architectures and synthesis algorithms for power-efficient bus interfaces. *IEEE Trans. Computer-Aided Design*, 19(9):969–980, Nov. 2000.

[BNYT93] R. Burch, F.N. Najm, P. Yang, and T.N. Trick. A Monte Carlo approach for power estimation. *IEEE Trans. VLSI Syst.*, 1(1):63–71, March 1993.

[BS91] I.N. Bronstein and K.A. Semendjajew. *Taschenbuch der Mathematik.* B.G. Teubner Verlagsgesellschaft, 25 edition, 1991. (in German).

[BTR00] D.K. Brock, E.K. Track, and J.M. Rowell. Superconductor ICs: The 100-GHz second generation. *IEEE Spectrum*, Dec. 2000.

[C$^+$98] D.I. Cheng et al. A hybrid methodology for switching activities estimation. *IEEE Trans. Computer-Aided Design*, 17(4):357–366, Apr. 1998.

[CA90] K.-T. Cheng and V. Agrawal. An entropy measure for the complexity of multi-output Boolean functions. In *Proc. Design Automation Conf.*, pages 302–305, 1990.

[CB95] A. P. Chandrakasan and R. W. Brodersen. *Low Power Digital CMOS Design.* Kluwer, Norwell, MA, 1995.

[CMD97] J.C. Costa, J.C. Monteiro, and S. Devadas. Switching activity estimation using limited depth reconvergent path analysis. In *Proc. Int. Symp. Low Power Electronics and Design*, pages 184–189, 1997.

[CSY96] D.-S. Chen, M. Sarrafzadeh, and G.K.H. Yeap. State encoding of finite state machines for low power design. *J. Circuits, Systems and Computers*, 6(6):649–661, Dec. 1996.

[CT91] T.M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley and Sons, 1991.

[CZV65] M Combet, H. Zonneveld, and L. Verbeek. Computation of the base two logarithm of binary numbers. *IEEE Trans. Electron. Comput.*, EC-14:863–867, Dec. 1965.

[DTP98] C.-S. Ding, C.-Y. Tsui, and M. Pedram. Gate-level power estimation using tagged probabilistic simulation. *IEEE Trans. Computer-Aided Design*, 17(11):1099–1107, Nov. 1998.

[GDKW92] A. Ghosh, S. Devadas, K. Keutzer, and J. White. Estimation of average switching activity in combinational and sequential circuits. In *Proc. Design Automation Conf.*, pages 253–259, 1992.

[Ger96] N. Gershenfeld. Signal entropy and the thermodynamics of computation. *IBM Systems Journal*, 35(3&4):577–586, 1996.

[Hel72] L. Hellermann. A measure of computational work. *IEEE Trans. Comput.*, C-21:439–446, May 1972.

[Hey99] A.J.G. Hey, editor. *Feynman and Computation: Exploring the Limits of Computers.* Perseus Books, 1999.

[HMPS96] G.D. Hachtel, E. Macii, A. Pardo, and F. Somenzi. Markovian analysis of large finite state machines. *IEEE Trans. Computer-Aided Design*, 15(12):1479–1493, Dec. 1996.

[Hwa79] K. Hwang. *Computer Arithmetic: Principles, Architecture and Design.* Wiley & Sons, 1979.

[HYH99] M.C. Hansen, H. Yalcin, and J.P. Hayes. Unveiling the ISCAS-85 benchmarks: A case study in reverse engineering. *IEEE Design & Test of Comput.*, pages 72–80, July-Sept. 1999.

[IP97] S. Iman and M. Pedram. *Low Power Design in Deep Submicron Electronics*, chapter Combinational Circuit Optimization. Kluwer, 1997.

[JN84] N.S. Jayant and P. Noll. *Digital Coding of Waveforms.* Prentice-Hall, 1984.

[Kae00] H. Kaeslin. *VLSI II: Design of Very Large Scale Integration Circuits.* Lecture notes, Integrated Systems Laboratoty, ETH Zürich, 2000.

[KAK99] A. Kabbani and A.J. Al-Khalili. Estimation of ground bounce effects on CMOS circuits. *IEEE Trans. Components and Packaging Technology*, 22(2):316–325, June 1999.

[KB98] M. Kahrs and K. Brandenburg, editors. *Applications of Digital Signal Processing to Audio and Acoustics.* Kluwer Academics, 1998.

[Key01] R.W. Keyes. Fundamental limits of silicon technology. *Proc. of the IEEE*, pages 227–239, March 2001.

[Kon94] A.M. Kondoz. *Digital Speech: Coding for Low Bit Rate Communications Systems.* John Wiley & Sons, 1994.

[KS99] L.M. Krauss and G.D. Starkman. The fate of life in the universe. *Scientific American*, pages 36–43, Nov. 1999.

[Lan61] R. Landauer. Irreversibility and heat generation in the computing process. *IBM J. Res. Develop.*, 5:183–191, 1961.

[Lew90] D.M. Lewis. An architecture for addition and subtraction of long word length numbers in the logarithmic number system. *IEEE Trans. Comput.*, 39(11):1325–1336, Nov. 1990.

[LNC96] J.T. Ludwig, S.H. Nawab, and A.P. Chandrakasan. Low-power digital filtering using approximate processing. *IEEE J. Solid-State Circuits*, 31(3):395–400, 1996.

[LR90] H. Leff and A. Rex, editors. *Maxwell's Demon: Entropy, Information, Computing.* Princeton University Press, 1990.

[MD00] J.D. Meindl and J.A. Davis. The fundamental limit on binary switching energy for terascale integration (TSI). *IEEE J. Solid-State Circuits*, 35(10):1515–1516, Oct. 2000.

[MDG+97] J. Monteiro, S. Devadas, A. Ghosh, K. Keutzer, and J. White. Estimation of average switching activity in combinational logic circuits using symbolic simulation. *IEEE Trans. Computer-Aided Design*, 16(1):121–127, Jan. 1997.

[Mei95] J.D. Meindl. Low power microelectronics: Retrospect and prospect. *Proc. of the IEEE*, 83(4):619–635, Apr. 1995.

[Mit62] J.N. Mitchell. Computer multiplication and division using binary logarithms. *IRE Trans. Electron. Comput.*, EC-11:512–517, 1962.

[MMP95] R. Marculescu, Diana Marculescu, and M. Pedram. Efficient power estimation for highly correlated input streams. In *Proc. 32nd Design Automation Conf.*, pages 628–634, San Francisco, CA, 1995.

[MMP96] D. Marculescu, R. Marculescu, and M. Pedram. Information theoretic measures for power analysis. *IEEE Trans. Computer-Aided Design*, 15(6):599–610, June 1996.

[MMP97] R. Marculescu, D. Marculescu, and M. Pedram. Composite sequence compaction for fine-state machines using block entropy and high-order Markov models. In *Proc. Int. Symp. Low Power Electronics and Design*, pages 190–195, 1997.

[MMP98] R. Marculescu, D. Marculescu, and M. Pedram. Probabilistic modeling of dependencies during switching activity analysis. *IEEE Trans. Computer-Aided Design*, 17(2):73–83, Feb. 1998.

[MMP99] R. Marculescu, D. Marculescu, and M. Pedram. Sequence compaction for power estimation: Theory and practice. *IEEE Trans. Computer-Aided Design*, 18(7):973–993, July 1999.

[MPS98] E. Macii, M. Pedram, and F. Somenzi. High-level power modeling, estimation, and optimization. In *IEEE Trans. Computer-Aided Design*, volume 17, pages 1061–1079, Nov. 1998.

[NN96a] M. Nemani and F.N. Najm. High-level power estimation and the area complexity of Boolean functions. In *Proc. Int. Symp. Low Power Electronics and Design*, pages 329–334, 1996.

[NN96b] M. Nemani and F.N. Najm. Towards a high-level power estimation capability. *IEEE Trans. Computer-Aided Design*, 15(6):588–598, June 1996.

[NN99] M. Nemani and F.N. Najm. High-level area and power estimation for VLSI circuits. *IEEE Trans. Computer-Aided Design*, 18(6):697–713, 1999.

[NOC$^+$97] S.H. Nawab, A.V. Oppenheim, A.P. Chandrakasan, J.M. Winograd, and J.T. Ludwig. Approximate signal processing. *J. VLSI Signal Processing Systems*, 15:177–200, Jan. 1997.

[NS99] L.S. Nielsen and J. Sparso. Designing asynchronous circuits for low power: An IFIR filter bank for a digital hearing aid. *Proc. of the IEEE*, 87(2):268–281, 1999.

[Opp70] A.V. Oppenheim. Realization of digital filters using block floating point arithmetic. *IEEE Trans. Audio and Electroacoustics*, 18:130–136, 1970.

[Pan00] C.J. Pan. A stereo audio chip using approximate processing for decimation and interpolation filters. *IEEE Trans. Solid-State Circuits*, 35(1):45–55, Jan. 2000.

[PM75] K. Parker and E. McClusky. Probabilistic treatment of general combinational networks. *IEEE Trans. Electron. Comput.*, C-24(6):668–670, 1975.

[RB97] K. Ralev and P. Bauer. Implementation options for block floating point digital filters. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP-97)*, volume 3, pages 2197–2200, 1997.

[RSH99a] S. Ramprasad, N.R. Shanbhag, and I.N. Hajj. Decorrelating (DECOR) transformations for low-power digital filters. *IEEE Trans. Circuits and Syst.*, 46(6):776–787, June 1999.

[RSH99b] S. Ramprasad, N.R. Shanbhag, and I.N. Hajj. Signal coding for low power: Fundamental limits and practical realizations. *IEEE Trans. Circuits and Syst.-II*, 46(7):923–929, July 1999.

[S$^+$95] D. Singh et al. Power conscious CAD tools and methodologies: A perspective. *Proc. of the IEEE*, 83(4):570–594, Apr. 1995.

[SA75] E.E. Swartzlander and A.G. Alexopoulos. The sign/logarithm number system. *IEEE Trans. Comput.*, C-24:1238–1242, Dec. 1975.

[SBG99] S.L. SanGregory, C. Brothers, and D. Gallagher. A fast, low-power logarithm approximation with CMOS VLSI implementation. In *Proc. 42nd Midwest Symposium on Circuits and Systems*, 1999.

[Sch99] A. Schaub. *Hearing aid.* Patent P1573, Bernafon AG, Switzerland, 1999.

[SF96] T. Sasao and M. Fujita, editors. *Representation of Discrete Functions.* IFIP WG 10.5 Workshop on Application of the Reed-Muller Expansion in Circuit Design. Kluwer, 1996.

[Sha48] C.E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423,623–656, July, Oct. 1948.

[Sha97] N.R. Shanbhag. A mathematical basis for power-reduction in digital VLSI systems. *IEEE Trans. Circuits and Syst.*, 44(11):935–951, Nov. 1997.

[SK96] P.H. Schneider and S. Krishnamoorthy. Effects of correlations on accuracy of power analysis - an experimental study. In *Proc. Int. Symp. Low Power Electronics and Design*, pages 113–116, Monterey, CA, 1996.

[SP00] J. Satyanarayana and K. Parhi. Theoretical analysis of word-level switching activity in the presence of glitching and correlation. *IEEE Trans. VLSI Syst.*, 8(2):148–159, Apr. 2000.

[SQA] Sound quality assessment material. European Broadcasting Union. CDROM (Cat.No 422 204-2).

[SR00] A.M. Steane and E.G. Rieffel. Beyond bits: The future of quantum information processing. *IEEE Computer*, pages 38–45, Jan. 2000.

[SRB97] N. Sankarayya, K. Roy, and D. Bhattacharya. Algorithms for low power FIR filter realization using differential coefficients. In *Proc. 10th Int. Conf. on VLSI Design*, pages 174–178, Jan. 1997.

[SSW96] P.H. Schneider, U. Schlichtmann, and B. Wurth. Fast power estimation of large circuits. *IEEE Design & Test of Comput.*, pages 70–78, Spring 1996.

[SW86] H. S. Stark and J.W. Woods. *Probability, Random Processes, and Estimation Theory for Engineers.* Prentice-Hall, 1986.

[Szi29] L. Szilard. On the decrease of entropy in a thermodynamic system by the intervention of intelligent beings. *Z. f. Physik*, 53:840–856, 1929. (in German).

[T^+95] C-Y. Tsui et al. Power estimation methods for sequential logic circuits. *IEEE Trans. VLSI Syst.*, 3(3):404–416, 1995.

[TTSG97] S. Theoharis, G. Theodoridis, D. Soudris, and C. Goutis. A new method for switching activity estimation of logic level networks. In *Proc. 7th Int. Workshop Power and Timing Modeling, Optimization and Simulation (PATMOS '97)*, pages 131–140, September 8–11 1997.

[WKF00] J. Wassner, H. Kaeslin, and N. Felber. Experimental speech data analysis with application to low-power DSP. Technical report, ETH Zurich, Integrated Systems Laboratory, 2000.

[WKFF99] J. Wassner, H. Kaeslin, N. Felber, and W. Fichtner. Spectral transformation of Boolean functions for probabilistic power estimation. In *Proc. 9th Int. Workshop Power and Timing Modeling, Optimization and Simulation (PATMOS'99)*, pages 169–178, Greece, Oct. 1999.

[WKFF00] J. Wassner, H. Kaeslin, N. Felber, and W. Fichtner. Speech coding for energy-efficient digital signal processing. In *CDROM-Proc. 43rd IEEE Midwest Symposium on Circuits and Systems (paper 05/001)*, Lansing, MI, USA, Aug. 2000.

[WKFF01] J. Wassner, H. Kaeslin, N. Felber, and W. Fichtner. Waveform coding for low-power digital filtering of speech data. submitted to *IEEE Trans. Signal Processing*, 2001.

[XN94] M.G. Xakellis and F.N. Najm. Statistical estimation of the switching activity in digital circuits. In *Proc. 31rd Design Automation Conf.*, pages 728–733, San Diego, CA, 1994.

[Yea98] G.K. Yeap. *Practical Low Power Digital VLSI Design.* Kluwer Academic Publishers, 1998.

[YTK98] L.-P. Yuan, C.-C. Teng, and S.-M. Kang. Statistical estimation of average power dissipation using nonparametric techniques. *IEEE Trans. VLSI Syst.*, 6(1):65–73, March 1998.

Seite Leer /
Blank leaf

Curriculum Vitae

Jürgen Wassner was born in Lauchhammer, Germany, on April 9, 1967. He received a technical degree in electronics before enrolling in Electrical Engineering at the Technical University of Dresden, Germany, in 1990. From the same institution he received the Dipl.-Ing. (M.Sc.) degree (with distinction) in 1995 for his diploma thesis on graph-theoretical methods in control system analysis.

During 1994-95 he spent six months as research scholar at the Department of Electrical and Computer Engineering at the University of Wisconsin, USA. There he contributed to the design of new algorithms for the modeling of electric power systems.

In 1995 he joined the European subsidiary of Silicon Valley Group (SVG) Inc. as System Engineer, working on lithographic equipment for the semiconductor industry. In this position he was promoted to supervise the technical support and advancement of SVG supplies at the semiconductor fab MOS4YOU of Philips Inc., Netherlands.

Since 1997 he has been working as research and teaching assistant at the Integrated Systems Laboratory of the Swiss Federal Institute of Technology (ETH) Zürich, Switzerland, towards his Doctoral degree. His research interest embraces all aspects of digital low-power VLSI design, specifically those related to data statistics, digital signal processing, and computer arithmetic.